WISE VIDEO: USING IN-BAND WIRELESS LOSS NOTIFICATION TO IMPROVE RATE-CONTROLLED VIDEO STREAMING

Athina Markopoulou, Eric Setton, Mark Kalman *John Apostolopoulos*

Stanford University
{amarko,esetton,mkalman}@stanford.edu

HP Labs, Palo Alto, CA
japos@hpl.hp.com

ABSTRACT

Both data and multimedia applications over the Internet are expected to perform some kind of congestion control, typically using packet loss as an indication for congestion. However, when packets are lost due to wireless errors, decreasing the rate unnecessarily harms the application's performance. This paper proposes to use an in-band notification mechanism, called WiSE, to distinguish wireless errors from congestion losses and improve the performance of rate-controlled video streamed over wireless links. A WiSE agent on the wireless network identifies wireless errors and piggy-backs this information onto other video packets. The WiSE-aware video source can benefit from this notification (1) by avoiding unnecessary decreases in the sending rate in response to wireless errors, and (2) by accurately adjusting the error resilience for the wireless link. Simulations demonstrate that WiSE provides significant improvement in video quality over a wide range of conditions.

1. INTRODUCTION

Wireless links are becoming an increasingly important part of the Internet, especially for last hop access. In general, they are characterized by lower bandwidth and higher loss rates, due to contention, retransmissions, interference, and intermittent connectivity due to handoffs. On the other hand, the dominant part of today's Internet traffic is carried by TCP, which interprets packet loss as an indication of congestion and decreases its sending rate to alleviate the congestion. Similarly, video traffic sent over the Internet is also expected to conform to the congestion avoidance rules.

It is well known that TCP performs suboptimally when packets are lost not due to congestion but due to corruption on wireless link(s). Proposals within the networking community [1, 2, 3] have shown that it is possible to identify the cause of packet loss at the network layer and improve the TCP reaction appropriately. However, this idea has received limited attention for sending video over hybrid wired/wireless links. Intuitively, identifying the cause of packet loss enables the video source to (i) avoid unnecessary rate reduction, and associated performance degradation, in case of wireless loss, and (ii) enable accurate error resilience reactions, such as fast retransmits of lost packets.

This paper examines wireless loss notification for rate-controlled video, using the in-band signaling mechanism called WiSE (*"Wireless Signaling via ECN"*, proposed in [3]) and considering two different streaming protocols for rate adaptation. We demonstrate that the use of WiSE can bring significant improvement in video quality for a large range of conditions, i.e. for various wireless error rates and loss patterns, congestion loads, and two streaming protocols. The paper continues by describing the two streaming protocols we consider and their associated rate-control algorithms. Section 3 describes the WiSE wireless loss notification mechanism. Section 4 describes the simulation setup. Section 5 presents simulation results that demonstrate the performance gains under a wide range of conditions, and Section 6 concludes the paper.

2. RATE-CONTROLLED VIDEO SCHEMES

With our goal of improving rate-controlled video, the choice of rate control algorithms is of critical importance. From the wide range of congestion control mechanisms that are employed in practice, this paper examines two protocols: SSRC and HTTP streaming, which are at two ends of the spectrum for TCP friendliness.

Switch stream or *SSRC* is a reasonably generic video - centric protocol over UDP. We consider the common case where the sender stores multiple copies of the same content at different bit-rates (e.g. using different quantization) and switches between streams within a single session, in reaction to feedback from the receiver. The receiver sends a negative acknowledgment (NACK) immediately when it detects a gap in the sequence numbers. To limit the load on the back channel only NACKs, and not ACKs, are sent by the receiver. Upon reception of a NACK, the sender switches to a lower bitrate stream, at the beginning of the next GOP. The change in video rate is limited to one stream, even if multiple NACKs are received in the same GOP. If no loss is reported within n_1 GOPs, the sender switches to a higher stream. To avoid oscillations, we do not allow rate increases beyond the maximum supportable rate given the bandwidth and the load, nor increases soon after the last (n_2 GOPs) increase.

