A Selective Encryption for Heterogenous Color JPEG Images Based on VLC and AES Stream Cipher

José Marconi Rodrigues, William Puech, Adrian G. Bors

▶ **To cite this version:**

José Marconi Rodrigues, William Puech, Adrian G. Bors. A Selective Encryption for Heterogenous Color JPEG Images Based on VLC and AES Stream Cipher. CGIV: Colour in Graphics, Imaging and Vision, Jun 2006, Leeds, United Kingdom. pp.34-39. lirmm-00122734

HAL Id: lirmm-00122734

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00122734

Submitted on 4 Jan 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Selective Encryption for Heterogenous Color JPEG Images Based on VLC and AES Stream Cipher

J.M. Rodrigues[a], W. Puech[a] and A.G. Bors[b]
[a]Laboratory LIRMM, University of Montpellier II, France
[b]Dept. of Computer Science, University of York, U.K.
jose-marconi.rodrigues@lirmm.fr, william.puech@lirmm.fr, adrian@cs.york.ac.uk

Abstract

Nowadays, the most important engine to provide confidentiality is encryption. Therefore, the classical and modern ciphers are not suitable for huge quantity of data in real-time environment. Selective encryption (SE) is an approach to encode only the most important portion of the data in order to provide a proportional privacy and to reduce computational requirements. The objective of our work is to leave free the low-resolution image and give full-resolution access only for authorized person. This approach is based on AES stream ciphering using VLC (Variable Length Coding) of the Huffman's vector. The proposed scheme allows decryption of a specific region of image and result in a significant reduction in encrypting and decrypting processing time. It also provides a constant bit rate and keep the JPEG bit-stream compliance. We have illustrated our method on a digital painting of the Louvre Museum of Paris, France.

Introduction

The explosion of the Internet popularity brought the necessity of protection of binary information in many fields such as medical, work-of-art, law enforcement and military. There is a wide sort of demands of secure multimedia transmission according to the target. The military strategy and the law enforcement, for example, demand full encryption. However there is a huge spectrum of applications that demands security on a lower level, it means partial or selective encryption. We can bring up several applications that portions of image data must be visible to allow database searching and classification. Applications in education field where images need be partially identified without disclosing the total information. Paintings (work-of-art) that must be exhibited in a scalable visual resolution. Some personal photographs taken from cellular telephones. Medical pictures (teleradiology) taken from a mobile capturing device. For vital reasons, these kinds of images must be sent quickly and no full encryption is needed. Therefore, the search for fast encoding/decoding procedures specifically appropriate for specific necessities is mandatory for multimedia performance and security. Selective encryption is an approach to reduce the computational requirements for huge volumes of multimedia data in networks with different client device capabilities [6].

Basically there are two kinds of selective encryption. The first the decoder can render a version of the media, with a quality severely degraded, but still discernible. The second the decoder can not render the image because some bits of the compressed variable-length data is changed. When those bits are encrypted, it becomes impossible to parse and recognize the meaning of all the subsequent non-encrypted bits [8]. The problem of the second approach is the bit-stream compliance format.

In this work we propose a new approach for selective VLC (Variable Length Coding) encryption with a stream cipher for JPEG images. It is based on AES (Advanced Encryption Standard) stream ciphering applied in the Huffman coding. The employment of selective encryption and compression together will save machine resource and keep the format compliance and compression rate. It also allows versatility in decode process. In Section *Previous works*, we introduce the main ideas, previous works, review basic terms such as JPEG and AES algorithms and discuss a possible application scenario. In Section *The proposed method*, we introduce the proposed method. Finally, Section *Experimental results*, we show our experiments applied on a digital paintings.

Previous works

The confidentiality in lower power environment is generally taken into account by cipher programs. Thus, for image processing applications it is always worth to minimize the computational overhead. However the software implementations of the classical ciphers are usually too slow to process image and video data in commercial systems [6]. The selective encryption can match applications requirements without the overhead of full encryption because only the minimum necessary data is ciphered. However the security of SE is always lower when compared to full encryption. Because selective encryption only protects the visually most important parts of an image or video to minimize computational efforts in real-time applications. The only reason to accept this drawback is the substantial computational reduction regarding the total encryption time. Thus, the reasonable utilization of SE should be investigated thoroughly in order to decide whether its use is appropriate for the environment and confidentiality required.

