Analysis of Channel Error upon Selectively Encrypted H.264 Video

Mamoona Naveed Asghar, Mohammed Ghanbari, Martin Fleury, and Martin J. Reed
School of Computer Science and Electronic Engineering
University of Essex, Colchester, CO4 3SQ, United Kingdom

***Abstract*—For real-time video streaming applications it is important to preserve both privacy and delay sensitivity. To achieve, this selective encryption exploits the entropy coding stage of hybrid codecs. As streaming applications move towards mobile networks, they become vulnerable to bit errors in the wireless channel. This paper is a pioneering analysis of the relative impact of bit errors on the two H.264 codec entropy coders, CAVLC and CABAC. The joint impact of compression and encryption on objective video quality is determined in experiments, when it is found that CAVLC is much more vulnerable to bit errors compared to CABAC. This finding will guide future bit error protection schemes.**

Keywords- entropy coding; selective encryption; video streaming

I. INTRODUCTION

While the processing and reproduction of analog media are time-consuming and results in an observable reduction in quality, digital media are freely processed and exactly copied. For real-time video streaming applications, to preserve privacy and confidentiality, it is necessary to encrypt the content, in addition to the usual compression to conserve bandwidth. However, interactive video streaming applications are very sensitive to the additional processing delay arising from simply encrypting the compressed stream with a block cipher such as the Advanced Encryption Standard (AES), a process known in the video community as naïve encryption [1]. Due to this sensitivity, these applications are transported by the User Datagram Protocol (UDP), which does not retransmit data if packets are lost or received in error. The same applies to classic one-way streaming of video-on-demand and Internet Protocol TV (IPTV), though there is also a strong recent trend to transport by the Transmission Control Protocol (TCP) underlying Dynamic Adaptive Streaming with HTTP (DASH) [2].

To resolve the problem of additional delay from encryption, researchers have turned to selective encryption [3], based on the realization that if a video is strongly distorted then it will be unattractive to the copier anyway. This is particularly so for 'infotainment' applications of video, for which it has been suggested that a display of what is missing (from lacking the encryption key) may result in the viewer purchasing the right to that key. In selective encryption, the encryption is embedded in the compression process itself, so that the video can still be decoded even if it is distorted. A selective encryption method should meet the following requirements: compatibility with the H.264 standard decoder[1], that is be decodable even when encrypted; no increase in bit-rate as a result of encryption; transparency to intermediate processing such as transcoding; and have the ability to be applied to scalable coding.

However, video streaming applications, along with many others, are transferring to mobile devices such as notebooks and smartphones. This implies that a selectively encrypted video stream will be susceptible to wireless channel errors. To reduce the impact of: 1) bit errors; and 2) packet drops on unencrypted video streams, video decoders incorporate error concealment, though the form of error concealment supported has been made implementation-dependent in the H.264 standard. To date, little or no research has been conducted on how to reduce the impact of channel errors, according to a recent survey of this field [3]. For all streaming using UDP transport, there is a threat from these errors and the impact on selectively encrypted video quality is unknown. This situation may be because researchers have previously concentrated on solving the problem of embedding encryption within the codec and in doing so have assumed perfect channel conditions (or reliable TCP transport with retransmissions).

Therefore, the contribution of the present paper is to analyze the impact of channel bit errors, which can be severe, upon a selectively encrypted video stream. In particular, the effect of choice of entropy coder is analyzed. To avoid affecting the bitrate, selective encryption is best applied at the final output stage of a hybrid encoder, the entropy coder stage, after the frequency transform of predictive residuals [4], and just after quantization of those transform coefficients (apart from the DC value, which is treated separately in intra-coded mode). As the quantized coefficients are mostly zeroes, it is possible to compactly code the characteristics of these coefficients, e.g. the number of zeroes, the signs of any 1s, the magnitudes (levels) of nonzero coefficients. In selective encryption, those characteristics that have a uniform distribution are randomly changed in value according to the encryption algorithm.