We also consider the widely used *HTTP Streaming*, where a single video bitstream is delivered using traditional TCP. This is a network - centric scenario: it is entirely TCP friendly but the window-based congestion control and the abrupt rate changes are unnatural for video traffic.

These two protocols serve as case studies to demonstrate the benefit from WiSE. We chose them to be at the two ends of the TCP friendliness spectrum: SSRC, although adaptive, has no notion of TCP friendliness, while HTTP streaming uses TCP itself. In future work, we plan to apply WiSE to streaming protocols whose rate control lies in-between, e.g. equation-based approaches [4].

3. WISE: WIRELESS LOSS NOTIFICATION

The idea of wireless loss notification for TCP is well understood within the networking community. Ref. [1, 2, 3] describe mechanisms for handling wireless errors. Ref. [1] shields the sender from the wireless link by using local layer retransmissions. Ref. [2, 3] both signal the cause of loss to the sender. There are different ways to detect wireless errors, depending on the environment. In the case of 802.11, a packet may be dropped after a number of retransmissions, but the IP header is available to the transmitting access point

Table 1. Foreman encoded into five different streams.

stream number	0	1	2	3	4
bitrate (kbps)	50	78	116	184	346
avg PSNR (dB)	27.4	30.3	32.9	35.7	39.1

or station. In general, corruption can be detected from an incorrect checksum or from the absence of an ACK.

In the context of video streaming, the idea of wireless loss notification is relatively unexplored. Recent work in [5], has proposed a proxy at the border between wired network and 3G wireless network, to monitor existing flows and send statistical feedback back to the sender. This approach has been extended in [6] with timely reports and in [7] with an R-D optimization framework. Our main difference from the above approaches is our in-band notification mechanism. We also explore a wide range of realistic network conditions.

We choose to use the WiSE mechanism, proposed in [3], that provides in-band signaling from the network to the application. It uses the ECN bits of the TCP/IP header, which are lightly utilized in practice, to opportunistically signal extra information. Ref. [3] showed that, with appropriate coding, ECN bits on packets of the same flow can be viewed as a *multiplexed ECN channel* and the spare capacity can be used without effecting the standard ECN messages. They studied the capacity of the ECN channel, and showed the performance boost that WiSE brings to TCP traffic and its peaceful coexistence with WiSE-unaware routers and flows. WiSE consists of two components: (1) a WiSE-Agent at the queue that detects the wireless error and appropriately piggy-backs this information on the ECN field of subsequent packets of the corrupted flow, and (2) a WiSE-aware application that can decode the messages sent by the WiSE agent and appropriately adjust its sending policy. Due to lack of space, we refer the reader to [3] for details on the encoding/decoding of messages by WiSE . In summary, WiSE provides the following benefits. It is simple, it does not introduce extra messages (bandwidth overhead) or modifications to the existing structure of packet headers, and it is incrementally deployable (it does not interfere with WiSE-unaware traffic and routers). Furthermore, it can gracefully generalize to a network-to-application notification framework for signaling a wide range of information, e.g. packet-specific or of statistical nature.

In this paper, we use the WiSE mechanism for signaling wireless errors to the video application, which can take advantage of this information in two ways: (i) by avoiding unnecessary decreases in the sending rate, and (ii) by accurately increasing the error resilience on the wireless link. In particular, WiSE can help the two streaming schemes of the previous section as follows. A WiSE-Agent attached on the wireless link detects which packets are corrupted and signals the information to the receiver. The SSRC receiver includes this information in its NACKs. Upon reception of a wireless NACK, the SSRC sender not only does not switch to a lower stream but also immediately retransmits the corrupted packets. HTTP streaming can benefit from WiSE in the same way that TCP did in [3]: by avoiding to half the window every time a packet is lost due to wireless errors. We call these modified versions of SSRC and HTTP streaming which make use of WiSE, *WiSE-SSRC* and *WiSE-HTTP* respectively.

