International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:11, 2008

Embedding a Large Amount of Information Using High Secure Neural Based Steganography Algorithm

Nameer N. EL-Emam

***Abstract*—**In this paper, we construct and implement a new Steganography algorithm based on learning system to hide a large amount of information into color BMP image. We have used adaptive image filtering and adaptive non-uniform image segmentation with bits replacement on the appropriate pixels. These pixels are selected randomly rather than sequentially by using new concept defined by main cases with sub cases for each byte in one pixel. According to the steps of design, we have been concluded 16 main cases with their sub cases that covere all aspects of the input information into color bitmap image. High security layers have been proposed through four layers of security to make it difficult to break the encryption of the input information and confuse steganalysis too. Learning system has been introduces at the fourth layer of security through neural network. This layer is used to increase the difficulties of the statistical attacks. Our results against statistical and visual attacks are discussed before and after using the learning system and we make comparison with the previous Steganography algorithm. We show that our algorithm can embed efficiently a large amount of information that has been reached to 75% of the image size (replace 18 bits for each pixel as a maximum) with high quality of the output.

***Keywords*—**Adaptive image segmentation, hiding with high capacity, hiding with high security, neural networks, Steganography.

I. Introduction

STEGANOGRAPHY literally means "covered message" and involves transmitting secret messages through seemingly innocuous files. The goal is that not only the message remains hidden, but also that a hidden message is even sent. There are many tools available that can hide messages in images, audio and video files. Steganography is now in common use while cryptography has been the preferred tool for sending secret messages, relying on complex ciphers to prevent detection in the huge bandwidth of the Internet which offers an alternative or complementary approach. Steganography supports hiding messages amongst the huge volume of Internet traffic, in media files where the addition of a hidden message is difficult to detect with the human eye even if the file is viewed [1].

Both approaches can be combined to produce better protection for the information, in this case, when the steganography fails and the message can not be detected if a cryptography technique is used too. Hiding information inside images is a popular technique nowadays.

To hide a message inside an image without changing its visible properties, the cover source can be altered in "noisy" areas with many colour variations, so less attention will be drawn to the modifications.

The most common methods to make these alterations involve the usage of the least-significant bit (LSB) developed by Chandramouli *et al.* [2], masking, filtering and transformations on the cover image. Dumitrescu *et al.* [3], construct an algorithm for detecting LSB Steganography. Von Ahn *et al.* [4] are the information theoretic scientists; they construct mathematical frameworks based on complex-theoretic view to seek the limits of steganography. Other construction is used by HweeHwa Pang [5]; his scheme used hash value obtained from a file name and password and a position of header of hidden file is located. This approach is used by the present work with new modifications. The next interesting application of steganography is developed by Miroslav Dobsicek [6], where the content is encrypted with one key and can be decrypted with several other keys, the relative entropy between encrypt and one specific decrypt key corresponds to the amount of information. Because of the continual changes at the cutting edge of steganography and the large amount of information involved, steganalysists have suggested using machine learning techniques to characterize images as suspicious or non- suspicious developed by Mittal *et al.* [7]. Pavan *et al.* [8] used entropy based technique for detecting the suitable areas in the document image where information can be embedded with minimum distortion. C. Zhang *et al*. [13] hides indirectly the secured binary bits along with some selected graphical image bits, based on the neural network algorithm, to get cipher bits. This approach is used by the present work. When using a 24 bit colour image, a bit of each of the red, green and blue colour components can be used, so a total of 3 bits can be stored in each pixel. Thus, an 800 × 600 pixel image can contain a total amount of 1.440.000 bits (180.000 bytes) of secret information. But using just 3 bit from this huge size of bytes is wasting in size. So the main objective of the present work is how to insert more than one bit at each byte in one pixel of the cover-image and give us results like the LSB [2], [3] (message to be imperceptible). This objective is satisfied by building new steganography

Manuscript received March 1, 2007.

Nameer N. EL-Emam is with Applied Computer Science Department, Faculty of Information Technology, Philadelphia University, Jordan (e-mail: Nemam@philadelphia.edu.jo).

algorithm based on an intelligent system to hide large amount of any type of information through bitmap image[4], [6] by using maximum number of bits per byte at each pixel. We discuss two kinds of attacks to be sure that our process for embedded information is worked well. The first discussion concerns to work against visual attacks [7], [9] to make the ability of humans is unclearly discern between noise and visual patterns, and the second discussion concerns to work against statistical attacks[3], [8], [10], [11] to make it much difficult to automate.

