Multimedia Tools and Applications (2023) 82:19323–19346
https://doi.org/10.1007/s11042-022-14166-3

Developing an adaptive DCT-based steganography method using a genetic algorithm

Vajiheh Sabeti[1] · Adeleh Aghabagheri[1]

Received: 7 June 2021 / Revised: 7 April 2022 / Accepted: 27 October 2022 /
Published online: 17 November 2022

Abstract

Steganography is an appropriate approach to establish a secure connection between the sender and the receiver. Data embedding in Discrete Cosine Transform (DCT) coefficients for JPEG images is one of the most practical approaches nowadays. In this paper, a new method called GA-Shield is proposed, in which, instead of using fixed embedding capacity, embedding a different number of bits in the quantized DCT coefficients according to the magnitude of the coefficient is used to spread bits of secret message in the most suitable coefficients. In addition, this method uses a genetic algorithm to minimize the distortion due to embedding. This minimization is performed by deciding on the best formula to calculate coefficient value after embedding. In this phase, PSNR is used as the metric to measure the amount of distortion in the cover image to produce the stego image. As these changes decrease, the value of PSNR would be optimized, and the stego image would have better quality. The proposed method can embed 300 to 20,000 bits of data (on average) in the cover image and produce the stego image with a PSNR value in the range of 65 to 40 and a SSIM value of more than 0.985. The consequences of comparisons with the state-of-the-art show that despite the fact that the proposed technique has less embedding capacity than some of the current ones, the superiority of stego image quality and security of the proposed technique, mainly at low embedding levels, is significant.

Keywords Steganography · Steganalysis · JPEG images · Discrete cosine transform · Genetic algorithm

✉ Vajiheh Sabeti
v.sabeti@alzahra.ac.ir

Adeleh Aghabagheri
adeleh.aghabagheri@gmail.com

[1] Department of Computer Engineering, Faculty of Engineerning, Alzahra University, Tehran 1993893973, Iran

1 Introduction

Although the expansion of the digital world and pervasive use of the internet as means of data transfer have led to the establishment of comfortable digital communication, providing information security on a network is still a crucial challenge. To solve this issue, various approaches have been presented by dividing the issue into two major parts; information encryption and information hiding. In information encryption which is known as cryptography, the structure of the message is altered such that it is meaningless and incomprehensible. Information hiding is on the other hand divided into two sections, steganography and watermarking. Although these branches are relatively similar, there are some differences between their applications. In watermarking, the message is precisely related to the cover image, whereas, in steganography, the cover image is only a means for hiding communication [19].

Steganography is the art and science of keeping communications hidden. In other terms, steganography is aimed at embedding and transferring secret messages by getting advantage of digital media that is indiscernible for any third party. Steganography and steganalysis are a pair of antagonistic techniques, wherein the former conceals secret messages within cover media, and the latter looks for embedding artifacts to reveal the presence of secret messages within stego images [41].

Three requirements including perceptual transparency, hiding capacity, and security are needed while designing an appropriate steganography approach with a passive warden scenario. Perceptual transparency means that stego media and cover media should not be perceptually different. Hiding capacity refers to the maximum number of bits that can be embedded into a cover media and security is aimed at being unrecognizable for the resultant stego media of the approach by existing steganalysis methods [17]. Considering the active warden scenario, the robustness against active attacks (such as rotation, compression, etc.) is also added to the list of basic requirements. Usually, the first scenario is used in steganography methods and the second scenario is used in watermarking methods. Embedding efficiency is one of the metrics that has recently been proposed as an indicator of higher stego image quality and lower message detectability [3].

In space domain steganography methods, pixel intensity values of cover media are used for the direct or indirect embedding of hidden data. The Least Significant Bit (LSB) is one the most famous approaches in the space domain including two useful methods of LSB Matching (LSBM) and LSB Flipping (LSBF). The main idea is formed by assuming that the least significant bits of the image are less important and any alteration in these bits would not be recognized by human eyes. Methods with the idea of bitplanes index manipulation [1] and pixel intensity value decomposition [2] are other examples of LSB-based methods. In transform domain algorithms, the secret message is embedded into transform coefficients. Compared to space domain techniques, transform domain approaches are more complex and have less capacity, and consequently, will be recognized with more difficulty [7].

In general, due to the large size of images, compression is indispensable. This issue may lead to a lack of integrity for the secret message and irreversibility for the approach. In order to solve this problem, the secret message is embedded through a compression process. Discrete Cosine Transform (DCT) is one of the most important transforms which embeds data during a compression process like JPEG. This is the most widespread format among images, which is produced by digital cameras and scanners, and thus, DCT methods have a special importance among all transform domain methods [8].

Most of the existing methods assumed the embedding ratio in all non-zero DCT coefficients to be 1 bit or otherwise, no solution was provided with the choice of variant embedding ratio to overcome the decline in the stego image quality and security level. One of the main ideas of the proposed method is to choose an adaptive embedding rate for all non-zero DCT coefficients, based on their quantity. Therefore, data is spread over the whole image and it attempts to use the most appropriate coefficients for embedding, and improve the security of the approach. To these ends, a genetic algorithm (GA) is used to enhance the PSNR of the stego image, which led to raising security as well. As PSNR is increasing, the quality of stego image is also improving. As a result, embedding effects occur less in the stego image and therefore, the security of the approach is acceptable.

The following is a summary of the important contributions of this article:

- Proposal of a new steganography method based on variant embedding rate for all non-zero DCT coefficients, according to their quantity.
- The proposed method employs a genetic algorithm to increase the PSNR value by selecting the best principal for embedding data at each non-zero DCT coefficient.
- The performance of the proposed method was measured for different numbers of bits that can be embedded in the coefficients. Numerous criteria have been used to evaluate the quality of the stego images and the security of the proposed method.

The remainder of this paper is organized as follows: Section 2 describes previous works around DCT domain steganography. In Section 3, we introduce our proposed approach by dividing it into the embedding and extraction algorithm. Experimental results and comparisons are provided in Section 4. Finally, we conclude our work in Section 5.

2 Related works

Considering that the new method presented in this paper is applied on DCT coefficients using GA, related articles are reviewed in two subsections. In the first part, the existing methods in the field of DCT are examined from the perspective of the purpose of the method, and in the second part, a history of the application of optimization algorithms in transform domain steganography is presented.

2.1 DCT-based steganography methods

DCT is a transformation that takes an image block in a spatial domain and transforms it into a frequency domain. JPEG compression is an example of where DCT is used and its process is shown in Fig. 1. There are various DCT-based steganography methods with different goals. One of the main approaches is increasing embedding capacity. DCT-based methods usually use two approaches to achieve this goal: making changes in the quantization table [10, 12, 18, 32] and using a different number of bits for embedding [6, 25, 33]. In most cases, these methods did not maintain the quality and security of stego images.

One approach which aimed to raise the hiding capacity is the Shield algorithm [6]. This technique uses the idea of classification of DCT coefficients based on their quantity, such that unlike using a fixed number of least significant bits for embedding data in all DCT coefficients, it uses a different number of bits according to the value of the coefficient. Therefore, the

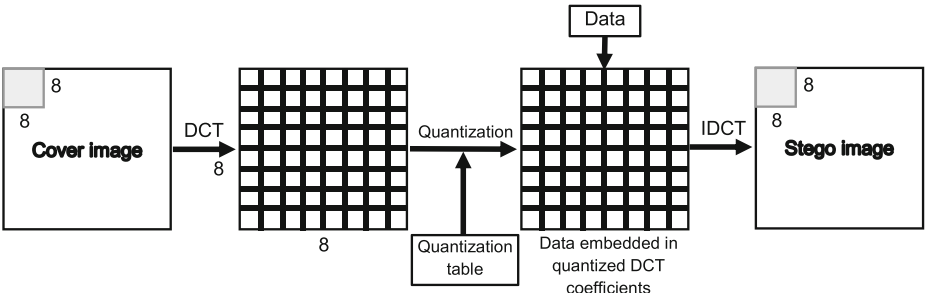

Fig. 1 DCT-based data-hiding using the JPEG compression model [27]

number of employed bits is increased by raising the amount of coefficient. However, this method does not provide any acceptable PSNR.

Some methods aimed to improve the quality of stego images. These methods usually use fewer coefficients for embedding and, consequently, embedding capacity is reduced. For instance, in [24], low- and high-frequency DCT coefficients are not changed and the embedding in LSB of medium frequency DCT coefficients is proposed. Keeping the low-frequency coefficients constant, maintains the quality of the uniform areas of the cover image, while by keeping the high-frequency coefficients constant, the quality of the edges of the cover image is maintained. This method is referred to as MedFreq.