There are two forms of entropy coder in H.264/AVC. Context Adaptive Variable Length Coding (CAVLC) and Context Adaptive Binary Arithmetic Coding (CABAC). The

[1] H.264 is the latest of a series of codec standards. It comes in two flavours: Advanced Video Coding (AVC) and the Scalable Video Coding (SVC) extension to H.264/AVC. The base layer of H.264/SVC is compatible with single-layered H.264/AVC.

former is less computational complex than CABAC and is suitable for hardware implementation. However, CABAC *can* result [5] in a 5-15% reduction in bitrate for the same video quality. Both coders are adaptive, which means that compression takes place depending on the pattern of coefficients that occurs upon extraction (zig-zag scanning) from the coefficient matrix However, CALVC selects from a limited context, while CABAC has over 400 context models available in the H.264 standard. CABAC uses arithmetic coding on the binary representation of syntax elements, which gives sub-integer probability estimation but is computationally demanding. It also operates on syntax elements other than the transform coefficients such as the relative motion vector. Both CAVLC and CABAC are a lossless form of coding in which there is a close data dependency between elements in the output bitstream. This dependency at the bit-level contributes to the potential fragility of a selectively-encrypted bitstream during its passage across a wireless link after packetization. It is this fragility that makes this coding vulnerable to bit errors and it is the relative vulnerability of CAVLC and CABAC that this paper investigates.

Section II of this paper introduces the background needed to understand the way experiments were set-up in Section III and conducted and evaluated in Section IV. Finally, Section V finishes with some concluding remarks.

II. BACKGROUND

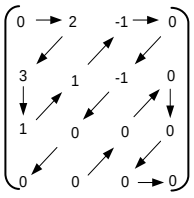
This Section gives essential background for an understanding of the analysis, though the description is necessarily precise, because of the restrictions of space in the paper.

A. CAVLC entropy coding

CAVLC uses both Exp-Golomb codes and VLC for coding of syntax elements. All the syntax elements other than residuals (e.g. motion vectors, their addresses and types) are encoded by the Exp-Golomb mapping, while the residual transform data is encoded by VLC codewords. The residual data is the predictively coded data that has first been transformed and quantized. It is this data which is the subject of selective encryption.

Fig. 1 shows just one sub-macroblock within a video picture [4] after transform, and quantization. In order, to group the more significant low-spatial-frequency transform coefficients at the start of the output, zig-zag scanning takes place before selecting coefficients for subsequent VLC. This scan also has the effect of grouping runs of zero-valued coefficients, and these runs can be summarized through run-length coding. Trailing ones (T1s) normally occur at the end of the set of non-zero coefficients (NZs) and up to three of these are coded separately.

The CAVLC codes the residual block encoding in five sequential steps: (1) the number of NZ coefficients and T1s (coeff_token), (2) the signs of each trailing ones (signs of T1s), (3) the levels of the remaining non-zero coefficients (levels), (4) the total number of zeros before the last coefficient (total_zeros), and (5) each run of zeros (run_before). In the first of these steps, one of four look-up tables is selected according to the number of NZs. The second step simply sets a



Zig-zig Scanning order: 0,2,3,1,1,-1,0,-1,0,0,...
Total NZs (coeff_token) = 6 (2,3,1,1,-1,-1)
levels = 3 (2,3,1)
T1s = 3 (1,-1,-1)
(T1s are 4 but maximum can be 3 only)
total_zeros = 2

NZ =non-zero coefficient, T1 = trailing ones.

Fig. 1. Illustrative example of 4×4 sub-macroblock coefficients coded by VLC

bit for the sign. The third step adaptively thresholds the level according to previous levels. The fourth step assigns a code according to the frequency of that number of zeroes. Finally, the fifth step codes each run of zeroes according to its context within the set of such runs.

