Error Resilient Video Transmission over Wireless Networks*

Gang Ding[1], Halima Ghafoor[2], and Bharat Bhargava[1]
1 Department of Computer Sciences 2 Department of Electrical and Computer Engineering
Purdue University, West Lafayette, IN 47907
{dingg, ghafooh, bb}@cs.purdue.edu

***Abstract*— An error resilient architecture for video transmission over mobile wireless networks is presented. Radio link layer, transport layer, and application layer are combined to deal with high error rate in wireless environments. The algorithms for both sender and receiver are given. An adaptive algorithm is presented to automatically adjust parity data length in error control. The performance of the proposed algorithm is analyzed through experimental studies.**

***Index Terms*—Error Control, Wireless Network, Video.**

I. INTRODUCTION

THE traditional cellular mobile networks are used for low-rate audio communications. New generation of cellular networks is emerging to transfer data traffic at much higher bit-rate. This motivates the demand for multimedia communication over wireless networks. Video is one of the killer applications in the emerging networks. It requires higher bandwidth and lower response time than text, audio or static image. The major challenges of video transmission over wireless networks are to deal with low bandwidths and high transmission error rates.

We propose an error resilient architecture for video transmission over wireless networks. It involves modified radio link layer error control, modified UDP (called UDP Lite), and a general frame for error control in application layer. The main objective of these techniques is to overcome the error-prone nature of wireless links. In contrast to previous research works, the cooperation among different layers and the performance analysis is studied. Based on an idea that "the higher layer, more the intelligent", we depend on the upper layer to coordinate error control, while lower layers provide as much information as possible for the upper layer to make the decision. This paper is organized as follows. Next section gives background and related works. Section III explains the proposed algorithms in detail. Theoretical performance analysis is conducted in section IV. Simulation experiments are explained in section V. Section VI concludes the paper and suggests future works.

II. BACKGROUND AND RELATED WORKS

In order to deal with high error rate in wireless networks, error control techniques have been employed in many ways. Two basic approaches are Forward Error Correction (FEC) and Automatic Repeat Request (ARQ). Many alternatives to FEC and ARQ have also been proposed in [1]. One of the popular error coding techniques is Reed-Solomon coding, which can deal with burst error. If the original message length is K, then after adding parity data, the codeword is of length N > K and can recover errors of length up to (N – K) / 2. We will briefly introduce video over wireless networks from the bottom to the top, with an emphasis on error control.

Wireless networks involve different kinds of radio access networks, such as mobile cellular networks and wireless local area networks (WLAN. 3G cellular networks standardized by 3GPP and 3GPP2 are expected to provide high data rate up to 384 Kbps or even 2 Mbps. IEEE 802.11 standards of WLAN can provide data rate up to 54 Mbps in local areas. Radio Link Protocol (RLP) located above MAC layer is used to deal with transmission errors at lower layers. The typical size of a radio unit in RLP layer is from 300 to 600 bits. A unit contains a CRC header in order to detect error bits inside the unit. RLP uses a particular ARQ mechanism to recovery errors. It can work in transparent mode, when no error control mechanism is applied. Previous research ([2], [3]) uses transparent mode and resort to upper layer error control techniques to overcome transmission errors. But in this case, the error recovery function of RLP is not fully used.

TCP is a network protocol which provides reliable transmission of data. However, for most video communications in wireless and wired networks, the application can tolerate data errors to some extent. The transport protocol, UDP, is widely used for video transmission. UDP has a checksum to verify the integrity of received packet. A modified version of UDP, called UDP Lite, is introduced in [4], which allows partial checksums on packet data by enabling application layer to specify how many bytes of the packet are sensitive and must be checksummed. If bit errors occur in the sensitive region, the receiver drops the packet; otherwise it is passed up to the application layer.