Another group of methods is proposed to improve security. These methods usually have a very low embedding capacity. In [47], an improved LSB-based steganography technique called GPM1, which uses GA on JPEG images, has been proposed. This method preserves the first-order statistical properties. GA is employed to choose whether to increase or decrease the coefficient by one unit for embedding the data.

In [4], a new algorithm called DCT-M3 is proposed. In this method, two DCT coefficients are selected from each 8×8 block. Then it uses modulus 3 of the difference between two DCT coefficients to embed two bits of the secret message. DCT-M3 minimizes the changes caused by embedding data. However, this method has drastically reduced the embedding capacity.

In [39], a method called MPS is proposed. In this method, four triple sets of DCT coefficients are considered in each block. In each set, one of the coefficients is randomly selected as the index. No data is stored in the index coefficient. Of the remaining two coefficients, embedding is performed in the smallest coefficient. Furthermore, the number of bits that could be embedded in the selected coefficient is agreed as a key between the sender and receiver. This number was allowed to be between 1 and 8.

Some methods try to use the idea of adaptive embedding. In these methods, more embedding is done in coefficients that belong to the edge areas of the image and thus can maintain the security of the method to some extent. However, in case the number of bits embedded in the coefficients is increased (with the aim of increasing the embedding capacity), the quality of the stego image would be decreased and the probability of detecting the stego image would be increased. In [9], in order to increase the hiding capacity and preserve first-order properties, the LSB-based method is improved and a different number of bits are used for embedding. Then, the order of embedded bits is changed according to the chaos algorithm and the best sort would be chosen by GA such that the stego image provides the highest PSNR.

Another technique to increase the embedding capacity is IA-LSB [38], which had the major idea of adaptive embedding. In this method, DCT blocks are divided into two parts and a different number of bits are used in each section. The Least Modification Rule (LMR) is used to decrease the effects on the stego image. Thus, there would be two resultant values for embedding every size of data into every coefficient, but the selected value was the one with the least fluctuation before and after the embedding process. In this method, GA was used to alter the order of message bits to obtain the best order.

Among all categories, approaches focusing on improving the stego image quality as well as preserving security are the most applicable ones, since they meet two important requirements of hidden communication. Our proposed method is classified in this category. Like the approach presented in [6], a different number of bits is used for data embedding, and GA is employed to improve the PSNR of stego images. However, in order to have a better result, both the strategy, in which the number of bits for embedding is determined and the rules that should be calculated in GA, are different from previous studies. In [46], the number of bits for embedding, which is limited to only two various values for two divided parts of each DCT block, while in our approach, five disparate values are used according to the size of the DCT coefficient, which could lead to a higher hiding capacity and lower detectability.

2.2 Optimization algorithms in steganography

Evaluation of existing methods revealed that one of the successful ideas is the application of optimization algorithms in the data embedding stage. Given the three objectives, increasing the embedding capacity, enhancing stego image quality, and increasing security, in the design of steganography methods, some methods have modeled their choices as a search problem and have solved it using optimization methods.

Application of optimization algorithms in steganography methods is usually performed in three stages; before embedding, during embedding, and after embedding. In the pre-embedding stage, the optimization algorithm is usually used to find the best place to embed or modify message bits [20, 29, 40, 42, 43]. In the second stage, the optimization algorithm is used to determine how the data is stored and the value of the stego image pixel [48], and in the third stage, the optimization algorithm helps to reduce the changes resulting from the embedding [22, 36]. In some methods, two optimization algorithms were used together [35]. The most common and successful optimization algorithm used in the literature is the GA, while there is limited use of other optimization algorithms, such as PSO [26, 30, 31], Ant Colony [23], or Artificial Bee Colony [5, 21]. For this reason, GA is used in the present research.

3 Proposed method

In this section, a new adaptive DCT-based steganography is introduced. The proposed method benefits from the advantages of the Shield algorithm [6] to adaptively choose the embedding capacity. In this algorithm, instead of using only 1 bit for embedding, it uses a different number of bits for embedding according to the DCT coefficient value. On the other hand, because of the reverse impact of increasing hiding capacity on security, it is necessary to choose a technique that can preserve intended security for the steganography method. To this end, a GA was used and the proposed method is called GA-Shield. Figure 2a shows the

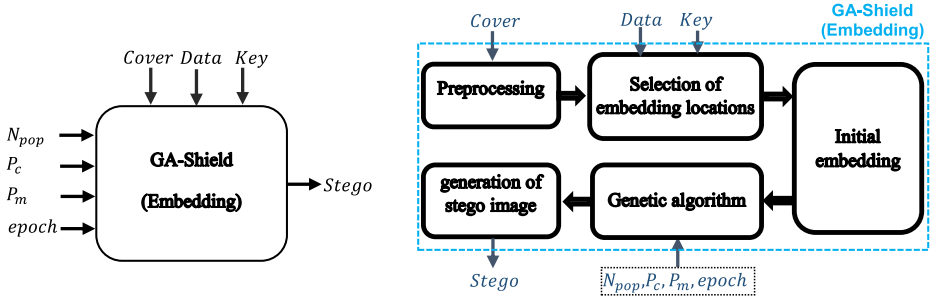

Fig. 2 GA-Shield algorithm. **a** Inputs and outputs, **b** steps of embedding algorithm

necessary inputs and outputs of this method. GA-Shield receives the cover image and the secret data from the input and produces the stego image as the output. In addition, this method has other inputs and outputs that are introduced in the description of the algorithm. Figure 2b shows the embedding algorithm for GA-Shield. In addition, the details of each step are described below. The pseudo-code for embedding algorithm is shown in Algorithm 1 to Algorithm 4, and a list of variables used to describe the proposed method is shown in Table 1. Function names and commands are written in bold.

Algorithm 1 Preprocessing

Input: $Cover$
Output: $DCTCover$

```
[x, y] = size(Cover);
NumRows = x/8;
NumCols = y/8;
for i = 1 : NumRows
  for j = 1 : NumCols
      CoverSub = getSubMatrix(Cover, i, j);
      DCTSub = DCT2(coverSub);
      QuSub = Quantization(DCTSub);
      DCTCover = SetSubMatrix(QuSub, i, j);
  End
End
```

Table 1 A list of variables and functions used to describe the proposed algorithm

Name	Description	Name	Description	Name	Description
Cover	Cover image	P_m	Probability of mutation	*getSubMatrix*	Get a 8×8 blocks of the *Cover*
DCTCover	quantized DCT coefficients of *Cover*	*Epoch*	Number of iteration	*DCT***2**	Apply DCT
x, *y*	Dimensions of *Cover*	*ChangeLoc*	Location of coefficients that needs to be changed	*Quantization*	quantize using a quantization table
NumRows	Number of blocks per row	*LenChr*	Length of chromosome	*SetSubMatrix*	Set *QuSub* in *DCTCover*
Numcols	Number of blocks per column	*NoCross*	Number of crossover	*Length*	Compute length of *Data*
CoverSub	A block of the cover image	*NoMut*	Number of mutation	*rng*	Set the seed of PRNG function
DCTSub	A DCT block	*Pop*	Population	*randperm*	Get a random permutation
QuSub	A quantized DCT block	*Popc*	Population resulting from crossover	*IsAC*	Is this coefficient an AC coefficient?
Data	Secret data	*Popm*	Population resulting from mutation	*bi***2***dec*	Convert a binary number to decimal number
Key	Seed for PRNG	*BestSolution*	Number of the best formula for each coefficients that needs to be changed	*Matching*	Determine coefficients that are used for embedding data and need to be changed
EmbLoc	Selected embedding locations	*Stego*	Stego image	*Initialize*	Random Initialization of the population
DecData	the decimal equivalent of a specific part of the message	*DCTStego*	quantized DCT coefficients of the stego image	*FitnessCal*	Calculation of the fitness of each chromosome
LenData	The length of all secret data	*c*	quantized DCT coefficient value	*Crossover*	Crossover function
RandLoc	Random embedding locations	*n*	number of bits used for embedding	*Mutation*	Mutation function
r	One of the random embedding locations	*L*	The decimal equivalent of *n* LSB of DCT coefficient	*Sort*	Sort chromosomes based of their fitness
index	Start index of data for a coefficient	*m*	the decimal equivalent of a specific part of the message	*Remove*	Remove additional chromosome
k	Start index of data for a coefficient	*s*	New value of coefficients after embedding	*Readjustment*	Alter *s* if *s* is not placed within the intended range
N_{pop}	Number of population	*DeqStego*	Dequantized DCT coefficients of the stego image	*Dequantization*	Dequantize using a quantization table
P_c	Probability of crossover	*size*	Get size of image	*IDCT***2**	Apply inverse DCT

