Tatra Mt. Math. Publ. **45** (2010), 65–74
DOI: 10.2478/v10127-010-0006-9

TATRA MOUNTAINS
Mathematical Publications

IMAGE-BASED JPEG STEGANOGRAPHY

Matúš Jókay — Tomáš Moravčík

ABSTRACT. This paper deals with the steganographic algorithm LSB (modification of the Least Significant Bits) in JPEG images. The focus is on minimizing of the number of modified DCT coefficients using $(2^k - 1,\ 2^k - k - 1)$ Hamming codes. Experimental part of the paper examines the dependencies between the coding, efficiency and saturation.

1. Introduction

Steganography systems transfer information over the public communication channels in such a way that an attacker is unable to identify the transmission of secret information on the background of a public communication [1]. In general, efficiency of the secret information detection in any steganography system is related to the size of the hidden secret data.

The steganographic systems use media with fixed size for transfer. It is thus possible to make the upper estimate of the maximum size of secret information that can be inserted into the transmission medium. Determination of the maximum capacity is a non-trivial problem even for the simplest methods of the image-based steganography. Chandramouli [2] presents theoretical analysis to determine the capacity for the secret information embedded into images using LSB algorithm in spatial domain. Recently, Jessica Fridrich [3] has derived a more accurate estimate using two statistics stegoanalysis. It appears that the selection of the particular steganographic medium is important, because it significantly affects the steganographic system design and security.

Due to the data redundancy of a lossless image format, such as BMP, is the best one to implement image-based steganography. However its obvious redundancy is a disadvantage as well, because it is one of the most suspicious transport

2010 Mathematics Subject Classification: 68P30, 94B05, 94A08.
Keywords: steganography, LSB algorithm, Hamming codes.
This material is based upon work supported under the grant NIL-I-004 from Iceland, Liechtenstein and Norway through the EEA Financial Mechanism and the Norwegian Financial Mechanism and also under the grant VEGA 1/0244/09.

channels. The LSB insertion method can be used also with data-loss compression. The most popular public steganographic systems, like *JSteg*, *Outguess* and *JP Hide & JP Seek*, use the JPEG compression. The secret information is hidden in LSBs of DCT coefficients. Their main disadvantage is that they need to change as many DCT coefficients as the number of secret information bits that do not match the LSBs of the reference DCT coefficients.

In this paper we describe the implementation of the JPEG-based steganographic system, which is able to insert more information than existing systems modifying the minimum number of DCT coefficients. The implementation is based on the coding from [4] and ZIP compression. Selection of modified DCT coefficients is provided by a random number generator.

The structure of the article is as follows: Section 2 deals with the basic principles of the JPG image files structure and the process of the secret information embedding. Part 3 describes the functions of the proposed steganographic system. In Section 4 we present measurements and their evaluation we conclude in the final (5) part of this document.

2. JPEG image file and secret information embedding process

2.1. JPEG image file

JPEG files use data–loss compression. A redundant graphical information is discarded by this method without a significant impact on the picture. It is possible to achieve much better compression ratio that way than with the lossless compression.

JPEG is a standard [6] that prescribes a sequence of operations that are performed with visual data. These operations are:

(1) the color subsampling,

(2) the discrete cosine transform,

(3) the quantization of DCT coefficients,

(4) the final variable length code word (VLC) encoding.

JPEG defines four modes that can be supported by encoders and decoders: sequential DCT-based, progressive DCT-based, hierarchical, and lossless mode.

We focus on the most commonly used sequential mode (Figure 1). It uses 8-bit color samples, the calculation of discrete cosine transform, quantization, and the result of this process is encoded with a VLC. Any encoder and decoder have to support this mode of processing image data. Other modes are only optional.

The number of applications and libraries working with the JPG format uses this fact and does not support them.

