

Comparison of Multimedia Transport Schemes over Markovian Wireless Channels

Syed Ali Khayam

NUST Institute of Information Technology (NIIT)
National University of Sciences & Technology (NUST)
Rawalpindi 46000, Pakistan
khayam@niit.edu.pk

Hayder Radha

Dept. of Electrical & Computer Engineering (ECE)
Michigan State University (MSU)
East Lansing, MI 48824, USA
radha@egr.msu.edu

Abstract— Contemporary wireless networks drop all corrupted packets at the receivers. Such a strategy adversely affects multimedia applications which have stringent throughput and delay constraints. Recently, a header estimation scheme was proposed to reduce packet drops at wireless receivers [1], [2]. These corrupted packets are recovered using FEC, which incurs lesser delay and bandwidth wastage than retransmissions. In this paper, we derive lower bounds on the expected FEC redundancy required by ideal header estimation, UDP and UDP-Lite over an arbitrary-order Markov wireless channel. We show that under realistic wireless channel conditions, header estimation always requires lesser redundancy than UDP and UDP-Lite. We then propose a practical minimum distance decoding (MDD) header estimation scheme, which is receiver-based, low complexity and highly accurate. Trace-driven wireless video simulations demonstrate that MDD header estimation requires significantly lesser FEC redundancy and renders better video quality than existing schemes.

I. INTRODUCTION

In existing wireless multimedia transport protocols (e.g., UDP [3]), corrupted packets are dropped by a wireless receiver's protocol stack. These dropped packets are recovered using retransmissions, which incur bandwidth wastage and delays. Moreover, emerging multimedia applications are error-resilient and can, therefore, tolerate some errors in the received content. Consequently, recent wireless multimedia studies have proposed that corrupted packets should be relayed to a wireless receiver's application layer. The application layer can then correct these packets using forward error correction (FEC).

Most of the recent studies deploy a UDP-Lite based protocol stack [4]–[8] which only drops packets with header errors, while payload errors are ignored. UDP-Lite requires modifications to the protocol stacks of wireless senders, receivers and intermediate multicast/multi-hop nodes. Recently, a receiver-based *header estimation* framework was proposed to reduce wireless packet drops [1], [2]. This framework ignored payload errors and corrupted packet headers were corrected using decision-theoretic tools. Significant benefits of header estimation were experimentally shown in [1], [2].

In this paper, we derive analytical conditions for the regions of operation under which header estimation performs better/worse than UDP and UDP-Lite when operating over an arbitrary-order Markov wireless channel. We show that for any realistic wireless system, the minimum expected FEC

redundancy required by header estimation is always lower than UDP and UDP-Lite protocols.

Existing header estimation schemes have somewhat high (training and run-time) complexities and memory requirements [1], [2]. In this paper, we propose a practical and low-complexity minimum distance decoding (MDD) header estimation scheme. We use trace-driven video simulations at 2, 5.5 and 11 Mbps data rates of an 802.11b LAN to show that, despite its extremely low-complexity, performance of the MDD header estimation scheme is comparable to prior header estimators. We also demonstrate that the proposed MDD header estimation scheme requires much lesser FEC redundancy than UDP and UDP-Lite at all 802.11b data rates. Finally, we show that for fixed-rate FEC, MDD header estimation provides much better video quality than UDP and UDP-Lite.

II. SYSTEM DESCRIPTION, ASSUMPTIONS AND NOTATION

In this section, we first describe the bit- and packet-level Markov models that will be employed in this paper. We then explain the abstract FEC scheme that is used to analyze UDP and UDP-Lite in subsequent sections. System assumptions and notation are also detailed in this section.

A. Bit-Level High-Order Markov Channel

We assume a transmitter that packetizes and transmits binary symbols or bits on a single-hop K -th order binary Markov channel¹. A brief description of the bit-level channel model follows.