Algorithm 2 Selection of embedding locations

Input: $DCTCover, Data, Key$
Output: $EmbLoc, DecData$

$[x, y] = \textbf{size}(DCTCover)$;
$LenData = \textbf{Length}(Data)$;
$\textbf{rng}(Key)$;
$RandLoc = \textbf{randperm}(1, x \times y)$;
$k = 1$;
$i = 1$;
while $(k <= LenData)$
  $index = k$;
  $r = RandLoc(i)$;
  if $(\textbf{IsAC}(r)\ and\ \textbf{abs}(DCTCover(r)) > 1)$
    if $(\textbf{abs}(DCTCover(r)) < 8)$ $k = k + 1$;
    elseif $(\textbf{abs}(DCTCover(r)) < 16)$ $k = k + 2$;
    elseif $(\textbf{abs}(DCTCover(r)) < 32)$ $k = k + 3$;
    elseif $(\textbf{abs}(DCTCover(r)) < 64)$ $k = k + 4$;
    else
      $k = k + 5$;
    End
    $EmbLoc(i) = r$;
    $DecData(i) = \textbf{bi2dec}(Data(index: k - 1))$;
    $i++$;
  End
End

Algorithm 3 Genetic Algorithm

Input: $DCTCover, DecData, EmbLoc, N_{pop}, P_c, P_m, epoch$
Output: $ChangeLoc, BestSolution$

$ChangeLoc = \textbf{Matching}(DCTCover, DecData, EmbLoc)$;
$LenChr = \textbf{Length}(ChangeLoc)$;
$NoCross = P_c \times N_{pop}$;
$NoMut = P_m \times N_{pop}$;
$Pop = \textbf{Initialize}(N_{pop}, LenChr)$;
$\textbf{FitnessCal}(Pop)$;
for $i = 1 : epoch$
  $Popc = \textbf{Crossover}(Pop, NoCross)$;
  $Popm = \textbf{Mutation}(Pop, NoMut)$;
  $Pop = [Pop\ Popc\ Popm]$;
  $\textbf{FitnessCal}(Pop)$;
  $\textbf{Sort}(Pop)$;
  $\textbf{Remove}(Pop, N_{pop})$;
End
$BestSolution = Pop(1)$;

Algorithm 4 Generation of stego image

Input: $DCTCover, DecData, ChangeLoc, BestSolution$
Output: $Stego$

```
DCTStego = DCTCover;
LenChr = Length (ChangeLoc);
for i = 1: LenChr
   r = ChangeLoc(i);
   c = DCTCover(r);
   n = Range(c);
   L = LSB(c, n);
   m = DecData(i);
   if BestSolution(i) == 0
      s = c + m − L;
    else
      s = c – (2^n – (m − L));
    End
   Readjustment(s);
   DCTStego(r) = s;
End
DeqStego = Dequantization(DCTStego);
Stego = IDCT2 (DeqStego);
```

3.1 Preprocessing

Whereas the proposed method requires quantized DCT coefficients of the cover image for embedding, three steps of blocking, applying DCT, and quantization should be performed on the input cover image. In Fig. 2b, the set of these three steps is called the preprocessing step, and *Algorithm* 1 corresponds to this step of the proposed algorithm. If the input image is in JPEG format, appropriate tools can be used to access the quantized DCT coefficients and there is no need to perform the preprocessing step.

3.2 Selection of embedding locations

In the next step, a number of coefficients must be selected to embed the data (*Algorithm* 2). For this, three main questions need to be answered:

1) Which coefficients are suitable for embedding? Among each block of DCT coefficients, the upper-left coefficient is called the DC coefficient and the remaining 63 coefficients are called AC coefficients. In the GA-Shield method, in order to prevent the image quality from decreasing, the AC coefficients of each block with values other than 0, 1, and − 1 are used for embedding.

2) How many bits of data can be embedded in each coefficient? In order to achieve the goal of proper distribution of data in the coefficients, a different number of bits are embedded in each coefficient according to its magnitude. The intervals used to classify quantized DCT coefficients based on their values, and the number of bits used to embed each interval are specified in Table 2.

Table 2 The capacity of coefficients according to their size in the proposed method

Quantized DCT coefficients	2–7	8–15	16–31	32–63	≥64
Number of bits	1	2	3	4	5

3) What is the order of selecting coefficients for embedding? In each image, according to the coefficients with embedding conditions and embedding capacity of each coefficient, the maximum embedding capacity can be calculated. If the desired data length is less than the maximum capacity, a number of these coefficients must be selected for embedding. The GA-Shield method uses a Pseudorandom Number Generator (PRNG) function to randomly select DCT coefficients. If the selected coefficient has embedding conditions, it will be used for embedding. The selection of new coefficients by the PRNG continues until the total embedding capacity of the selected coefficients is less than the length of data bits. The PRNG requires an initial seed, the value of which as a key has already been agreed between the sender and receiver. In this way, the receiver can use this seed to reconstruct the embedding order used by the sender and extract the embedded data without any error.

3.3 Initial embedding

After selecting the coefficients, the sender must determine the new value of the coefficients according to the data corresponding to each coefficient in such a way that the receiver can extract the desired data. In this approach, in order to embed any number of bits in quantized DCT coefficients according to their value, two major formulas are used. These rules are useful when it is required to embed a different number of bits in various DCT coefficients and they are determined as Formulas (1) and (2).

$$s_i^{'} = c_i + m_i - LSB(c_i) \tag{1}$$

$$s_i^{\}} = c_i - \left(2^l - \left(m_i - LSB(c_i) \right) \right) \tag{2}$$

In which, l is the number of bits used for embedding according to the size of quantized DCT coefficient and c_i, m_i, and $LSB_l(c_i)$ are quantized DCT coefficient value, the decimal equivalent of a specific part of the message selected for embedding in this coefficient, and the decimal equivalent of l least significant bits of DCT coefficient, respectively. $s_i^{'}$ and $s_i^{''}$ are also coefficients after embedding.

These formulas are used if $LSB_l(c_i)$ is different from m_i, otherwise, there is no need to change c_i and the same value will be placed in the corresponding coefficient in the stego image. Noteworthy, if the resultant coefficient after embedding ($s_i^{'}$ and $s_i^{''}$) is not placed within the intended range, it will be added to or subtracted from 2^l such that it leads to being at the intended range, in which 1 is the number of used bits for embedding. If the resultant value is still not at the intended range, the first amount in the mentioned range will be chosen, in which

the corresponding LSB is equal to the determined message bits and the quantized DCT coefficient will be replaced by that.

To illustrate how to embed data using these formulas, Table 3 lists some examples. The goal is to embed *Data* = 11,001,011,100 in the six selected quantized DCT coefficients. The first coefficient is 5 ($c_i = 5$), and according to Table 2, one bit of data must be embedded in this coefficient ($l = 1$). The first bit of the data sequence is also 1 ($m_i = 1$). Since the data bit is equal to the LSB of the desired coefficient ($LSB_1(5) = 1$), this coefficient does not need to be changed. Therefore, without using Formulas (1) and (2), the value 5 is placed in the stego image. The value of the next selected coefficient is 12 ($c_i = 12$). According to Table 2, its embedding capacity is 2 bits ($l = 2$), and thus the two-bit data ($m_i = (10)_2 = 2$) must be embedded in it. Two LSBs of this coefficient are equal to 00 ($LSB_2(12) = (00)_2 = 0$). According to Formula (1), the value of the first alternative is 14 ($s_i' = 14$) and according to Formula (2), the value of the second alternative is 10 ($s_i'' = 10$). After calculating the alternative values in the third and fourth rows of Table 3, an out-of-range error occurs for one of the alternative values, where according to the mentioned explanation, the appropriate alternative value is generated by adding or subtracting 2^l.

3.4 Genetic algorithm

To complete the embedding process, the final value in the stego image for the coefficients must be specified, for which the two alternatives are calculated. The strategy of random selection between two alternative values is the simplest solution, but this choice can be made to achieve a specific goal.

The GA-Shield method uses a GA in order to increase the PSNR value by selecting the best principal for embedding data in each DCT coefficient. PSNR is one of the criteria for stego image quality, which shows the number of changes in stego image compared to the cover image. Fewer changes increase PSNR and thus improve the quality of the stego image. Steganalysis attacks can detect data using changes made to the stego image. Therefore, it is probable that by reducing changes, attacks will be less successful in detecting the embedding approach and thus the security will be increased.