Before to present previous works of selective encryption, we review basic terms of JPEG and AES algorithms.

The JPEG algorithm

The standard JPEG format decomposes the image in blocks of 8×8 pixels. These blocks are transformed from the spatial to the frequency domain by the Discrete Cosine Transform (DCT). The goal of this transformation process is to decorrelate the pixels of each block, or to pack as much information as possible into the smallest number of transformed coefficients. Then, each DCT coefficient is divided by its corresponding constant in a standard quantization table and rounded to the nearest integer. Then, the quantized DCTed coefficients are scanned in a predefined zigzag order according to the increasing spatial frequency Figure 1. Then, this sequence of quantized coefficients is used in the entropy encoding which is depicted in the next section.

The principal characteristic of JPEG is that it can be implemented in hardware or in software. To be JPEG compatible, the algorithm or product, must include support for the sequential baseline system [5]. That means (DCT-based process, only 8-bit

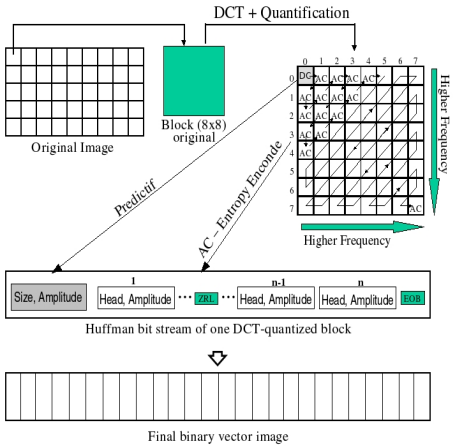

Figure 1. *JPEG algorithm.*

images, sequential and Huffman coding). The baseline system is required for all DCT-based decoders.

Entropy encoding and Huffman coding

In the Huffman coding block, the quantized coefficients are coded by the pair {(HEAD),(AMPLITUDE)}. The HEAD contains the controllers provided by the Huffman's tables. The AMPLITUDE is a signed-integer that is the amplitude of the nonzero AC, or in the case of DC is the difference between two neighbor DC coefficients. For the AC coefficients the HEAD is composed of (RUNLENGTH, SIZE), while for the DC coefficients it is made up only by SIZE. Because DC are highly predictable, they are treated separately in the Huffman coding. Our approach is essentially based on encrypting of some AC coefficients thus, we review some concepts of Huffman encoding.

For the AC coding, JPEG uses a method based on combining run-length and amplitude information. It aggregates zero coefficients into runs of zeros. RUNLENGTH is a consecutive number of zero-valued AC coefficients which precede nonzero-value in the zigzag sequence. The SIZE is the amount of necessary bits to represent the AMPLITUDE. Two extra codes that correspond to (RUNLENGTH, SIZE) = (0, 0) and (15, 0) are used for symbolizing EOB (End-Of-Block) and ZRL (Zero Run Length) respectively. The EOB is transmitted after the last nonzero coefficient in a quantized block. The ZRL symbol is transmitted whenever RUNLENGTH is greater than 15 and represents a run of 16 zeros. One of the objectives of our method is not to change those codes.

The AES encryption algorithm

The Advanced Encryption Standard (AES) is symmetric block cipher and had as objective to substitute the vulnerable DES (Data Encryption Standard). The choice AES over other algorithms was based primarily on its efficiency and low memory requirements because it was designed to use only simple whole-byte operations.

AES algorithm

The AES algorithm consists of a set of steps repeated a number of times called rounds. The number of rounds depends on the size of the key and the size of the data block. The number of rounds is 9, if both the block and the key are 128 bits long. It is 11, if either the block or the key is 192 bits long, and neither of them is longer than that. It is 13, if either the block or the key is 256 bits long. Given a sequence $X_1, X_2 \ldots X_n$ bit plaintext blocks, each X_i is encrypted with the same secret key k producing the ciphertext blocks $Y_1, Y_2 \ldots, Y_n$ respectively, with $Y_i = E_k(X_i)$, as described Fig.2.