II. New Steganography Algorithm with Four Layers of Security

New Steganography algorithm by using four layers of security has been constructed. These layers are developed from the previous works [1], [10] to acquire high security and they work independently to provide unbreakable security wall Fig. 1.

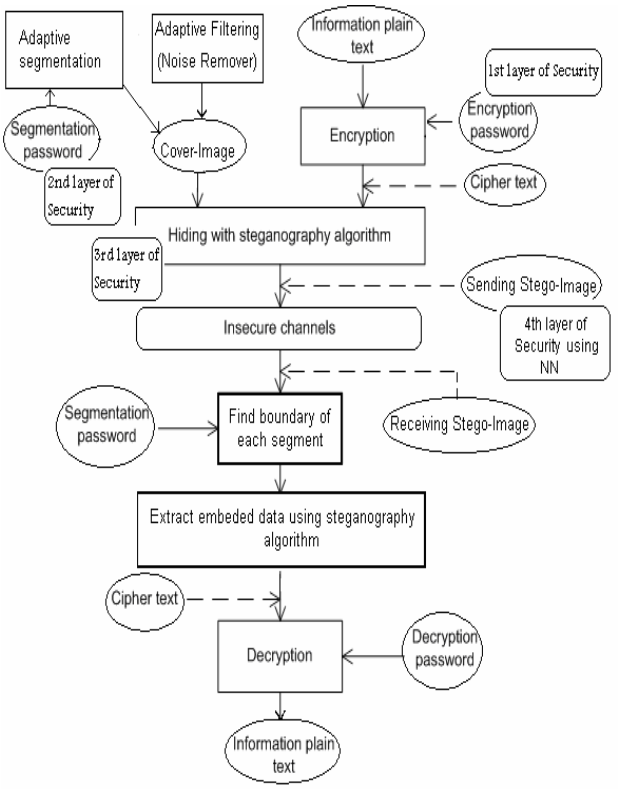

Fig. 1 Steganography with security layers

Before describing the present algorithm, we need to show the following approaches:

A. Encryption Approach

The encryption mechanism is the first layer of security for information protection by using AES algorithm [1], [5]. This layer is used to encrypt the password before hiding input information and it gives a powerful Steganography algorithm.

B. Adaptive Segmentation Approach

We introduce new concept of image segmentation by using adaptive segment to support the second layer of security. The cover-image from type bitmap is segmented into random number of non-uniform segments according to the value of password or any other key that is provided by the information owner Fig. 2. At section five of this work, we have shown how to calculate a number of segments by using non-uniform segmentation which is more secure than uniform segments [2], [4], [5] to carry the input information. In addition, to avoid sending files of this enormous size, a compression scheme has been employed on BMP Stego image which is known as *lossless* compression.

Fig. 2 Adaptive segmentation on a cover-image using uniform or non-uniform segmentations

C. Pixel Selection Approach

This approach is the third layer of security as shown in Fig. 1. We perform random selection of the proper byte at each pixel of the cover-image according to the color characteristic to embed secret information. This type of selection is implemented by using new concepts which are called main cases and sub cases that have been described in the next section. These new concepts are powerful in the sense of reducing the noise of a stego-image.

III. Steganography Algorithm Approach

The present Steganography algorithm has two parts (information hiding at the sender side and information extracting at the receiver side). These parts are constructed and implemented to satisfy the following requirements:

a) The algorithm must reduce the chances of statistical detection.
b) The algorithm must provide robustness against a variety of image manipulation attacks.
c) The stego-image must not have any distortion artifacts.
d) The algorithm must not sacrifice embedding capacity in order to achieve the above requirements.