4. SIMULATION SETUP

As our video sequences, we use the first 10sec of Foremam (QCIF, 30 fps), encoded using H.264 (jm61e encoder) into five different streams (using different quantization). The rates and qualities of these streams are shown in Table 1. In order to allow SSRC to quickly switch between streams, we have frequent I frames- one every 15 frames (the GOP size). RTP packetization is done according to [8], with 33MB/slice. When a slice is lost, the entire frame is lost and prior frame error concealment is used, i.e. the lost frame is estimated as the last correctly received frame. With streaming applications in mind, we consider low (fixed) playout delays in the order of seconds, in particular 1sec for SSRC and 3sec for HTTP.

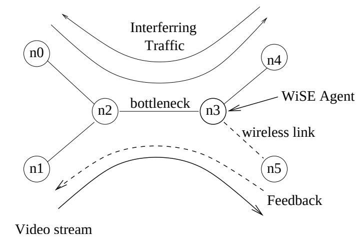

Fig. 1. Simulation Topology

We use the Network Simulator (NS) [9], to simulate the topology and our streaming protocols. The topology is shown in Fig. 1: it is a classic (shared wired bottleneck - last hop wireless) topology. Video is streamed from n1 to n4 through path n1-n2-n3-n4. Interfering traffic (if any) goes from n0 to n5 through path n0-n2-n3-n5, sharing the bottleneck link n2-n3 with the video stream. All links are wired (with bandwidth 10Mbps and delay 10ms) and only the last hop in the video path (n2-n3) is wireless (with bandwidth 1.1Mbps and delay 100ms). Apart from the constant propagation delays, there is also jitter introduced by queueing delay in the bottleneck link.

Two models are used to simulate wireless loss: Bernoulli and a two-state Markov model with average burst length of 3 packets, as per [10, 3]. The average loss rate is a parameter that we vary. For a given sequence of lost packets, we calculate the PSNR for each frame and the average PSNR across frames. Futhermore, we consider different realizations (random seeds) of the same average loss rate and we compute the average PSNR over them. Specifically, in Fig. 3 and Fig. 6, for each loss rate in the wireless, we consider ten realizations (ten random seeds). In Fig. 4, to obtain a PSNR value for one pair of (% congestion, % corruption), we also consider ten realizations (ten pairs of random seeds) for each pair (% congestion, % corruption) and we average over them. In all figures, except Fig. 4 that will be discussed in length, we simulate interfering traffic by using a realistic traffic mix of long FTP flows and short HTTP web sessions that fill up the bottleneck bandwidth. Flows start at uniformly distributed random times and have Pareto distributed file sizes, as in [11].

We also implement in NS the rate control schemes described in the previous section. We implemented SSRC from scratch in NS and we are in the process of making the module publicly available. For HTTP streaming, the TCP transport is built-in in NS and is implemented in great detail. However, in the current version of NS, TCP does not support variable packet sizes, and therefore we were forced to “pad” each video packet to fill up 1500B size packets. The WiSE agent sitting at the border of wired and wireless network (queue n3), the WiSE encoding/decoding modules and the WiSE modifications in TCP have also been implemented in NS by the authors of [3], who kindly shared their code with us.

5. RESULTS

WiSE performance is examined over a wide range of wireless loss and congestion conditions, and two rate control mechanisms.

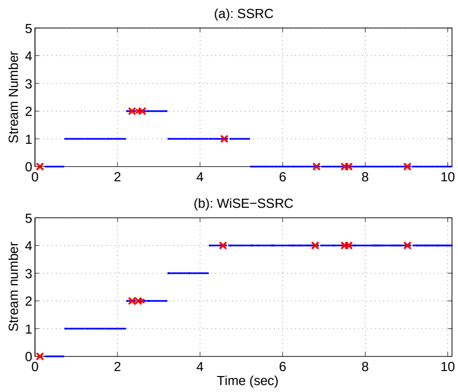

Fig. 2. Example of SSRC and WiSE-SSRC switching streams at 2% wireless loss.