In the application layer, ITU-T H.263 recommendation is the first standard to offer a solution for very low bit-rate (<64 Kbps) teleconferencing applications. The recently adopted H.263+ improves coding efficiency of H.263. Similarly, the recent ISO MEPG-4 standard is robust in error-prone environments, which is achieved by inserting resynchronization markers, partitioning macroblocks, using header extension code and reversible variable-length coding. Many novel error resilient video codecs are being invented. Here we do not investigate a particular video codec. Instead,

*This work is support by NSF grants ANI 0219110 and CCR 0001788.

we will use standard codecs, such as MPEG-4, as the source coding in application layer. We focus on the networking issue of video transmission in wireless network environments.

III. ERROR RESILIENT VIDEO TRANSMISSION OVER WIRELESS NETWORKS

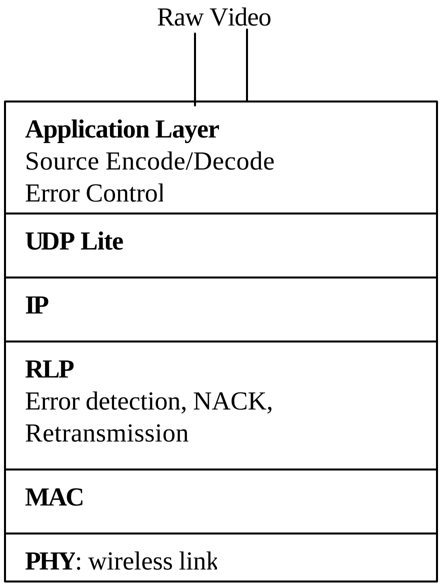

Fig. 1. Protocol stack

We propose a general protocol stack for video transmission over wireless links in Fig. 1. The error control algorithms for sender and receiver are given as follows.

Sender's Algorithm:

1. *At application layer, when the raw video comes, appropriate source codec is applied to get encoded video bitstream. We don't specify a particular source coding algorithm, but an error resilient algorithm is preferred. For the error control at application layer, the encoded video data is fragmented to the size of $N = 255$ bytes, including partial data of size R bytes. So the actual data length is $N – R$ bytes.*
2. *Add UDP Lite header where checksum only covers the header.*
3. *Add IP header with IP checksum.*
4. *At RLP layer, fragment packets to equal length radio units and add CRC for error detection. Set timer for sent unit and retransmit the unit if timeout or NACK is received from the receiver. If sender still gets NACK after n_{max} times of retransmission, reset the link.*
5. *Send radio units through MAC and wireless link to receiver.*

Receiver's algorithm:

1. *At RLP layer, detect error of received packet units, assemble units and send up to IP layer. If error is detected, send a NACK to sender and set timer for retransmission of NACK.*
2. *At IP layer, calculate checksum of IP header. If checksum error, discard the packet, otherwise forward it up to UDP layer.*
3. *At UDP layer, calculate checksum of UDP header. If checksum error, discard the packet, otherwise forward it up to error control layer.*
4. *At application layer, apply error coding algorithm to correct errors, assemble fragments to a complete video frame and buffer it until its time is up and retrieved by upper applications. Notice that if transmission error size at receiver is not greater than the error control capacity, for example R/2 for RS codes, the entire real video packet can be recovered. Otherwise, signal RLP layer to send one more NACK immediately. There is still a chance to receive the correct data from retransmission before playing. But when this frame's time is up, before forwarding it to applications, make sure to clear all corresponding timers in the RLP layer because we do not need the data any more. We refer to the source decoding to deal with remaining errors and the video quality of received data might be degraded.*

Followings are several remarks to the above algorithm

A. Parity data length

By error control coding, i.e. adding extra parity data of length R bytes, at least part of errors can be recovered by receiver. Obviously, the larger the value of R, more the errors will be corrected. On the other hand, parity data introduces more traffic to the limited network bandwidth and may even cause packet loss due to congestion. So R should not be large. Choosing an appropriate R to trade-off error correction and network traffic should be considered carefully. In the above algorithms, we assume that the parity data length R is given *a priori* and fixed. A better way is to adapt R based on the available information of networks and errors. To this end, we introduce an R adaptation algorithm as follows.