Most wireless bit-error random processes are bursty with a memory-length greater than one bit. To make such a process comply with the Markov property, wireless studies define a high-order Markov chain such that at each time instance the process is characterized by as many bits as its memory-length [11]. Specifically, at each time instance, a new bit is added to the memory-window and the oldest bit is dropped from the memory-window. Thus the states of a K -th order (memory-length= K) Markov chain comprise 2^K possible combinations of K consecutive bits; that is, each Markov state corresponds to the decimal equivalent of a unique

¹We operate on the *residual channel* [9] observed at a wireless receiver's MAC layer after physical layer processing.

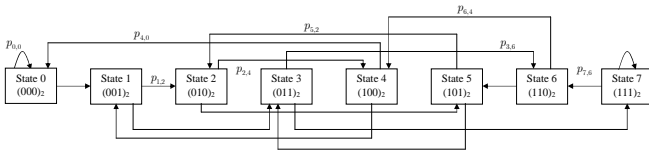


Fig. 1. A 3-rd order (memory-length=3) bit-level Markov channel.

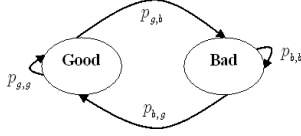


Fig. 2. The packet-level Gilbert Markov channel.

K bit sequence. Transition probabilities between states are computed by (bit-by-bit) sliding a K bit memory-window over the training data and counting the number of times a bit-pattern $[x_1, x_2, \dots, x_K]$ is followed by another bit-pattern $[y_1, y_2, \dots, y_K]$. We use such a K -th order Markov channel for all bit-level theoretical analysis. Since we parameterize the bit-level channel's memory-length in our analysis, our theoretical deductions are applicable to any Markov channel of arbitrary order.

Throughout this paper, $p_{i,j}$ denotes the probability of transitioning from current state i to state j , where i and j are decimal equivalents of the binary state indices. We use π_i to represent the steady-state probability of being in state i . All probability subscripts (i.e., state indices) are implicitly assumed to be modulo 2^K . The bit-level wireless channel's output is treated as a binary time series with elements $x(i) \in \{0, 1\}$, where $x(i) = 0 \Rightarrow$ error-free bit and $x(i) = 1 \Rightarrow$ corrupted bit. In this paper, states corresponding to even (odd) decimal numbers are referred as even (odd) states. Using the above notation, an example 3-rd order Markov chain is shown in Fig. 1; only transitions to even states are labelled. If the Markov chain is in an even state, the last received bit (i.e., the least significant bit position in the memory-window) must be error-free. Similarly, a Markov chain in the odd state implies that the last bit was corrupted.

Note in Fig. 1 that due to the binary nature of the underlying wireless bit-error process, each state can transit to only two other states. This is due to the process' definition in which the memory-window at each time instance is left-shifted by one bit and a one or a zero bit is added to the least-significant bit position. Thus from state i , the Markov chain can transit either to even state $(2i) \bmod 2^K$ or to odd state $(2i+1) \bmod 2^K$. Since the sum of all transitions from a Markov state must sum to one, for any state i we have $p_{(2i+1) \bmod 2^K} = 1 - p_{(2i) \bmod 2^K}$. It should also be emphasized that once a corrupted bit is received, a K -th order Markov chain will return to state 0 (i.e., the no error state) only from state 2^{K-1} after K transitions; see in Fig. 1 that at state $2^{3-1} = 4$, the Markov chain wraps around to state 0.

B. Packet-Level Gilbert Markov Channel

It has been shown in many prior wireless studies that the high-order bit-level channel memory does not persist beyond packet boundaries [10]–[12]. Therefore, despite the high-order bit-level memory, the packet-error process can be adequately modeled using a memory-less channel or a first-order² Gilbert channel.

In this paper, we assume a Gilbert channel of Fig. 2 at the packet level. The Gilbert channel is an order-1 two-state Markov chain with a *good* state and a *bad* state. In the good state, the process only outputs good packets, whereas only corrupted packets are observed in the bad state. Thus the packet-level Gilbert channel does not provide any indication of the number of errors in a corrupted packet. At each transition, this process only indicates a binary decision of whether or not a packet is corrupted.

