International Journal of
Applied & Experimental Mathematics
Research Article Open Access

A Secret Key Information Hiding in Digital Images Protocol based on Steganographic Methods in Image Frequency Domain

Panos M. Pardalos[1,2], Nikolaos Giovanopoulos[3], Panayotis E. Nastou[1,3*] and Yannis C. Stamatiou[4,5]

[1]*Center for Applied Optimization, University of Florida, Gainesville, USA*
[2]*Department of Industrial and Systems Engineering, University of Florida, Gainesville, USA*
[3]*Department of Mathematics, University of Aegean, Samos, Greece*
[4]*Department of Business Administration, University of Patras, Patra, Greece*
[5]*Computer Technology Institute and Press ("Diophantus"), Patras, 26504, Greece*

Abstract

In this paper, we present and evaluate experimentally a stego-key steganographic method operating in the frequency domain of digital images and an information hiding protocol based on this method. According to this protocol, the two communicating parties share a set of candidate cover images and a secret key, which is divided into two subkeys needed by the stego-key steganographic method. The first one is used for hiding the hidden message size while the second one is used to guide the message hiding process. After the selection of the cover image, the message embedding method can utilize one of three schemes for the collection of appropriate image blocks, based on the Discrete Cosine Transform coefficients, in which the message can be hidden with minimal visual image distortion. The recovery of the hidden message is possible even if the cover image is subjected to the JPEG (lossy) image encoding standard. The use of the keys in the embedding process makes the method especially robust as the attacker has two challenges to overcome in order to extract the hidden message, i.e. the extraction of the message size and, then, the extraction of the hidden message itself. We provide experimental evidence demonstrating the efficiency of the proposed cover block selection schemes of the embedding and disembedding methods both in terms of image distortion as well as robustness under image compression.

Publication History:

Received: October 31, 2016
Accepted: June 28, 2017
Published: June 30, 2017

Keywords:

Stego-key steganography, Information hiding, Image processing, Protocols, DCT-based compression

Introduction

In today's digital ecosystems, secret messages can be easily embedded into innocent-looking carriers [1] which can, in principle, be any imaginable data item, a WEB page, a sound file, a video file, an image file, a data packet, a video frame etc. The process of embedding messages into carriers has its roots in the ancient times. The first reported method of such an embedding was a method where a secret message was camouflaged into a hare corpse [1,2].

Nowadays, *steganography* is the scientific field that provides methods of hiding a message or any data item in general into any kind of carrier digital medium. In contrast, *cryptography* makes a message unintelligible but does not hide the existence of the message itself (RFC2828). The unit of data where a message is to be embedded is called a cover. The embedded secret message is called *steganogram* while the unit of data that conveys the covert message is called *stego.* Steganographic methods can be easily applied in order to provide secret communication among several parties. The main characteristics of a steganographic method are: i) it uses innocent-looking covers, which means that a cover object is almost impossible to be considered a suspicious object by third parties that eavesdrop the communcation and ii) its effectiveness is based on the camouflaging capability of the algorithm that mixes a steganogram with parts of the cover [1]. Another characteristic concerns the capability of the method to withstand attempts of detecting the embedded message. This means that a stego object must look like the cover object to the casual observer.

The "prisoners' problem" was the first model for invisible communication [8]. Conceptually, the problem refers to two agents, say Alexander and Peter, who are under the surveillance of a third party and they want to develop a secret plan. Alexander and Peter have access to computer systems and they can exchange unencrypted messages according to the third party's rules. If encrypted messages are discovered, the penalty is to forbid the further exchange of messages between Alexander and Peter. Consequently, Alexander and Peter have to communicate invisibly so that they escape the third party's attention. A steganographic protocol can be a solution to their problem. Alexander embeds a secret message into an innocent digital image and trasmits the stego image to Peter. Peter, upon its receiption, disentangles the message from the image and responses in a similar way. The camouflaging capablity of the used steganographic algorithm guarantees that the stego image will not raise suspicions from the third party.

A simple steganographic system or protocol does not require any prior exchange of secret information between the communicating parties [8]. However, such a system is not secure since anyone that knows the details of the steganographic method can extract the message. Greater security in a steganographic system can be achieved by the use of some piece of secret information, called the *stego-key*, that the two communicating parties should exchange prior to the establishment of their steganographic communication. In this case, the extraction of the secret message is impossible without knowledge of the shared secret information. Such a system or protocol is

***Corresponding Author:** Dr. Panayotis E. Nastou, Department of Mathematics, University of Aegean, Samos, Greece; E-mail: pnastou@aegean.gr

Citation: Pardalos PM, Giovanopoulos N, Nastou PE, Stamatiou YC (2017) A Secret Key Information Hiding in Digital Images Protocol based on Steganographic Methods in Image Frequency Domain. Int J Appl Exp Math 2: 117. doi: https://doi.org/10.15344/2456-8155/2017/117

called a *stego-key steganographic system/protocol*. Alternatively, the security of a steganographic system can be achieved by *public key steganography*. A public key steganographic system uses a pair of keys. The public key is used to embed the message into the cover object while the private key is used to extract the message from the stego object. Typically, breaking a steganographic system means that a transmitted object is caught to convey a hidden message which can be extracted. However, a steganographic system is also considered insecure if an eavesdropper can even only prove that a message exists in a transmitted object regardless if it can be extracted or not.

Steganographic systems can be classified either according to the used cover object or the type of modifications to the cover that can be applied during the message hiding process. Based on the second classification approach, i.e. the type of applied cover modifications, the steganographic systems can be classified into the following categories [8]:

- *Substitution systems*, where redundant parts of a cover object are substituted with a secret message.
- *Transform domain techniques*, where the cover can be any kind of signal and the steganogram is intermixed with cover entities in the frequency domain.
- *Spread Spectrum techniques*, where the steganogram is spread over a wide range of signal frequencies. The most popular information hiding variants in this class are the direct-sequence and frequency hopping procedures.
- *Statistical steganographic techniques*, where a secret message is com- bined with cover elements in such a way that some statistical characteristics of the cover signal are changed.
- *Distortion techniques*, where a sender hides a message by modifying the signal and the the receiver retrieves the message by discovering the modified parts of the signal. These techniques require from the receiver and the sender to share the cover signals that will be used for information hiding.
- *Cover generation systems*, where instead of using an existing cover object the system constructs a tailor-made one for carrying the secret information.

Information hiding methods that operate in the frequency domain of a signal are considered more robust than those that operate in the time domain. Thus, the majority of the proposed robust steganographic systems operate in the frequency domain [8]. In [6], a digital watermarking technique that spreads a watermark throughout the spectrum of an image is presented. Initially, an image is transformed into the frequency domain. During a second transformation stage, visibly significant regions in image spectrum, where the watermark can be stored without affecting the quality of the image, are detected. The watermark is stored in these regions by modifying their frequency coefficients while the stego image can be obtained by applying the inverse transform.

Most of the steganographic systems pay particular attention to their robustness under the JPEG compression attack, which is a standard image compression process often employed to destroy the hidden information. In [5] and [3], a reversible information hiding method is presented where a number L of message bits are stored in an 8×8 image block such that L sequences of more than two successive zero DCT-quantized coefficients of the medium-frequency components exist, by modifying coefficients of these sequences. In [4], the embedding process is based on the modification of certain entries of the used JPEG quantization table and their corresponding DCT-quantized coefficients. Specifically, some quantization table entries are divided by an integer k while the corresponding DCT-quantized coefficients are multiplied by k in order to create sufficient range for hiding message bits on these coefficients.