5.1. Using WiSE to improve Switch Stream

Fig. 2 shows an example of how SSRC switches streams, in a scenario with 2% wireless loss. The corrupted packets are represented by Xs and the received packets by dots. Plain SSRC, shown on top, interprets all wireless losses as signs of congestion and unnecessarily switches to lower streams. In contrast, for the exact same wireless loss scenario, WiSE-SSRC, shown on the bottom, switches to higher streams and also retransmits the lost packets. This results in a larger total number of packets delivered, and thus to a higher video quality, e.g. in this case WiSE-SSRC sends approximately 25% more packets (including retransmissions) and 257 Kbps on average vs. 90 Kbps without WiSE. Furthermore, in this scenario (no congestion loss), WiSE will follow the same policy for any wireless loss rate and pattern (the only difference will be the retransmitted packets) and switch gradually from stream 0 to stream 4, thereby offering the highest quality (36.28dB) provided by the coded video.

Fig. 3 examines the performance over a range of wireless loss rates from 0 to 10%. First, we consider the case of Bernoulli wireless loss, without interfering traffic, shown by the solid lines in Fig. 3(a). WiSE provides significant gains on the order of 5-8dB. Second, we consider the same scenario but with congestion on the shared bottleneck link n2-n3. In particular, we simulate interfering traffic in the forward direction (100 FTP flows and 100 web sessions from n0 to n5), as well as reverse traffic (100 FTP flows from n5 to n0). This congestion results in lower video quality (30.8dB even in the absence of wireless loss) that degrades further for higher wireless loss rates. Once again, WiSE-SSRC outperforms plain SSRC for all wireless loss rates, as shown by the dotted lines. For plain SSRC, the two curves with and without congestion converge for large wireless loss rates. This is expected: when enough packets are dropped due to either cause, SSRC never switches above stream 0; furthermore no lost packets are retransmitted, so the overall quality is unacceptably low. On the contrary, WiSE-SSRC is robust to wireless loss both in the congested and in the non-congested case. Fig. 3(b) shows that WiSE-SSRC also outperforms plain SSRC in the case of bursty wireless loss with and without congestion. The PSNRs in the congested case are similar to the congested case under uniform loss. However, the PSNRs in the uncongested case are higher than those for the same uniform loss rate. This is because of the copy loss concelament and the fact that the bursty model causes a smaller number of consecutive loss events for the same number of lost packets.

Fig. 4 shows contours of quality as a function of both wired and

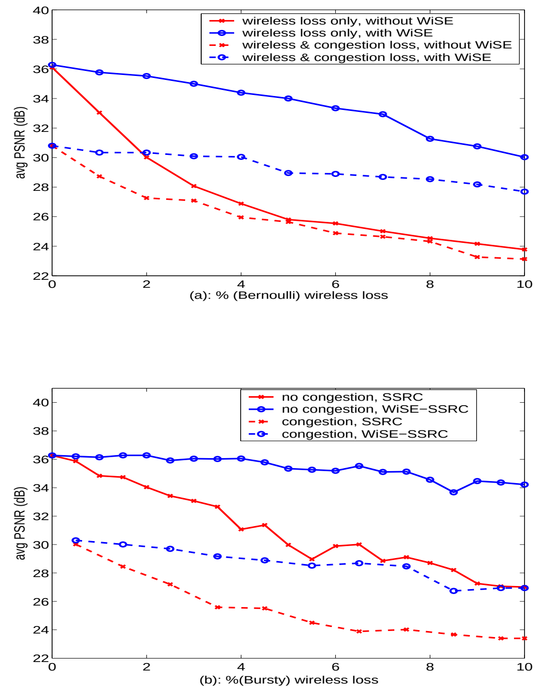

Fig. 3. SSRC and WiSE-SSRC streaming for different wireless loss rates and loss patterns (Bernoulli, top, and bursty, bottom).

