Fast Protection of H.264/AVC by Selective Encryption of Cabac

Zafar Shahid, Marc Chaumont, William Puech

FAST PROTECTION OF H.264/AVC BY SELECTIVE ENCRYPTION OF CABAC

Z. SHAHID, M. CHAUMONT and W. PUECH

LIRMM,UMR CNRS 5506, University of Montpellier II,
161, rue Ada, 34392 Montpellier CEDEX 05, France
zafar.shahid@lirmm.fr, marc.chaumont@lirmm.fr, william.puech@lirmm.fr

ABSTRACT

This paper presents a novel method for the protection of copyrighted multimedia content. The problem of selective encryption (SE) has been addressed alongwith compression for the state of the art video codec H.264/AVC. SE is performed in the context-based adaptive binary arithmetic coding (CABAC) module of video codec. For this purpose, CABAC is converted to an encryption cipher. It has been achieved through scrambling of equal length binarized code words. In our scheme, CABAC engine serves the purpose of encryption cipher without affecting the coding efficiency of H.264/AVC by keeping exactly the same bitrate and by generating completely compliant bitstream, and requires insignificant computational cost. Nine different benchmark video sequences containing different combinations of motion, texture and objects are used for experimental evaluation of the proposed algorithm.

Index Terms— Video encryption, CABAC, H.264/AVC, selective encryption.

1. INTRODUCTION

With the rapid evolution of digital media, growth of processing power and availability of network bandwidth, many multimedia applications have emerged in the recent past. As digital data can easily be copied and modified, the concern about its protection and authentication have surfaced. Data encryption is used to restrict access of digital data to only authenticated users. For video data, SE is used in order to encrypt only a small part of the whole bitstream [1]. In this work, we have transformed CABAC module of H.264/AVC into encryption cipher. We have achieved this by scrambling of part of Exp-Golomb suffix of non-zero quantized coefficients (NZs) and sign bits of all NZs.

SE of H.264/AVC has been studied in [2] who has done encryption of some fields like intra-prediction mode, residual data, inter-prediction mode and motion vectors. Carillo *et al.* have presented an idea of encryption for H.264/AVC [3]. They do permutations of the pixels for those macroblocks (MBs) which lie in *region of interest* (ROI). The drawback of this scheme is that bitrate increases when the size of ROI increases. This is due to changes in the statistics of ROI as it is no more a slow varying region which is the basic assumption for video signals. The use of general entropy coder as encryption step has been discussed in the literature in [4]. This method encrypts NZs by using different Huffman tables for each input symbols. The tables, as well as the order in which they are used, are kept secret. This technique is vulnerable to known plaintext attack as explained in [5]. Key-based interval splitting of arithmetic coding (KSAC) [6] follows an approach in which intervals are partitioned in each iteration of arithmetic coding based on secret key. The number of sub intervals in which an interval is divided should be kept small as it increases the size of bitstream. Randomized arithmetic coding [7] is aimed at arithmetic coding but instead of partitioning of intervals like in KSAC, secret key is used to scramble the order of intervals. Both of these techniques make the bitstream non-compliant and thus not suitable for SE. [8] have presented an encryption approach for Context-base Adaptive Variable Length Coding (CAVLC) [9] of H.264/AVC wherein he encrypts scrambles only equal length syntax elements of MB header, thus keeping exactly the same bitrate.

We organize our work as follows. In Section 2, overview of H.264/AVC and CABAC is presented. It explains the working of CABAC along with its limitations from encryption point of view. We explain the whole system architecture in Section 3. Section 4 contains its experimental evaluation and performance analysis including its analysis over the whole rate-distortion (RD) curve and its efficiency for different benchmark video sequences. In Section 5, we present some concluding remarks.

2. PRELIMINARIES

2.1. Overview of H.264/AVC

H.264/AVC [10] is state of the art video coding standard of ITU-T and ISO/IEC. It offers better compression as compared to previous video standards. Like previous video standards, an input video frame is processed into blocks of 16x16 pixels in H.264/AVC, called MB and each of them is encoded separately. Each MB can be encoded as *intra* or *inter*. In *intra* frame, current MB is predicted spatially from MBs which have been previously encoded, decoded and reconstructed (MB at top and left). In *inter* mode, motion compensated

prediction is done from previous frames. The difference between original and predicted frame is call residual. This residual is coded using integer transform coding followed by quantization and zigzag scan. In the last step, either of the entropy coding techniques named context-based adaptive variable length coding (CAVLC) [9] or CABAC [11] is used. H.264/AVC has some additional features as compared to previous video standards. Discrete cosine transform (DCT) transform has been replaced by integer transform (IT) which does not need any multiplication operation and can be implemented by only additions and shifts. In *baseline* profile, H.264/AVC has 4x4 transform in contrast to 8x8 transform of previous standards. It uses a uniform scalar quantization. For *Inter* frame, H.264/AVC supports variable block size motion estimation, quarter pixel accuracy, multiple reference frames, improved skipped and direct motion inference. For *Intra* frame, prediction has been shifted to spatial domain. H.264/AVC provides two modes for entropy coding which are CAVLC and CABAC [11]. Owing to all these additional features, H.264/AVC outperforms previous video coding standards.