For the the bit-level high-order model, $p_{i,j}$ denotes the probability of transiting to state j from state i and π_i represents the steady-state probability of being in state i . $\pi_b^{(pkt)}$ denotes the steady-state probability of the bad state in the packet-level Gilbert model.

C. FEC Construction

We assume that a bit-level block-based maximum distance separable (MDS) FEC scheme capable of simultaneously correcting errors and erasures is operating at the communicating nodes' application layers. Each FEC block is packetized into l packets, with each packet having a fixed-length data payload of n_D bits. Before transmitting each packet, a header of size n_H bits is appended to the packet. Thus each packet has a fixed length of $n_H + n_D$ bits. The FEC algorithm only protects the data symbols, and hence the FEC block-length is $n_D l$ symbols. A total of r bits are redundant.

We assume that a packet dropped by a protocol below the application layer will be treated as a packet erasure by the FEC decoder. For the bit-level FEC decoder, each packet erasure will result in n_D bit erasures. Here we are assuming that before decoding, the FEC decoder can determine missing packets (i.e., packet erasures) in an FEC block. This can, for example, be achieved by transmitting an FEC-protected sequence number in each packet. Then before FEC decoding, missing sequence numbers can be flagged as erased packets.

Let X and Y be two random variables that respectively represent the number of errors and erasures observed at the wireless receiver before FEC decoding. An MDS code can recover all errors X and erasures Y as long as $2X + Y \leq r$, where r is the number of redundant symbols [13]. None of the corrupted or erased symbols are recovered if $2X + Y > r$.

We do not account for wireless MAC layer acknowledgements because the same acknowledgement overhead is incurred by both the protocols under consideration. We do not assume any symbol interleaving at the FEC encoder and decoder. Interleaving is used to remove channel memory,

²Here the term *order* is used for the packet-error process. In later discussions, the meaning or granularity of this term will be clear from the context.

and the interleaved symbols are coded to be transmitted over a memory-less channel. However, emerging service-aware protocols and applications perform packet-by-packet FEC, as assumed in this paper. While this strategy results in some loss of FEC efficiency, it reduces the delay incurred due to symbol interleaving. Finally, we inherently assume that the UDP-Lite protocol is supported on the communicating nodes. We acknowledge that (unlike UDP) support for UDP-Lite requires changes at the multimedia transmitters. We, however, ignore these architectural issues in this paper and focus on quantifying the theoretical gains.

III. FEC REDUNDANCY LOWER BOUNDS FOR UDP, UDP-LITE AND HEADER ESTIMATION

In this section, we derive theoretical bounds on the efficiency of an ideal header estimation scheme with application layer FEC for data payload operating on a K -th order Markov wireless channel. Here the term ‘‘ideal header estimation’’ implies that all corrupted packets are passed to the application layer. We derive lower bounds on the expected amount of FEC redundancy required to successfully decode one FEC block. Naturally, we want the amount of redundancy to be as low as possible for efficient utilization of scarce wireless bandwidth.

A. Bound on a UDP based Protocol Stack

A UDP based wireless protocol stack performs a checksum on the entire packet and drops all packets that fail the checksum. Thus the number of errors in the received data is always zero, $\Pr\{X=0|\text{UDP}\} = 1$. Therefore, we focus on deriving the expected value of the number of erasures, Y , observed on a UDP protocol stack. For UDP, an n_D -symbol erasure will occur whenever a received packet has one or more bit-errors. As defined earlier, $\pi_b^{(pkt)}$ denotes the average or steady-state probability of observing a packet erasure on a UDP protocol stack. Using the bit-level model, and under the realistic assumption of $K < n_H + n_D$, $\pi_b^{(pkt)}$ can be written as:

$$\begin{aligned} \pi_b^{(pkt)} &= \Pr\{\text{pkt error}\} = \Pr\{\text{pkt erasure}|\text{UDP}\} \\ &= 1 - \sum_{i=0}^{2^{K-1}-1} \left(\frac{\pi_{2i} \prod_{j=1}^K p_{2i,2^j(2i)} (p_{0,0})^{n_H+n_D-K} + \pi_{2i+1} \prod_{j=1}^K p_{2i+1,2^j(2i+1)} (p_{0,0})^{n_H+n_D-K}}{\pi_{2i} \prod_{j=1}^K p_{2i,2^j(2i)} + \pi_{2i+1} \prod_{j=1}^K p_{2i+1,2^j(2i+1)}} \right) \\ &= 1 - (p_{0,0})^{n_H+n_D-K} \sum_{i=0}^{2^{K-1}-1} \left(\frac{\pi_{2i} \prod_{j=1}^K p_{2i,2^j(2i)} + \pi_{2i+1} \prod_{j=1}^K p_{2i+1,2^j(2i+1)}}{\pi_{2i} \prod_{j=1}^K p_{2i,2^j(2i)} + \pi_{2i+1} \prod_{j=1}^K p_{2i+1,2^j(2i+1)}} \right). \end{aligned} \quad (1)$$

To simplify notation, let

$$\mathfrak{S}_{\pi,p} = \sum_{i=0}^{2^{K-1}-1} \left(\pi_{2i} \prod_{j=1}^K p_{2i,2^j(2i)} + \pi_{2i+1} \prod_{j=1}^K p_{2i+1,2^j(2i+1)} \right) \quad (2)$$

and hence $\pi_b^{(pkt)}$ can be written as:

$$\pi_b^{(pkt)} = 1 - (p_{0,0})^{n_H+n_D-K} \mathfrak{S}_{\pi,p}. \quad (3)$$

The expected value of packet erasures in a block of l packet is:

$$\begin{aligned} E\{\text{pkt erasures}|\text{UDP}\} &= l\pi_b^{(pkt)} \\ \Rightarrow E\{\text{bit erasures}|\text{UDP}\} &= E\{Y|\text{UDP}\} = n_D l \pi_b^{(pkt)}. \end{aligned}$$

Since $\Pr\{X=0|\text{UDP}\} = 1$, $E\{X|\text{UDP}\} = 0$. Thus the expected amount of redundancy required to recover one FEC block over a UDP protocol stack is

$$r_{udp} \geq n_D l \left[1 - (p_{0,0})^{n_H+n_D-K} \mathfrak{S}_{\pi,p} \right]. \quad (4)$$

B. Redundancy Bound on a UDP-Lite based Protocol Stack

Recall that a UDP-Lite based protocol stack performs a partial checksum that only covers packet headers, while payload errors are corrected using application layer FEC [2]. Based on prior derivations, the expected number of UDP-Lite erasures is:

$$E\{Y|\text{UDP-Lite}\} = n_D l \left[1 - (p_{0,0})^{n_H-K} \mathfrak{S}_{\pi,p} \right].$$

In addition to erasures, a UDP-Lite protocol stack will also have errors in the application layer payload. The average proportion of bits in error is characterized by the long-run proportion of time spent in the error Markov states; i.e., the odd states which have a one in the least significant bit position. For a Markov chain, the long-run proportion of time spent in an odd state $2i+1$ is simply the steady-state probability of the state π_{2i+1} . Hence the expected number of errors in the unerased payload symbols is:

$$E\{X|\text{UDP-Lite}\} = (n_D l - E\{Y|\text{UDP-Lite}\}) \sum_{i=0}^{2^{K-1}-1} \pi_{2i+1}.$$

For ease of notation, let $\pi_b^{(bit)} = \sum_{i=0}^{2^{K-1}-1} \pi_{2i+1}$ represent the average probability of bit-error on the channel. Then:

$$E\{X|\text{UDP-Lite}\} = (n_D l - E\{Y|\text{UDP-Lite}\}) \pi_b^{(bit)},$$

and the total expected redundancy required to recover the errors and losses in an FEC block over a UDP-Lite protocol stack is

$$\begin{aligned} r_{lite} &\geq n_D l \left[1 - (p_{0,0})^{n_H-K} \mathfrak{S}_{\pi,p} + 2\pi_b^{(bit)} (p_{0,0})^{n_H-K} \mathfrak{S}_{\pi,p} \right] \\ &\geq n_D l \left[1 - (p_{0,0})^{n_H-K} \mathfrak{S}_{\pi,p} \left(1 - 2\pi_b^{(bit)} \right) \right]. \end{aligned} \quad (5)$$