A method of embedding a message in the frequency domain that is based on the modulation of the distance between two DCT coefficients in 8×8 image blocks is presented in [8]. Each message bit is embedded in a block using the DCT transform G of this block. The method selects two DCT-coefficients of the middle frequencies that is to be divided by equal or similar quantization values of the quantization table. If $G(u_1, v_1)$ and $G(u_2, v_2)$ are these coefficients, then a block conveys a 1 if $G(u_1, v_1) > G(u_2, v_2)$, otherwise 0. These coefficients should be modified so as $|G(u_1, v_1) - G(u_2, v_2)| > Threshold$ for a certain positive constant *Threshold* so that the embedded bit can be retained after the JPEG quantization. A similar method where three DCT coefficients of each block are involved in the embedding and extraction methods is presented in [7].

In this paper, we present an information hiding protocol that is based on a stego-key steganographic method in the frequency domain of images. The two communicating agents share a set of cover images and a secret key. The protocol selects a cover image for hiding the message and, then, it invokes the embedding method. The secret key is divided into two subkeys. The method, at first, using the first key collects a set of "good" blocks in the sense that these blocks can convey message bits with minimal image distortion and that the bits can be retrieved even if the image is compressed using the JPEG standard. Using these blocks and the second key, the method hides the message itself.

We propose three schemes for selecting the blocks which are capable of hiding bits, where a block is considered as a "good" block only if the relation between the particular DCT-coefficients are resistent to a set of processing steps that are involved in the JPEG compression even if the DCT-coefficients change location after a swap operation. The next step of the embedding phase is to embed both the message size and the message itself in the selected blocks using the two secret keys. As the method does not modify the coefficients of the involved blocks by adding values to them, it modifies slightly the quality of the cover image only in cases where a swap between coefficients is mandatory for hiding a bit. In this way, it is not easy to perceive that an image conveys a message. Moreover, the method is robust under JPEG compression i.e. the message is retrieved correctly even after the image has undergone a JPEG compression. Finally, the fact that the embedding and extraction methods employ two secret keys for the cover and the stego block retrieval, makes almost impossible for the attacker to extract the message from the stego image. This is due to the fact that the attacker first has to discover the message size by finding the first key and then to try to extract the message by searching for the second key. As for the size of the key, it could be suffieciently large so as the protocol is resistent to brute force attacks.

Image Processing in the Frequency Domain

As in most digital image processing applications, steganographic applications in the frequency domain deal with discrete, digital signals. Specifically, when digital images are to be used as stego objects steganography applications deal with two dimensional discrete signals.

A digital image is obtained after sampling and quantizing an analog image [11]. It is organised as a two dimensional N × M grid of pixels (picture elements) and it can be color or gray-scale. In gray-scale images, a pixel value is a single byte that represents the luminance of the spot with coordinates $1 \le x \le N$ and $1 \le y \le M$. Thus, a digital image is considered either as a function, $f : X \times Y \to \{0,1\}^8$ where X and Y are the coordinates or indices in a 2-dimensional matrix for computational purposes.

Image Transforms

Among the most important image transforms are the Fourier, Walsh, Hadamard, discrete cosine and Haar transforms. All these transforms express an image in the frequency domain. The two dimensional discrete fourier transform (2D-DFT for short) is widely used in image filtering operations and image analysis. Its co-domain is the set C of complex numbers. The two dimensional discrete cosine transform (2D-DCT for short) is one of the image transforms that is widely used in image compression. The fact that its co-domain is the set R of reals, makes the 2D-DCT computationally elegant. A very popular application where the 2D-DCT plays a central role is the JPEG compression. We briefly present the 2D- DFT and 2D-DCT of an image since they are involved in steganographic methods in frequency domain.

If $f(x, y)$ is a $N \times M$ gray-scale image, then the two dimensional discrete fourier transform is defined by the following equation,

$$F(u,v) = \sum_{a=0}^{N-1}\sum_{b=0}^{M-1} f(a,b) e^{-i2\pi\left(\frac{au}{N}+\frac{bv}{M}\right)} \quad (1)$$

for all $0 \le u \le (N\text{-}1)$ and $0 \le v \le (M\text{-}1)$. The original image can be obtained by its *DFT* $F(u, v)$ by applying the inverse *DFT* which is given by the following equation,

$$f(a,b) = \frac{1}{NM}\sum_{u=0}^{N-1}\sum_{v=0}^{M-1} F(u,v)\, e^{i2\pi\left(\frac{au}{N}+\frac{bv}{M}\right)} \quad (2)$$

for all $0 \le a \le (N-1)$ and $0 \le b \le (M-1)$ ([9-11] for details). A $N \times M$ image f (a, b) can be handled, computationally, as a two dimensional matrix. The most common practice is to calculate the 2D-DFT of an N × M image f (a, b) (where N and M are powers of 2) by applying the one dimensional radix-2 FFT to the rows of $f(a, b)$ first and then to the columns of the resulting matrix.

The two dimensional discrete cosine transform $G(u, v)$ of a $N \times M$ image $f(a, b)$ (see in [10]) can be calculated by applying a three stage computational procedure. At the first stage, a $2N \times 2M$ image $g(a, b)$ is constructed by f (a, b) using the following formula

$$g(a,b) = \begin{cases} f(a,b) & a \le N-1, b \le M-1 \\ f(2N-1-a,b) & a > N-1, b \le M-1 \\ f(\alpha, 2M-1-b) & a \le N-1, b > M-1 \\ f(2N-1-a, 2M-1-b) & a > N-1, b > M-1 \end{cases} \quad (3)$$

At the second stage, the computation of the two dimensional discrete fourier $C(u, v)$ of $g(a, b)$ takes place, i.e. $C(u, v)$ =2D-DFT[$g(a, b)$]. Finally, at the third stage, the two dimensional cosine transform $G(u, v)$ of the image $f(a, b)$ is obtained by performing the following computations,

$$G(u,v) = W_{2N}^{\frac{u}{2}} W_{2M}^{\frac{v}{2}} C(u,v)$$

where $W_{2N} = e^{-j\frac{2\pi}{2N}}, 0 \le u \le (N-1), 0 \le v \le (M-1)$. On the other hand, the original image can be obtained by applying the inverse two dimensional DCT to $G(u, v)$ which is a computational process of three stages. The first stage consists of the following computations,

$$C(u,v) = \begin{cases} W_{2N}^{\frac{-u}{2}} W_{2M}^{\frac{-v}{2}} G(u,v) & 0 \le u \le N-1, \\ & 0 \le v \le M-1 \\ -W_{2N}^{\frac{-u}{2}} W_{2M}^{\frac{-v}{2}} G(2N-u,v) & N+1 \le u \le 2N-1, \\ & 0 \le v \le M-1 \\ -W_{2N}^{\frac{-u}{2}} W_{2M}^{\frac{-v}{2}} G(u,2M-v) & 0 \le u \le N-1, \\ & M+1 \le v \le 2M-1 \\ W_{2N}^{\frac{-u}{2}} W_{2M}^{\frac{-v}{2}} G(2N-u,2M-v) & N+1 \le u \le 2N-1, \\ & M+1 \le v \le 2M-1 \\ 0 & u = N \text{ or } v = M \end{cases}$$

At the second stage, the computation of the inverse two dimensional discrete fourier is applied to the previously calculated $C(u, v)$, i.e. $g(a, b)$ = 2D-IDFT[$C(u, v)$]. Finally, the original image is obtained by the following equation,

$$f(a,b) = \begin{cases} g(a,b) & 0 \le a \le N-1, 0 \le b \le M-1 \\ 0 & otherwise \end{cases}$$

JPEG: A DCT-based Image Coding

An entity that wishes to transmit a stego image may decide to compress it. A popular way to accomplish this, is by using the JPEG standard [12]. Due to the fact that the JPEG compression is a lossy compression method, it is highly probable that the secret message will be destroyed. Since our goal in this paper is to provide a JPEG-resistent steganography protocol, we need to understand how this protocol works. In order to avoid secret message distortion, the design of the algorithm that embedes a message should consider carefully the way that compression works.