In step 4 of receiver's algorithm, the receiver sends one more NACK back to the sender when it can not correct all errors. At the sender's side, this second NACK can be regarded as an indication of the failure of error correction. On receiving the second NACK, sender increases R in order to improve the error correction capacity:

$$R = R_0 + 2^k$$

where R_0 is the initial value for R, which can be arbitrarily chosen. k is increased by 1 for each following NACK. For regular case when there is no second NACK, however, sender keeps deceasing R in order to reduce the extra traffic to the network by

$$R = R – R_{step}$$

where R_{step} is the decreasing step.

This adaptation algorithm can help us to choose an appropriate parity data length R. But it may introduce oscillation of R. We can run this adaptation algorithm at the beginning. After certain amount of time, we can use the average value of R as a constant for R and turn off the adaptation algorithm. Adaptation can be started when necessary.

B. *The higher layer, more the intelligent*

When designing the above algorithm, we follow an important idea borrowed from intelligent control theory: *the higher layer, more the intelligent*. For example,

- The error control algorithm at application layer is responsible for recovering errors, adjusting R, requesting to transmit NACK and clearing timers at RLP layer.
- But at RLP layer, it only conducts error detection and semi-ARQ. Its timer is cleared by signals from upper layer error control.
- At both RLP and UDP layer, data is forward to the upper layer even if there is some error in the payload, they just simply resort to the upper layers to make decision.

The reason to follow these ideas is that at higher layers, there is more information collected from below so that better decisions can be made. For the video transmission over wireless network, the final decision is made by the highest layer: the human being who watches the video. The lower layers just try to collect as much as information as possible instead of discarding it.

C. *General algorithm*

We are trying to make the proposed algorithm general. For instance, instead of thinking of one particular wireless network, we only take a common characteristic of all wireless networks: unreliability. We take the wireless link as just an error generator which may generate different kinds of errors, such as constant rate error, variable rate error, burst error, etc. We do not specify a particular video source coding algorithm. But we prefer standardized codec schemes which have been tested over time.

IV. PERFORMANCE ANALYSIS

In order to analyze the above algorithm, we first specify the error model for wireless link. There is no agreeable standard model for the underlying wireless link, mainly due to the highly time-varying and non-stationary nature of wireless networks. It involves fast channel fading and slow channel fading, as well as the mobility pattern, the location of the mobile node, and so on. But in all cases, the wireless link can just be modeled as an error generator. For the sake of convenience, we assume that the bit error probability is given by p_b. We then give theoretical expressions for the error probability and efficiency at RLP, UDP and application layers.

A. *RLP layer analysis*

Since the bit error probability is p_b, it is easy to find out the RLP radio unit error probability

$$p_R=1-(1-p_b)^{M1+H1} \sim (M_1+H_1)p_b$$

where M_1 and H_1 are the length of data body and header, respectively. In order to reflect the effect of retransmission on the system, we define a transmission efficiency parameter for radio unit as the ratio of times transmitting a radio unit without loss and with loss-and-retransmission.

$$g_R=\frac{(M_1+H_1)}{\sum_{n=1}^{n_{\max}-1}[nM_1+(2n-1)H_1]P_{n0}+[n_{\max}M_1+(2n_{\max}-1)H_1](1-\sum_{k=1}^{n_{\max}-1}p_{n0})}$$

where $P_{n0}=p_R^{n-1}(1-p_R)$ represents the probability of n-1 radio units getting errors before a success (re)transmission. We assume that the transmission rate is constant during retransmission and the size of an NACK packet is just the header length. It is obvious that the more the retransmissions, the lower the efficiency.

B. *Transport layer analysis*

For UDP, considering both the header H_2 and payload M_2, error probability is

$$p_{UDP}=1-(1-p_R)^{\frac{M2+H2}{M1}}$$

and efficiency is

$$g_{UDP}=\frac{(M_2+H_2)}{\frac{M_2+H_2-M_1}{M_1}(M_1+H_1)+\frac{1}{1-p_{UDP}}}$$

where $P_n=p_{UDP}^{n-1}(1-p_{UDP})$. For simplicity, we assume that only one radio unit in a UDP packet is retransmitted.

