SELECTIVE ENCRYPTION OF C2DVLC OF AVS VIDEO CODING STANDARD FOR I & P FRAMES

Z. SHAHID, M. CHAUMONT and W. PUECH

LIRMM,UMR CNRS 5506, University of Montpellier II,
161, rue Ada, 34095 Montpellier Cedex 05, France
Email: zafar.shahid@lirmm.fr, marc.chaumont@lirmm.fr, william.puech@lirmm.fr

ABSTRACT

In this paper, a novel method for fast protection of AVS video coding standard alongwith compression is presented. Here the problems of compression and selective encryption (SE) have been simultaneously addressed for AVS part-2 Jizhun profile. It is performed in the context-based 2D variable length coding (C2DVLC) module of video codec. SE is performed by using the Advanced Encryption Standard (AES) algorithm with the Cipher Feedback (CFB) mode on a subset of codewords. C2DVLC serves the purpose of encryption step without affecting the coding efficiency of AVS by keeping the bitrate unchanged, generating completely compliant bitstream and utilizing negligible computational power. Nine different benchmark video sequences containing different combinations of motion, texture and objects are used for experimental evaluation of the proposed algorithm.

***Keywords*—** AVS video coding standard, video protection, C2DVLC, selective encryption

1. INTRODUCTION

With the rapid growth of processing power and 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. Encryption is used to restrict access of digital data to only authenticated users. For video data, the concept of SE has evolved in which only a small part of the whole bitstream is encrypted [1]. In this work, we have transformed C2DVLC module of AVS into crypto-compression module by the encryption of non-zero coefficients (NZs).

AVS is a new standard and it still is to be analyzed for encryption but SE of other video standards like H.264/AVC has been discussed in literature. Lian et *al.* have done partial encryption of some fields of H.264/AVC as intra-prediction mode, residue data, inter-prediction mode and motion vectors [2]. Carrillo et *al.* [3] have also presented an idea of encryption for H.264/AVC. They do permutations of the pixels of those macroblocks (MBs) which are in ROI. The drawback of this scheme is that the bitrate increases as the size of ROI increases. This is due to change 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 an encryption step has been studied in [4]. It 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]. For H.264/AVC, entropy coding based SE has been discussed for context adaptive variable length coding (CAVLC) [6] and context adaptive binary arithmetic coding (CABAC) [7] which fulfills real-time constraints by producing format-complaint encrypted bitstream without changing the bitrate.

This paper has been presented as follows. In Section 2, overview of AVS and C2DVLC is presented. Comparison of AVS with H.264/AVC is also presented in this section. We have explained the proposed algorithm in Section 3. Section 4 contains its experimental evaluation, followed by the concluding remarks in Section 5.

2. PRELIMINARIES

2.1. Overview of AVS Video Coding Standard

AVS [8] is the state of the art video coding standard of China and is based on motion compensated hybrid framework. It has slightly less performance but is much less complex than H.264/AVC [9], which is the state of the art video coding standard of ITU-T and ISO/IEC. A video frame is processed into blocks of 16x16 pixels, called macroblock (MB). Each MB can be encoded as *intra* or *inter*.

In *intra* frame, spatial prediction is performed from reconstructed (i-e top and left) MBs. It is performed on blocks of 8x8, in contrast to 4x4 and 16x16 block size in H.264/AVC. Spatial prediction in AVS is less complex with only five modes for luma, as compared to thirteen modes in H.264/AVC. Reference pixels, which are to be used for prediction, are first low pass filtered. In *inter* mode, motion compensated prediction is done from previous frames. It supports variable block size motion estimation up to 8x8 block, quarter pixel motion estimation and multiple reference frames in *inter* frame.