2.2. Context-based Adaptive Binary Arithmetic Coding

In entropy coding, quantized transformed coefficients are scanned in reverse order as shown in Fig. 1. In this paper, we are presenting an encryption scheme based on CABAC. It is designed to better exploit the characteristics of NZs, consumes more processing and offers about 10% better compression than CAVLC on average. Run length encoding has been replaced by Significant Map (SM) coding which specifies the position of NZs in 4x4 block. Binary arithmetic coding module (BAC) of CABAC using many context models to encode NZs and context model for a specific NZ depends on the number of NZs which have been already coded in the current block.

Bitstream compliance is a required feature for some direct operations (displaying, time seeking, cutting, etc.). In each MB, header information in encoded first, which is followed by the encoding of MB data. To keep the bitstream compliant and the bitrate unchanged, we cannot encrypt MB header data, since it is used for prediction of future MBs. MB data contains NZs and can be encrypted. A MB is further divided into 16 blocks of 4x4 pixels to process it by IT module. For each 4x4 block inside MB, if *coded block pattern* and *macroblock mode* are set, it indicates that this block is encoded, *coded block flag* (CBF) is encoded first. If CBF is zero, no further data is transmitted; otherwise, SM is encoded. Finally, the absolute value of each NZ and its sign are encoded. Similar to MB, header information of 4x4 block which includes CBP and SM, should not be encrypted for the sake of bitstream compliance.

CABAC consists of multiple stages as shown in Fig. 2. First of all, *binarization* is done in which, non binary syntax elements are converted to binary numbers which are more

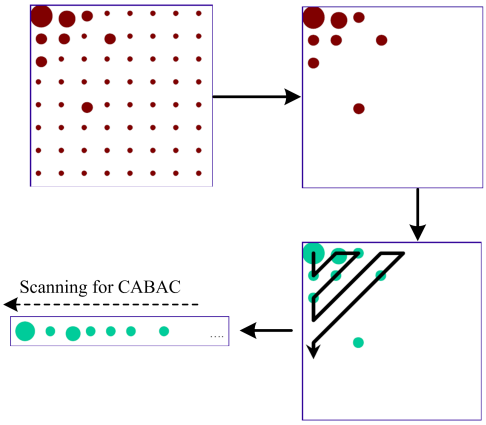

Fig. 1. Scanning order of NZs in CABAC.

amenable to compression by BAC. In CABAC, there are four basic code trees for *binarization* step, namely *the unary code, the truncated unary code, the kth order Exp-Golomb code and the fixed length codes*.

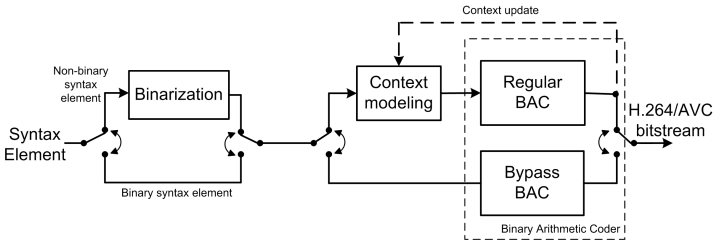

Fig. 2. Block diagram of CABAC of H.264/AVC.

For an unsigned integer value $x \geq 0$, *the unary code* consists of x 1's plus a terminating 0 bit . *The truncated unary code* (TU) is only defined for x with $0 \leq x \leq s$. For $x < s$ the code is given by the unary code, whereas for $x = s$ the terminating 0 bit is neglected. *The kth order Exp-Golomb code* (EGk) is constructed by a concatenation of a prefix and a suffix code word and are suitable for binarization of syntax elements that represent prediction residuals. *The fixed length code* is applied to syntax elements with a nearly uniform distribution or to syntax elements, for which each bit in the fixed length code work represents a specific coding decision e.g., in luma part of the coded block pattern symbol.