C. Bound on a Header Estimation based Protocol Stack

Under an ideal header estimation protocol stack, there are no erasures since all (corrupted and error-free) packets are passed to the FEC decoder. That is, $\Pr\{Y=0|\text{HdrEst}\} = 1 \Rightarrow E\{Y|\text{HdrEst}\} = 0$. Based on previous derivations, the expected number of symbol errors is $E\{X|\text{HdrEst}\} = n_D l \pi_b^{(bit)}$. Thus the total (expected) amount of redundancy

required by an ideal header estimation scheme that passes all packets to the FEC decoder is

$$r_{h_{drest}} \geq 2n_{Dl}\pi_b^{(bit)}. \quad (6)$$

Comparison of the above bound with the bounds for UDP and UDP-Lite reveals that minimum expected redundancy required by header estimation is independent of channel memory, K . The redundancy is simply a function of the probability of bit-error. Thus the performance of header estimation will remain unchanged with changes in channel memory. On the other hand, the redundancy required by UDP and UDP-Lite is high for low-memory channels and the redundancy decreases with an increase in channel memory.

D. Comparison of Redundancy Bounds

First, let us compare minimum expected FEC redundancies of UDP-Lite and header estimation:

$$\begin{aligned} & \min(r_{lite}) - \min(r_{h_{drest}}) \\ &= n_{Dl} \left[\left(1 - (p_{0,0})^{n_H - K} \mathfrak{S}_{\pi,p} \left(1 - 2 \sum_{i=0}^{2^{K-1}-1} \pi_{2i+1} \right) \right) \right] > 0, \end{aligned}$$

for $\pi_b^{(bit)} < 0.5$. That is,

$$\min(r_{lite}) > \min(r_{h_{drest}}) \text{ if } \pi_b^{(bit)} < 0.5. \quad (7)$$

This condition implies that header estimation should perform better than UDP-Lite as long as the average probability of bit-error is less than 0.5. This condition is always true because the probability of bit-error is much smaller than 0.5 for any reasonable wireless channel.

We now compare minimum expected redundancies of UDP and header estimation:

$$\begin{aligned} & \min(r_{udp}) - \min(r_{h_{drest}}) \\ &= n_{Dl} \left[1 - (p_{0,0})^{n_H + n_D - K} \mathfrak{S}_{\pi,p} - 2 \sum_{i=0}^{2^{K-1}-1} \pi_{2i+1} \right] \end{aligned}$$

Based on the above comparison, we have

$$\min(r_{udp}) > \min(r_{h_{drest}}) \text{ if } \pi_b^{(pkt)} > 2\pi_b^{(bit)}. \quad (8)$$

Since all combinations of one or more bit-errors result in a packet-error, on any realistic channel the probability of packet-error is much greater than the probability of bit-error, $\pi_b^{(pkt)} \gg \pi_b^{(bit)}$. Thus header estimation requires lesser redundancy than UDP over realistic wireless channels.

IV. HEADER ESTIMATION USING MINIMUM DISTANCE DECODING

In this section, we propose and evaluate a practical header estimation scheme that employs minimum distance decoding (MDD) to estimate the correct values of corrupted header fields. We only estimate the *critical header fields* (CHF) [1] that can uniquely identify a multimedia session at a receiver and are not liable to change during the session. A list of active CHF (i.e., CHF of multimedia sessions that are currently being received) is maintained by the header estimation module. This list can be generated using global or local packets observed

on a wireless channel. These two variants are referred to as global and local statistics variants (GSV and LSV) [1].