B. CABAC entropy coding

CABAC coding [6] is based on three steps: 1) Binarization; 2) Context Modeling; and 3) Binary Arithmetic Coding (BAC). In binarization, any input non-binary syntax elements, such as the quantized transform coefficients, macroblock type specifiers, and motion vector components are converted into unique binary codewords known as bin strings for a given syntax element. There are five basic code trees: the unary code; the truncated unary code (*TU*); the kth order Exponential-Golomb code (*EGk*); the fixed length (*FL*) code; and the Concatenation of the first and third schemes (*UEGk*). Which code (or combination of codes) is applied depends on the characteristics of the syntax element. The bit position in each bin string is known as a bin. Each bin is then passed to one of the two coding decision modes: regular coding mode and by-pass coding mode. The bins in regular coding mode are passed to the next step, context modeling/probability distribution and then encoded by the regular BAC engine. The bins from the bypass coding mode skip the context modeling step and directly enter the bypass BAC engine for the encoding process.

C. Encryption

AES was introduced after a competition to replace the broken Data Encryption Standard (DES). The popularity of AES is due to its minimum memory usage, fast computation in both software and hardware and a simple implementation. AES can use 128 bits, 192 bits and 256 bits length of keys with 10, 12 and 14 encryption round steps respectively. AES algorithm is based on four different byte-oriented transformations:

1) Byte substitution using a substitution table or S-box
2) Shifting rows of the state array by different offsets
3) Mixing the data within each column of the state array
4) Adding a Round Key (Cipher key) to the state array

The AES is basically a symmetric key block cipher using 128-bit block size but it can be used as stream cipher in the Output Feedback (OFB) mode of operation selected in this study. In selective encryption, a small quantity of data is encrypted at a time. So the AES acting as a stream cipher is a valid choice for selective encryption.

D. Gilbert-Elliott channel model

To model the wireless channel in this study, the Gilbert-Elliott (G-E) two-state model [7] was employed. The attraction of this widely-adopted model is that, despite its relative simplicity and the fact that it does not model the physical processes involved, it reproduces the error bursts that are experienced by the application.

It is a discrete-time Markov chain model with two states *G* (for good or gap) and *B* (for bad or burst). We have implemented the fixed G-E model in this study by assuming that the probability of error in the G state is fixed. The two-state transition matrix is determined by the value of *PGG* (probability that the next state is again G) and *PBB*, as represented by *M*:

$$M = \begin{pmatrix} PGG & PGB \\ PBG & PBB \end{pmatrix} \quad (1)$$

The mean state sojourn times, TG and TB, are calculated as follows:

$$TG = \frac{1}{1-PGG} \qquad TB = \frac{1}{1-PBB} \quad (2)$$

The steady state probabilities of being in the G and B states are obtained by (3).

$$PGG = \frac{TG}{TG+TB} \qquad PBB = \frac{TB}{TB+TG} \quad (3)$$

The mean bit-error rate (BER) (M_{BER}) can then be calculated as:

$$M_{BER} = (PGG \times Pg) + (PBB \times Pb) \quad (4)$$

where *Pg* and *Pb* are the probabilities of error occurrence in G and *B* states respectively. For our study *Pg* = 0, which means that the *G* state is error free.

Fig. 2 is an example of the transition states acting on the transmission channel, making the concept of the G-E channel model in this study easier to understand.

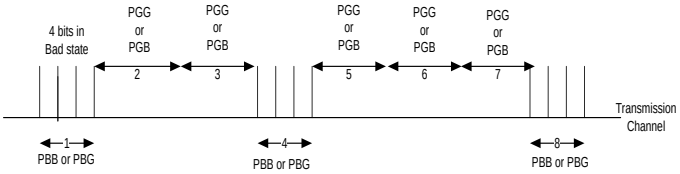


Fig. 2. G-E model state transitions over a transmission channel

III. EXPERIMENTAL METHODOLOGY

On the basis of our analysis in a previous paper [8], three types of codeword are chosen in this paper to fulfill our purpose of selective encryption for CAVLC:

i) Signs of MVD in *se(p)* mapping,
ii) Signs of T1s, and
iii) *EG0* suffix of NZ levels including signs of the NZ levels.