To encipher a block of data X_i in AES, see Fig.2, you first perform an AddRoundKey step (XORing the secret key with the block). The incoming data and the key are added together in the first AddRoundKey step. After, we entry in the round operation. Each regular round operation involves four steps. The first is the "SubBytes" step, where each byte of the block is replaced by its substitute in a S-box. The next one is the "ShiftRows" step where the rows are cyclically shifted over different offsets. The next step is the "MixColumns", where each column is multiplied over GF(2^8) by a matrix. The last step of the round operation is another "AddRoundKey". It is a simple XOR with the actual data and the subkey for the current round. Before producing the final ciphered data Y_i, the AES performs an extra final routine that is composed of (SubBytes, ShiftRows and AddRoundKey) steps.

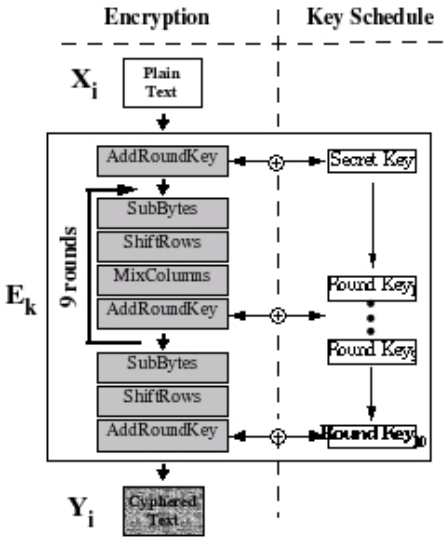

Figure 2. *AES general outline.*

The process over the plaintext data X_i is independent of the process over the secret key, and this last one is called Key Schedule. It is made up of two components: the Key Expansion and the Round Key Selection. The Expanded Key is a linear array of 4-byte words and is denoted by $W[N_b * (N_k + 1)]$, where N_b is the number of columns of the data block and N_k is the number of column of the cipher key. The first N_k words contain the cipher key and all other words are defined recursively. The key expansion function depends on the value of N_k. The cipher key is expanded into an Expanded Key. Round Keys are taken from this Expanded Key in the following way: the first Round Key consists of the first N_b words, the second Round Key consists of the following N_b words and so on [2].

AES modes

The AES algorithm can support several cypher modes ECB, CBC, OFB, CFB and CTR. The first one, the (Electronic CodeBook) mode is the basic AES algorithm. With the ECB mode, each plaintext block X_i is encrypted with the same secret key k producing the ciphertext block Y_i, with $Y_i = E_k(X_i)$. The CBC (Cipher Block Chaining) mode adds a feedback mechanism to

a block cipher. Each ciphertext block Y_i is XORed with the incoming plaintext block X_{i+1} before being encrypted with the key k. An initialization vector IV is used for the first iteration. In fact, all modes (except ECB) require this initialization vector IV. In CFB (Cipher FeedBack) mode, the $IV = Y_0$ see Fig.3. The keystream element Z_i is generated by $Z_i = E_k(Y_{i-1}), i \geqslant 1$ and the ciphertext block is produced by $Y_i = X_i \oplus Z_i$. In OFB (Output FeedBack) mode as in CFB $Y_i = X_i \oplus Z_i$, but $IV = Z_0$ and $Z_i = E_k(Z_{i-1}), i \geqslant 1$. The input data is encrypted by XORing it with the outputted Z_i. The CTR (Counter) mode has very similar characteristics to OFB, but in addition it allows a random access property for decryption. It generates the next keystream block by encrypting successive values of a counter. The counter can be any simple function that produces a sequence which is guaranteed not to repeat for a long time. In this mode, the output of counter is input of the AES.