The difference between original and predicted frame is called a *residual*. This *residual* is coded using transform

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

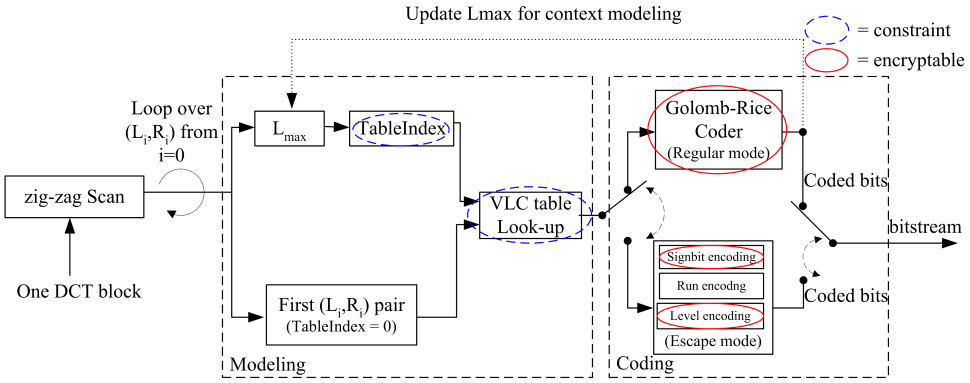

Fig. 1. Block diagram of C2DVLC showing constraints and encrypt-able blocks.

coding. In AVS, standard DCT transform has been replaced by 8x8 Integer Cosine Transform (ICT) [10] which does not need any multiplication operation and can be implemented by only additions and shifts. It is followed by quantization and zigzag scan. For quantization, QP value ranges 0-63 with a period of approximate 8.

In AVS Part-2, two modes for entropy coding are supported, namely C2DVLC in Jizhun profile and context-based binary arithmetic coding (CBAC) in Jiaqiang profile [11]. In the last step, either of the entropy coding techniques namely C2DVLC or CBAC is used.

On the decoding side, compressed bitstream is decoded by entropy decoding module, followed by inverse-zigzag scan. These coefficients are then inverse-quantized and inverse transformed to get the *residual* signal which is added to the predicted signal to reconstruct the original signal back. AVS decoder complexity is further reduced by moving the inverse scaling from decoder to encoder module.

In comparison to H.264/AVC, AVS Part-2 Jizhun profile has about 3% efficiency loss as compared to H.264/AVC main profile in terms of bit saving on HD progressive-scan sequences [12]. 8x8 transform coding, 8x8 spatial prediction, motion compensation up to 8x8 block, 8x8 in-loop deblocking filter and 2D variable length coding are major tools of AVS Part-2 which distinguishes it from H.264/AVC.

2.2. Context-based 2D Variable Length Coding

In this standard, an efficient context-based 2D-VLC entropy coder is designed for coding 8x8 block-size transform coefficients, where 2D-VLC means that a pair of Run-Level (L_i, R_i) is regarded as one event and jointly coded [13].

Fig. 1 illustrates the working of C2DVLC by a flowgraph. A DCT block contains several NZs which are transformed to (L_i, R_i) pairs after the zig-zag scan. C2DVLC starts the coding in reverse order using 2D-VLC table with $TableIndex = 0$. Every table has certain range for (L_i, R_i) as shown in Fig. 2. If the current (L_i, R_i) lies in the range of current 2D-VLC table, it is encoded by $regular\ mode$. Otherwise $escape\ mode$ is used.

In $regular\ \ mode$, the value of syntax element is firstly mapped to a non-negative integer $CodeNumber$ using a table look-up operation. These $CodeNumbers$ are then mapped to corresponding Exp-Golomb codewords. For (L_i, R_i) pairs having negative value for level, $CodeNumber$ is incremented by 1. For example, for $(L_i, R_i) = (2, 1)$ $CodeNumber$ is 11 and for $(L_i, R_i) = (-2, 1)$ $CodeNumber$ is 12. Exp-Golomb codes have regular structures, which means that any non-negative $CodeNumber$ can be mapped to a unique binary codeword using the regular code-constructing rule. Due to the regular codeword structure, the binary code for a given $CodeNumber$ can be constructed in coding process without involving high computational complexity. In AVS, it is a valuable feature that resolves the problem of high memory requirement for multiple VLC tables. In $escape\ mode$, L_i and R_i are coded separately using Exp-Golomb codewords. R_i and sign of L_i are jointly coded. While for the magnitude of L_i, a prediction is first performed and then the prediction error is coded.