Algorithm 3 corresponds to the GA step of the proposed algorithm. The first step in the GA is to define the structure of a chromosome as a member of the population. It was explained that if $LSB_l(c_i)$ is the same as m_i, there is no need to change c_i. The length of the chromosome is equal to the number of coefficients that are used for embedding data and need to be changed

Table 3 A few examples to illustrate how to use embedding rules (*Data* = 110010110)

c_i	l	m_i	LSB_l	s_i'	s_i''
5	1	$(1)_2=1$	$(1)_2=1$	×	×
12	2	$(10)_2=2$	$(00)_2=0$	12+2–0=14	12-(4-(2–0)))=10
7	1	$(0)_2=0$	$(1)_2=1$	7+0–1=6	7-(2-(0–1)))=4
18	3	$(101)_2=5$	$(010)_2=2$	18+5–2=21	18-(8-(5–2)))=13 13+8=21
8	2	$(11)_2=3$	$(00)_2=0$	8+3–0=11 11–4=7	8-(4-(3–0)))=7
−12	2	$(00)_2=0$	$(00)_2=0$	×	×

and each gene on a chromosome corresponds to a coefficient that needs to be changed during the embedding process.

$$\mathrm{P} = p_1 p_2 p_3, \ldots, p_{L_c} \quad (p_i \in \{0, 1\}, 1 \le i \le L_c) \tag{3}$$

In which, L_c is equal to the number of coefficients selected for embedding and needs to be changed. Each gene on a chromosome (p_i) can have values 0 or 1. If p_i is equal to 0, Formula (1) will be used for embedding and if it is equal to 1, Formula (2) will be selected.

If N_{pop} represents the number of chromosomes in the population of each generation, N_{pop} chromosomes must be generated randomly at the beginning. The fitness value of each chromosome is then calculated. The fitness function is selected as PSNR. In order to calculate the fitness of each chromosome, first, according to chromosome gene values, the embedding of secret bits will be conducted in DCT coefficients and then according to the resultant stego and cover image, PSNR value will be calculated and used as fitness value for the relevant chromosome. Formulas (4) and (5) indicate the calculation equations. Where *cover* represents the cover image and M and N are its dimensions. $stego_P$ represents the resultant stego image of chromosome P.

$$Fitness(P) = 20 * log_{10} \frac{255}{\sqrt[2]{MSE(P)}} \tag{4}$$

$$MSE\ (P) = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{(cover(i,j) - stego_P(i,j))^2}{M * N} \tag{5}$$

In each iteration of the GA, crossover and mutation stages are performed. Each of these operations has an event probability that is specified as the input to the algorithm. If P_c indicates the probability of crossover operation, $(P_c \times N_{pop})/2$ crossover operations must be performed on each iteration of the GA. In each crossover operation, two members of the current population are selected as parents, and from their combination, two new children are created. The fitness function value of each child is then calculated. In GA-Shield, a combination of three different operators including single-point crossover, two-point crossover, and uniform crossover is used. Then one of the crossover operators is selected by "Roulette Wheel Selection "in each step. If P_m indicates the probability of mutation, $P_m \times N_{pop}$ mutation(s) will be performed in each iteration of the algorithm. In each mutation operation, a random gene from the chromosome is inverted. If the value of the selected gene is zero, it becomes one, and vice versa. Then, the fitness function value will be calculated for resultant chromosomes as well.

After the reproduction of children and mutated populations, an approach should be considered to combine them with the initial population and choose members for creating a new generation population. The number of members is equal to the number of initial population members. There are various mechanisms for this operation, too. In the proposed algorithm "merge, sort, and remove "method will be used. In this approach, first, three intended populations will be combined. Then, they will be sorted in descending order so that members with the most fitness function value will be placed at the beginning of the population. Finally, the same number of members as the initial population will be selected from the beginning of the ordered population. Therefore, the best members with the highest value of fitness function will be chosen in every step. The condition for the end of the algorithm is to

repeat it a predetermined number of times (*epoch*). At the end of the algorithm, the best chromosome, the one with the highest value of the fitness function, is selected.

3.5 Generation of stego image

In this step, the final stego image must be generated using the output of the GA (Algorithm 4). Initially, all coefficients that were not selected for embedding are placed in the corresponding location in the Stego image. Next, the following steps should be applied to all selected coefficients:

1. According to the size of the selected DCT coefficient, an adequate number of message bits are chosen.
2. According to the data bits and previous description, if the selected coefficient does not need any modification, this coefficient is placed in the corresponding location in the stego image and step 1 is repeated for the next coefficient.
3. The corresponding gene of this coefficient is examined on the output chromosome of the GA. If its value is 0, Formula (1), otherwise, Formula (2) is used to embed the desired data in the selected coefficient and the alternative result is calculated. This value is then placed in the corresponding location in the stego image.
4. If the resultant coefficient after embedding is not placed in the intended range, it will be added to or subtracted from 2^l such that it leads to being at the intended range, in which l is the number of used bits for embedding. If the resultant value is still not within the intended range, the first amount from the mentioned range will be chosen, in which the corresponding LSB is equal to the determined message bits and the previous result in the stego image will be replaced by it.

Considering that the quantized DCT coefficients were extracted from the cover image and the embedding steps were performed in these coefficients, so far the quantized DCT coefficients of the stego image have been obtained and the stego image must be generated in the end. First, dequantization and Inverse DCT (IDCT) should be applied on 8×8 blocks, and then the stego image should be reconstructed by placing these blocks with each other.

3.6 Extraction algorithm

Each embedding algorithm requires an extraction algorithm that the receiver uses to extract the message. The block diagram of this algorithm is shown in Fig. 3a. The receiver needs three sets of information to execute the extraction algorithm; the stego image, the key used in the embedding step, and the embedded data length. Using this information and executing the extraction algorithm, the receiver can fully extract the embedded data. Figure 3b shows the steps of the extraction algorithm.

Since the data is embedded within the data in quantized DCT coefficients of the cover image, the receiver must divide the stego image into 8×8 blocks. Next, DCT will be applied to corresponding values in each block and resultant coefficients will be quantized. The set of these three steps is called the preprocessing step (Fig. 3b).

The purpose of the second step is to identify the location of the coefficients that the sender used to embed. If these coefficients are not identified correctly, complete data extraction is not possible. In the GA-Shield method, the PRNG function was used to distribute the data in the

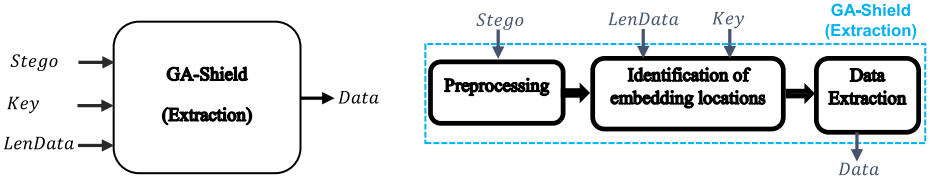

Fig. 3 GA-Shield algorithm. **a** Inputs and outputs, **b** steps of extraction algorithm

image, and thus the receiver could fully identify the location of the coefficients by knowing the initial seed of the PRNG function (*Key*). Each coefficient that was selected using this function must be checked for the data to be embedded in it. If this coefficient is a DC coefficient with a value other than 0, 1, and − 1, one or more data bits will be embedded in this coefficient and must be extracted in the next step. Otherwise, this coefficient should be ignored and go to the next identified coefficient by the PRNG function. This process continues until the receiver has identified all the coefficients containing the data based on the knowledge of the length of the data.

In the last step to extract the message from the selected coefficients, the following operation is repeated for each coefficient:

According to the value of the desired coefficient and the interval listed in Table 2, the number of bits embedded in the coefficient is determined. If $c_i^{'}$ is the desired DCT coefficient from the stego image and l is the number of bits embedded in the corresponding coefficient, Formula (6) can be used to extract the message bits. $m_i^{'}$ represents the decimal equivalent of data.

$$m_i^{'} \leftarrow mod\left(c_i^{'}, 2^l\right) \tag{6}$$

After repeating the above steps for all selected coefficients, an intact hidden message will be extracted.