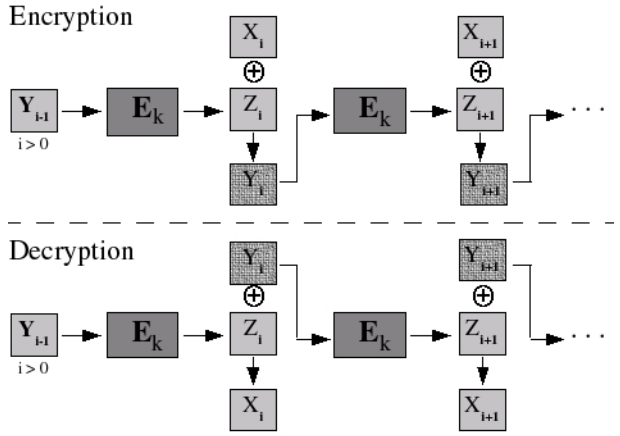

Figure 3. *CFB stream cipher encryption/decryption.*

Although AES is a block cipher, in the OFB, CFB and CTR modes it operates as stream cipher. These modes do not require any special measures to handle messages whose lengths are not multiples of the block size since they all work by XORing the plaintext with the output of the block cipher. Each explained mode has different advantages and disadvantages. In ECB and OFB modes for example any modification in the plaintext block X_i causes the corresponding ciphered block Y_i to be altered, but other ciphered block are not affected. On the other hand, if a plaintext block X_i is changed in CBC and CFB modes, then Y_i and all subsequent ciphered blocks will be affected. These properties mean that CBC and CFB modes are useful for authentication purposes and ECB and OFB modes treat separately each block. Therefore, we can notice that OFB does not spread noise, while the CFB do that.

In Fig.3, it is important to notice that the encryption function $E_k(X_i)$ is used for both encryption and decryption process in CFB mode.

Selective encryption of JPEG images

Despite the appearance of the JPEG2000, the JPEG is a commonly used standard method of image compression. It is still largely employed in picture processing, security communication and industry [7]. Nowadays, JPEG format has a huge quantity of images and hardware/firmware dedicatedly manufactured for it such as digital cameras, portable telephones, scanners and mobile machines. Those devices already exist and suggestions that optimize JPEG format concerning the confidently are welcome.

Several selective encryption methods have been proposed by authors, specially encryption of DCT-based coded images.

- Tang [9] proposed a technique called zigzag permutation applicable to DCT-based videos and images. Although his method offers more confidentiality, it increases the overall bit rate.
- Fisch et al [3] suggested a technique that encrypts a selected number of AC coefficients. In their method the DC coefficients are not ciphered because they carry important visible information and they are highly predictable. In spite of the bit rate be constant and preserve the bitstream compliance, it is not scalable and the compression and the encryption process are separated, consequently it leads to an operating cost.
- Fisch et al [4] have proposed a method whereby the data is organized in a scalable bit-stream form. These bit-streams are constructed with the DC and some AC coefficients of each block and then arranged in layers according to their visual importance. The partial encryption process is made over these layers.
- Some works were also proposed for DCT-based videos [12, 1, 10].
- Recently A. Said has measured the strength of partial encryption showing attacks that exploit information from non-encrypted bits and availability of side information [8].

The proposed method

Let $E_k(X_i)$ be the notation of the encryption of a n bit block X_i using the secret key k with AES cipher in CFB mode described in Fig. 3. For practicability, we assume that $n = 128$ and X_i is a nonempty plaintext. Let $D_k(Y_i)$ be the decryption of a ciphered text Y_i using the secret key k (with $D_k() = E_k()$ for CFB mode). The use of CFB mode will optimize the decryption process. Only the previous block is necessary to decrypt the current one.

Encryption procedure

The proposed method works in the entropy encoding process during the creation of the Huffman's vector. Therefore, our method can be applied for all JPEG modes that use Huffman's table. The main idea of the proposed method is shown in Fig.4 and summarized below.

1. Take the non-zero AC coefficients in the Huffman's bitstream from the highest to the lowest frequencies to build the plaintext vector X_i.
2. Encode X_i with AES in CFB mode.
3. Substitute the original Huffman's bit-stream with the ciphered information.

It is important to mention that these operations above are made separately in each quantized DCTed block.

Before depicting the method, it is interesting provide some considerations.

- The reason to take the path from the highest to the lowest frequencies (the order contrary of the zigzag) is because the most important visual characteristics of the image are placed in the lower frequencies, while the details are located in the higher frequencies. The HVS (Human Visual System) is more sensitive the lower frequencies than the higher ones. Therefore, we can calibrate the visual appearance of the resulted image. This means, we have a progressive encryption and it increases as much as we go up in the DC direction (lower frequencies).