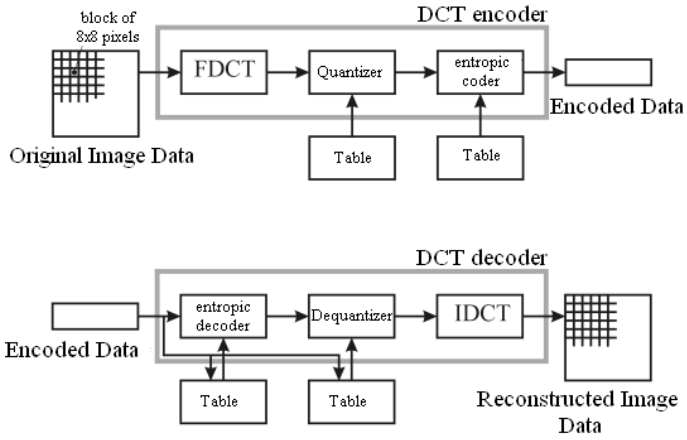

FIGURE 1. Block diagram of JPEG encoder and decoder.

2.2. Description of the JPEG encoding

Image data are represented by pixels. Each pixel contains three color components, namely red, green and blue (RGB color model). These pixels are first transformed into the YCbCr model, where Y is the brightness and CbCr are color channels. After conversion from RGB to YCbCr model, values of the pixels are divided into blocks of dimension 8x8 and transformed using discrete cosine transform (DCT). Then the DCT coefficients are quantized. This step represents a major lossy process: all the DCT coefficients are divided by the values from the quantization table and rounded up. Quantized DCT coefficients are then prepared for entropy coding.

When the discrete cosine transform and quantization process is done, each field size of 8x8 DCT coefficients is reorganized into the "zig–zag" sequence. The coefficients corresponding to the lowest frequencies are moved from the top left of the DCT matrix to the beginning of the sequence. Other coefficients (which are mostly zeroes) form the end of the sequence. RLE (Run Length Encoding) is applied on the sequence, so the runs of zeroes are efficiently encoded. Finally, the VLC is applied to get the compressed JPG image file. More information about JPG compression can be found in [5].

2.3. Embedding secret information into DCT coefficients

From the steganographic point of view JPG compression algorithm can be divided into two main parts: the first part P_1, which calculates quantized DCT

coefficients with the given compression ratio

$$P_1 : (\textit{carrier image}, \textit{compression ratio}) \longrightarrow \textit{DCT coefficients}$$

P_1 is a lossy compression process. Thus P_1 is not a bijection. If we apply D_1, which is a decompression algorithm for P_1, we usually do not reconstruct the original carrier image exactly.

$$D_1 : (\textit{DCT coefficients}, \textit{compression ratio}) \longrightarrow \textit{original image}$$

P_2 is the second part of the calculation. It converts the quantized coefficients into the binary string b_i

$$P_2 : (\textit{DCT coefficients}) \longrightarrow b_i, b_i \in Z_2^*.$$

P_2 is a lossless compression process. Thus P_2 is a bijection (VLC is reversible).

Since P_1 is not a bijection, we cannot hide secret information in the first part of the compression algorithm. This means that secret information may only be hidden in the second part of the compression algorithm—in quantized DCT coefficients.

3. Steganographic algorithm

The proposed steganographic system (Fig. 2) consists of several modules.

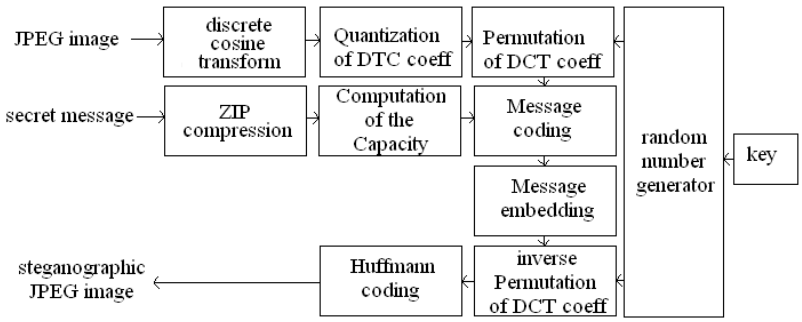

FIGURE 2. Schematics of the steganographic system.

The main goal of this research is to increase the number of input bits and minimize the number of modified DCT coefficients. We have set the following criteria:

(1) the system has to embed secret information into JPG image file format correctly; so that all graphic applications are able to handle it,

(2) the system must embed secret information in such a way that the insertion is not detectable by the human perception system,

(3) the system security will be based on a secret key, which initializes the pseudo-random generator for the DCT coefficients selection,

(4) appropriate Hamming code is used, if the capacity of the medium allows it,

(5) the system can include a compression algorithm that compresses the secret information before the insertion; this compression can reduce the size of secret information to be inserted.