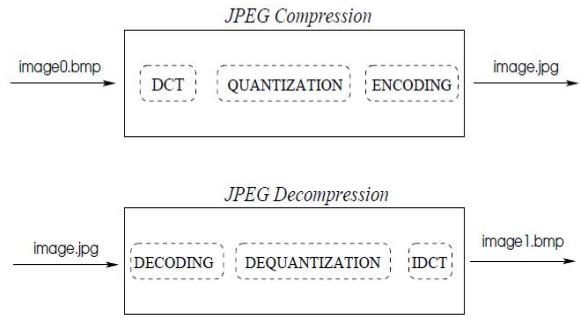


Figure 1: The three stages of the JPEG image compression.

Complying with the saying *a chain is no stronger than its weakest link*, it is mandatory to explain how each part of the JPEG compression works. The JPEG compression method consists, mainly, of three stages which are depicted in Figure 1. These steps aim to reduce the redudant information in the image file. Briefly, these stages are the 2D-DCT calculation of all 8 × 8 image blocks that an image is divided, the quantization of the DCT coefficients and the final entropy coding. More specifically, an image is divided into 8 × 8 image blocks. Each block is inserted in the JPEG-compression round and its 2D-DCT is computed. This stage provides a representation of the image block in the frequency domain. Actually, for each block a set of 64

cosine coefficients, $G(u, v)$ where $0 \le u \le 7$ and $0 \le v \le 7$, is derived corresponding to different components. At the second stage, each coefficient $G(u, v)$ is quantized by dividing it with a predefined value $QV(u, v)$ that corresponds to $G(u, v)$, rounding the result of the division (Equation 5). The goal of the quantization is to minimize the amount of information required for image spectrum description. The predefined values that are used in the quantization stage are stored in a table. The quantization table consists of 64 *quantization values* each one corresponding to a particular coefficient of the 8 × 8 image block. There are different quantization tables and one can choose the most appropiate for the application in hand. The table below is the one proposed in [12] for luminance components.

(u, v)	0	1	2	3	4	5	6	7
0	16	11	10	16	24	40	51	61
1	12	12	14	19	26	58	60	55
2	14	13	16	24	40	57	69	56
3	14	17	22	29	51	87	80	62
4	18	22	37	56	68	109	103	77
5	24	35	55	64	81	104	113	92
6	49	64	78	87	103	121	120	101
7	72	92	95	98	112	100	103	99

Table 1: Quantization Table in the JPEG compression for luminance components.

$$NG(u,v) = roundL\left(\frac{G(u,v)}{QV(u,v)}\right) \quad (5)$$

During the last stage of JPEG compression, the coefficients $NG(u, v)$ provided by the quantization process are forwarded to the encoding stage. Using Huffman coding, this stage is a lossless stage and it does not affect the quality of the image. Consequenlty, this stage does not contribute to the distortion of the steganogram.

The Proposed Two Secret Key Information Hiding in Digital Images Protocol

An $N \times M$ digital cover image can be divided into regions or blocks of size $n \times n$, where n is a common divisor of N and M. This actually means that a cover image consists of $PB = \frac{N \times M}{n^2}$ *cover regions or blocks.*

Let us consider the set of symbols $\Sigma = \{0, 1, \ldots, 9, A, B, C, D, \ldots, Z\}$ and let Σ^* be the set of all sequences of symbols from Σ. A sequence $X = x_1 x_2 \ldots x_k$ where $k > 0$ and $x_i \in \Sigma$, $1 \le i \le k$ is defined as a message of length k that is to be hidden in a cover image. Thus, the set Σ is the alphabet while Σ^* is the message domain.

Initially, the information hiding process encodes a message X into the set $\{0, 1\}^*$. In order to define such an encoding function, the correspondence $g: \Sigma \to \mathbf{Z}$, defined by $0 \to 0, \ldots, 9 \to 9, A \to 10, B \to 11, \ldots, Z \to 35$, is adopted. Since $|\Sigma| = 36$, we define an encoding function $f: \Sigma^* \to Z$, such that

$$x = f(X) = g(x_1)36^{k-1} + g(x_2)36^{k-2} \ldots + g(x_k)36^0$$

where $g(x_i)$ is the integer that corresponds to the symbol x_i. It is obvious that f is inversible, i.e. the corresponding decoding function f^{-1} exists, and consequently any integer x can be decoded into Σ^*. By this way, the problem of hiding a message $X \in \Sigma^*$ in an image is reduced to the problem of embedding the bits of the binary representation of the integer $x = f(X)$ into an image. In practice, the encoding and decoding functions f and f^{-1} can be implemented using the functions *mpz_set_str* and *mpz_get_str* of the *Gnu Multiple Precision Arithmetic Library*, gmpLib for short (see in [13] for details).

In [8], a method of embedding a message in the frequency domain that is based on the modulation of the distance between two specific DCT coefficients in 8 × 8 image blocks is presented. Specifically, an image is divided into 8 × 8 blocks of pixels. For each message bit X_i, an unused block CB_i (i.e. a block that was not used earlier for another message bit) is choosen and the DCT transform G_i of this block is calculated, i.e. $G_i = 2D\text{-}DCT(CB_i)$. Since DCT coefficients of middle frequencies represent significant parts of a signal and they may have similar magnitudes, it is possible that they are good candidates for hiding a message bit. In the standard quantization table, as it appears in Table 1, there are pairs of entries that correspond to DCT coefficients of middle frequencies whose coordinates are equal e.g. $(QV(4, 1), QV(3, 2))$, $(QV(3, 0), QV(1, 2))$ and $(QV(6, 1), QV(5, 3))$. Consequently, a good candidate pair of DCT coefficients is $(G_i(4, 1), G_i(3, 2))$.