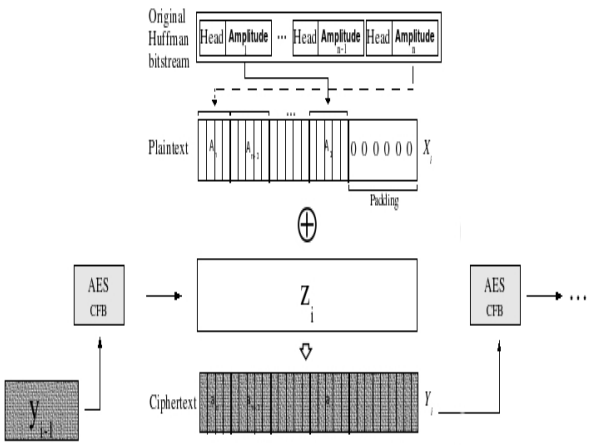

Figure 4. *Global overview of the proposed method.*

- The Huffman's vector is composed of a set of pair {HEAD, AMPLITUDE} or some control marks ZRL and EOB. These control marks are not compulsory, but they can appear in the following cases. If the last AC coefficients in the zigzag order are zeros, the Huffman's bit-stream for this block must contain the mark EOB. In turn, the ZRL control mark is found every time that sixteen successive zeros happen in the zigzag path and there is still at least one non-zero AC coefficient. In our method, we do not make any change in the HEAD and in the mentioned control marks. To guarantee a full compatibility with any JPEG decoder, the bit-stream should only be altered at places where it does not compromise the compliance to the original format.
- In cryptography, padding is the practice of adding values of varying length to the plaintext. It is because the cipher works on units of a fixed size, but messages can come in some lengths. Several padding schemes exist, but we will use the simplest one, that is to append null bits to the plaintext to bring its length up to the block size. Historically, padding was used to make cryptanalysis more difficult, but it is now used for more technical reasons with block ciphers, cryptographic hashes and public key cryptography.
- A constraint concerning the greatest quantity of bits used to build the plaintext X_i is taken in account. It graduates the level of ciphering and the visual quality of resulted image. If it is not stipulated, its value is the size of the cipher block $n = 128$.
- As much as homogeneous is a block in the original image as much as zeroed it is in quantized DCTed phase. The Discrete Cosine Transform separates the image into spectral sub-bands. Region of image that is almost monotone, most of the high-frequency DCT coefficients are near zero and after quantization the truncation became them zero [11].

In summary, the method works in three main steps. The construction of the plaintext X_i, the ciphering of X_i to create Y_i and the substitution of the original Huffman's vector with the ciphered information.

Construction of plaintext X_i

For constructing the plaintext X_i, we take the non-zero AC coefficients of the current block i by accessing the Huffman's vector from the end to the beginning to create the {HEAD, AMPLITUDE} pairs. From each HEAD is extracted the length of AMPLITUDE in bits. These values are computed and tested according to the Equation 1. As shown in global overview of the proposed method (Fig.4), only the AMPLITUDEs $(A_n, A_{n-1}...A_1)$ are taken to build the vector X_i. The final plaintext length L_{X_i} depends on both the homogeneity ρ of the block and the given constraint C:

$$f(\rho) \;<\; L_{X_i} \;\leq\; C, \tag{1}$$

where ρ is the homogeneity of the block, $f(\rho) = 0$ for $\rho \to \infty$ and $C \in \{128, 64, 32, 16, 8\}$ bits.

This constraint C specifies the maximum quantity of bits that must be taken in each block (VLC). In other hand the homogeneity depends on the content of image and it specifies the minimum quantity of bits. That means, a block with great ρ will produce small L_{X_i}. The Huffman's vector is accessed while $L_{X_i} \leq C$ and the DC coefficient is not reached. Then, we apply the padding function $p(j) = 0$, where $j \in \{L_{X_i}, \ldots, 128\}$, to fill out with zeros the vector X_i.