The first part is used to hide information file inside bitmap according to the following actions:

a) Accept encryption password from the sender.
b) Find a maximum size (number of bytes) that is accepted by the cover-image.
c) Perform compressions on secret information file to increase the amount of hiding secret information.
d) Perform encryption on secret information file.

e) Perform the following processes on the cover-image:
   1) Adaptive segmentation according to the password.
   2) Adaptive filtering (noise remover) by using Minimum Mean-Square Error (*MMSE*) [2] as in (1) to reduce number of colors at the cover-image.

$$\underset{\forall i,j}{MMSE} = C_{ij} - \frac{\sigma_n^2}{\sigma_l^2}\left(C_{ij} - m_{ij}^{l}\right). \quad (1)$$

Where
C_{ij} = color byte at the Cover-image
σ_n^2 = noise variance.
σ_l^2 = local variance (in the segment under consideration).
m^l = local mean (average in segments under consideration) .

f) Perform scanning to select suitable pixels on each segment by extracting image characteristics. The candidate pixels are used to embed secret information.
g) Perform hiding of the secret information into bitmap images according to color characteristics.

The second part is used to extract information from bitmap image at the receiver side in conformity with the following actions:
a) Extracting password.
b) Scanning segment's pixels according to the password.
c) Extracting information file.
d) Decrypt the extracting information file by using the password.

The present Steganography algorithm is implemented on colour image with pixel size 24– bits (3 bytes of *RGB* colors), each byte is separated into two nibbles (four bits). The left nibble contains the highest value in the byte while the right nibble contains the lowest value in the byte. Therefore, any changes in the right nibble will cause a minimal change in a byte value. We can specify the byte effective according to the left nibble value. The nibble value is fixed by the interval [0, 15], so that we conclude that we have 16 levels of priority, each one represents one main case (*MC*) out of 16. In the present work we use (2) to determine the index of *MC* that must be implemented to perform hiding in a bitmap cover-image.

$$MC = Int\left(\frac{ByteColor}{16}\right) + 1. \quad (2)$$

where $ByteColor \in \{ByteRed, ByteGreen, ByteBlue\}$ represents the value of color in decimal notation. To hide large amount of information, all three colors of one pixel are checked by the present Steganography algorithm to perform hiding information by depending on new concept which is called sub-case (*SC*). This concept is used to organize the pixel architect. We have defined six *SCs* that are specified according to the following definition:

Let us define $MC^{color} = index$, where $1 \le index \le 16$ and $color \in \{R, G, B\}$ where *R, G, B* are pixel's color equal to *ByteRed, ByteGreen* and *ByteBlue* respectively. Assume that the *MC* of the current color is *C* and *X,Y* are the *MCs* of the rest colors at the same pixel, then we can define the index of *SC* according to the following conditions:

$$SC\_Index = \begin{cases} 1 \Leftrightarrow X, Y < C \\ 2 \Leftrightarrow (X = C \,\&\, Y < C) | (X < C \,\&\, Y = C) \\ 3 \Leftrightarrow X, Y = C \\ 4 \Leftrightarrow (X > C \,\&\, Y < C) | (X < C \,\&\, Y > C) \\ 5 \Leftrightarrow (X > C \,\&\, Y = C) | (X = C \,\&\, Y > C) \\ 6 \Leftrightarrow X, Y > C \end{cases} \quad (3)$$

To embed information in the proper pixels, it is necessary to use the priority value *Pr(MC)* which is calculated as follows:

$$Pr(MC) = |MC - 17| \quad (4)$$

The priority level depends on the subscript of MC as in the following formula:

$$Pr(MC_i) > Pr(MC_{i+1}) \quad \forall i \ni 1 \le i \le 15 \quad (5)$$

Before we describe the present Steganography algorithm, let us define the following concepts:

A. Type of Pixels

Assume we have the following notations.
a) *CP* is the Current pixel that includes the set of colors *{R, G, B}*.
b) NP is the next pixel of the CP.

B. Pixels Properties

a) Selected color:

$$Sel_{color} = \underset{\forall color \in CP}{arg\,min}(MC_{color}) \quad (6)$$

b) Main case of the selected color in the current pixel:

$$Cmc = MC_{Sel_{color}} \ni color \in CP \quad (7)$$

c) Main case of the rest colors in the current pixel:

$$MC_{RC} = \underset{\forall Color \in RC}{argmin}(MC_{Color}) \ni Color \in CP \,\&\, RC \subset CP \quad (8)$$

d) One of the rest colors:

$$E = RC_i \quad \forall i = 1,2 \quad \& \quad RC_i \in RC \quad (9)$$

e) Select one of the rest colors in the current pixel whose main case is greater than main case of the selected color (Sel_c) :

$$Pr(MC_H) < Pr(MC_{Sel_C}) \ni H \in RC, C \notin RC \quad (10)$$

f) R, G and B are the red, green, and blue colors in one pixel respectively.