To determine the capacity of the JPG image file we use the following formula

$$Capacity = \sharp DCT - \sharp DCT_1 - \sharp DCT_0,$$

where $\sharp DCT$ is the number of DCT coefficients, $\sharp DCT_1$ is the number of DCT coefficients with a value of 1 and $\sharp DCT_0$ is number of zero DCT coefficients.

In the process of a message insertion we use a key–dependent permutation (depends on a user password) that mixes all the DCT coefficients. Secret bits are inserted in the order given by the permutation. Decoder is able to repeat the same permutation only when the correct key is used. It is impossible to determine which DCT coefficient hides a secret bit of information without the knowledge of the key. This process spreads the secret bits of information through an entire JPG image file in order to avoid detection using χ–square test.

To reduce the number of modified DCT coefficients we use $(2^k - 1, 2^k - k - 1)$ Hamming codes, for $k \geq 1$ according to [4]. Using these codes the system inserts k-bits of a secret information (denot. $\bar{\mathbf{s}}$) by $2^k - 1$ LSBs of DCT coefficients (denot. $\bar{\mathbf{x}}$). At most one of them is changed. Selection of the modified DCT coefficient (denot. c) is performed by the following relations

$$h(x) = \mathbf{H}_c * \bar{\mathbf{x}}^{\mathsf{T}},$$
$$i = \bar{\mathbf{s}} \otimes h(\bar{\mathbf{x}}),$$

where $\mathbf{H}_c$ is the control matrix of a Hamming code $(2^k - 1, 2^k - k - 1)$ and $\otimes$ is the binary bit operation *xor*.

Coding in the above process does not change any bit if

$$(\mathbf{H}_c * \bar{\mathbf{x}}^{\mathsf{T}}) = \bar{\mathbf{s}}^{\mathsf{T}}.$$

When

$$\mathbf{H}_c * \bar{\mathbf{x}}^{\mathsf{T}} \neq \bar{\mathbf{s}}^{\mathsf{T}},$$

the value of the LSB of the DCT coefficient i in this group is changed. This increases the efficiency of inserting the secret information, because it does not modify $2^k - 2$ or $2^k - 1$ DCT coefficients for each k-bit block of the secret information.

The parameter k determines which of two embedding algorithms is selected. If $k = 1$, then the system inserts the secret information using the following sequential algorithm. Each bit of the secret information is inserted into a single DCT coefficient. In this case, we need as many DCT coefficients, as many bits we need to embed the secret information. In case of the pseudo-random secret information (i.e., when the message is encrypted), approximately half of the used DCT coefficients are modified.

If $k > 1$, then the system inserts a secret information using a Hamming $(2^k - 1, 2^k - k - 1)$ code. The k-bit block of the secret information is encoded by $2^k - 1$ LSBs of the DCT coefficients. Using the Hamming codes ensures that at most one of the LSB of the DCT coefficients is modified in each block.

Algorithm 1 (The embedding process)**.** This process consists of the following steps:

(1) Perform JPEG compression with optional compression ratio and get the value of quantized DCT coefficients.

(2) Compress the secret message using ZIP algorithm.

(3) Calculate the estimated capacity of the JPEG image.

(4) Initialize the pseudo-random number generator with a secret password and form a random sequence by which will be selected DCT coefficients.

(5) If $k > 1$, then:
   (a) divide the secret information into groups of k-bits length; they will be placed in groups of size $2^k - 1$ LSB bits of the DCT coefficients,
   (b) compute the information $h(\bar{\mathbf{x}})$ from the code word $\bar{\mathbf{x}}$,
   (c) if $h(\bar{\mathbf{x}})$ is not equal to k-bits block of secret information, modify the value of the appropriate LSB bit of the DCT coefficient c in this group.

(6) If $k = 1$, then insert the bits of secret message without coding by following rule: if value of the DCT coefficient is zero or one, skip the actual coefficient and select the next one, which value is not zero or one.

(7) Apply the Huffman coding for all the DCT coefficients.

(8) Insert the size of secret message and the value of parameter k to the information field of the JPEG file.