1. **for** each message bit X_i do:
2. **select** an unused block CB_i
3. compute $G_i = 2D\text{-}DCT(CB_i)$
4. if $(X_i = 0)$
5. if $(G_i(4, 1) > G_i(3, 2))$ $G_i(4, 1) \rightleftarrows G_i(3, 2)$
6. else if $(G_i(4, 1) < G_i(3, 2))$ $G_i(4, 1) \rightarrow G_i(3, 2)$
7. modify $G_i(4, 1)$ and $G_i(3, 2)$ so as $|G_i(4, 1) - G_i(3, 2)| >$ Threshold
8. **compute** $CB_i^* = 2D\text{-}IDCT(G_i)$
9. **end for**

In general, if $G_i(4, 1) > G_i(3, 2)$ holds, the block conveys 1, otherwise 0. Thus, if a message bit X_i is 1, while $G_i(4, 1) < G_i(3, 2)$ in the selected block, then the coefficients should be swapped. Correspondingly, if a message bit X_i is 0, while $G_i(4, 1) > G_i(3, 2)$ in the selected block, then the coefficients should be swapped too. However, since the DCT coefficients of the block will be quantized, it is possible that the relation between $G_i(4, 1)$ and $G_i(3, 2)$ will change. For this reason, the authors in [8] adjust these coefficients by modifying them so as $|G_i(4, 1) - G_i(3, 2)| > Threshold$, for a certain positive constant *Threshold*. The pseudocode above describes the presented embedding procedure.

Let l be the number of bits of the binary representation of x. Our information hiding method also hides the bits of the binary representation of l in the same cover image. Consequently, if d is the number of bits of l, the information that is to be hidden in an image is the concatenation of the binary representations of l and x, denoted by the tuple (l, x), where its size is equal to $lf = d + l$. Upon the reception of the stego image, the receiver should, first, extract the d bits so as to know the exact number of message bits that he or she will extract from the stego image. Since d is unknown for the receiver of the stego image, we consider a fixed upper bound D for d which could be sufficiently large to accommodate large messages (in practice $D = 16, 32$ are sufficiently large). Thus the total number of bits that is to be hidden in a cover image is $lf = D + l$.

Our information hiding protocol is based on a secret key steganographic method in image frequency domain. The two agents share a set of cover images in a bitmap format and a secret key. The secret key is divided into two keys. The first key is used for searching goods blocks while the second key will be used for hiding the message.

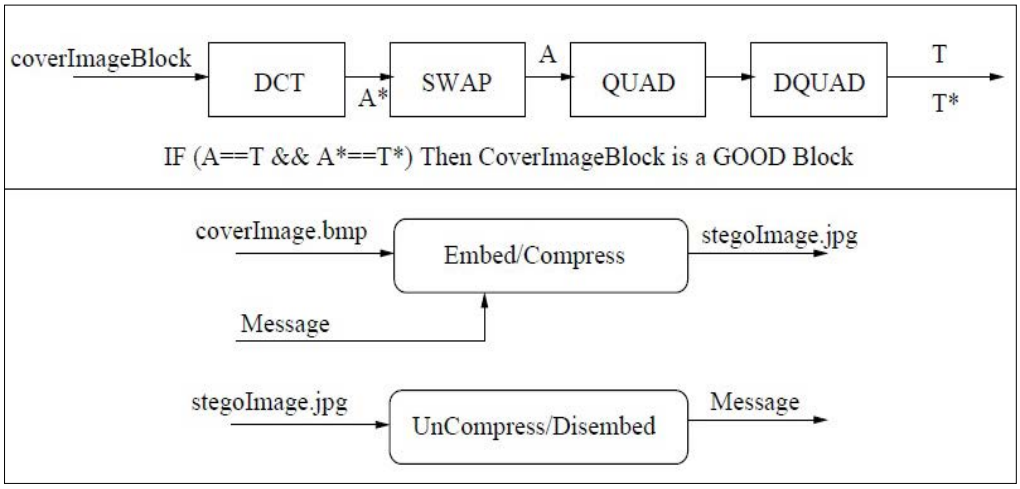


Figure 2: "Good" block selection with scheme 1.

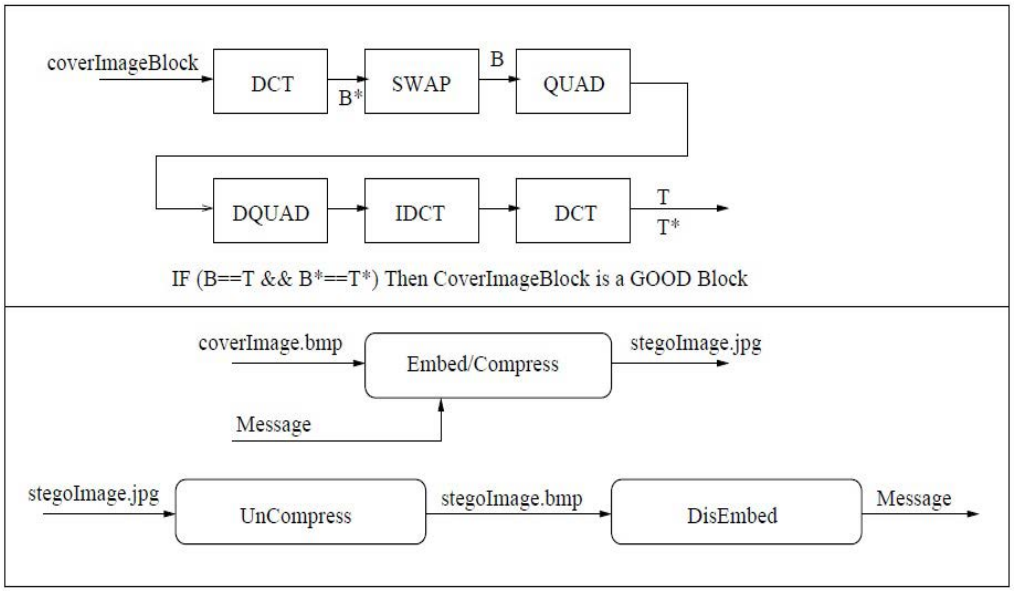


Figure 3: "Good" block selection with scheme 2.

When an agent is to transmit a message to her peer agent the protocol selects a bitmap cover image for hiding the message and it calls the embedding method. The output of the embedding method is always in jpeg format. The embedding method, consists of three stages. At the first stage, the method searches for a set of lf good blocks in the sense that these blocks can convey message bits without a loss due to jpeg compression and with minimal image distortion. We propose three schemes for selecting blocks capable for hiding bits.

The first scheme is used when the message is embedded and extracted as a part of the JPEG compression and decompression respectively. Specifically, using the first key, an 8 × 8 block is selected among the PB cover image blocks. The 2D-DCT transform of the block is computed, the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in A^* (i.e $A^* = G_i(4, 1) > G_i(3, 2)$ or $A^* = G_i(4, 1) < G_i(3, 2)$), the coefficients of the block are quantized and dequantized and finally the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T^*. If the relation $A^* == T^*$ is true, the process continues by checking a second criterion otherwise the block is considered as a "bad" block, as the relation between the coefficients is broken after the above computation steps. In the second step, the 2D-DCT transform of the block is computed, the values of the coefficients $G_i(4, 1)$ and $G_i(3, 2)$ are swapped, the relation between them is stored in A, the coefficients of the block are quantized and dequantized and finally the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T. If the relation $A == T$ is true, then the block is considered as a "good" block for conveying a message bit. The described computation steps are depicted in Figure 2.