Ciphering of X_i with AES in CFB mode

In the ciphering step the former ciphered block Y_{i-1} is used as input of the AES cipher to create Z_i. Then, the plaintext X_i is XORed with Z_i to generate Y_i. For the initialization, the vector IV is created from the secret key k according to the following strategy. The secret key k is used as seed of PRNG (Pseudo-Random Number Generator). The k is divided in 16 portions of 8 bits each. The PRNG produces 16 random numbers that define the order of formation of the IV vector. After generating the vector $IV = Y_0$, it will be used in AES to produce Z_1, see Fig. 3.

Substitution of the original Huffman's bit-stream

The final step is the substitution of the ciphered information in the Huffman's vector. As in the first step (construction of the plaintext X_i), the Huffman's vector is accessed from the end to the beginning, but the ciphered vector Y_i is accessed from the beginning to the end. Known the length in bits of each AMPLITUDE $(A_n, A_{n-1}...A_1)$, we start cutting these portions in Y_i to substitute the AMPLITUDE in the Huffman's vector. The total quantity of replaced bits is L_{X_i}.

Decryption procedure

The decryption process in CFB mode is very simple and works as follows. The previous block Y_{i-1} is used as argument to AES cypher to generate the Z_i. Then Z_i is XORed with the current block Y_i to generate the X_i. Therefore, the same procedures for encryption depicted in the previous section are used. The difference is that, the input of the cypher process is the ciphered Huffman's vector. This ciphered vector is also accessed from the end to the beginning to construct the plaintext Y_{i-1}. Then, it will be used in AES to generate Z_i and X_i, see procedure Fig. 3. The resulted plaintext vector is cut in portion to substitute the AMPLITUDE in the ciphered Huffman and to generate the original Huffman's vector.

Experimental results

For all of our experiments, we have used the algorithm JPEG in the baseline sequential mode with a Quality Factor (QF) of 100%. We have applied five constraints for C $(128, 64, 32, 16$ and $8)$. For the cipher, we have used the AES in CFB (Cipher Feedback Block) mode with block and key of 128 bits long. However they can be used with the other possible combination of key and block sizes. We have applied our method on a digital painting of the Louvre Museum of Paris, France. For this painting color image, we have applied the same process depicted in the previous section only in the plan Y. We did not use the Cr and Cb plans.

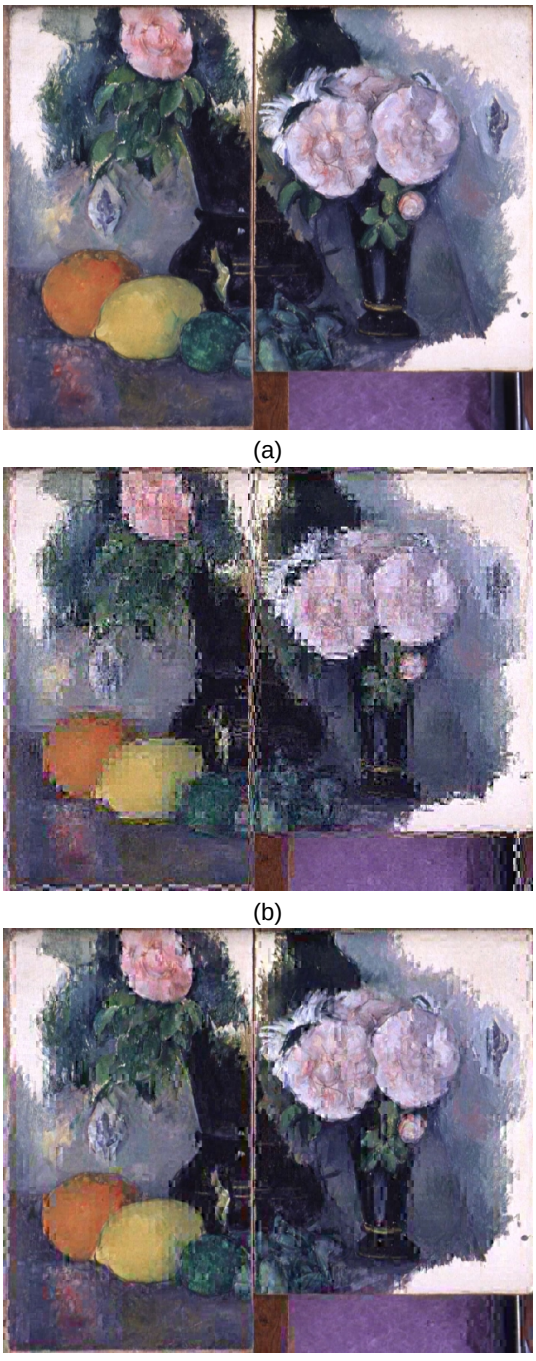

Figure 5. *a) Work-of-art original image* 512×640*, b) Ciphered image for* $C = 128$*, c) Ciphered image for* $C = 8$*.*