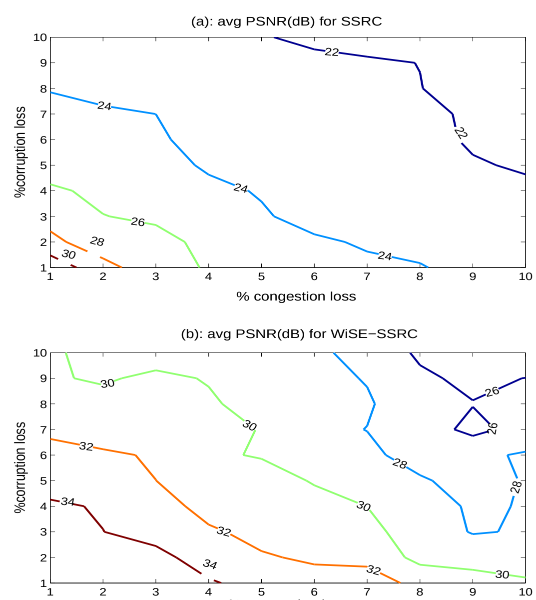

Fig. 4. Performance of (a) SSRC and (b) WiSE-SSRC for all pairs of (wired, wireless) loss rates. Contours of constant PSNR values are plotted.

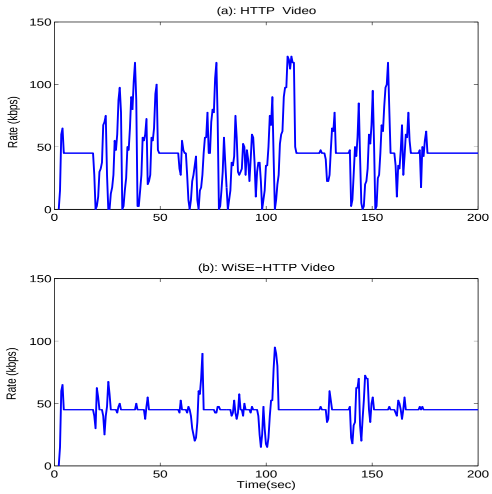

Fig. 5. Sending rate for (a) HTTP video and (b) WiSE-HTTP video, in the presence of wireless (and congestion) loss

wireless loss rates, for (a) SSRC and (b) WiSE-SSRC. Notice that the curves in Fig. 3 are special cases of Fig. 4 for a fixed amount of wired loss. E.g. the top curve in Fig. 3(a) is obtained for zero congestion and by varying the corruption rate from 0 to 10 %; therefore, it corresponds to the y-axis in Fig. 4(a). For the purpose of fair comparison between wired and wireless, the same (Bernoulli) loss model was used to simulate loss on both the wired and the wireless links. Fig. 4(a) shows the contours of the average PSNR achieved by plain SSRC: quality degrades from 36.28dB, in the case of no loss, to as low as 20 dB, for high congestion and corruption rates. Similarly, Fig. 4(b) shows the average PSNR achieved by WiSE-SSRC: it is significantly and consistently higher than the one achieved by plain SSRC, for all combinations of (% wired loss, % wireless loss).

5.2. Using WiSE to improve HTTP Streaming

This section compares WiSE-HTTP to plain HTTP. Stream number 1 of Foreman is repeated 20 times and provides a stream of total duration 200 sec, in order to better capture the dynamic behavior of TCP and the impact of the location of losses on various parts of the sequence. Wireless loss is modeled with the two-state (bursty) model. Video is streamed directly over TCP, as is the interfering traffic.

Fig. 5(a) and Fig. 5(b) show the different sending rates (computed over 0.3sec intervals) of HTTP and WiSE-HTTP Video respectively, in the presence of 0.6% bursty wireless loss and interfering traffic. In general, the sending rate appears constant due to the padded 1500B video packets sent at 30fps. Wireless losses results in sudden dips in the HTTP rate, because plain HTTP (i.e. TCP) reduces its sending rate at each wireless loss. In contrast, WiSE-HTTP video experiences significantly less variation due to the wireless errors. There remain a few variations which are due to the initiation or termination of interfering FTP flows at those times. Otherwise, the sending rate of WiSE-HTTP is not affected by wireless losses.