However, in order to not impact on visual quality too much, the number of encrypted parameters is reduced somewhat compared to [8].

The *se(p)* mapping codeword has separate sign and magnitude bits for motion vector differences (MVD). The least significant bit (LSB) of the *se(p)* codeword is the sign bit. The signs of the T1s are encoded separately and the *EG0* suffix part (suffixlength) of non-zero (NZ) levels are encoded according to the rules of NZs prefix value, while the LSB of the suffix length is the sign bit for NZs. The *EG0* suffixes are encrypted both in the regular and escape modes of CAVLC.

Similarly from previous work in [9] for CABAC we encrypted the following parameters:

i) UEG3 suffix, and
ii) Signs of the NZ-TC levels.

The *UEG3* suffix consists of the MVD sign bits if two conditions hold i.e. $|MVD| \geq 9$ and $0 < |MVD| < 9$, the sign bits of TC levels. Again and for the same reason as for CAVLC selective encryption, fewer parameters were encrypted than in [9].

The UDP-lite protocol [10] was assumed to transport the selectively encrypted video stream. This protocol allows the packet payload to be passed up the protocol stack without checksum validation. However, all header content is checked. Therefore, headers were made error free to ensure compatibility with an H.264/SVC decoder. In fact, a single bit error in a header would cause the decoder to crash. The advantage of the UDP-lite protocol is that unlike in UDP transport, which can reject a packet after just one bit error, a packet's payload may still be passed to the decoder, rather than be completely dropped. If a packet is dropped the simplest form of error concealment, previous frame replacement, may be used for simplicity and speed. However, this does not account for motion between video frames. A secondary objective of this study was to judge the extent of bit errors without bit error correction (or error concealment). Other studies [11] have investigated the possibility of bit error correction, which could subsequently also be applied by us.

Bit-errors were introduced into the compressed bit-stream on the basis of following parameters for G-E model. The M_{BER} was varied from 0.01 to 1%, calculated over the total number of bits transmitted per sequence. As previously mentioned *Pg* was set to zero (0) and *Pb* was set to 0.8. The probabilities of being in a good and bad state, *PGG* and *PBB*, are dependent on M_{BER}.

Selective encryption was applied on an H.264/SVC single layer bitstream and tested with the SVC reference software (Joint Scalable Video Model) JSVM 9.19.10 version encoder. For the evaluation of results, three test sequences were chosen with different combinations of features such as motion, colors, objects and texture. The experiments were performed on Common Intermediate Format (CIF) (352 × 288 pixels/frame) video resolutions. For CAVLC encoding, the Baseline profile was used, while for CABAC encoding, the Main/High profile was chosen. The Intra and Inter frames were selectively encrypted in sequence of their occurrence in the bit-stream with same Group of Pictures (GOP) size and Intra period, equal to 16. The video quality was analyzed by taking different quantization parameter (QP) values.

IV. EVALUATION

This Section begins by illustrating the effect of selective encryption (SE) upon video sequences, before examining the effect of bit errors.

A. *Effect of selective encryption on PSNR*

To demonstrate our selective encryption scheme, we have encoded 300 frames of CIF resolution sequences at 30 fps. Tables I and II compare the average Peak Signal-to-Noise Ratio (PSNR) of 300 frames (I+P+B) with and without selective encryption, when encoded with CAVLC and CABAC respectively. Selective encryption is applied with AES-OFB mode with same encryption key of 128-bit length for both entropy coders. Because the selectively encrypted are codec compatible they can be reconstructed *without* decryption.