In Tab. *Result table for color work-of-art*, we show the results of our method employed in a color painting 512×640 pixels illustrated Fig. 5.a. The original compressed and the ciphered files have the same size, *i.e.* 258 Kb. For the constraint $C = 128$, we have ciphered 305033 AC coefficients and 733300 bits. The percentage of bits ciphered in whole image was 34.70% and that give us 326686 pixels altered, Fig. 5.b, this means 99.70% of pixels ciphered. The PSNR (Peak Signal Noise Ratio) was 19.31 *dB*. For $C = 8$ the quantity of AC coefficients and bits encoded were 20805 and 60956 respectively. Only 2.88% of whole image bits was ciphered. For $C = 8$, the selective encrypted image is illustrated in Fig. 5.c and the PSNR attained was 25.87 *dB*.

Result table for color work-of-art 512×640 **pixels.**

	Total ciphered				
C	Coeffi cients	Bits	% Bits	PSNR (dB)	Altered Pixels %
128	305033	733300	34.70	19.31	99.70
64	172902	449369	21.26	19.67	99.68
32	82399	238212	11.27	20.40	99.63
16	39735	121292	5.74	21.65	99.34
8	20805	60956	2.88	25.87	98.25

The greatest advantage of our method is the possibility of deciphering only one 8×8 block or a group of them. This comes from the fact that we have used the AES in CFB mode, and have applied it over each block. Fig. 6 shows the decryption of regions of interest (ROI). In this example the regions were 100% decrypted, but each region can be decrypted in a scalable way too, with $C = 16$ or $C = 32$ for example. It is important to note that the ROI must be defined in unit of block 8×8, default of JPEG format. In Fig. 6 two ROIs were decrypted, in the left bottom corner of the image we have decrypted a windows of 296×224 pixels. The other one is in the right top corner, 240×240 of image.

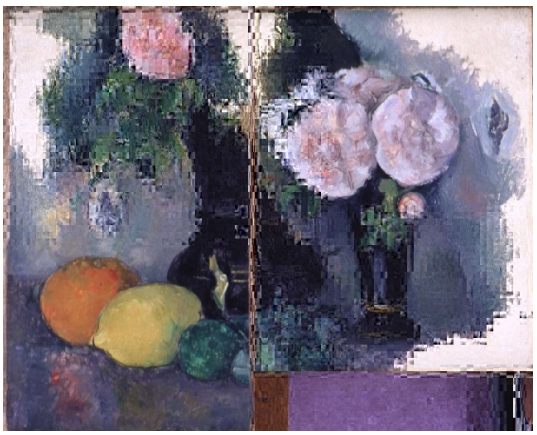

Figure 6. *Decoding of two regions.*

As can be seen in the resulted images, the selective cipher over the JPEG format yield the effect of blocking artifacts. The blocking artifacts are the discontinuity at block boundaries, which is often annoying to human eyes. Since the transform and quantization of pixel blocks are performed independently, the continuity in pixel values through neighboring blocks are broken in coded images.

Conclusion

In this paper, we have proposed new scheme for selective encryption for JPEG images based on AES cipher in CFB mode. We can enumerate some advantages of our approach such as portability, constant bit rate, JPEG format compliance, scalable selective encryption and a gradual decryption of region of interest. The main advantage of this method is this progressive decoding of a particular areas of the image. Those characteristics are useful for a large range of applications such as educational field, work-of-art and marketing. The experiments have shown that our scheme provides satisfactory PSNR, sufficient security and an acceptable and selective confidentiality results.

Acknowledgments

This investigation was in part supported by the TSAR project which is a french national project ANR-05-SSIA-0017-05 of the ANR ARA SSIA (*Agence Nationale de la Recherche, Action de Recherche Amont, Sécurité Systèmes Embarqués et Intelligence Ambiante*).