The second scheme is used when the message is embedded as a part of JPEG compression and disembedded in two steps: retrieve the bitmap image from the transmitted JPEG image and then extract the message from the bitmap image. Specifically, using the first key, an 8 × 8 block is selected among the *PB* cover image blocks. The 2D-DCT transform of the block is computed, the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in B^* (i.e $B^* = G_i(4, 1) > G_i(3, 2)$ or $B^* = G_i(4, 1) < G_i(3, 2)$), the coefficients of the block are quantized and dequantized, the inverse 2D-DCT and the 2D-DCT are applied successively and, finally, the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T^*. If the

relation $B^* == T^*$ is true, the process continues by checking a second criterion otherwise the block is considered as a "bad" block since the relation between the coefficients is broken after the above computation steps. In the second step, the 2D-DCT transform of the block is computed, the values of the coefficients $G_i(4, 1)$ and $G_i(3, 2)$ are swapped, the relation between them is stored in B, the coefficients of the block are quantized and dequantized, the inverse 2D-DCT and the 2D-DCT are applied successively and, finally, the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T. If the relation $B == T$ is true then the block is considered as a "good" block for conveying a message bit. The described computation steps are given in Figure 3.

Finally, the third scheme is used when the message is embedded in two steps, where the message is embedded in the bitmap image and the stego image is compressed and extracted as a part of the JPEG decompression. Specifically, using the first key, an 8 × 8 block is selected among the PB cover image blocks. The 2D-DCT transform of the block is computed, the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in C^* (i.e $C^* = G_i(4, 1) > G_i(3, 2)$ or $C^* = G_i(4, 1) < G_i(3, 2)$), the inverse 2D-DCT and the 2D-DCT are applied successively, the coefficients of the block are quantized and dequantized, and finally the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T^*. If the relation $C^* == T^*$ is true, the process continues by checking a second criterion, otherwise the block is considered as a "bad" block since the relation between the coefficients is broken after the above computation steps. In the second step, the 2D-DCT transform of the block is computed, the values of the coefficients $G_i(4, 1)$ and $G_i(3, 2)$ are swapped, the relation between them is stored in C, the inverse 2D-DCT and the 2D-DCT are applied successively, the coefficients of the block are quantized and dequantized, and finally the relation between $G_i(4, 1)$ and $G_i(3, 2)$ is stored in T. If the relation $C == T$ is true then the block is considered as a good block for conveying a message bit. The described computation steps are depicted in Figure 4.

The above schemes assure that the selected blocks that fulfill the above constraints can convey any message bit even if JPEG compression is used. Moreover, they assure that each block will be modified slightly, only if a swap of the coefficients will take place, which means that the image distorion will be small enough so as it will not be easily perceptible that an image conveys a message. Moreover, it is obvious that not all *PB* blocks of a cover image can convey successfully a message bit under the previously described constraints. Only a fraction of the *PB* possible blocks can be considered as candidates for conveying message of bits. The value of this fraction depends on cover image characteristics.

Let *FB* be the number of feasible blocks of a cover image, i.e. the number of blocks that satisfy the above constraints. The problem of embedding a tuple (l, x) into a particular cover image can be solved if and only if the inequality lf < FB holds. Thus, having a cover image with FB feasible cover blocks and a sequence of lf bits to be embedded in the image, the number of feasible solutions is given by,

$$P(FB, lf) = \binom{FB}{lf} \times (lf!).$$

Consequently, for every message X the embedding procedure should choose an image with *FB* cover blocks so that lf < FB. Since FB determines the size of the space of feasible solutions, it is obvious that FB should be sufficiently large so as message retrieval with exhaustive enumeration of all possible solutions will not be computationally feasible.

The proposed embedding procedure presented in Figure 5 uses two secret keys and it consists of three stages. In the first stage, using the first secret key and one of the previously described good blocks selection scheme, a pool of lf out of *FB* feasible blocks is formed. In the second stage, the first *D* blocks are reserved for the bits of the message size while the rest $l = lf$ - D feasible blocks of the pool are randomly permuted using the second secret key so as to increase the difficulty of message retrieval by a third party. The blocks of the message size are excluded from the second phase since the knowledge of blocks that are used for hiding the size of the message could not provide any valuable information to an attacker. Finally, in the third stage, the provided sequence of blocks is used so as to embed the tuple (l, x).

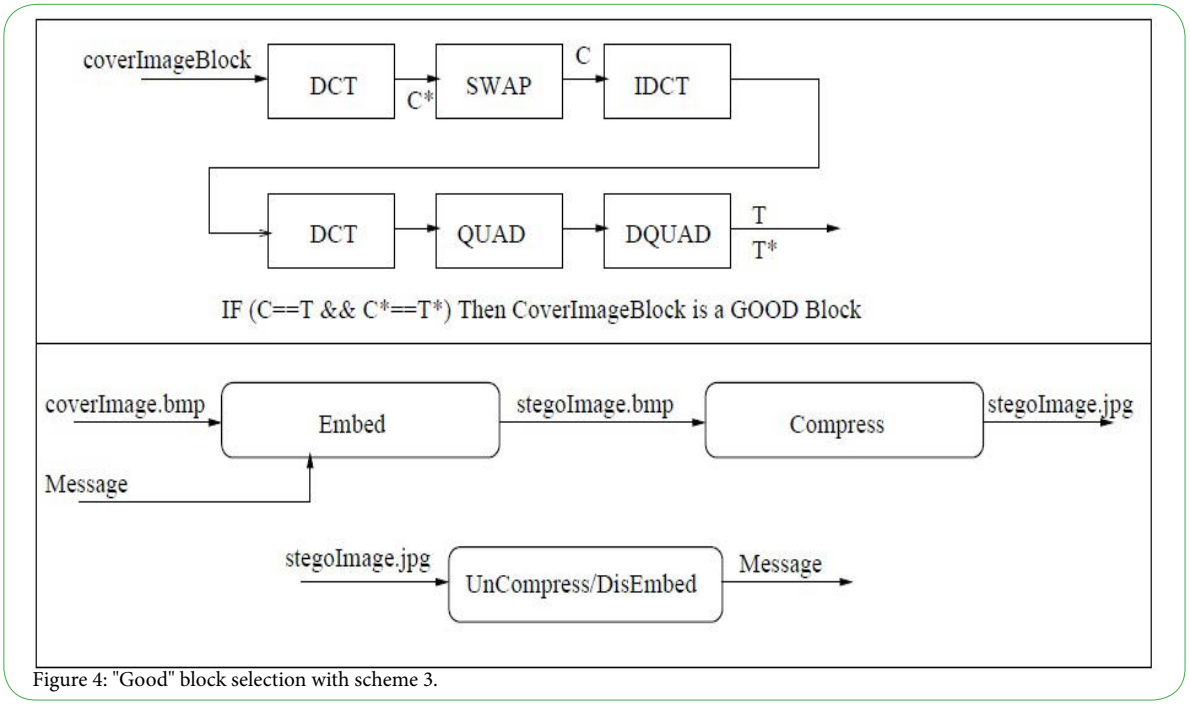


Figure 4: "Good" block selection with scheme 3.

