July 2002

Compression performance of JPEG encryption scheme

C. Kailasanathan
University of Wollongong

R. Safavi-Naini
University of Wollongong, rei@uow.edu.au

P. Ogunbona
University of Wollongong, philipo@uow.edu.au

Compression performance of JPEG encryption scheme

Abstract

Recent development in the Internet and Web based technologies require faster communication of multimedia data in a secure form. A number of encryption schemes for MPEG have been proposed. In this paper, we evaluate the compression performance of JPEG which has been encrypted with the zig-zag permutation algorithm, suggest a security enhancement to the scheme, and propose an alternative to entropy coding recommended by JPEG to compensate for the compression drop occurring due to permutation.

Disciplines

Physical Sciences and Mathematics

Publication Details

This article was originally published as: C. Kailasanathan, R. Safavi-Naini & P. Ogunbona, 'Compression performance of JPEG encryption scheme', 14th International Conference on Digital Signal Processing, 1-3 July 2002, Vol 2, pp. 1329-1332. Copyright IEEE 2002.

Compression Performance of JPEG Encryption Scheme

C.Kailasanathan and R.Safavi Naini,
Centre for Computer Security Research,
Department of Computer Science, University of Wollongong,
Northfield Avenue,NSW 2522, Australia.
e-mail: ck12@uow.edu.au and rei@uow.edu.au
P.Ogunbona
Motorola Australian Research Centre,
Level 3,12 Lord Street, Botany, NSW 2019,
Australia.
e-mail: pogunbon@arc.corp.mot.com

February 13, 2002

Abstract

Recent development in the Internet and Web based technologies require faster communication of multimedia data in a secure form. A number of encryption schemes for MPEG have been proposed. In this paper, we evaluate the compression performance of JPEG which has been encrypted with Zig-Zag Permutation Algorithm, suggest a security enhancement to the scheme, and propose an alternative to entropy coding recommended by JPEG to compensate for the compression drop occurred due to permutation.

1 Introduction

Adding encryption to compression algorithm is an attractive proposition which could result in combined security and compression. The main challenge is to design secure systems without degrading compression performance. A number of algorithms have been proposed to provide security to MPEG [6, 7, 8, 9]. Out of all these algorithms, the one which has been analysed for security and compression performance is the Zig-Zag Algorithm proposed by Tang [8]. However, an attack to this scheme has also been proposed in [9].

Without degrading the picture quality, improving the compression performance is possible, only if the lossless compression such as entropy coding is replaced with a different one. A possible alternative is to use a dynamic Huffman coding scheme instead of a static one recommended by JPEG.

Huffman coding is an optimal compression algorithm that in its simplest form uses probabilities of characters in the input sequence as a *model* for input source, and produces codewords whose lengths are inversely proportional to their probabilities. In *Dynamic Huffman Coding* (DHC) the model is updated and the codeword representing a symbol changes with its position in the input stream.

In this paper, after reviewing the only JPEG encryption scheme known to date, we examine the compression performance of JPEG which has been applied a Zig-Zag Permutation Algorithm, propose a security enhancement to the current scheme, and suggest an alternative to static Huffman recommended by the JPEG in order to improve on compression performance.

In the next Section we review DHC, and Section 3 reviews the only known JPEG encryption scheme. Section 4 describes the Zig-Zag Permutation Algorithm. In Section 5, we examine the compression performance of JPEG which has been applied a Zig-Zag Permutation Algorithm. Section 6 concludes the paper.

2 Dynamic Huffman Coding

Dynamic Huffman coding was independently proposed by Faller [1] and Gallager [2] and considerably improved by Knuth [4] and Vitter [3]. The algorithm uses a *code tree* which is a weighted binary tree capturing the statistics of the input stream and forming the coder's model. Each leaf of the tree corresponds to one alphabet symbol and the weight of the leaf represents the current frequency of that symbol in the input stream representing the weights of each alphabets. Two nodes with a common parent in a code tree are called *siblings*. The weight of an internal node is the sum of the weights of its siblings. Nodes are numbered in increasing order starting from the bottom left node to bottom right node, followed by the nodes in the layers above again from left to right, until the root node

0-7803-7503-3/02/$17.00 ©2002 IEEE

is reached. *A binary tree is said to have sibling property* if listing the nodes using the above ordering results in a non-decreasing sequence. Gallager [2] proved that a binary prefix code is a Huffman code if, and only if, the code tree has the sibling property. In a dynamic Huffman tree the structure will be maintained in such a way that sibling property is preserved.