For UDP Lite, only considering UDP header, error probability is

$$p_{UDPLite}=1-(1-p_R)^{\frac{H2}{M1}}$$

The efficiency of UDP Lite is of the same form as UDP, with p_{UDP} replaced by $p_{UDPLite}$. The $p_{UDPLite}$ is much smaller than p_{UDP}, so the efficiency of UDP Lite is actually higher than that of UDP.

C. *Application layer analysis*

If there is no error control layer above the UDP layer, all error data is forwarded to the application layer, where the packet error rate is

$$p_{APP1}=1-(1-p_b)^{\frac{M2+H2}{M1}(M1+H1)}\approx\sum_{n=1}^{M_2-1}\binom{M_2}{n}p_b^{M_2-n}(1-p_b)^n$$

When error control is added, it can recover R bytes of error out of a packet of M_2 bytes. In this case, the error probability is represented as

$$p_{APP2}=\sum_{n=1}^{M_2-R-1}\binom{M_2}{n}p_b^{M_2-n}(1-p_b)^n$$

This implies that error probability is decreased when the error control is added. We should notice that the actual throughput is also decreased because after adding error control, M_2 - R bytes of data instead of M_2 is transmitted every time. For a video packet of size M_2, the efficiency is represented as follows.

$$g_{APP}=\frac{M_2}{\sum_{n=1}^{\infty}[\frac{M_2+H_2}{M_1}(M_1+H_1)+(n-1)(M_1+H_1)]P_n}$$

V. SIMULATION

We simulated the proposed algorithm and tested it using a benchmark MPEG-4 trace. The video trace file comes from the movie "Jurassic Park" with duration of 60 minutes and is publicly available for testing of the algorithm's network performance, especially for wireless networks ([5], [6]). Some important parameters for the video are as follows. Resolution: QCIF 176*144. Frame rate: 25 frames/sec. Frame sequence: IBBPBBPBBPBB. Compression ratio: YUV:MP4 49.96. We choose parameters M_1 = 50 bytes, M_2 = 255 bytes, H_1 = 2 bytes, and H_2 = 28 bytes.

A. *UDP vs. UDP Lite*

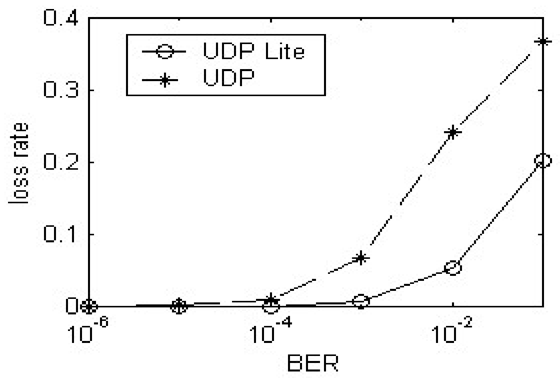

Fig. 2. Video frame loss rate

We first compare packet loss rate for UDP and UDP Lite. The result is shown in Fig. 2 where total frame loss is displayed for Bit Error Rate (BER) p_b from 10^{-6} to 10^{-1}. It is obvious that UDP Lite (solid) performs much better than UDP(dash). This justifies the theoretical results for p_{UDP} and $p_{UDPLite}$ in the last section.

B. *Error control*

The data loss due to payload data error is shown in Fig. 3. where we choose a high BER p_b = 0.01. To test our error control algorithms, we choose parity data length R = 16 bytes. The corresponding data loss is shown in Fig. 4. It is evident that when parity data is added, data loss is greatly reduced. However, the disadvantage of adding parity data is that more redundant data will be transmitted though the network. As discussed in Section III.B, we further add the adaptation algorithm to adjust parity data length R. In this case, we choose an initial length R_0 = 16 bytes, with R_{step} = 2 bytes, R_{min} = 0 bytes and R_{max} = 32 bytes. We finally get an average R = 10.7 bytes, with average data loss of 1.63 bytes/frame. The simulation result is displayed in Fig. 5.