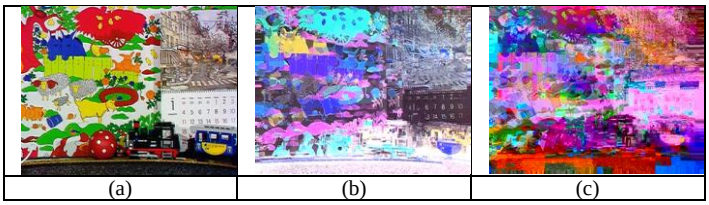
The mean PSNR [dB] value of luma (Y) is lower for all tested sequences when selectively encrypted. The minimum (18) and maximum (44) QP values correspond respectively to high and low objective video quality respectively. The color or chroma components (U and V) are compressed separately

TABLE I
COMPARISON OF PSNR (dB) OF PLAIN AND ENCRYPTED 300 FRAMES (I+P+B) ENCODED WITH CAVLC

Sequences	QP value	Plain Y	SE Y	Plain U	SE U	Plain V	SE V
Foreman	18	40.4	9.08	45.2	24. 01	46.6	24.11
	44	29.9	7.67	36.9	10.61	37.7	10.35
Mobile	18	38.9	12.6	41.5	13.2	41.4	11.3
	44	25.2	8.8	31.1	14.3	30.6	11.3
News	18	43.5	8.6	46.5	20.7	47.5	26.1
	44	30.5	5.7	36.1	18.8	36.9	22.9

TABLE II
COMPARISON OF PSNR (dB) PLAIN AND ENCRYPTED 300 FRAMES (I+P+B) ENCODED WITH CABAC

Sequences	QP value	Plain Y	SE Y	Plain U	SE U	Plain V	SE V
Foreman	18	40.4	9.08	45.2	25.20	46.6	24.89
	44	29.8	7.78	36.9	10.43	37.7	10.33
Mobile	18	38.9	6.9	41.5	12.6	41.4	13.3
	44	25.0	7.2	31.1	12.3	30.6	13.6
News	18	43.5	4.9	46.5	15.9	47.6	21.2
	44	30.4	5.2	35.9	15.9	36.9	21.4



(a) (b) (c)

Figure 3. Impact of selective encryption on the PSNR of *Mobile* (CIF) sequence encoded with 300 frames (I+P+B) and QP 18. (a) Frame # 156 [Y=38.9, U=41.5, V=41.4 dB]. (b) Proposed SE on CABAC [Y=6.9, U=12.6, V=13.3 dB]. (c) Proposed SE on CAVLC [Y=12.6, U=13.2, V=11.3 dB].

from the luminance. The subjective impact of selective encryption on the *Mobile* sequence is shown in Fig. 3. For such short sequences, the gain from using CABAC compared to CAVLC is hardly noticeable in the plain (without selective encryption) results. However, the quality of *Mobile* and *News* appears better after decoding without decrypting for CAVLC compared to CABAC. However, as the intention is to reduce the quality, CABAC appears better.

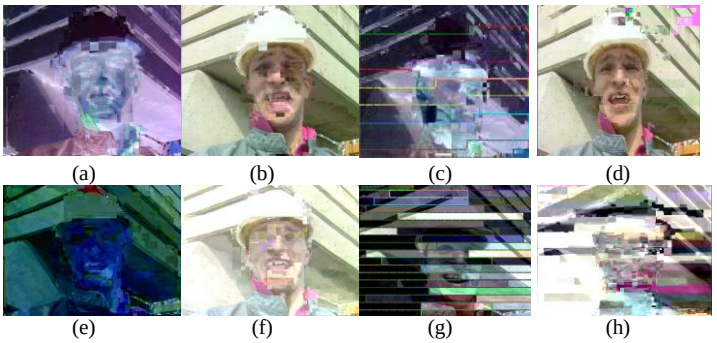
B. *Effect of bit- errors on PSNR*

Experiments to analyze the impact of bit-errors over the selectively encrypted bit-streams encoded with CABAC and CAVLC were performed with H.264-VISA (an analyzer tool) and the JSVM decoder. H.264-VISA will repair bit errors to the extent that invalid codewords are made valid before presentation to the decoder. Without this precaution, the decoder would simply crash when encountering an invalid codeword. The bit-errors were introduced by the G-E model of Section II.D by varying *PBB* according to the desired M_{BER}, which was varied. Multiple runs are taken for each desired M_{BER} to determine the impact of channel errors. Notice that fixed values for *Pg* and *Pb* were given in Section III.