3.7 An example of the proposed method

After an explanation of different steps of the proposed approach, in Fig. 4 an example is presented for better understanding in this section. In this figure different steps of the proposed method to embed data in an 8 × 8 dimension image (an image containing one block) are shown. Firstly, DCT transform and quantization have been applied to the image block. Then embedding order of various coefficients has been specified based on a PRNG function. Coefficients with values 1, 0, and − 1 would not be used for embedding. According to the value of the coefficients eligible for embedding, the number of bits of message data for embedding and their corresponding decimal equivalent would be determined. As an example, the first coefficient which is used for embedding is in the 11_{th} index and its corresponding value is 9. According to Table 1, two bits of data should be embedded in this coefficient. Therefore, the first two bits of message data including the "10" value should be used for embedding. Since the current value of this coefficient is "01" which is different from the related bits of embedded data, it needs to be changed. Generally, among 13 coefficients that are eligible for embedding, 10 of them should be altered. Thus, chromosomes containing 10 genes (with values 0 or 1) are needed in the genetic algorithm and the response of the algorithm would determine which formula should be used to maximize the corresponding PSNR of the stego image for changing every coefficient. For instance, *BestSolution* which is

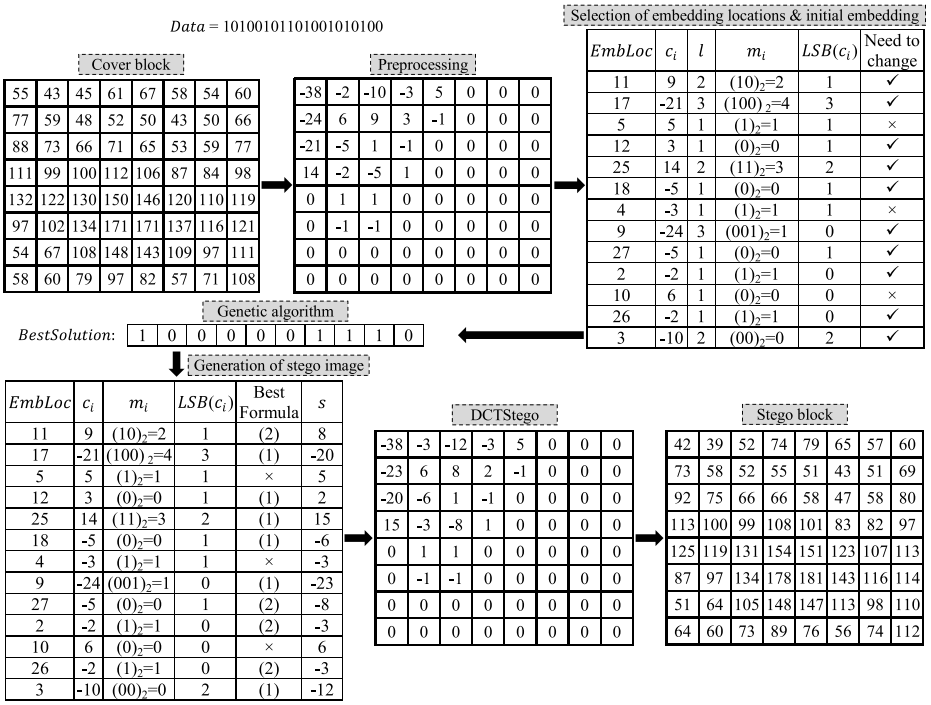

Fig. 4 An example of GA-Shield algorithm

resulted from the genetic algorithm is 1 which demonstrates that Formula 2 should be used for changing the corresponding coefficient located in the 11th index (in case the response was equivalent to 0, then we should use the first formula). Consequently, the value of the coefficient after altering would be changed to 8. After obtaining new values for all ten coefficients which are required to change, these values would be replaced with the previous ones. Ultimately, dequantization and inverse DCT would be applied to the intended block to generate the stego image.

4 Analysis and evaluation

The proposed method was implemented in MATLAB 2016. There are different criteria for comparing steganography methods, which can be divided into three main categories; stego image quality, embedding capacity, and attack resistance. In addition to GA-Shield (our proposed method), other methods used for the comparison are Jsteg (as a basic method in the field of DCT), GPM1 [47] (as a GA-based method in the field of DCT), Shield [6] (as an adaptive multi-bit embedding method in the DCT coefficients), and three newer methods including DCT-M3 [4], MPS [39], MedFreq [24] and ADCT_LC [37]. The test results provided for all methods are the average of 200 test images via http://lear.inrialpes.fr, which contains multiple JPEG images of nature, objects, water, and fire. The obtained color images had different dimensions and were converted into grayscale images before the embedding algorithm was implemented. The JPEG toolbox is used to access the DCT coefficients of the images. Random secret data with different lengths were used to perform various tests. As

previously explained, the embedding algorithm of the proposed method requires four input parameters to run the GA. The results presented in this section are obtained using the following values for these parameters.

$$N_{pop} = 30, P_c = 0.7, P_m = 0.2, epoch = 50$$

4.1 The effect of interval on the performance of the proposed method

To describe the proposed method, Table 2 was presented to determine the number of bits that can be embedded in the coefficients. In this section, in order to investigate the effect of this interval on the performance of the proposed method, three different modes were investigated. In the first mode, the GA-Shield$_1$, the capacity of the coefficients from 1 to 5 bits was selected based on its value and according to Table 2. In the second mode, the GA-Shield$_2$, the capacity of the coefficients is 1–3 bits, and Table 4 was used to classify quantized DCT coefficients. In the third mode, the GA-Shield$_3$, the embedding capacity of all non-zero coefficients, was considered to be 2 bits and the coefficient value had no effect on determining the capacity.

For the initial test, four test images were selected (Fig. 5). Also, the PSNR of stego images obtained by performing three different modes of the proposed method is presented in Table 5. Evaluation of the results showed that among these three methods, the GA-Shield$_3$, which did not consider the size of the coefficients in the embedding process, was weaker than the other two methods. The GA-Shield$_1$ and GA-Shield$_2$ methods had almost the same performance at low embedding percentages, but as expected, with the embedding percentage increased, the GA-Shield$_2$ method was superior in most cases. The GA-Shield$_2$ method embeds a maximum of 3 bits per coefficient, so the range of coefficient variation is less than that in the GA-Shield$_1$ method. Consequently, the GA-Shield$_2$ method produces a stego image with a higher PSNR. Collectively, only GA-Shield$_1$ and GA-Shield$_2$ methods were used in the subsequent comparisons.

4.2 Quality of the stego image

There are several criteria to compare the quality of stego images. The MSE and PSNR criteria were explained in Formulas (4) and (5). MSE represents the mean square error, and the error refers to the difference between the cover and stego images. The closer the MSE value to zero, the lower the error, while larger PSNR values indicate a better image quality. SSIM shows the structural similarity of the stego image to the cover image. Structural information refers to the interdependence of pixels, especially in the case of pixels that are very close together. Ideally, this value is close to 1.

To investigate the effect of data length on the quality of stego images, data with different lengths from 300 bits to 18,000 bits are embedded in four sample images, and stego image quality parameters are measured. Figure 6 shows PSNR and SSIM changes due to embedding data of different lengths in four test images Lena, Baboon, Barbara, and Peppers. Examination

Table 4 The capacity of coefficients according to their size in the GA-Shield$_2$

Quantized DCT coefficients	2–31	32–63	≥ 64
Number of bits	1	2	3

Fig. 5 Four test images

of the diagrams in Fig. 6 shows that stego image quality is directly related to the length of embedded data in it. Fewer changes are made to the image by embedding smaller data, and the stego image has better quality. As the length of the data increases, this quality decreases.

The average of MSE, PSNR, and SSIM criteria for the proposed method and test methods at three embedding levels of 0.001, 0.01, and 0.05 are presented in Table 6. The Jsteg method has been successful in producing high-quality stego images due to a maximum of 1 unit change in each DCT coefficient, and the DCT-M3 method [4] due to the embedding of only 2 bits per block. The proposed method, using the optimization algorithm to minimize the necessary changes to embed data, is more successful than all test methods and is superior to them in all three criteria. Notably, as the percentage of embedding increases, this superiority becomes more apparent.

In another test, the quality of the stego image in the proposed method was compared with some of the most successful reversible steganograpgy methods for JPEG images. The similarity of these two areas is the importance of the stego image quality, but there are also differences in their requirements. Table 7 shows the PSNR for Lena, Baboon, Barbara, and Lena for the four methods [15, 16, 44, 45] and the proposed method at four embedding levels. These results indicate the superiority of the proposed method in the PSNR criterion. In addition, by increasing the embedded data length, the superiority of the PSNR criterion obtained from the proposed method becomes more apparent than the previous methods.