VI. CONCLUSION AND FUTURE WORKS

We proposed an error resilient video transmission architecture over wireless networks. Theoretical performance analysis and simulation results demonstrate the effectiveness of the proposed algorithm. Future research is needed with issues such as more accurate error models and their effect on system performance, video transmission over both internet and wireless networks.

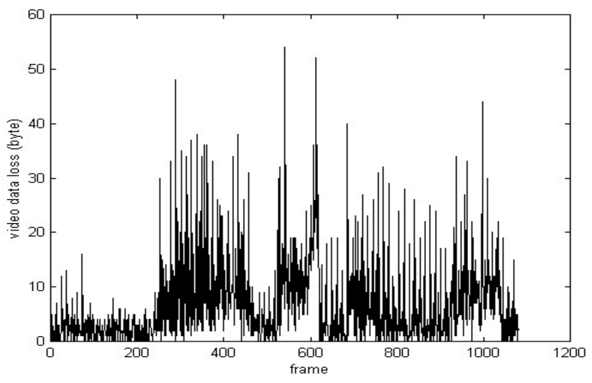

Fig. 3. Video data loss without error control (BER=0.01, Average data loss = 6.5 bytes/frame)

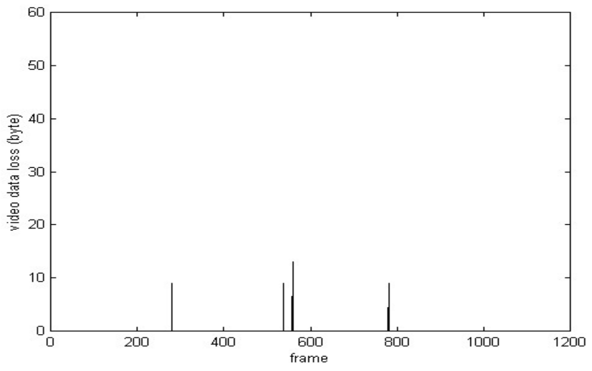

Fig. 4. Video data loss with error control (BER=0.01, R = 16 bytes, Average data loss = 0.068 bytes/frame)

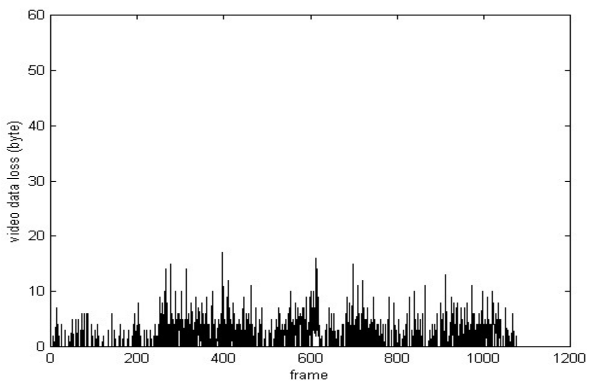

Fig. 5. Video data loss with adaptive parity data length (BER=0.01, Average data loss = 1.63 byte/frame, Average R = 10.7 bytes)

REFERENCES

[1] H. Liu, et al., "Error control schemes for networks: an overview," *Mobile Networks and Applications*, 2, pp. 167-182, 1997.
[2] A. Singh. et al., "Performance evaluation of UDP Lite for cellular video". NOSSDAV'01, 2001.
[3] H. Zheng and J. Boyce, "An improved UDP protocol for video transmission over internet-to-wireless networks". IEEE Trans. on Multimedia. 3(3): 356-365. 2001.
[4] L Larzon, "The UDP Lite Protocol," internet draft, Jan. 2002.
[5] MPEG-4 and H.263 Video Traces for Network Performance Evaluation, http://www-tkn.ee.tu-berlin.de/research/trace/trace.html
[6] F. Fitzek and M. Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," IEEE Network, 15(6): 40-54, 2001.