To further elaborate the adopted method, it is discussed by example. Suppose, for 0.1% M_{BER}, the errors were introduced in multiple different places by changing the *PBB* transition state probability i.e. in the first run the errors were introduced in byte 65, 200, 350 and so on, in the next run the errors were introduced into different bytes. The experiments were performed by taking ten runs for each M_{BER}, and finally the arithmetic mean of the PSNRs was calculated for all the runs.

The test sequences were encoded with and without slices for both entropy coders and the errors were introduced in the same places for both the CABAC and CAVLC encoded bit-stream. The slices were set to a size of 1 kB and selective encryption was applied on slices to confine error propagation to within slices. (A slice is a sub-video frame component which can be decoded independently of other slices.)

The results of a 0.2% M_{BER} upon the *Foreman* sequence are shown in Fig. 4, encoded with and without slices. As far as



(a) (b) (c) (d)
(e) (f) (g) (h)

Figure 4. Impact of 0.2% BER on the PSNR of Foreman sequence (Frame # 57) encoded with 300 frames (I+P+B). (a) Encrypted CABAC encoded without slices [Y=7.0, U=18.8, V=22.7 dB]. (b) Decrypted CABAC encoded without slices [Y=27.8, U=35.5, V=32.4 dB}. (c) Encrypted CABAC encoded with slices [Y=6.6, U=20.9, V=22.8 dB]. (d) Decrypted CABAC encoded with slices [Y=26.0, U=35.1, V=33.9 dB]. (e) Encrypted CAVLC encoded without slices [Y=7.8, U=23.7, V=17.8 dB]. (f) Decrypted CAVLC encoded without slices [Y=14.9, U=35.6, V=28.1 dB}. (g) Encrypted CAVLC encoded with slices [Y=7.1, U=21.8, V=21.6] dB. (h) Decrypted CAVLC encoded with slices [Y=9.2, U=22.9, V=23.4 dB].

these individual frames are concerned, there is a small gain from using slices for CABAC but no gain for CAVLC. The

video quality after decryption appears better for CABAC, whether using slices or not. However, examination of a single frame's PSNRs is only indicative. However, this is remedied by the graphs of Fig. 5, which are plotted on the basis of the mean Y-PSNR [dB] against varying M_{BER}. The mean PSNR was taken after the decryption of erroneous bit-streams for both entropy coders.

The results show that the presence of slices is not significant as far as video quality is concerned in these tests. However, beyond an error threshold video quality for CAVLC plummets while CABAC maintains is less affected by errors. Therefore, even at the low error rates represented by Fig. 5, selectively encrypted CAVLC streams are extremely fragile. That is they are very easily affected by bit errors, once the mean error rate rises slightly.

The impact of 1% M_{BER} upon the original (plain) *Foreman* and *Mobile* sequences without selective encryption was also compared with the impact upon the encrypted sequences for both entropy coders. These results are illustrated in Table III. Firstly, it was only possible decode a tenth of the frames after errors when using the CAVLC coder, even when not selectively encrypting the sequence. When selectively encrypting, no frames could be decoded for both *Foreman* and *Mobile*. Therefore, as Fig. 5 indicated, CAVLC is very sensitive to bit errors. However, though there is a relative drop in quality between plain and encrypted video after the introduction of bit errors, using a CABAC coder renders the selectively encrypted sequences relatively immune to bit errors. This implies that if bit error correction algorithms were to be introduced, the most gain will come from application to the CABAC coder.