4.3 Hiding capacity

Each steganography method has a Maximum Hiding Capacity (MHC). MHC can be expressed in three different units; bits, bpnz, and bpc. MHC in bits is equal to the maximum number of bits that can be embedded in the image using this method. Another way is to express MHC in bpnz, which expresses the ratio of the maximum number of embedded bits to the number of

Table 5 The PSNR value on the four test images and three modes of the proposed method

Hiding Capacity$_{(bpnz)}$	Method	Lena	Barbara	Baboon	Peppers
0.001	GA-Shield$_1$	67.75	64.65	64.58	66.32
	GA-Shield$_2$	67.92	65.83	65.74	67.85
	GA-Shield$_3$	65.54	62.85	62.89	65.75
0.01	GA-Shield$_1$	58.94	56.67	56.50	56.96
	GA-Shield$_2$	60.33	58.38	58.30	61.84
	GA-Shield$_3$	55.31	55.25	55.23	55.46
0.05	GA-Shield$_1$	56.36	52.95	52.89	51.54
	GA-Shield$_2$	56.75	54.18	54.11	56.05
	GA-Shield$_3$	52.96	50.34	50.22	51.78

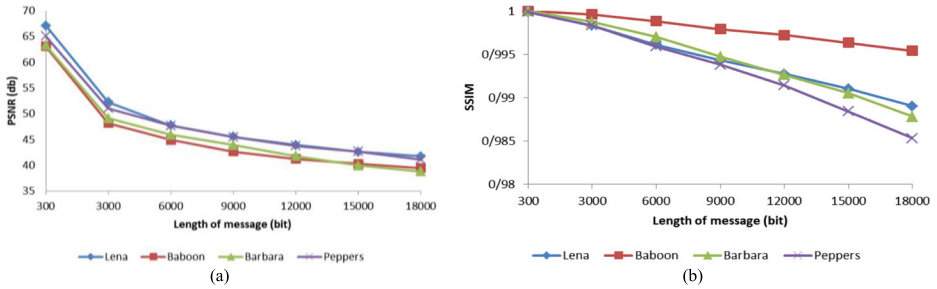

Fig. 6 Dependence of the stego quality on the length of message (**a**) PSNR (**b**) MSE

non-zero DCT coefficients of the image. The third method is the expression of MHC in terms of bpc, which is the result of dividing the maximum number of embedded bits by the number of DCT coefficients of the image.

To compare the maximum embedding capacity of the proposed methods and previous methods, four sample images with dimensions of 512 × 512 (Fig. 5) were used. In addition, the average of MHC was calculated for 200 sample images, and the results of this test are provided in Table 8. For example, the Jsteg method has the ability to embed 29,522 bits in the Lena image. This image has 262,144 DCT coefficients, 41,184 of which are non-zero. Hence, the MHC of the Lena image is 0.71 bpnz or 0.11 bpc.

Examination of the presented results in Table 8 showed that the DCT-M3 [4] method had the lowest capacity, while the MedFreq [24] had the highest capacity. This outcome, based on their embedding algorithm, was expected though. DCT-M3 [4], MPS [39], and MedFreq [24] methods had a similar embedding capacity in images with the same dimensions, but other methods had variable capacities, relative to the image characteristics. The MHC resulting from the Shield algorithm was equal to GA-Shield$_1$. The Jsteg method did not embed in 0 and 1 coefficients, nor in −1, and since a large number of coefficients had the values of 0, 1, and − 1, the MHC of the proposed method was less than the Jsteg method. According to Tables 8 and 9, the capacity of the GA-Shield$_2$ method was less than GA-Shield$_1$. It should be noted that, according to the algorithm of the proposed method, it is possible to embed in coefficients 1 and − 1, but the security of the method is lower.

Notably, according to ongoing methodologies for steganalysis, it is impossible to use the maximum capacity of steganography approaches. These types of methods will be easily detected by existing attacks. Thus, steganography techniques should try to use as high a

Table 6 The average intended metric values resulting from implementing the method on the dataset

	0.001_{bpnz}			0.01_{bpnz}			0.05_{bpnz}		
Method	PSNR	MSE	SSIM	PSNR	MSE	SSIM	PSNR	MSE	SSIM
Jsteg	63.79	0.0319	0.9998	57.01	0.1336	0.9989	50.10	0.2531	0.9947
GPM1 [48]	63.82	0.0320	0.9998	57.55	0.1298	0.9991	50.85	0.2298	0.9950
Shield [6]	66.32	0.0189	0.9999	51.90	0.1018	0.9981	44.91	0.2076	0.9908
DCT-M3 [4]	65.90	0.0182	0.9996	56.09	0.1626	0.9986	×	×	×
MPS [39]	63.36	0.0313	0.9997	52.24	0.3941	0.9954	45.46	1.8697	0.9804
MedFreq [24]	59.20	0.0855	0.9990	48.79	0.9198	0.9690	41.80	4.6148	0.8776
ADCT_LC [37]	65.55	0.0145	0.9998	54.62	0.2797	0.9984	47.61	1.39	0.9922
GA-Shield$_1$	66.68	0.0165	1.0000	59.01	0.0905	0.9998	55.53	0.2033	0.9995
GA-Shield$_2$	66.70	0.0145	1.0000	59.53	0.0746	0.9998	56.17	0.1617	0.9996

Table 7 The PSNR value on the four test images for four reversible methods and proposed method

Image	Method	Embedding capacity (bits)			
		6000	9000	12000	15000
Baboon	[16]	44.05	41.73	39.54	37.81
	[44]	41.82	39.81	38.43	37.48
	[15]	44.67	42.02	39.89	38.04
	[45]	44.82	42.27	40.25	38.53
	GA-Shield$_2$	44.93	42.68	41.22	40.30
Barbara	[16]	45.99	42.47	39.63	37.04
	[44]	44.87	42.67	40.98	39.61
	[15]	46.59	43.39	40.96	38.69
	[45]	46.77	43.72	41.36	39.63
	GA-Shield$_2$	45.95	43.87	41.82	39.96
Lena	[16]	47.02	44.35	42.08	40.19
	[44]	45.97	43.97	41.83	40.38
	[15]	47.35	44.67	42.65	40.67
	[45]	47.57	45.04	42.88	40.99
	GA-Shield$_2$	47.76	45.56	43.94	42.60
Peppers	[16]	47.00	44.55	42.72	41.21
	[44]	46.94	44.22	42.72	41.44
	[15]	47.58	45.30	43.39	41.75
	[45]	47.82	45.39	43.50	41.78
	GA-Shield$_2$	47.88	45.50	43.80	42.66

capacity rate as possible to make the detection by steganalysis methods more difficult. Therefore, the security of every algorithm is of more importance than its hiding capacity.

4.4 Resistance against attacks

A steganography method produces a stego image by making changes to the cover image. Steganalysis attacks use these changes to detect a method. As the changes increase, the method is more likely to be discovered. One of the numerical parameters indicating the accuracy of each attack is the AUC value, which is the area under the ROC diagram. This area is normalized to a value of 1 for a successful detection method. If the AUC value is closer to 0.5, the attack is less successful and therefore the embedding method is safer. SPAM [34] has been used to compare the security of the proposed methods and previous methods. The AUC value of this attack for different methods and at three embedding levels is presented in Table 9. As an example, the ROC output of the attack for the GA-Shield$_1$ is shown in Fig. 7.

Methods proposed in [4, 24, 39] have a very high probability of discovery due to the choice of embedding locations in different images regardless of the characteristics of that location. In addition, these methods due to embedding in zero DCT coefficients, are easier to discover. The Shield method has low security against attack due to increasing the embedding capacity of coefficients, while the Jsteg method has low security due to changing a large number of coefficients. Conversely, GA-Shield$_1$ and GA-Shield$_2$ methods are much more secure than other methods because by using GA, an attempt has been made to reduce image changes.