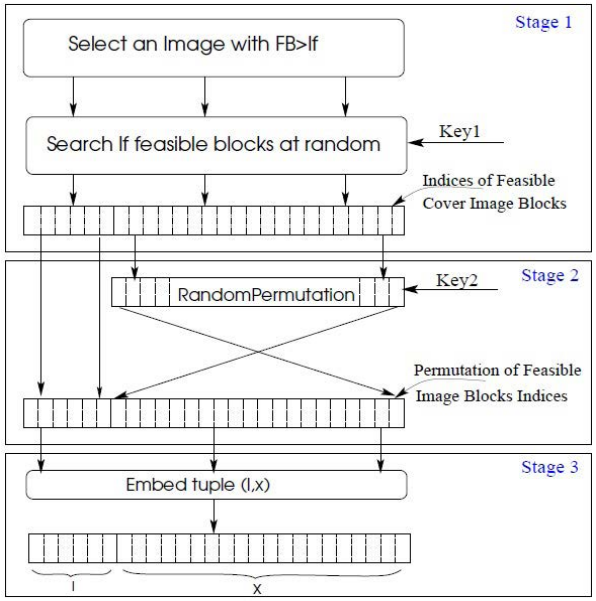


Figure 5: The proposed two secret key Embedding Procedure.

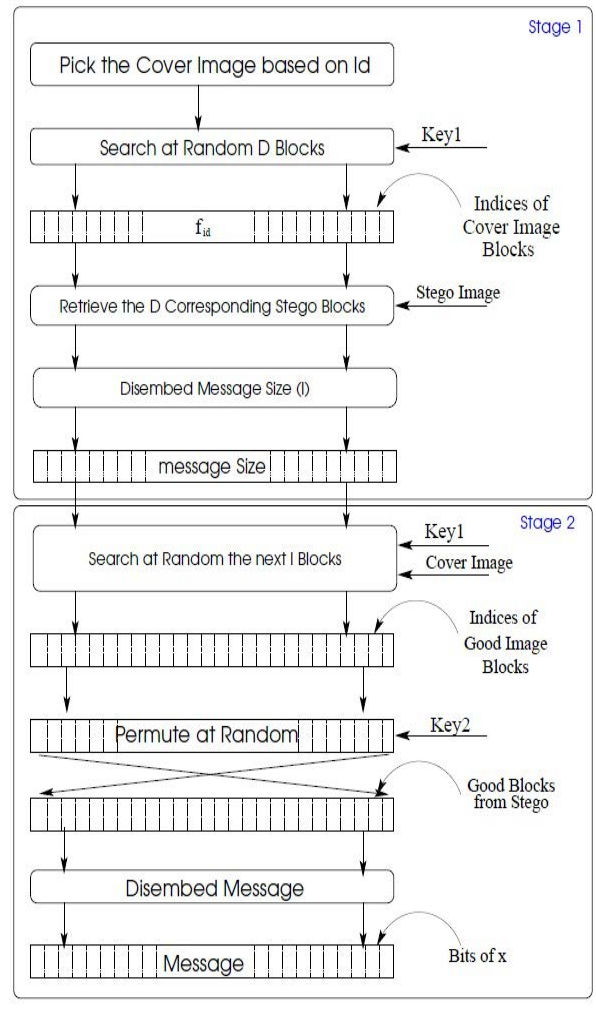


Figure 6: The proposed two secret key Disembedding Procedure.

Upon the reception of the stego image, the two secret key disembedding procedures presented in Figure 6 are applied for message retrieval. The disembedding procedure needs both the stego image and the corresponding cover image. In the first stage, using the cover image and the first secret key, the indices of the first D feasible blocks are retrieved. The next step is to retrieve the D feasible blocks from the stego image using the previously retrieved indices and to apply the extraction method. At the end of the first stage, the number of message bits l is retrieved.

In the beginning of second stage, the extraction procedure retrieves the next l feasible cover image blocks where the hidden message resides. It constructs a permutation of the indices of the retrieved image blocks using the second secret key. Finally, the permuted image blocks are used by the extraction method so that the integer x that encodes the message X is retrieved. Thus, by applying the decoding function f^{-1} described above, the original message X can be retrieved.

The presented embedding and extracting procedures are the basic elements of the information hiding protocol depicted in Figure 7. A pair of agents agree to use for their communication a set of cover images with various number of feasible blocks FB where each image is described by an identification number ID. Initially, the two agents establish a pair of secret keys that the embedding and disembedding procedures are using. For this establishment, any public key scheme can be used. Whenever an agent A needs to send a hidden message to its peer B, it simply sends the cover image ID to B at first and initiates the embedding procedure. Upon the reception of the ID, the agent B initiates the first stage of the disembedding procedure to retrieve the first D feasible blocks where the message size resides. The agent A, upon the completion of the embedding procedure, transmits the stego image to agent B. The suspended disembedding procedure in B is resumed and the message X is retrieved.

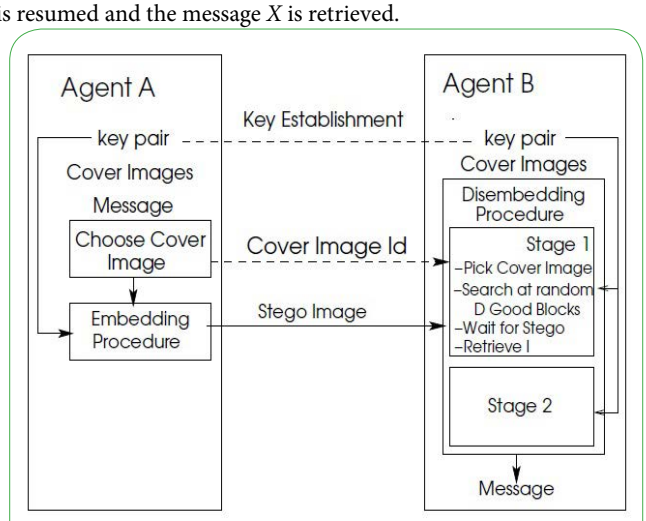


Figure 7: The Proposed Information Hiding Protocol.

Experimental Results

This section presents experimental results for the evaluation of the block selection schemes and the corresponding embedding and disembedding methods that the proposed protocol employs. The six gray-level images that appear in Figure 8 were selected as the set of cover images on which the proposed embedding and disembedding methods were applied. Our experiments were run on the high performance computing cluster named Pythagoras of the Department of Mathematics of the Aegean University. The operating system

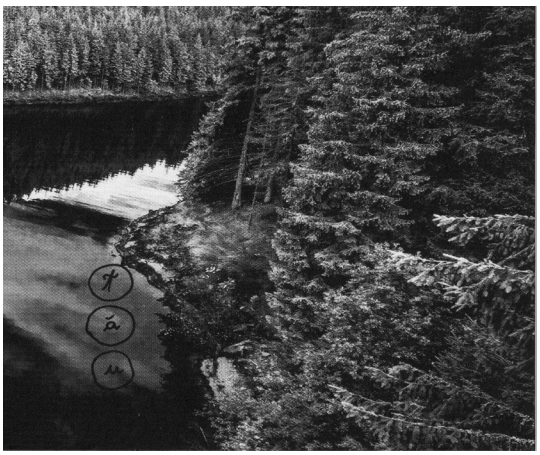
environment of Pythagoras is the Rocks Cluster Distribution (version Sidewider) on the CentOS Linux distribution. There are 120 processor logical cores (64 physical) available for computations distributed in 5 computing nodes and 24 logical cores for managing the computing cluster at the front-end node. Each node has a secondary storage while both the computing nodes and the front-end node share a filesystem through Network File System (NFS). The high throughput computing environment is provided by Condor which is the standard grid middleware on Rocks Clusters.