C2DVLC switches the 2D-VLC tables based on the maximum magnitude of the previously coded levels. Let L_{max} be the maximum magnitude of the previously coded levels. The $TableIndex$ for coding of next (L_i, R_i) is updated if L_{max} is greater than the threshold of the current table as given below:

$$TableIndex = \quad j, \quad \text{if } (Th[j+1] > L_{max} \geq Th[j]) \quad (1)$$

with the threshold for each table given as:

$$Th[0\ldots7] = \begin{cases} (0,\ 1,\ 2,\ 3,\ 5,\ 8,\ 11,\ \infty) & \textit{intra\_luma} \\ (0,\ 1,\ 2,\ 3,\ 4,\ 7,\ 10,\ \infty) & \textit{inter\_luma} \\ (0,\ 1,\ 2,\ 3,\ 5,\ \infty,\ \infty,\ \infty) & \textit{chroma} \end{cases} \quad (2)$$

This process is repeated for all the (L_i, R_i) pairs. At last the EOB flag is coded to signal the end of block.

2D-VLC entropy coding has already been used in former video coding standards such as MPEG-2/4. But it has two main differences here. First, Huffman coding has been replaced by Exp-Golomb coding in AVS. Second, former video coding standards are not adaptive and use single VLC table to code a certain type of transform blocks, e.g. one table for intra blocks,

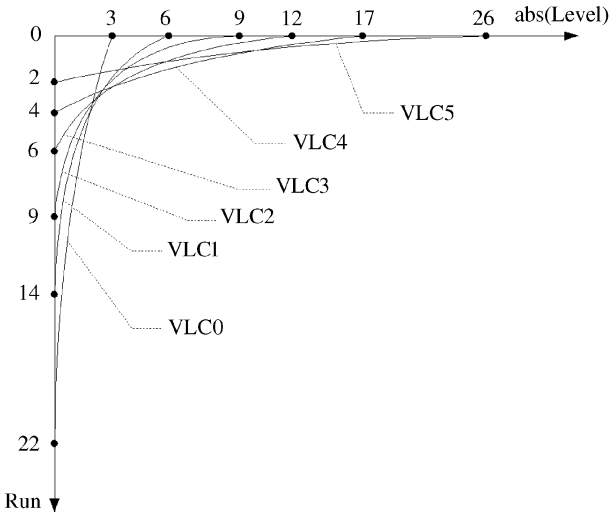

Fig. 2. Limit of each 2D-VLC table of C2DVLC.

one table for inter blocks, etc.

In AVS, 19 2D-VLC tables have been introduced for coding of residual coefficients and the memory requirement is only about 1k bytes. This method gives gain up to 0.23 dB compared to one-table-for-one-type-of-block coding method [14]. For further details about C2DVLC, please refer to [11].

2.3. C2DVLC vs. CAVLC

The common thing between C2DVLC of AVS and CAVLC of H.264/AVC is that both of them are adaptive to the local statistics of DCT coefficients and coding efficiency of both of them is similar. Otherwise C2DVLC is substantially different as compared to CAVLC. In CAVLC, Exp-Golomb coding is used only for coding of syntax elements. Transform coefficients are converted to levels and runs, which are coded **separately** using multiple VLC tables.

While in AVS, Exp-Golomb coding is used for coding all syntax elements including transform coefficients. Transform coefficients are first converted to (L_i, R_i) pairs. These pairs are mapped to $Codenumber$ which is coded using Exp-Golomb codes in $regular\ mode$. In $escape\ mode$, L_i and R_i are coded separately using Exp-Golomb codes.