Three syntax elements are binarized by concatenation of these trees, namely the coded block pattern, the NZ and the motion vector difference (mvd). Binarization of absolute level of NZs is done by concatenation of TU and EG0. TU constitutes the prefix part with cutoff value $S = 14$. Binarization and subsequent arithmetic coding process is applied to the syntax element $coeff\_abs\_value\_minus1 = abs\_level - 1$, since NZs with zero magnitude are encoded using SM. Kth order Exp-Golomb code (EGk) consists of a prefix and a suffix codeword. For a given unsigned integer value

$x > 0$, the prefix part of the EGk codeword consists of a unary code corresponding to the value of $l(x) = \left[log_2(\frac{x}{2k} + 1)\right]$. The EGk suffix part is computed as the binary representation of $x + 2^k(1 - 2^{l(x)})$ using $k + l(x)$ significant bits. Consequently for EGk binarization, the number of symbols have the same code length of $2l(x) + k + 1$. When k = 0, $2l(x)+k+1 = 2l(x)+1$. So for each NZ with $\|NZ\| > 14$, encryption is done by scrambling of $l(x)$ bits to encrypt the EG0. It is followed by encryption of $coeff\_sign\_flag$ (sign of levels) of all non-zero levels.

3. SYSTEM ARCHITECTURE

To keep the size of the bitstream unchanged, we scramble the NZ with only those NZs whose EG0 code have the same length. We initialize pseudo-random number generator (PRNG) with a secret key. The scrambling space is dependent on the absolute value of NZ. EG0 codes, having same code length, constitute the scrambling space which is $log_2(n + 1)$ where n is the absolute value which is binarized using EG0. The block diagram of our scheme is shown in Fig. 3.

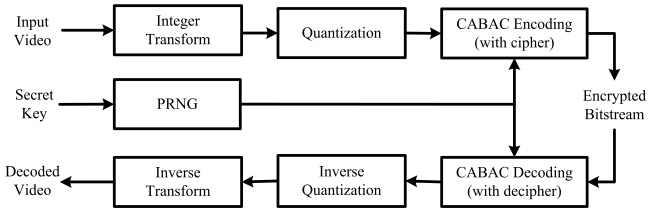

Fig. 3. Encryption and decryption process in H.264/AVC.

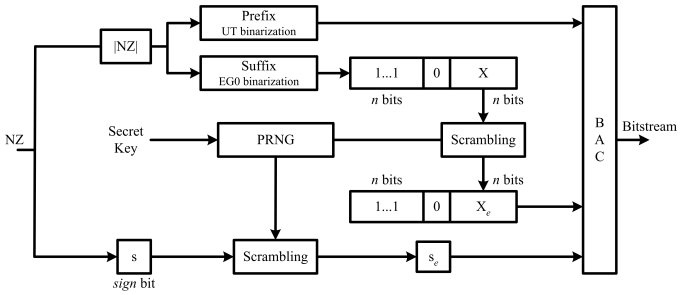

Fig. 4. Encryption of NZs in CABAC of H.264/AVC.

3.1. Encryption Process

The encryption process is shown in block diagram in Fig. 4. Let x be a suffix part of absolute level of NZ which is encoded using EG0 and is to be encrypted with the encrypted coefficients y can be given by:

$$\gamma = rand() \mod log_2(x + 1), \quad (1)$$

$$y = (x + \gamma)) \mod log_2(x + 1). \quad (2)$$

3.2. Decryption Process

For the decryption of NZ in H.264/AVC decoder, the process can be performed in reverse order in Context-based Binary Arithmetic Decoder (CABAD) module of H.264 decoder. Same secret key will be used as seed for PRNG to produce γ. Original value of absolute level of NZ can thus be extracted using encrypted NZ by using the formula:

$$x = (y + log_2 n) - \gamma)) \mod log_2(x + 1). \quad (3)$$

4. EXPERIMENTAL RESULTS

For the experimental results, nine benchmark video sequences have been used for the analysis in QCIF format. Each of them represents different combinations of motion, color, contrast and objects. The video sequences 'bus', 'city' and 'foreman' contain camera motion while 'football' and 'soccer' contain camera panning and zooming along with object motion and texture in background. The video sequences 'Harbour' and 'ice' contain high luminance images with smooth motion. 'Mobile' sequence has a foreground motion with complex still background. To demonstrate the efficiency of our proposed scheme, we have compressed 100 frames as INTRA of each sequence at 30 fps.

Fig. 5. Decoding of encrypted video *"foreman"*: first frame with QP equal to a) 18, b) 24, c) 30, d) 36.

Fig. 5 shows the encrypted video frames at different *quantization parameter* (QP) values of the *foreman* video sequence. Their PSNR values are given in Table 1 and they are compared with the PSNR obtained for the same video frames without encryption. One can note that, whatever is the QP value, the quality of the encrypted video remains in the same range (below 10 dB on average for luma component) which shows that our algorithm is equally efficient over the whole RD curve. Table 2 compares the average PSNR of 100 frames of all benchmark video sequences at a QP value of '18' without encryption and with SE. It confirms that this

Table 1. Comparison of PSNR without encryption and with selective encryption (SE) for *"foreman"* sequence at different QP values.