As noted, a high embedding capacity has an insignificant role by itself, which can be evident upon attacks. Due to the advancement of attacks, a high embedding rate cannot be

Table 8 The MHC in four sample images and an average of 200 test images

	Lena			Barbara			Baboon			Peppers			Avg.		
	bits	bpnz	bpc	bits	Bpnz	bpc	Bits	Bpnz	bpc	bits	bpnz	bpc	bits	bpnz	bpc
Jsteg	29,522	0.71	0.11	46,370	0.74	0.17	65,077	0.75	0.24	30,174	0.71	0.11	40,614	0.74	0.15
GPM1 [48]	41,184	1	0.16	62,662	1	0.23	86,726	1	0.33	42,498	1	0.16	58,366	1	0.18
Shield [6]	23,977	0.58	0.09	38,836	0.62	0.14	56,710	0.65	0.21	24,008	0.56	0.09	33,810	0.61	0.12
DCT-M3 [4]	8192	0.19	0.03	8192	0.13	0.03	8192	0.09	0.03	8192	0.19	0.03	8192	0.15	0.03
MPS [39]	16,384	0.39	0.06	16,384	0.26	0.06	16,384	0.18	0.06	16,384	0.38	0.06	16,384	0.30	0.06
MedFreq [24]	131,072	3.18	0.50	131,072	2.10	0.50	131,072	1.517	0.5	131,072	3.10	0.50	131,072	2.39	0.50
ADCT_LC [37]	13,748	0.33	0.05	23,208	0.37	0.09	34,307	0.39	0.13	14,134	0.33	0.05	21,349	0.35	008
GA-Shield$_1$	23,977	0.58	0.09	38,836	0.62	0.14	49,254	0.56	0.18	24,008	0.56	0.09	33,810	0.61	0.12
GA-Shield$_2$	18,347	0.44	0.06	31,031	0.49	0.11	43,926	0.51	0.16	18,100	0.42	0.06	27,025	0.49	0.10

Table 9 AUC results for all three approaches with different hiding capacities

Hiding Capacity$_{(bpnz)}$	Jsteg	GPM1 [48]	Shield [6]	DCT-M3 [4]	MPS [39]	MedFreq [24]	ADCT_LC [37]	GA-Shield$_1$	GA-Shield$_2$
0.001	0.5608	0.5325	0.5444	0.7004	0.7029	0.9435	0.5228	0.5161	0.5112
0.01	0.7969	0.7558	0.8508	0.9344	0.9583	1	0.7001	0.6061	0.6226
0.05	0.9152	0.8829	0.9624	×	0.9891	1	0.7601	0.6703	0.6950

expected from the steganography method and these rates are not practical as the method can be easily detected by attacks.

It is necessary to mention that the passive warden scenario is considered in the proposed approach. Therefore, although resistance against manipulation attacks (including cropping, scaling, etc) is not considered a significant factor for the evaluation of steganography methods [3], according to the adaptive characteristics of the proposed method and spreading the hidden data through the whole image, modification of stego image would affect the error in the extraction of embedded data and therefore low resistance of the algorithm against manipulation attacks.

4.5 Comparison with other state-of-the-art methods

In the previous sections, the performance of the proposed method was evaluated from different aspects, and comparisons were made with successful and new embedding methods in DCT coefficients at the same embedding levels. Given that the proposed method is a frequency-domain method, in the final test a number of frequency-domain methods with different transformations whose embedding level range is close to the proposed method have been selected and the PSNR range of these methods are listed in Table 10. The proposed method is an adaptive method and therefore the maximum embedding capacity for various images is different and the average maximum hiding capacity of the proposed method is mentioned in the Table 10. Examination of these results shows that the proposed method is completely superior to [11, 13, 28] methods. However, the [14] method, which is one of the newest embedding methods in DCT coefficients, has a higher embedding capacity than our proposed method, but the PSNR range of this method shows that at low embedding levels, stego images resulted from the proposed method have better quality. Due to the existence of strong steganalysis methods and sensitivity to high embedding levels, so our proposed method is more practical than the [14] method.

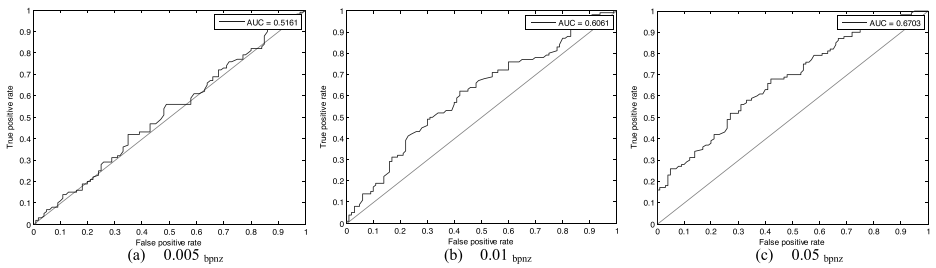

Fig. 7 The ROC diagram resulting from implementing the attack on the GA-Shield$_1$

Table 10 Comparison of the proposed method with existing methods

Algorithm	Embedding domain	Capacity (bit)	PSNR
Chen et al. [11]	DWT	2048–16384	44.90–44.20
Evsutin et al. [13]	DFT	2048–40960	48.03–36.87
Melman et al. [28]	DCT	10000–50000	39–31.5
Evsutin et al. [14]	DCT	300–100000	57.69–36.29
Our algorithm	DCT	300–20,000(adaptive)	65–40

5 Conclusion

Image steganography with data embedding in DCT coefficients is one of the most attractive areas of information hiding. In this paper, a new method is proposed that uses optimization algorithms to embed in these coefficients. In the proposed algorithm, embedding is performed into the LSBs of quantized DCT coefficients with absolute values greater than or equal to 2 and different number of bits are used for embedding, based on the coefficient value. The proposed method uses a genetic algorithm to select a new value for the coefficients that need to be changed. Comparison results show that the proposed method generates stego images with high PSNR (65–40 db) and SSIM (≥0.985), and it is more successful than the existing methods in generating high-quality stego images in most cases. The detection accuracy of the proposed method using SPAM attacks at different embedding capacities is less than 70%. Although the maximum embedding capacity of the proposed method is less than that of some the existing methods, the proposed method is practically possible; because it has good security against existing attacks as well as appropriate quality of stego image, and it provides an image steganography solution with a short embedding key.

Funding No funding was received to assist with the preparation of this manuscript.

Data availability Not applicable.

Code availability Not applicable.

Declarations

Conflict of interest The authors have no financial or proprietary interests in any material discussed in this article.

References

1. Abdulla AA, Jassim SA, Sellahewa H (2013) Secure steganography technique based on bitplane indexes. In: Proc. - 2013 IEEE Int Symp Multimedia, ISM 2013, pp. 287–291. https://doi.org/10.1109/ISM.2013.55
2. Abdulla AA, Sellahewa H, Jassim SA (2014) Steganography based on pixel intensity value decomposition. 9120:19–27. https://doi.org/10.1117/12.2050518
3. Abdulla AA, Sellahewa H, Jassim SA (2019) Improving embedding efficiency for digital steganography by exploiting similarities between secret and cover images. Multimed Tools Appl 78(13):17799–17823. https://doi.org/10.1007/S11042-019-7166-7