It is also worth noting, that if a bit-error damages the codeword of more sensitive syntax elements [12], the decoder will crash immediately and not decoded the full bit-stream. Otherwise the decoder decodes the stream with some concealment methods but the video quality suffers. Experiments were performed to find how many frames are decoded after a bit error. Fig. 6 (for the *Foreman* sequence) shows the number of frames decoded after a bit error along

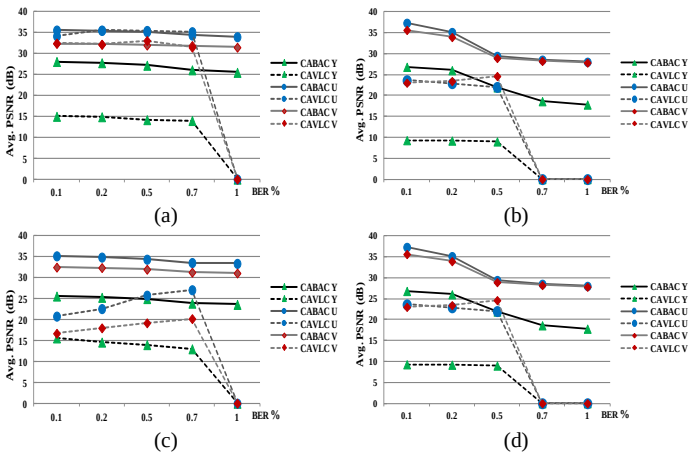


Figure 5. Impact of bit errors on PSNR of test sequences with varying BER. (a) *Foreman* average PSNR of CABAC and CAVLC encoded bit-streams without slices. (b) *Foreman* average PSNR of CABAC and CAVLC encoded bit-streams with slices. (c) *Mobile* average PSNR of CABAC and CAVLC encoded bit-streams without slices. (d) *Mobile* average PSNR of CABAC and CAVLC encoded bit-streams with slices.

TABLE III
COMPARISON OF PSNR [dB] AND DECODED FRAMES FOR ERRONEOUS TEST SEQUENCES WITH AND WITHOUT ENCRYPTION

Sequence with 1% M_{BER} encoded without slices	Avg. PSNR Y	Avg. PSNR U	Avg. PSNR V	Average no. of decoded frames
Erroneous *Foreman* without SE (CABAC)	28	35.5	36.1	299
Erroneous *Foreman* after decryption (CABAC)	25.5	34	31.5	290
Erroneous *Foreman* without SE (CAVLC)	31.05	41.7	42.5	30
Erroneous *Foreman* after decryption (CAVLC)	0	0	0	0
Erroneous *Mobile* without SE (CABAC)	25.6	35.2	32.5	299
Erroneous *Mobile* after decryption (CABAC)	23.6	33.4	31.1	289
Erroneous *Mobile* without SE (CAVLC)	32.6	41.5	41.0	25
Erroneous *Mobile* after decryption (CAVLC)	0	0	0	0

with error bars showing one standard deviation around the mean for the ten tests used. It is confirmed that error recovery from CABAC is much better than from CAVLC. The work in [12] considered discarding corrupted packets if the bit error(s) were to more sensitive syntax errors but retaining packets if less sensitive syntax errors had taken place. Fig. 6 suggests that it is more important to select a CABAC coder, when the effect of bit errors on syntax elements is limited. An interesting feature of Fig. 6 is that employing slices has a worse effect that no using slices. Though slices include synchronization headers to allow the decoder to restart entropy coding, the penalty of slicing is that there are more sensitive syntax elements on a per slice basis than on a per video frame basis. Therefore, there is a greater probability of causing a fatal bit error, e.g. to the resynchronization header, when slicing is used.

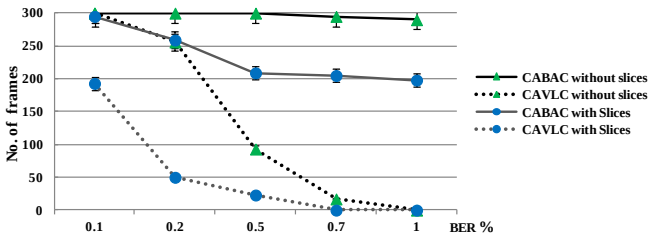


Figure 6. The mean number of frames decoded after a bit error in the *Foreman* sequence when using the CABAC or CAVLC entropy coders (error bars show one standard deviation).