QP	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
	Without SE	With SE	Without SE	With SE	Without SE	With SE
18	44.43	8.42	45.62	23.87	47.42	22.14
24	39.40	8.38	41.70	24.87	43.86	22.70
30	34.93	8.92	39.38	24.60	40.99	22.71
36	30.80	8.89	37.33	24.65	38.10	22.90

Table 2. Comparison of PSNR without encryption and with SE of benchmark video sequences at QP 18.

Seq.	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
	Without SE	With SE	Without SE	With SE	Without SE	With SE
bus	44.26	7.73	45.22	25.19	46.50	26.86
city	44.28	11.52	45.83	30.50	46.76	31.86
crew	44.81	9.39	45.81	23.80	45.66	19.90
football	44.59	11.46	45.70	15.79	45.98	23.10
foreman	44.43	8.42	45.62	23.87	47.42	22.14
harbour	44.10	9.48	45.60	23.82	46.63	31.20
ice	46.56	10.37	48.70	25.42	49.19	19.73
mobile	44.45	8.42	44.14	13.47	44.04	11.11
soccer	44.26	10.84	46.59	19.69	47.82	24.83

algorithm works well for various combinations of motion, texture and objects while utilizing negligible computational power.

Despite the fact that encryption is performed during the *binarization* step, performance of BAC remains unaffected and bitrate remains exactly the same. It is due to the fact that syntax elements which are encrypted are encoded by bypass coding engine for which CABAC uses fixed context model.

5. CONCLUSION

In this paper, a novel framework for SE of H.264/AVC based on CABAC has been presented. Real-time constraints have been handled successfully by having exactly the same bitrate and by having a compliant bitstream and it is equally efficient over the whole RD curve. The experiments have shown that we can achieve the desired level of encryption in each frame, while maintaining the full H.264/AVC bitstream compliance, under a minimal set of computational requirements. The proposed system can be extended to protect not only the ROI in a video [12] for video surveillance but can also be applied to medical image transmission [13]. In future, this work will be extended for P frames while maintaining same bitrate and bitstream compliance.

Acknowledgment

This work is in part supported by the VOODDO (2008-2011) project of the french Agence Nationale pour la Recherche.

6. REFERENCES

[1] A. Uhl and A. Pommer, *Image and Video Encryption: From Digital Rights Management to Secured Personal Communication*, Springer, 2005.

[2] S. Lian S.and Z. Liu, Z. Ren, and Z. Wang, "Selective Video Encryption Based on Advanced Video Coding," *Lecture notes in Computer Science, Springer-verlag*, , no. 3768, pp. 281–290, 2005.

[3] P. Carrillo, H. Kalva, and S. Magliveras, "Compression Independent Object Encryption for Ensuring Privacy in Video Surveillance," in *ICME*, 2008.

[4] C.-P. Wu and C.-C.J. Kuo, "Design of Integrated Multimedia Compression and Encryption Systems," *IEEE Transactions on Multimedia*, vol. 7, pp. 828–839, 2005.

[5] G. Jakimoski and K.P. Subbalakshmi, "Cryptanalysis of Some Multimedia Encryption Schemes," *IEEE Transactions on Multimedia*, vol. 10, no. 3, pp. 330–338, April 2008.

[6] W. Jiangtao, K. Hyungjin, and J.D. Villasenor, "Binary arithmetic coding with key-based interval splitting," *IEEE Signal Processing Letters*, vol. 13, no. 2, pp. 69–72, Feb. 2006.

[7] M. Grangetto, E. Magli, and G. Olmo, "Multimedia Selective Encryption by Means of Randomized Arithmetic Coding," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 905–917, Oct. 2006.

[8] C. Bergeron and C. Lamy-Bergot, "Complaint selective encryption for h.264/avc video streams," 30 2005-Nov. 2 2005, pp. 1–4.

[9] G. Bjontegaard and K. Lillevold, "Context-Adaptive VLC Coding of Coefficients," in *JVT Document JVT-C028*, Fairfax, VA, May 2002.

[10] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[11] D. Marpe, H. Schwarz, and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, July 2003.

[12] J.-M. Rodrigues, W. Puech, and A.G. Bors, "Selective Encryption of Human Skin in JPEG Images," in *Proc. IEEE Int. Conf. on Image Processing, Atlanta, USA*, Oct. 2006, pp. 1981–1984.

[13] W. Puech and J.M. Rodrigues, "A New Crypto-Watermarking Method for Medical Images Safe Transfer," in *Proc.* 12^{th} *European Signal Processing Conference (EUSIPCO'04)*, Vienna, Austria, 2004, pp. 1481–1484.