3. THE PROPOSED ALGORITHM

Encrypted bitstream (EB) compliance is a required feature for several operations (fast forward, fast backward, parsing etc.). To keep the bitrate unchanged alongwith the EB format compliance, we perform encryption of C2DVLC while fulfilling the following constraints:

- In the (L_i, R_i) pair, only L_i can be encrypted. R_i value must not be changed otherwise the bitstream will not be decodable.
- From equation (1), the encrypted symbol should be such that L_{max} remains in the same interval, thus selecting the same context for the next (L_i, R_i).
- For Exp-Golomb coding, the length of the encrypted codeword must be equal to that of original codeword.

Encryption of C2DVLC is not straight forward like that of CAVLC and pose number of challenges. In CAVLC, codespace is always full and we have specific bits which can be encrypted. In case of C2DVLC, for $regular\ mode$, we can encrypt only the levels and their sign bits while taking into account the constraints described above. In this mode, codespace is not full because of two major limitations. First, we do not have specific bits to be encrypted and the encryption space (ES) is not a power of 2. Second, non-consecutive $CodeNumbers$ are assigned to consecutive levels.
In $escape\ mode$, we can encrypt the sign bit and suffix of the Exp-Golomb codeword. Here code space is not garunteed to be full because of 2^{nd} constraint. Fig. 1 encircles the functional blocks with constraints. It also shows the encryptable functional blocks.

3.1. Encryption Process

AES algorithm consists of a set of steps repeated for a number of iterations called rounds [15]. In CFB mode, AES is a stream cipher. In this mode, each ciphertext block Y_i is XORed with the incoming plaintext block X_{i+1} before being encrypted with the key k. For the first iteration, Y_0 is substituted by an initialization vector (IV). The keystream element Z_i is then generated and the ciphertext block Y_i is produced as:

$$\begin{cases} Z_i &= E_k(Y_{i-1}), for\ i \geq 1 \\ Y_i &= X_i \oplus Z_i \end{cases}, \qquad (3)$$

where $\oplus$ is the XOR operator.
According to equation (3), the previous encrypted block Y_{i-1} is used as the input of the AES algorithm in order to create Z_i. Then, the current plaintext X_i is XORed with Z_i in order to generate the encrypted text Y_i. The encryption and decryption processes of AES are shown in Fig. 3.

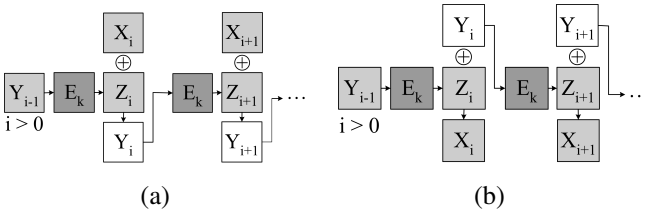

Fig. 3. CFB stream cipher: (a) Encryption, (b) Decryption.

For the initialization, the IV is created from the secret key k according to the following strategy. The secret key k is used as the seed of the pseudo-random number generator (PRNG). Firstly, the secret key k is divided into 8 bits (byte) sequences. The PRNG produces a random number for each byte component of the key, which defines the order of IV formation. Then, we substitute Y_0 with the IV, and Y_0 is used in AES to produce Z_1. With the CFB mode of the AES algorithm, the generation of the keystream Z_i depends on the previous encrypted block Y_{i-1}. Consequently, if two plaintexts are identical $X_i = X_j$ in the CFB mode, then always the two corresponding encrypted blocks are different, $Y_i \neq Y_j$. To handle the limitation that encryption space is not equal to 2^n, if encryption codeword lies outside the valid range, the encryption of codeword is performed again.