V. CONCLUSION

This paper tested the robustness of an H.264 decoder to bit errors when using either a CABAC or CAVLC entropy coder as the final processing stage at the encoder. As selective encryption, at the very least, is essential to preserve privacy and some measure of confidentiality, the paper compared the relative impact of selectively encrypting the sequence prior to sending it over a wireless channel. Though we examined relatively good channel conditions, it was found that the CAVLC coder, whatever its merits in terms of ease of

implementation, fared very badly. The decoder was easily interrupted by bit errors and quality was very poor once the bit error level passed a low threshold. Notice also that we also used the OFB mode of block encryption, that is probably the most robust mode in terms of recovery from errors. This implies that any future work to correct bit errors after decryption but before decoding should concentrate on using the CABAC coder. The value of this approach, rather than simply discarding packets if even one error is check-summed is that no additional delay is introduced by packet retransmission and no assumptions are necessary as to what form of error concealment is present at the decoder.

REFERENCES

[1] B. Furht, D. Socek, and A.M. Eskicioglu, "Funadamentsof multimedia encryption techniques," in *Multimedia Security Handbook*, B. Furht and D. Kirovski, Eds. Boca Raton, FL: CRC Press, 2004, pp. 95-132.

[2] O. Oyman, and S. Singh, "Quality of experience for HTTP adaptive streaming services," *IEEE Common. Mag*., vol. 50, no. 4, pp. 20-27, 2012.

[3] A. Massoudi, F. Lefebvre, C. De Vleeschouwer, B. Macq, and J.-J., "Quisquater "Overview on selective encryption of image and video: Challenges and perspective," *EURASIP J. on Inf. Security*, vol. 2008, article no. 5, pp. 1-18, Jan. 2008.

[4] I.E.G. Richardson, H.264 and MPEG-4 video compression, Wiley & Sons, Chichester, U.K.

[5] J. Ostermann et al., "Video coding with H.264/AVC: Tools, performance, and complexity," *IEEE Circuits Syst. Mag*., vol. 4, no. 1, pp. 7-28, 2004

[6] D. Marpe, H. Schwarz, and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264 video compression standard," *IEEE Trans.Circuits Syst. Video Technol.*, vol. 13, no. 4, pp. 620-636, 2003.

[7] Ch. Jiao, L. Schwiebert, and B. Xu, "On modeling the packet error statistics in bursty channels," in *Proc. of IEEE Conf. on Local Computer Networks*, 2002, pp. 534- 541.

[8] M. N. Ashgar, M. Ghanbari, and M. Reed, "Sufficient Encryption with Codewords and Bin-strings of H.264/SVC," in *Proc. of IEEE Int. Conf. on Trust, Security and Privacy in Computing and Communications*., Liverpool UK, 2012.

[9] M. N. Ashgar, M. Ghanbari, M. Fleury, and M. Reed, "Efficient selective encryption with H.264/SVC CABAC Bin-STRINGS," in *Proc. Int. Conf. on Image Proc.*, Orlando FL, 2012.

[10] L. Larzon, M. Degermark, and S. Pink, "The lightweight user datagram protocol (UDP-Lite)," IETF, RFC 3828, 2004.

[11] J. Korhone, A. Perkis, and U. Reiter, "Congestion control in wireless links based on selective delivery of erroneous packets," *Signal Processing: Image Commun.*, vol. 26, pp. 106-115, 2006.

[12] A. M. Demirtas, A. R. Reibman, and H. Jafarkhani, "Performance of H.264 with isolated bit error: Packet decoded or discard?" in *Proc. Int. Conf. on Image Proc.*, 2011, pp. 949-952.