Initially, each of the proposed schemes was applied on every image of Figure 8. In Table 2, the space of feasible blocks produced by each scheme and for each image is presented. As we can see Scheme-1 gives, in general, more feasible blocks in comparison with Scheme-2 and Scheme-3. However, as we shall see later, Scheme-2 and Scheme-3 provide better feasible blocks that, also, minimize the distortion of an image after embedding a message. For large images like Salonica, Draw and Tau, Scheme-3 provides more feasible blocks than Scheme-2 while for smaller images like Lena, Cow, and Art both schemes have

(a) Art: 1024 × 1024

(b) Cow: 1024 × 1024

(c) Draw: 2048 × 2048



(d) Tau: 2048 × 2048

(e) Lena: 512 × 512

(f) Salonica: 2048 × 2048

Figure 8: A set of images that are used in our experiments.

comparable performance. Five messages of various lengths were used as candidate messages for the embedding and disembedding methods. Specifically, we used a small message of length m_0 = 93 bits (one sentence of 18 characters), two of medium size m_1 = 217 (one sentence of 42 characters) and m_2 = 496 bits (five senctences of 96 characters in total) and two large messages m_3 = 723 (five sentences of 140 characters in total) and m_4 = 1307 bits (10 sentences of 253 characters in total). Using 100 different keys, for every message m_i, each of the three schemes was applied on every image in order to find a set of good blocks that will be used by the embedding method. In Tables 3,4, and 5, the average running time (in seconds) of each scheme for each message on every image is presented. It is obvious that as the message size increases linearly, the running time of a scheme also increases linearly. Moreover, as the number of feasible blocks that an image provides decreases, the running time of searching good blocks for embedding a message increases. For example, the image entitled Draw provides 3088 blocks under scheme-1, 1224 under scheme-2 and 2563 under scheme-3 as it appears in Table 2. Searching good blocks for embedding the message m_0 using scheme-1 took 0.17 seconds, using scheme-3 took 0.81 seconds (since there are less good blocks) and using scheme-2 took 2.032 seconds where there are few good blocks.

Image	number of FBs		
	Scheme-1	Scheme-2	Scheme-3
Art	8309	8307	8117
Cow	6240	5782	5777
Draw	3088	1224	2563
Lena	806	805	787
Salonica	7546	2675	4353
Tau	24319	17105	21972

Table 2: Number of Feasible Blocks in certain images using the proposed selection schemes.

Image	Message size in bits				
	m_0 = 93	m_1 = 217	m_2 = 496	m_3 = 723	m_4 = 1307
Art	0.022	0.046	0.099	0.145	0.270
Cow	0.025	0.054	0.122	0.182	0.371
Draw	0.170	0.391	0.831	1.221	2.446
Lena	0.045	0.105	0.287	0.632	-
Salonica	0.086	0.175	0.364	0.523	0.934
Tau	0.031	0.067	0.136	0.194	0.339

Table 3: Running time of searching good blocks for various messages using Scheme-1.

The impact on the image of a steganographic method can be measured by calculating the peak signal-to-noise ratio (PSNR). The peak signal-to- noise ratio (PSNR) is defined by the following equation

$$PNSR = 10log_{10}\frac{255^2}{MSE} \quad (7)$$

Image	Message size in bits				
	m_0 = 93	m_1 = 217	m_2 = 496	m_3 = 723	m_4 = 1307
Art	0.076	0.186	0.467	0.689	1.163
Cow	0.131	0.314	0.651	0.994	1.606
Draw	2.032	4.395	10.393	17.144	-
Lena	0.185	0.414	1.130	2.554	-
Salonica	1.081	2.082	4.843	6.966	13.573
Tau	0.185	0.426	0.914	1.176	1.985

Table 4: Running time of searching good blocks for various messages using Scheme-2.

Image	Message size in bits				
	m_0 = 93	m_1 = 217	m_2 = 496	m_3 = 723	m_4 = 1307
Art	0.074	0.170	0.434	0.651	1.142
Cow	0.102	0.265	0.614	0.861	1.441
Draw	0.813	1.635	3.346	4.912	10.144
Lena	0.141	0.381	1.055	2.529	-
Salonica	0.564	1.105	2.142	2.955	4.921
Tau	0.140	0.300	0.416	0.600	1.074

Table 5: Running time of searching good blocks for various messages using Scheme-3.

where the mean square error (MSE) for an N × M grayscale cover image *cImage* and stego image *sImage* is calculated using the following formula

$$MSE = \frac{\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}(sImage(x,y) - cImage(x,y))^2}{N \times M} \quad (8)$$

Let x_1 and x_2 be the PSNRs of images I_1 and I_2 respectively after embedding a message m into them. If it holds $x_1 < x_2$ then it can be easily proved that the MSE of I_1 is greater than the MSE of I_2. Thus, the distortion in image I_2 is smaller than the distortion in image I_1.

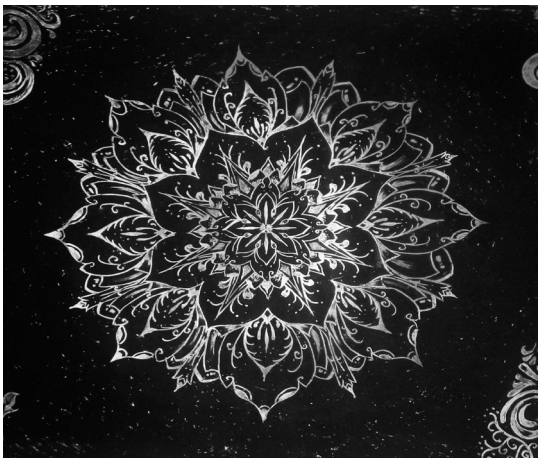
In order to evaluate the quality of the embedding and disembedding methods, the number of feasible blocks per message bit that an image provides under a scheme, denoted by FBB, is introduced. It can be easily calculated by $FBB = \left\lfloor \frac{FB}{m_i} \right\rfloor$ where FB is the total number of feasible blocks of an image under the operation of a scheme. Using 100 different keys, every message m_i is embedded and disembedded in each image of the selected set using the three selection schemes. In Tables 6, 7, and 8, the calculated PSNRs and FBB per image and message for every block selection scheme are presented. Obviously, for every scheme, as the message size increases, the distortion increases since the PSNR decreases. Moreover, although scheme-2 and scheme-3 provides smaller FBB than scheme-1, they have better PSNRs than Scheme-1 with a cost of greater running times due to a smaller FBB. Scheme-2

Images	Message size in bits									
	m_0 = 93		m_1 = 217		m_2 = 496		m_3 = 723		m_4 = 1307	
	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB
Art	50.0	89	46.7	38	43.3	16	41.7	11	39.2	6
Cow	45.2	67	41.8	28	38.4	12	36.8	8	34.2	4
Draw	42.6	33	39.3	14	35.9	6	34.3	4	31.8	2
Lena	46.2	8	42.8	3	39.3	1	37.8	1	-	0
Salonica	40.5	81	37.2	34	33.7	15	32.1	10	29.6	5
Tau	48.5	261	45.3	112	41.8	49	40.2	33	37.7	18

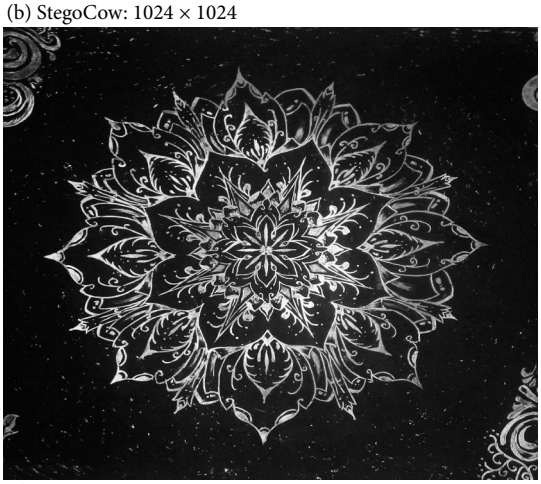
Table 6: PSNR and FBB per image for various messages using Scheme-1.