On receiving the first error-free packet of a new session, the multimedia application adds the new session's CHF information to the list of active multimedia sessions. Let $x_i \in \Lambda$ denote the CHF of active multimedia session i , where $\Lambda = \{x_1, x_2, \dots, x_N\}$ is the set of all active CHF; a global and a local method of generating the set of active CHF are described later. Whenever a corrupted packet with CHF y is received, the MDD-based header estimator picks an active CHF $x_i \in \Lambda$ to minimize the Hamming distance $d(y, x_i)$. Thus the CHF $x_i \in \Lambda$ that yields the smallest $d(y, x_i)$ is chosen as the CHF estimate for further packet processing.

MDD header estimation has many advantages. First, it is clearly receiver-based. Second, it has extremely low complexity – a highly desirable feature in resource-constrained wireless environments. Also, unlike prior schemes [1], [2], an MDD-based header estimator is stateless and non-probabilistic, and therefore does not require a priori training. Lastly, header estimation does not require any hardware modifications to existing wireless products. The following sections focus on evaluating the improvements provided by MDD header estimation.

A. Experimental Setup

We use 802.11b MAC layer bit-error traces collected over an operational LAN – and similar to the ones collected in [1] – to simulate the wireless channel. We simulate an 802.11b LAN with three unicast receivers, namely recvr-0, recvr-1 and recvr-2, in an infrastructure configuration. For each packet transmission, a 512 byte packet (452 bytes of video payload and 60 bytes of headers) is corrupted using the bit-error traces.

Video sequences are compressed using the H.264 video standard [14]. The sequences have a QCIF frame size and are encoded at a frame-rate of 30 frames/sec. The streams are encoded at different source coding bitrates ranging from 100 kbps to 750 Kbps. A slice mode with fixed number of 452 bytes per slice is used for encoding [14]. Each group of pictures (GOP) has 12 video frames. Each receiver is receiving 5 simultaneous video streams.

B. Comparison of FEC Redundancy

We use MDS Reed-Solomon (RS) FEC codes [13] with a packet block length of $N = 30$ bytes. Each FEC codeword is composed of one byte from a different packet. Thus each packet contributes to 452 separate FEC codewords, and each codeword spans over 30 packets. For performance evaluation, we use a *decodable probability* measure which is the ratio of decodable and total number of received FEC codewords.

Fig. 3 shows the decodable probability at all data rates. Clearly, both LSV- and GSV-based MDD header estimators completely outperforms UDP and UDP Lite; at all three data rates, UDP and UDP Lite require consistently and considerably more bandwidth for FEC recovery than MDD header estimation. Moreover, MDD header estimation, while providing significant complexity reduction, requires the same amount of redundancy as MAP [1] and ML [2] header estimators.

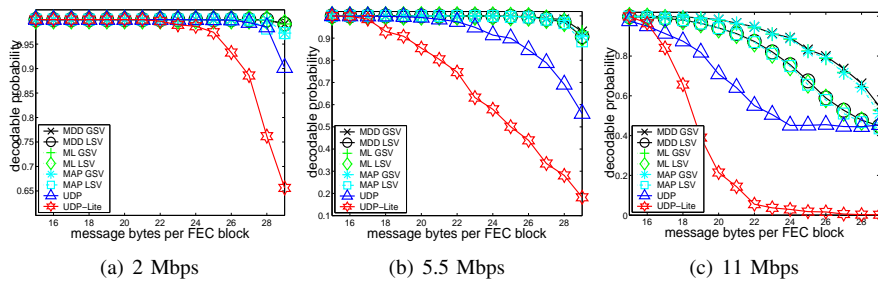


Fig. 3. FEC redundancy used by different schemes/protocols.

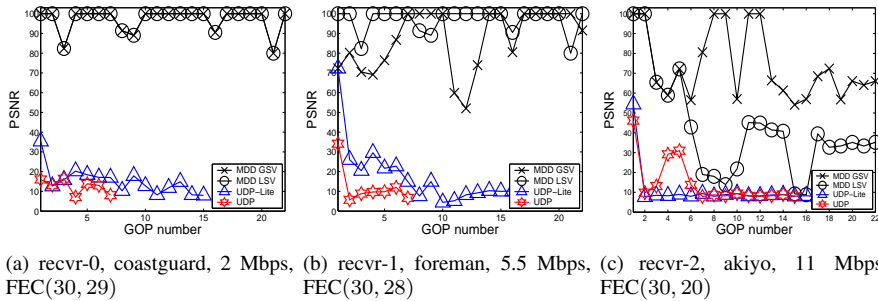


Fig. 4. PSNR of video sequences with fixed-rate FEC.