4. Attaby AA, Mursi Ahmed MFM, Alsammak AK (2018) Data hiding inside JPEG images with high resistance to steganalysis using a novel technique: DCT-M3. Ain Shams Eng J 9(4):1965–1974. https://doi.org/10.1016/J.ASEJ.2017.02.003
5. Banharnsakun A (2018) Artificial bee colony approach for enhancing LSB based image steganography. Multimed Tools Appl:77(20). https://doi.org/10.1007/s11042-018-5933-5.
6. Bansal D, Chhikara R (2014) An improved DCT based steganography technique. Int J Comput Appl 102(14):46–49. https://doi.org/10.5120/17887-8861
7. Baziyad M, Rabie T, Kamel I (2021) Toward stronger energy compaction for high capacity dct-based steganography: a region-growing approach. Multimed Tools Appl 80(6):8611–8637. https://doi.org/10.1007/S11042-020-10008-2
8. Bhattacharyya S, Khan A, Sanyal G (2014) DCT Difference Modulation(DCTDM) Image Steganography. Int J Inf Netw Secur 3(1)40–63. Accessed 23 Dec 2021. [Online]. Available: http://iaesjournal.com/online/index.php/IJINS
9. Biswas R, Bandyapadhay SK (2019) Random selection based GA optimization in 2D-DCT domain color image steganography. undefined 79(11–12):7101–7120. https://doi.org/10.1007/S11042-019-08497-X
10. Chang CC, Chen TS, Chung LZ (2002) A steganographic method based upon JPEG and quantization table modification. Inf Sci (Ny) 141(1–2):123–138. https://doi.org/10.1016/S0020-0255(01)00194-3
11. Chen ST, Huang HN, Kung WM, Hsu CY (2015) Optimization-based image watermarking with integrated quantization embedding in the wavelet-domain. undefined 75(10):5493–5511. https://doi.org/10.1007/S11042-015-2522-8
12. Eggers JJ, Baeuml R, Girod B (2002) Communications approach to image steganography. In: Proc. SPIE 4675, security and watermarking of multimedia contents IV. https://doi.org/10.1117/12.465284
13. Evsutin O, Kokurina A, Meshcheryakov R, Shumskaya O (2018) The adaptive algorithm of information unmistakable embedding into digital images based on the discrete Fourier transformation. Multimed Tools Appl 2018 7721 77(21):28567–28599. https://doi.org/10.1007/S11042-018-6055-9
14. Evsutin O, Melman A, Meshcheryakov R (2021) Algorithm of error-free information embedding into the DCT domain of digital images based on the QIM method using adaptive masking of distortions. Signal Process 179:107811. https://doi.org/10.1016/J.SIGPRO.2020.107811
15. Hou D, Wang H, Zhang W, Yu N (2018) Reversible data hiding in JPEG image based on DCT frequency and block selection. Signal Process 148:41–47. https://doi.org/10.1016/J.SIGPRO.2018.02.002
16. Huang F, Qu X, Kim HJ, Huang J (2016) Reversible data hiding in JPEG images. IEEE Trans Circuits Syst Video Technol 26(9):1610–1621. https://doi.org/10.1109/TCSVT.2015.2473235
17. Hussain M, Wahab AWA, Bin Idris YI, Ho ATS, Jung KH (2018) Image steganography in spatial domain: A survey. Signal Process Image Commun:65. https://doi.org/10.1016/j.image.2018.03.012.
18. Jiang C, Pang Y, Xiong S (2013) A high capacity Steganographic method based on quantization table modification and F5 algorithm. Circuits, Syst Signal Process 33(5):1611–1626. https://doi.org/10.1007/S00034-013-9703-3
19. Kadhim IJ, Premaratne P, Vial PJ (2020) Improved image steganography based on super-pixel and coefficient-plane-selection. Signal Process:171. https://doi.org/10.1016/j.sigpro.2020.107481.
20. Kanan HR, Nazeri B (2014) A novel image steganography scheme with high embedding capacity and tunable visual image quality based on a genetic algorithm. Expert Syst Appl 41(14):6123–6130. https://doi.org/10.1016/J.ESWA.2014.04.022
21. Kaur A, Kaur R, Kumar N (2016) Image steganography using Discrete Wavelet Transformation and Artificial Bee Colony Optimization. In: Proc. 2015 1st Int. Conf Next Gener Comput Technol NGCT 2015, pp 990–994. https://doi.org/10.1109/NGCT.2015.7375269
22. Khamrui A, Gupta DD, Ghosh S, Nandy S (2017) A spatial domain image authentication technique using genetic algorithm. Commun Comput Inf Sci 776:577–584. https://doi.org/10.1007/978-981-10-6430-2_45
23. Khan S, Bianchi T (2018) Ant colony optimization (ACO) based data hiding in image complex region. Int J Electr Comput Eng 8(1). https://doi.org/10.11591/ijece.v8i1.pp379-389
24. Khan S et al (2019) On hiding secret information in medium frequency DCT components using least significant bits steganography. C – Comput Model Eng Sci 118(3):529–546. https://doi.org/10.31614/CMES.2019.06179
25. Kumar KS, Raja KB, Chhotaray RK, Pattanaik S (2010) Bit length replacement steganography based on DCT coefficients. Int J Eng Sci Technol 2(8):3561–3570
26. Lima R, Gramacho W, Henrique A (2017) Optimizing image steganography using particle swarm optimization algorithm. Int J Comput Appl 164(7):1–5. https://doi.org/10.5120/ijca2017913686
27. McAteer I, Ibrahim A, Zheng G, Yang W, Valli C (2019) Integration of biometrics and steganography: a comprehensive review. Technol 7(2):34. https://doi.org/10.3390/TECHNOLOGIES7020034
28. Melman A, Petrov P, Shelupanov A (2020) An adaptive algorithm for embedding information into compressed JPEG images using the QIM method. Accessed 23 Dec 2021. [Online]. Available: https://arxiv.org/abs/2012.08742v1
29. Miri A, Faez K (2017) Adaptive image steganography based on transform domain via genetic algorithm. Optik (Stuttg) vol. 145. https://doi.org/10.1016/j.ijleo.2017.07.043

30. MK S, MK S (2018) An image steganography using particle swarm optimization and transform domain. Int J Eng Technol 7(2.24):474–477. https://doi.org/10.14419/ijet.v7i2.24.12139
31. Nipanikar SI, Hima Deepthi V, Kulkarni N (2018) A sparse representation based image steganography using Particle Swarm Optimization and wavelet transform. Alexandria Eng J 57(4). https://doi.org/10.1016/j.aej.2017.09.005
32. Noda H, Niimi M, Kawaguchi E (2006) High-performance JPEG steganography using quantization index modulation in DCT domain. Pattern Recogn Lett 27(5):455–461. https://doi.org/10.1016/J.PATREC.2005.09.008
33. Patel H, Dave P (2012) Steganography technique based on DCT coefficients. Int J Eng Res Appl 2(1):713–717. http://www.ijera.com
34. Pevný T, Bas P, Fridrich J (2010) Steganalysis by subtractive pixel adjacency matrix. IEEE Trans Inf Forensics Secur 5(2). https://doi.org/10.1109/TIFS.2010.2045842
35. Pramanik S, Singh RP, Ghosh R (2020) Application of bi-orthogonal wavelet transform and genetic algorithm in image steganography. Multimed Tools Appl 79(25–26). https://doi.org/10.1007/s11042-020-08676-1
36. Roy R, Laha S (2015) Optimization of Stego image retaining secret information using genetic algorithm with 8-connected PSNR. Procedia Comput Sci 60(1):468–477. https://doi.org/10.1016/J.PROCS.2015.08.168
37. Sabeti V, Ahmadi S (2020) Adaptive image steganography in the difference value of discrete cosine transform coefficients. J Soft Comput Inf Technol 9(3):55–66
38. Saidi M, Hermassi H, Rhouma R, Belghith S (2017) A new adaptive image steganography scheme based on DCT and chaotic map. Multimed Tools Appl 76(11):13493–13510. https://doi.org/10.1007/S11042-016-3722-6
39. Sajid Ansari A, Sajid Mohammadi M, Tanvir Parvez M (2017) JPEG image steganography based on coefficients selection and partition. undefined 9(6):14–22. https://doi.org/10.5815/IJIGSP.2017.06.02
40. Shah PD, Bichkar RS (2018) A secure spatial domain image steganography using genetic algorithm and linear congruential generator. Adv Intell Syst Comput 632:119–129. https://doi.org/10.1007/978-981-10-5520-1_12
41. Tang W, Li B, Barni M, Li J, Huang J (2021) Improving cost learning for JPEG steganography by exploiting JPEG domain knowledge. IEEE Trans Circuits Syst Video Technol. https://doi.org/10.1109/TCSVT.2021.3115600
42. Wang RZ, Lin CF, Lin JC (2001) Image hiding by optimal LSB substitution and genetic algorithm. Pattern Recogn 34(3):671–683. https://doi.org/10.1016/S0031-3203(00)00015-7
43. Wazirali R, Alasmary W, Mahmoud MMEA, Alhindi A (2019) An optimized steganography hiding capacity and imperceptibly using genetic algorithms. IEEE Access:7. https://doi.org/10.1109/ACCESS.2019.2941440
44. Wedaj FT, Kim S, Kim HJ, Huang F (2017) Improved reversible data hiding in JPEG images based on new coefficient selection strategy. Eurasip J Image Vid Process 2017(1):1–11. https://doi.org/10.1186/S13640-017-0206-1/TABLES/6
45. Xiao M, Li X, Ma B, Zhang X, Zhao Y (2021) Efficient reversible data hiding for JPEG images with multiple histograms modification. IEEE Trans Circuits Syst Vid Technol 31(7):2535–2546. https://doi.org/10.1109/TCSVT.2020.3027391
46. Xie J, Yang C, Huang D, Xie D (2008) A large capacity blind information hiding algorithm. Proc Int Symp Electron Commer Secur ISECS 2008:934–937. https://doi.org/10.1109/ISECS.2008.130
47. Yu L, Zhao Y, Ni R, Zhu Z (2009) PM1 steganography in JPEG images using genetic algorithm. Soft Comput 4(13):393–400. https://doi.org/10.1007/S00500-008-0327-7
48. Yu L, Zhao Y, Ni R, Zhu Z (2009) PM1 steganography in JPEG images using genetic algorithm. Soft Computing 13(4). https://doi.org/10.1007/s00500-008-0327-7