(a) Cow: 1024 × 1024

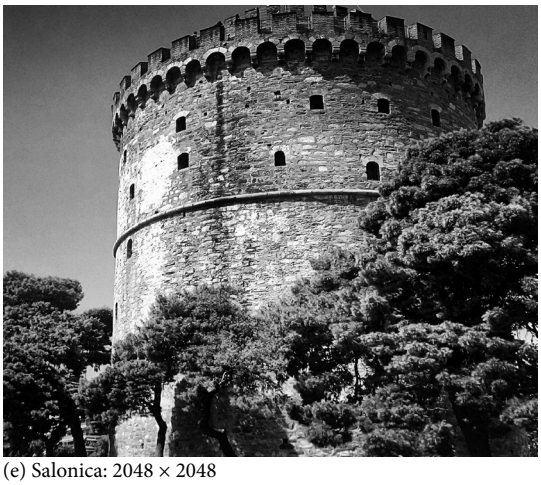
(b) StegoCow: 1024 × 1024



(c) Draw: 2048 × 2048



(d) StegoDraw: 2048 × 2048



(e) Salonica: 2048 × 2048

(f) StegoSalonica: 2048 × 2048

Figure 9: Cover and Stego Images after embedding a message of m_i = 1126 in certain Images using Scheme 2.

has better PSNRs than Scheme-3 but Scheme-3 provides always a solution even for large messages since *FBB* is large enough. Finally, the PSNRs that a scheme-2 provides for sufficiently large messages, where the corresponding FBBs are small, are better than those appearing in [5] where every image block is, potentially, capable of embedding a message bit (the case where $L = 1$), i.e. the corresponding FBB is greater than the *FBB* of scheme-2.

Thus, which scheme will be used each time for hiding a message, depends on the message size, the image characteristics, the set of available images and the time constraints of the protocol. An example where the existence of a large message in the images we used can hardly be detected visually appears in Figure 9. Using a random key, a message of size m =1126 bits was embedded in images Cow, Draw and Salonica using the block selection scheme 2. If we perform a 800% zoom in the images of the document, we can hardly see in Cow in the

Images	Message size in bits									
	$m_0 = 93$		$m_1 = 217$		$m_2 = 496$		$m_3 = 723$		$m_4 = 1307$	
	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB
Art	50.0	89	46.7	38	43.3	16	41.7	11	39.2	6
Cow	52.5	62	49.2	26	45.8	11	44.3	7	41.7	4
Draw	50.4	13	47.1	5	43.6	2	42.0	1	-	0
Lena	46.3	8	42.9	3	39.4	1	37.8	1	-	0
Salonica	49.5	28	46.1	12	42.5	5	41.0	3	38.4	2
Tau	52.1	183	48.8	78	45.3	34	43.7	23	41.2	13

Table 7: PSNR and FBB per image for various messages using Scheme-2.

Images	Message size in bits									
	$m_0 = 93$		$m_1 = 217$		$m_2 = 496$		$m_3 = 723$		$m_4 = 1307$	
	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB	PSNR	FBB
Art	49.4	87	46.4	37	43.2	16	41.6	11	39.1	6
Cow	48.5	62	45.9	26	42.9	11	41.5	7	39.0	4
Draw	44.1	27	40.5	11	36.9	5	35.3	3	32.8	1
Lena	44.3	8	41.6	3	38.6	1	37.2	1	-	0
Salonica	44.3	46	40.8	20	37.2	8	35.6	6	33.1	3
Tau	49.3	236	45.9	101	42.4	44	40.7	30	38.2	16

Table 8: PSNR and FBB per image for various messages using Scheme-3.

right-hand, lower corner a few black dots over the straws, in Salonica in the left-hand, lower corner a few white dots over the tree trunk while in Draw in the left-hand, lower corner in the black area there are a few white dots.

Conclusions and Future Work

In this paper, a stego-key steganographic method operating in the frequency domain of digital images and an information hiding protocol based on this method were presented. The cover image block selection schemes of the embedding method were evaluated experimentally.

The two communicating agents share a set of cover images and a secret key. A cover image for hiding the message is selected and the sender invokes the embedding method. The secret key is divided into two subkeys. The method using the first key and one of the proposed three schemes for selecting a set of "good" blocks in the sense that these blocks can convey message bits with minimal image distortion and that the bits can be retrieved even if the image is compressed using the JPEG standard. Using these blocks and the second key, the method hides the message.

Our plan is to implement the proposed protocol and to simulate its use in a computing cluster environment using the Message Passing Interface (MPI). An agent can be simulated by a single process and the MPI library can provide the communication routines for exchanging messages. Then, the protocol can be deployed over any communication network. Moreover, steganographic attacks and analysis focus on detecting, extracting and deleting or changing hidden information. We plan to investigate the resistance of the proposed embedding method on stego-only and chosen-stego attacks.

Competing Interests

The authors declare that they have no competing interests.

Funding

P.M. Pardalos was partially supported by the Paul and Heidi Brown Preeminent Professorship at the Department of Industrial and Systems Engineering (ISE), University of Florida.

References

1. Zielinska E, Mazurczyk W, Szczypiorski K (2014) Trends in Steganography, Communications of ACM, 57: 86-95.
2. De Selincourt A (1954) Herodotus: The Histories, Penguin.
3. Sharda S, Budhiraja S (2013) Image Steganography: A Review. International Journal of Emerging Technology and Advanced Engineering 3: 707-710.
4. Wang K, Lu ZM, Hu YJ (2013) A high capacity lossless data hiding scheme for JPEG images. The Journal of Systems and Software 86: 1965-1975.
5. Chang CC, Lin CC, Tseng CS, Tai WL (2007) Reversible hiding in DCT-based compressed images. Information Sciences 177: 2768-2786.
6. Cox IJ, Kilian J, Leighton T, Shamoon T (1996) A Secure, Robust Watermark for Multimedia. Workshop on Information Hiding, pp 185-206.
7. Zhao J, Brands EKS (1995) Embedding Robust Labels Into Images For Copyright Protection. In Proc. of Int. Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Technologies, Vienna.
8. Katzenbeisser S, Petitcolas FAP (2000) Information Hiding Techniques for Steganography and Digital Watermarking, Artech House Inc.
9. Gonzalez RC, Woods RE (2000) Digital Image Processing, Prentice Hall, 3rd edn.,
10. Lim SJ (1989) Two Dimentional Signal and Image Processing, Prentice Hall PTR.
11. Manolakis D, Ingle V (2011) Applied Digital Signal Proccessing: Theory and Practice, Cambridge University Press.
12. Wallace GK (1992) The JPEG still picture compression standard. IEEE Trans- actions on Consumer Electronics 38: 1.
13. The Gnu Multiple Precision Arithmetic Library.