C. Properties of Embedding Secret Message

Create Object (*DataBits*) which holds information about the embedded message (the content and the length of the secret message).

D. Properties of MC

Assume that every *MC* has a number of *SCs*, (see Table I). Table I illustrates the *MC* with their *SCs*. Practically, we conclude that SC_2 and SC_4 are neglected from any *MC*. This restriction is used to avoid sudden change of the color in the stego-image.

TABLE I
MC WITH THEIR CORRESPONDING SC

MC	*SC*	*SC* Description
1	$\{SC_3, SC_5, SC_6\}$	$\{\forall\ SC\ \ \exists\ C = Sel_{Color} \in CP\ \ \&\ RC_i \in CP\ \ \forall\ i = 1,2\ \ \ni\ C \le RC_i\}$
2-15	$\{SC_1, SC_3, SC_5, SC_6\}$	$\{\forall\ SC\ \ \exists\ C = Sel_{Color} \in CP\ \ \&\ RC_i \in CP\ \ \forall\ i = 1,2\ \ \ni\ C \le RC_i \mid C \ge RC_i\}$
16	$\{SC_1, SC_3\}$	$\{\forall\ SC\ \ \exists\ C = Sel_{Color} \in CP\ \ \&\ RC_i \in CP\ \ \forall\ i = 1,2\ \ \ni\ C \ge RC_i\}$

The following pseudo code shows step by step how the present steganography algorithm can select the suitable pixel efficiently to embed information to be imperceptible from both visual and statistical attacks.

```
// Read information file bits and, Perform Searching on all MC for each segment //starting from MC=1,…16
Foreach(MCi, i=1,…,16 ) {
  Foreach(segment in the Cover-image){
   Foreach(Sel(Color) ∈ Cmc)  {
     If (Cmc >= 2 AND SC=1)  {
// Check the MC of rest colors
   If ( RC = 1)  {
// Check if all current pixel property is equal to the next pixel property
   If ( CP.property = NP.property ) {
  // Hide 2 information bits in the selected color from the current pixel
     Hide(CP.C, DataBits(2));
  // Hide 2 information bits in the selected color byte from the next pixel
 Hide(NP.C, DataBits(2));    }   }   }
Else if (Cmc >= 1 AND Cmc <= 16 AND SC=3 ) {
// Hide 1 information bit in the Red color byte
   Hide(CP.R, DataBits(1) );
// Hide 1 information bits in the Green color byte
   Hide( CP.G, DataBits(1) );
// Hide 2 information bits in the Blue color byte
   Hide( CP.B, DataBits(2) );  }
Else if(Cmc >= 1 AND Cmc <= 15 AND SC=5 ) {
// Hide 2 information bits in the selected color byte from the current pixel
   Hide( CP.C, DataBits(2) );
// Hide information bits in the greater or equal MC byte for the selected color min //case from the rest pixel bytes
   Hide( CP.E, DataBits(2) );   }
Else if(Cmc >= 1 AND CP.Cmc <= 15 AND SC=6 ) {
// Hide 2 information bits in the selected color byte from the current pixel
   Hide( CP.C, DataBits(2) );
// Hide information bits in the greater or equal MC byte for the selected color min //case from the rest pixel byte's
   Hide( CP.H, DataBits(2) );  }
  }// end foreach Sel(color)
 } //end foreach segment
}// end foreach MC
```

IV. Neural Networks Learning System

We use a learning system through a neural network that includes (n-p-n) Perceptron layers architecture.that is it has n neurons in the first (input layer), p neurons in the second (hidden layer) and n neurons in the third (output layer) with full connection Fig. 3.