3.2. Example

Let us have a $(L_i, R_i) = (6, 0)$ with $TableIndex = 3$ as shown in Fig. 4. By encoding it with $regular\ \ mode$ of C2DVLC, its $CodeNumber$ will be 17 and its Exp-Golomb codeword will take 7 $bits$. Now let us examine the ES available for this (L_i, R_i) pair.
The 1^{st} constraint is that only L_i can be encrypted in the (L_i, R_i) pair. So the ES consists of the levels which have valid codeword in $TableIndex = 3$ with $R_i = 0$. These are $CodeNumbers$ {0, 2, 4, 9, 11, 17, 21, 25, 33, 39, 45, 55} related to levels $\{1, 2, \ldots, 12\}$.
The 2^{nd} constraint is that the magnitude of the encrypted level should be within the interval that creates the same $TableIndex$ for the next (L_i, R_i). From equation (2), we see that $L_i = 6$ will increase the $TableIndex$ from 3 to 4. So the encrypted L_i should also be in the same interval i-e {5, 6, 7}. From $Table[3]$, the $CodeNumbers$ for these levels are {11, 17, 21} for positive sign and {12, 18, 22} for negative sign.
The 3^{rd} constraint implies that out of these $CodeNumbers$, only those make the ES which have the same Exp-Golomb codeword length as the original level (7 $bits$). Out of the 6 $CodeNumbers$ which have been selected in the last step, five $CodeNumbers$ have the same length as $(6, 0)$. The only $CodeNumber$ whose length is different is 11. So $(6, 0)$ pair has ES of 5 in this example.

In CAVLC, $escape\ mode$ is rarely used. While in C2DVLC, it is very frequently used and it may be difficult to find a bock in which all the transform coefficients are coded using $regular\ \ mode$.For $regular\ \ mode$ of C2DVLC, ES ranges from 1 to 25 and ES is up to 2^n for $escape\ \ mode$, where n is the number of bits in the suffix of Exp-Golomb codeword, while respecting the 2^{nd} constraint.

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. We have used the AVS version RM 6.2c and have performed SE of C2DVLC for for *intra & inter* frames [16].

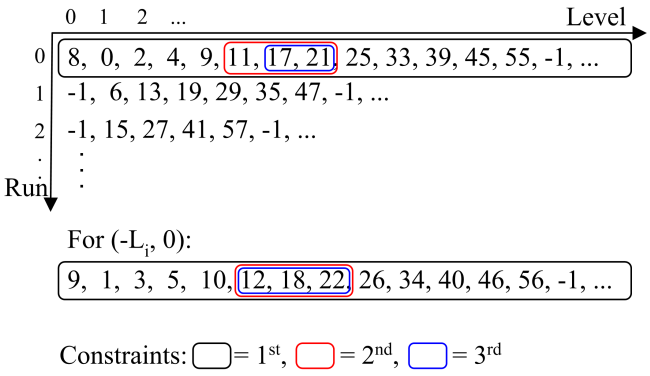

Fig. 4. Encryption of (L_i, R_i) pair in $regular\ \ mode$ of C2DVLC for $TableIndex = 3$.

4.1. *Intra* Frames

To demonstrate the efficiency of our proposed scheme for *intra* frames, we have compressed 100 frames of each sequence at 30 fps as *intra*. Table 1 compares the PSNR of 100 frames of nine video sequences at QP value of 28 without and with SE. One can note that the proposed algorithm works well for video sequences having various combinations of motion, texture and objects and is significantly efficient. The average PSNR of all the encrypted sequences is 10.22 dB for *luma*. Fig. 5 shows the encrypted video frames at different QP values for *foreman*. PSNR comparison over whole range of QP values is given in Table 2. One can note that, PSNR of the SE video remains in the same lower range (around 10 dB on average for *luma*) for all QP values.

4.2. *Intra & Inter* Frames

Video data normally consists of an *intra* and a trail of *inter* frames. *Intra* frames are inserted periodically to restrict the drift because of lossy compression and rounding errors. For experimental evaluation of *intra & inter* frames, *intra period* is set to 10 in a sequence of 100 frames. Table 3 verifies the performance of our algorithm for all video sequences for *Intra & Inter* frames at QP value of 28. Average PSNR of *luma* for all the encrypted sequences is 10.43 dB. Results shown in Table 4 verify the effectiveness of our scheme over the whole range of QP values for *foreman* for *intra & inter* frames.