We would like also to thank Mr Lahanier Christian of the C2RMF (*Centre de Recherche et de Restauration des Musées de France*) and the Louvre Museum for the digital paintings and for valuable discussions.

References

[1] H. Cheng and X. Li. Partial Encryption of Compressed Images and Videos. *IEEE Transactions on Signal Processing*, 48(8):2439–2451, 2000.

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

[3] M. Van Droogenbroeck and R. Benedett. Techniques for a Selective Encryption of Uncompressed and Compressed Images. In *Proceedings of Advanced Concepts for Intelligent Vision Systems (ACIVS) 2002, Ghent, Belgium*, Sept. 2002.

[4] M. M. Fisch, H. Stgner, and A. Uhl. Layered Encryption Techniques for DCT-Coded Visual Data. In *European Signal Processing Conference (EUSIPCO) 2004, Vienna, Austria*, Sep., 2004.

[5] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (2nd Edition)*. Pearson Education (2002), Elsevier, 2002.

[6] X. Liu and A. Eskicioglu. Selective Encryption of Multimedia Content in Distribution Networks:Challenges and New Directions. In *IASTED Communications, Internet & Information Technology (CIIT), USA*, November, 2003.

[7] W.B. Pennebaker and J.L. Mitchell. JPEG: Still Image Data Compression Standard. *Van Nostrand Reinhold, San Jose, USA*, 45, 1993.

[8] A. Said. Measuring the Strength of Partial Encryption Scheme. In *ICIP 2005, IEEE International Conference in Image Processing, Genova, Italy*, volume 2, pages 1126–1129, 2005.

[9] L. Tang. Methods for Encrypting and Decrypting MPEG Video Data Efficiently. In *ACM Multimedia*, pages 219–229, 1996.

[10] J. Wen, M. Severa, W. Zeng, M. Luttrell, and W. Jin. A Format-Compliant Configurable Encryption Framework for Access Control of Video. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(6):545–557, 2002.

[11] J. Yhang, H. Choi, and T. Kim. Noise Estimation for Blocking Artifacts Reduction in DCT Coded Images. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(7), 2000.

[12] W. Zeng and S. Lei. Efficient Frequency Domain Video Scrambling for Content Access Control. In *ACM Multimedia, Orlando, FL, USA*, pages 285–293, Nov. 1999.

Author Biography

José M. Rodrigues *was born in Brazil. He received his M.S degree in Computer Science in 2002 from the Federal University of Ceara, Brazil. Nowadays, he is a Ph.D. student at University of Montpellier II, France.*

W. Puech *was born in December 1967, in France. He received the diploma of Electrical Engineering from the University of Montpellier, France, in 1991 and the Ph.D. Degree in Signal-Image-Speech from the Polytechnic National Institute of Grenoble, France in 1997. He initialised its research activities in image processing and computer vision. He served as a Visiting Research Associate at the University of Thessaloniki, Greece. From 1997 to 2000, he had been an Assistant Professor at the University of Toulon, France, with research interests include methods of active contours applied to medical images sequences. Since 2000, he is Associate Professor at the University of Montpellier, France. He works now in the LIRMM Laboratory (Laboratory of Computer Science, Robotic and Microelectronic of Montpellier). His current interests are in the areas of security of digital image transfer (watermarking, data hiding, compression and cryptography) and edges detection applied to medical images and road security.*

Adrian G. *Bors received the M.S. degree in Electronics Engineering from the Polytechnic University of Bucharest, Romania, in 1992 and the Ph.D. degree in Informatics from the University of Thessaloniki, Greece in 1999. From 1992 to 1993 he was a research scientist at the Signal Processing Laboratory, Tampere University of Technology, Finland. From 1993 to 1999 he was a research associate at the University of Thessaloniki, Greece. In March 1999 he joined the Department of Computer Science, University of York, U.K., where he is currently a lecturer. Dr. Bors has been an associate editor of IEEE Trans. on Neural Networks since 2001 and a member of technical committees of several international conferences. He has authored and coauthored more than 60 papers in journals, edited books, and international conference proceedings.*