Fig. 6 plots the decoded PSNR for wireless loss rates 0-5%, with and without WiSE. Two cases are considered. The first is the *without congestion* case, where video is sent without any interfering traffic, and is shown in triangular marks. The second is the *with congestion* case, shown in round marks, where video is sent in the presence of interfering traffic. Clearly, WiSE (dashed lines) improves PSNR significantly over plain HTTP (solid lines), especially for high wireless loss rate and congestion. Without congestion, TCP's retransmissions succeed in delivering all the video packets before their playout deadlines, as long as the loss rate is below 3%. At higher loss rates, TCP's backoff leads to a significant fraction of the packets arriving after their 3 second deadline and therefore highly degraded performance. WiSE preserves the performance to higher loss rates. With congestion and when WiSE is not used, the combined effect of corruption and congestion drops causes high delay for wireless loss more than 0.2%. WiSE offers full protection for up to 1% loss, leading to an over 3dB improvement in decoded quality. Furthemore, WiSE gives this 3dB improvement over plain HTTP, for all examined rates up to 5%.

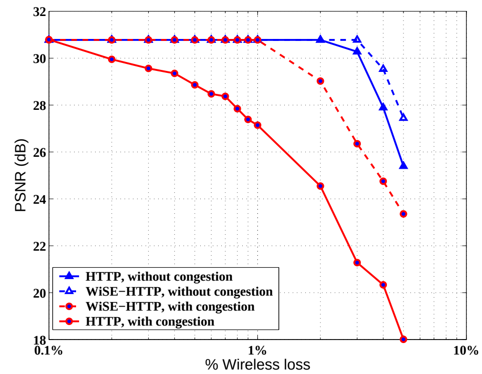

Fig. 6. HTTP & WiSE-HTTP streaming versus wireless loss rate.

6. CONCLUSION

This paper introduced the use of WiSE, an in-band network-to-application notification mechanism, for improving video streaming by making it aware of wireless losses. Simulations demonstrate significant performance improvements for a wide range of conditions: various wireless loss rates, isolated and bursty losses, various degrees of congestion with realistic cross traffic, and two different video streaming protocols: SSRC and HTTP. Furthermore, this benefit is achieved without compromising rate-control in the presence of real congestion.

7. REFERENCES

[1] H.Balakrishnan and R.Katz, "Explicit loss notification and wireless web performance," in *Globecom*, 1998.

[2] R.Krishnan, M.Allman, C.Partridge, and J.Sterbenz, "Explicit transport error notification for error-prone and satellitenetworks," Tech. Rep. 8333, BBN, Mar. 2002.

[3] M.Sharma, S.Katabi, R.Pan, and B.Prabhakar, "A general multiplexed ECN channel and its use for wireless loss notification," in *Sigcomm Posters*, Aug. 2003.

[4] M.Handley, S.Floyd, J.Padhye, and J.Widmer, "RFC 3448: Tcp friedly rate control (tfrc): Protocol specification," Jan. 2003.

[5] T.Yoshimura, "Streaming agent: a network proxy for streaming media in 3G," in *ICC*, Apr. 2002.

[6] G.Cheung and T.Yoshimura, "Streaming agent: a network proxy for media streaming in 3g wireless networks," in *Packet Video WorkShop*, Apr. 2002.

[7] G.Cheung, W.T.Tan, and T.Yoshimura, "Rate-distortion optimized application-level retransmission using streaming agent for video streaming over 3G wireless network," in *ICIP*, 2002.

[8] S.Wenger, M.Hannuksela, T.Stockhammer, M.Westerlund, and D.Singer, "RTP packetization of JVT video," Internet Draft, Dec. 2003.

[9] "The Network Simulator, ns-2," http://www.isi.edu/nsnam.

[10] G.T.Nguyen, B.Noble, R.Katz, and M.Satyanarayanan, "A trace based approach fro modeling wireless channel behavior," in *Winter Conf. in Simulation*, Dec. 1996.

[11] A.Feldmann, A.Gilbert, P.Huang, and W.Willinger, "Dynamics of IP traffic; a study of the role of variability and the impact of control," in *Sigcomm*, 1998.