5. CONCLUSION

In this paper, a novel framework for SE of AVS based on C2DVLC has been presented. Since all the constraints posed by the contexts and Exp-Golomb codewords for each NZ, have been fulfilled, encrypted bitstream is fully compliant to AVS format and is decodable by reference AVS decoder. Real-time

Table 1. Comparison of PSNR without encryption and with SE of benchmark video sequences at QP = 28 for *intra*.

	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
Seq.	Orig.	SE	Orig.	SE	Orig.	SE
bus	37.93	7.76	41.60	25.99	42.82	27.90
city	38.11	12.34	42.93	30.71	44.19	31.05
crew	39.45	10.20	41.81	24.99	40.83	22.19
football	39.10	11.89	41.48	16.26	42.32	24.06
foreman	38.93	9.12	42.09	23.82	43.87	26.23
harbour	37.80	9.81	42.16	24.39	43.65	32.49
ice	41.42	10.72	44.50	26.15	44.78	20.25
mobile	37.92	8.71	38.63	14.47	38.39	11.81
soccer	38.25	11.42	42.91	22.07	44.31	24.10
avg.	38.87	10.22	42.44	23.21	43.35	24.45

Table 2. Comparison of PSNR without encryption and with SE for *foreman* at different QP values for *intra*.

	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
QP	Orig.	SE	Orig.	SE	Orig.	SE
12	49.56	8.97	50.10	24.76	50.82	21.50
20	44.10	8.85	45.71	26.29	47.44	22.07
28	38.93	9.12	42.09	23.82	43.87	26.23
36	34.37	8.91	39.30	23.84	40.23	22.00
44	30.54	9.05	37.06	23.90	37.33	21.66
52	26.93	9.97	35.32	25.50	35.86	20.78

Table 3. Comparison of PSNR without encryption and with SE of benchmark video sequences at QP = 28 for *intra* & *inter*.

	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
Seq.	Orig.	SE	Orig.	SE	Orig.	SE
bus	36.49	7.96	41.84	25.22	43.07	27.94
city	36.87	12.06	43.20	31.08	44.44	31.69
crew	38.27	13.42	41.97	25.41	40.93	22.36
football	37.89	11.79	41.50	15.15	42.41	23.34
foreman	37.92	8.55	42.36	24.94	44.15	26.05
harbour	36.20	9.79	42.43	25.01	43.85	31.35
ice	40.20	10.32	44.70	26.39	44.98	18.56
mobile	36.06	8.53	38.78	14.84	38.46	12.33
soccer	37.15	11.48	43.05	20.39	44.47	24.15
avg.	37.45	10.43	42.20	23.16	42.97	24.20

Table 4. Comparison of PSNR without encryption and with SE for *foreman* at different QP values for *intra* & *inter*.

	PSNR (Y) (dB)		PSNR (U) (dB)		PSNR (V) (dB)	
QP	Orig.	SE	Orig.	SE	Orig.	SE
12	47.19	9.32	50.01	25.05	50.46	23.53
20	42.74	8.94	46.01	26.36	47.66	20.62
28	37.92	8.55	42.36	24.94	44.15	26.05
36	34.01	8.11	39.53	23.92	40.53	21.62
44	30.42	9.56	37.27	25.36	37.69	20.13
52	26.97	10.71	35.65	24.39	36.00	19.85