C. Video Performance

Many multimedia applications use fixed-rate FEC codes and do not change the amount of redundancy in real-time. To cater for such scenarios, it is important to compare the video quality rendered by the proposed scheme for fixed FEC parameters. PSNR plots for three video streams with fixed rate FEC are given in Fig. 4. (Results for MAP and ML estimators are skipped because their performance was comparable to MDD.) Clearly, PSNR of GSV-based MDD header estimation is much higher than UDP-Lite and UDP at all data rates. Similarly, LSV’s PSNR at 2 and 5.5 Mbps is much higher than UDP-Lite and UDP. LSV’s PSNR degrades at 11 Mbps with some GOPs having an average PSNR as low as UDP and UDP-Lite. However, even at 11 Mbps the average PSNR of LSV is substantially higher than UDP and UDP-Lite.

V. CONCLUSION

We analytically showed that an ideal header estimation scheme will always require lesser FEC redundancy than UDP and UDP-lite over realistic Markovian wireless channels. We proposed a minimum-complexity MDD header estimation scheme and used trace-driven 802.11b wireless video simulations to demonstrate that MDD header estimation requires lesser FEC redundancy than UDP, UDP-Lite, and prior header estimators. We used video experiments with fixed rate FEC to show that MDD header estimation provides significantly better multimedia quality than UDP and UDP-Lite.

REFERENCES

[1] S. A. Khayam, S. Karande, M. U. Ilyas, and H. Radha, “Header detection to improve multimedia quality over wireless networks,” *IEEE Trans. Mult.*, (9)2 377–385, Feb. 2007.

[2] S. A. Khayam and H. Radha, “Maximum-likelihood header estimation: A cross-layer methodology for wireless multimedia,” *IEEE Trans. Wireless Comm.*, to appear.

[3] J. Postel, “The User Datagram Protocol,” RFC 768, August 1980.

[4] L-A. Larzon, M. Degermark, S. Pink, L-E. Jonsson, and G. Fairhurst, “The Lightweight User Datagram Protocol (UDP-Lite),” RFC 3828, July 2004.

[5] E. Masala, M. Bottero, and J. C. De Martin, “MAC-level partial checksum for H.264 video transmission over 802.11 ad hoc wireless networks,” *IEEE VTC’05*.

[6] A. Servetti and J. C. De Martin, “Error tolerant MAC extension for speech communications over 802.11 WLANs,” *IEEE VTC’05*.

[7] A. Singh, A. Konrad, and A. D. Joseph, “Performance evaluation of UDP-Lite for cellular video,” *ACM NOSSDAV’01*.

[8] H. Zheng and J. Boyce, “An improved UDP protocol for video transmission over Internet-to-wireless networks,” *IEEE Trans. Mult.*, (3)3 356–365, 2001.

[9] M. Zorzi and R. R. Rao, “On the statistics of block errors in bursty channels,” *IEEE Trans. Comm.*, (45)6, 660–667, June 1997.

[10] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, “A Markov-based channel model algorithm for wireless networks,” *ACM Wireless Net.*, (9), 189–199, May 2003.

[11] S. A. Khayam, H. Radha, S. Aviyente, and J. R. Deller, Jr., “Markov-based Modeling of Wireless Local Area Networks,” *ACM MSWiM’03*.

[12] A. Willig, M. Kubisch, C. Hoene, and A. Wolisz, “Measurements of a wireless link in an industrial environment using and 802.11-compliant physical layer,” *IEEE Trans. Indus. Elec.*, (49)6 1265–1282, Dec. 2002.

[13] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, May 1984.

[14] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC), Doc. JVT-G050, Mar. 2003.