Algorithm 2 (The process of extraction)**.** To extract a secret message the steganographic system has to perform the following steps:

(1) Decode the Huffman code.

(2) Get the values of all the DCT coefficients.

(3) Initialize the pseudo-random number generator with a secret password and form the random sequence as in process of the secret message embedding.

(4) Get the size of the secret information and the value of parameter k.

(5) If coding was used,
   (a) divide all of the DCT coefficients in groups of $2^k - 1$ bits,
   (b) get LSBs of each group of DCT coefficients,
   (c) calculate the value $h(\bar{\mathbf{x}})$, which represents k-bits of the secret message.

(6) If coding was not used, get the LSBs of the selected DCT coefficients. If the value of the selected DCT coefficient is zero or one, skip it and select the next one, which value is not zero or one.

(7) Decompress the secret information using the ZIP algorithm.

(8) The result is the secret message.

4. Measurements

Definition 1. Let saturation denote a fraction of capacity of the media (in percentage), which we need for embedding of the secret information. The saturation, therefore, is given by:

$$s = \frac{\sharp DCT_{used}}{\sharp DCT_{total}} * 100\,\%\,,$$

$\sharp DCT_{used}$ is the number of used DCT coefficients for the secret message insertion and $\sharp DCT_{total}$ is the total number of DCT coefficients.

Using Hamming code

$$(2^k - 1, k, 1)$$

we change $2^k - 1$ bits in average for every $k * 2^k$ bits inserted.

Definition 2. Let us define the efficiency of the code *eff* as the ratio of inserted to changed bits. For the Hamming code we get:

$$eff = k + \frac{k}{2^k - 1}.$$

The second term goes quickly to zero with growing k. Using the efficiency we can compute for each k the theoretical saturation:

$$s_t = \frac{k}{2^k - 1} * 100\,\%\,.$$

The saturation affects not only the selection of the embedding algorithm, but also the optimal encoding of a secret message (choice of the theoretically optimal encoding parameter k_t):

$$k_t = ceil\left(\frac{2^k - 1}{N}\right),$$

k is parameter of Hamming code $(2^k - 1, k, 1)$ in interval $< 2; 11 >$, N is the total number of secret message bits.

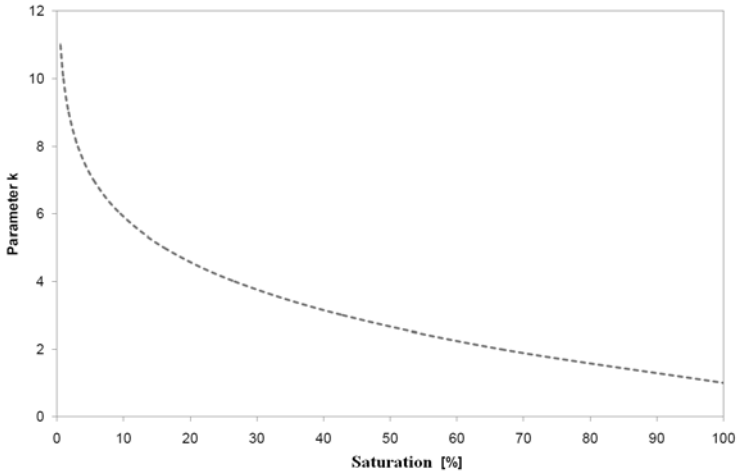

FIGURE 3. The dependency of the parameter k of the saturation.

TABLE 1. The theoretic saturation values.

parameter k	Hamming code	saturation s_t [%]
1	—	100
2	(3,2,1)	66.67
3	(7,3,1)	42.86
4	(15,4,1)	26.67
5	(31,5,1)	16.13
6	(63,6,1)	9.52
7	(127,7,1)	5.51
8	(255,8,1)	3.14
9	(511,9,1)	1.76
10	(1023,10,1)	0.98

Figure 3 depicts the choice of the optimal parameter k for a given saturation level. The numerical values are presented in Table 1. Figure 4 shows the comparison of the efficiency for different saturation levels between a classical LSB algorithm and our modification. The numerical values are summarized in Table 2.