To encode an input symbol, first the the current Huffman tree is used to generate the codeword and then the weight of the leaf corresponding to the incoming symbol is updated. This update will flow through the whole tree. If because of updating of the weights, the sibling property is violated a new round of tree updates that involves the exchange of the nodes and their corresponding subtrees, will be applied to reinstate the property. Details of the update can be found in [3].

3 JPEG Encryption Scheme

Although many proposals exist for encrypting MPEG, only a limited amount of research has been done in encrypting JPEG. This may be because JPEG has been considered as a subset of MPEG. Most of the schemes described for MPEG can also be applied to JPEG as well. The only known encryption scheme proposed for JPEG has been given in [5]. Their scheme has considered encrypting DC and low frequency AC coefficients of DCT of each block and the following shortcomings have been noted.

1. If only DC coefficients are encrypted, a good approximation to the original image can be obtained by assigning any value to the DC coefficients.

2. If only DC and low frequency AC coefficients are encrypted, a reasonable approximation to the image can be obtained. This is because the un-encrypted high frequency AC coefficients will partially reveal the edge information of the image.

The above shortcomings have led the authors to propose a spatial domain encryption scheme.

4 Zig-Zag Permutation Algorithm

The purpose of encrypting text information is to prevent an adversary without the secret key from obtaining the information. In this instance, the content of the information is either known completely or it is unknown. But there are two levels of security to digital images.

1. Scrambled image has poor image quality compared with original image, but the content of the original image is visible to the viewer - obscured image

2. Scrambled image is not comprehensible - incomprehensible image

There are some special properties unique to the JPEG and MPEG encoding. After DCT and the quantization procedure, the coefficients of every 8x8 block has the following properties.

1. All coefficients are in the range of [0, 255] which can be represented by 8-bit binary string.

2. The amplitude of the DC coefficient is much larger than that of every AC coefficient.

3. Big portion of AC coefficients are zeros.

The basic idea of Zig-Zag Algorithm is that, instead of mapping the 8x8 block to a 1x64 vector in "Zig-Zag" order, it uses a random permutation list to map the individual 8x8 block to a 1x64 vector.

Following experiments are conducted by Tang [8].

1. DC coefficient is mapped to the first element in the 1x64 vector and the rest of the elements are permuted. → Obscured image

2. DC coefficient of every bock is set to zero or a fixed value between 0 and 255 and rest of the elements are permuted. → Obscured image

3. DC coefficient is mapped to any other position other than the first position in the 1x64 vector, and the rest of the elements are randomly permuted → Incomprehensible image

4. AC63 coefficient is set to 0 → Degradation is negligible

5. Split the DC coefficient into two parts, first part remain in the same position, the second part is substituted for AC63 and randomly permute the list → Incomprehensible image

The basic Zig-Zag Permutation Algorithm is vulnerable to the ciphertext only attack. The attack is based on the fact that none-zero AC coefficients are gathered in the upper-left corner of the I block. Statistical analysis which count the number of non-zero DC and AC coefficients from all blocks in an I frame was conducted by Qiao and Nahrstedt [9]. The results show

1. DC coefficients always have the highest frequency of non-zero occurrence.

2. The frequency of AC1 and AC2 are among the top 6.

3. The frequency of AC3 to AC5 are among top 10.

The other method is to use so-called binary coin flipping sequence together with two different permutation lists. For each 8x8 block, a coin is flipped. If it is a tail, the permutation list 1 (key1) is applied to the block; if it is a head, the permutation list 2 (key2) is applied to the block. Key1 and Key2 are the secret keys. (L.Tang) This method is subject to known plain text attack. The idea is to select the key that has the tendency to gather AC coefficients in the upper left corner.

5 Compression Performance

In this section, we analyse the compression performance of JPEG, if a random permutation has been applied to the zig-zag ordered DCT coefficients. The same experiment had been done for MPEG and the results can be found in [10]. Their findings reveal that zig-zag permutation for MPEG increases the MPEG stream size by as much as forty-six percent. As we have described before, this scheme is also vulnerable to chosen-plaintext attack if the permutations are not randomly chosen. To enhance security we propose applying permutations randomly selected from a set of known permutations after permuting the blocks.

5.1 Experimental Results

We performed experiments using four different permutations on Una, House, Tree, and Lenna images. We looked at the compression rates of the original images (Table 1) and the compression rates after applying each of those permutations. These observations revealed that a maximum of seven percent drop in compression rate on Una image, and a maximum of three percent drop in compression rate on all the other images. We also experimented the compression drop when the permutations were randomly selected from a list of four different permutations for each block (Table 2). What we observed was a maximum of three percent compression drop.