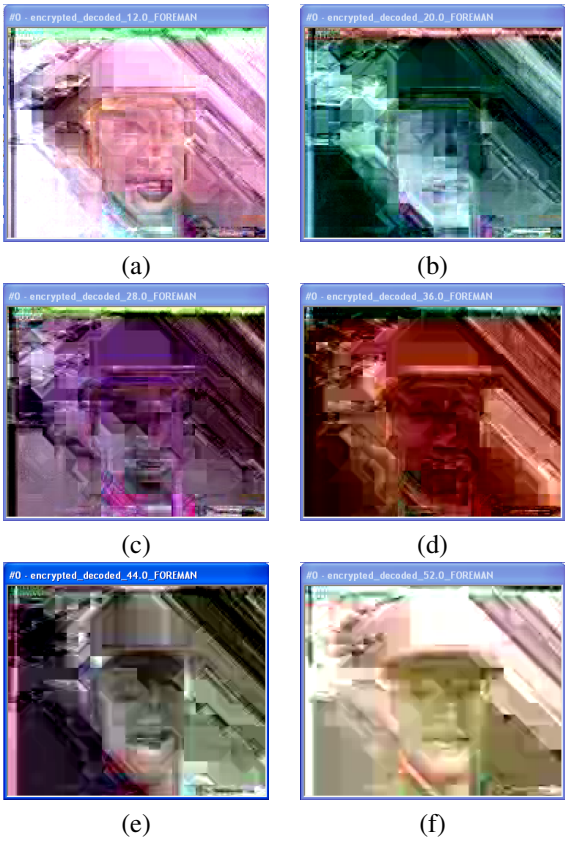

Fig. 5. Decoding of encrypted video *"foreman"*: first frame with QP equal to: a) 12, b) 20, c) 28, d) 36, e) 44, f) 52.

constraints have been successfully fulfilled by having exactly the same bitrate.

The experiments have shown that we can achieve the desired level of encryption in each frame, while maintaining the full AVS video coding standard compliance, under a minimal set of computational requirements. The proposed system can be extended to protect only ROI [17] in video surveillance and can be applied to medical image transmission [18].

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, 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 *Proc. IEEE International Conference on Multimedia and Expo*, Hannover, Germany, 2008, pp. 273–276.

[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] Z. Shahid, M. Chaumont, and W. Puech, "Fast Protection of H.264/AVC by Selective Encryption," in *Proc. SinFra 2009, Singaporean-French IPAL Symposium, Fusionopolis*, Singapore, 18-20 Feb. 2009.

[7] Z. Shahid, M. Chaumont, and W. Puech, "Fast Protection of H.264/AVC by Selective Encryption of CABAC for I & P frames," in *Proc.* 17^{th} *European Signal Processing Conference (EUSIPCO'09)*, Glascow, Scotland, 2009, pp. 2201–2205.

[8] L. Fan, S. Ma, and F. Wu, "Overview of AVS Video Standard," in *Proc. IEEE International Conference on Multimedia and Expo*, Taipei, Taiwan, 2004, pp. 423–426.

[9] "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 / ISO/IEC 14496-10 AVC)," Tech. Rep., Joint Video Team (JVT),Doc. JVT-G050, March 2003.

[10] S. Ma and W. Gao, "Low Complexity Integer Transform and Adaptive Quantization Optimization," *J. Comput. Sci. Technol.*, vol. 21, no. 3, pp. 354–359, 2006.

[11] L. Zhang, Q. Wang, N. Zhang, D. Zhao, X. Wu, and W. Gao, "Context-based Entropy Coding in AVS Video Coding Standard," *Image Commun.*, vol. 24, no. 4, pp. 263–276, 2009.

[12] X. Wang and D. Zhao, "Performance Comparison of AVS and H.264/AVC Video Coding Standards," *J. Comput. Sci. Technol.*, vol. 21, no. 3, pp. 310–314, 2006.

[13] Q. Wang, D.-B. Zhao, and W. Gao, "Context-based 2D-VLC Entropy Coder in AVS Video Coding Standard," *J. Comput. Sci. Technol.*, vol. 21, no. 3, pp. 315–322, 2006.

[14] Q. Wang, D. Zhao, S. Ma, Y. Lu, Q. Huang, and W. Ga, "Context-based 2D-VLC for Video Coding," in *Proc. IEEE International Conference on Multimedia and Expo*, June 2004, vol. 1, pp. 89–92 Vol.1.

[15] J. Daemen and V. Rijmen, "AES Proposal: The Rijndael Block Cipher," Tech. Rep., Proton World Int.l, Katholieke Universiteit Leuven, ESAT-COSIC, Belgium, 2002.

[16] "http://159.22.42.57/incoming/dropbox/video_software/Rm62c.zip," .

[17] 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.

[18] 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.