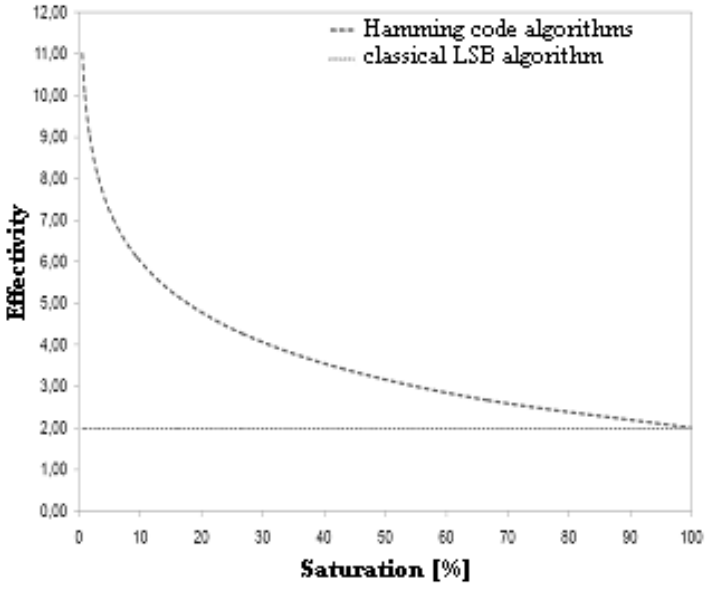

FIGURE 4. Efficiency of the embedding algorithms.

TABLE 2. Efficiency of the embedding algorithms.

parameter k	Efficiency *eff* [embedded bits per 1 modified DCT coefficient]
1	2.00
2	2.67
3	3.43
4	4.27
5	5.16
6	6.10
7	7.06
8	8.03
9	9.02
10	10.01

5. Conclusion

If the target saturation is 100 %, then it is possible to achieve the efficiency of 2 bits per each changed DCT coefficient. The embedding algorithm using Hamming codes with parameter k in interval $< 2; 5 >$ achieves the best results

for saturation levels between 16.13 % and 66.67 % (corresponding efficiency is 2.67–5.16).

If the embedding algorithm is used without the ZIP compression, the longest message can have approximately 20 % of the total capacity of the medium. If the embedded information is structured in some way (e.g., a text file), a ZIP compression allows us to use a relatively larger fraction of the capacity (when considering the original data, the ZIPped file still can only contain 20 % of the capacity of the medium).

Using the Hamming codes for embedding, and compression for preprocessing, it is possible to increase the number of inserted bits of the secret information, while reducing the number of modified DCT coefficients.

REFERENCES

[1] ARNOLD, M.—SCHMUCKER, M.—WOLTHUSEN, S.: *Techniques and Applications of Digital Watermarking and Content Protection*. Artech House, London, 2003.

[2] CHANDRAMOULI, R.—KHARAZZI, M.—MEMON, N.: *Image steganography and steganalysis: concepts and practice*, in: Internat. Workshop on Digital Watermarking––IWDW '03 (T. Kalker et al., eds.), Lecture Notes in Computer Science, Vol. 2939, Springer-Verlag, Berlin, 2004, pp. 35–49.

[3] FRIDRICH, J.—GOLJAN, M.—DU, R.: *Reliable detection of LSB steganography in grayscale*, in: Proc. of the 2001 Workshop on Multimedia and Security: New Challenges ACM New York, NY, 2001, pp. 27–30.

[4] ZHANG, W.—ZHANG, X.—WANG, S.: *Maximizing steganographic embedding efficiency by combining hamming codes and wet paper codes*, in: Information Hiding: 10th International Workshop—IH '08, Lecture Notes in Comput. Sci., Vol. 5284, Springer-Verlag, Berlin, 2008, pp. 60–71.

[5] MURRAY, J.: *Encyklopédie Grafických Formátů*, Computer Press, Brno, 1997. (In Czech)

[6] CCITT: *Digital compression and coding of continuous-tone still images, part 1: requirements and guidelines*, in: ISO/IEC IS 10918-1.

Received April 30, 2010

Department of Applied Informatics and Information Technology
Faculty of Electrical Engineering and Information Technology
Slovak University of Technology
Ilkovičova 3
SK–812-19 Bratislava
SLOVAKIA
E-mail: matus.jokay@stuba.sk
xmoravcik@stuba.sk