An adaptive Huffman coding algorithm was incorporated into JPEG instead of a static one, which had been recommended by JPEG, to see how much compression drop may occur. In fact compression rate increased by a maximum of four percent on Una image and by a maximum of three to two percent on all the other images (Table 3).

These experiments showed that adaptive Huffman coding achieved a better compression rate than the static one. Replacing static Huffman with a dynamic one in JPEG might compensate for the compression drop that occurred when random permutations had been applied to the zig-zag ordered DCT coefficients.

6 Conclusion

In this paper, we experimented the compression performance of JPEG when a random permutation had been applied to the zig-zag ordered DCT coefficients. We found that JPEG sequence increased by at most twenty percent compared to MPEG's forty-six percent. There was a fifty percent performance improvement for JPEG. We also noted a performance improvement in compression when static Huffman coding was replaced with an adaptive one.

Image	Compression(%)	Quality	PSNR
Una.pgm	65%	3	39.4978
Una.pgm	89%	25	27.933
Una.pgm	85%	14	30.7209
House.pgm	88%	3	25.9796
House.pgm	94%	25	20.0447
House.pgm	93%	14	22.3766
Tree.pgm	81%	3	39.2094
Tree.pgm	92%	25	30.2431
Tree.pgm	91%	14	32.3513
Lenna.pgm	88%	3	38.5824
Lenna.pgm	94%	25	31.5417
Lenna.pgm	93%	14	33.6306

Table 1: Compression percentage for JPEG

Image	Compression(%)	Quality
Una.pgm	62%	3
Una.pgm	89%	25
Una.pgm	84%	14
House.pgm	87%	3
House.pgm	93%	25
House.pgm	92%	14
Tree.pgm	79%	3
Tree.pgm	92%	25
Tree.pgm	90%	14
Lenna.pgm	87%	3
Lenna.pgm	93%	25
Lenna.pgm	92%	14

Table 2: Compression percentage for JPEG with four different permutations

References

[1] Faller.N. An adaptive system for data compression. In Record of the 7th Asilomar Conference on Circuit, Systems, and Computers. 1973,pp. 593-597.

Image	Compression(%)	Quality	PSNR
Una.pgm	69%	3	39.38
Una.pgm	91%	25	27.9374
Una.pgm	87%	14	30.7239
House.pgm	90%	3	25.8399
House.pgm	96%	25	20.3442
House.pgm	95%	14	22.7807
Tree.pgm	83%	3	39.1911
Tree.pgm	95%	25	30.2597
Tree.pgm	92%	14	32.3642
Lenna.pgm	90%	3	38.204
Lenna.pgm	96%	25	31.3555
Lenna.pgm	95%	14	33.4016

Table 3: Compression percentage for JPEG with adaptive Huffman coding

[2] Gallager.R.G. Variation on a theme by Huffman. IEEE Tans. Info. Theory IT-24,6 (Nov. 1978),668-674

[3] J.S.Vitter. Design and Analysis of Dynamic Huffman Codes. Journal of ACM,34(4):825-845, October 1987.

[4] Knuth.D.E. Dynamic Huffman coding. J. Algorithm 6(1985),pp 163-180

[5] Howard Cheng and Xiaobo Li, On the Application of Image Decomposition to Image Compression and Encryption, Communication and Multimedia Security 11, pages 116-127.

[6] T.B Maples and G.A.Spanos. Performance Study of a Selective Encryption Scheme for the Security of Networked Real-time Video. In Proceedings of 4th International Conference on Computer Communications and Networks, Las Vegas, Nevada, September 1995.

[7] L.Agi and L.Gong. An Emprical Study of MPEG Video Transmissions. In Proceedings of the Internet Society Symposium on Network and Distributed System Security, pages 137-144, San Diego, CA, Feb. 1996

[8] L.Tang Method for Encrypting and Decrypting MPEG Video Data Efficiently. In Proceedings of the Fourth ACM International Multimedia Conference (ACM Multimedia '96), pages 219-230, Bosten, MA, November 1996.

[9] Qiao and Nahrstedt Comparison of MPEG Encryption Algorithms. International Journal of Computers and Graphics, special issue: "Data Security in Image Communication and Network" vol.22 January 1998.

[10] Qiao and Nahrstedt, Is MPEG Encryption by Using Random List Instead of Zig-Zag Order Secure?,