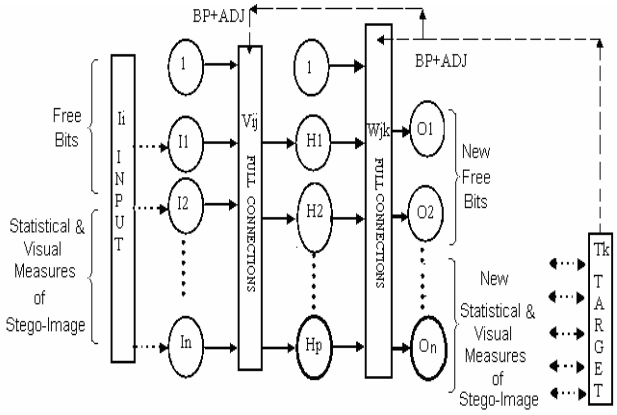

Fig. 3 A Multi-Layered Perceptron (n-p-n) Neural Networks

In Fig. 3, the solid arrow means many to one or one to many transition, whereas dotted arrow refers to one to one transition and dashed arrow shows the send action for adjustment process. We use back-propagation algorithm with adaptive neural network to apply training through three stages: the feedforword of the input training pattern, the back-propagation of the associated error, and the adjustment of the weights. In addition, we add adaptive smoothing error ASE [12] to speed up training process. The main objective of learning system is to add additional complexity for the statistical and visual attacks as in Fig. 4. The neural based Steganography training algorithm is shown through the following steps:

Step1: Input: Cover image and Secret message.
Step2: Implement the present Steganography algorithm to hide a secret message and produce a stego-image.

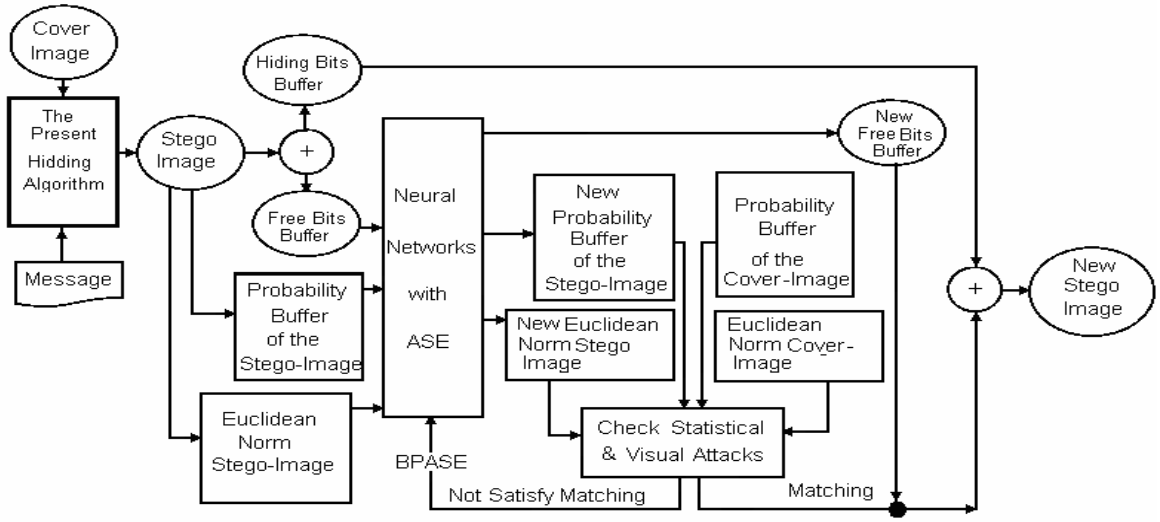

Fig. 4 Neural based Steganography training system architecture

Step3: Find statistical measures (Chi-square and probabilities) and visual measure (Euclidian) for each stego and cover images.

Step4: Extract all bits with their locations which are not used by the present steganography algorithm and saved into temporary buffer which is called free bits buffer.

Step5: Use neural networks with back propagation algorithm and adaptive smoothing error BPASE. The input layer contains (statistical and visual measures for stego-image and free bits buffer) while the output layer produces (new free bits buffer and new statistical and visual measures).

Step6: Check matching between statistical-visual measures of stego image and statistical-visual measures of cover images. If matching is satisfied, then build new setgo image by adding new free bits buffer to stego bits buffer, else adjust (ADJ) weights values Vij, W_{jk}. Fig. 3, then goto step 5 (Back propagation BP).

During feedforward, each input neuron I_i $(\forall i = 1,...,n)$ receives an input signal {free bits and probabilities of stego-image} and broadcasts this signal to each of the hidden neuron H_j $(\forall j=1,...,p)$ through weight of full connections V_{ij}. See (11).

$$H_j = \Phi\left(\sum_{i=0}^{n} I_i V_{ij}\right), \quad I_0 = 1 \tag{11}$$

Each hidden neuron computes its activation and sends its signal H_j to each of the output neurons O_k $(\forall k=1,...,n)$ through weight of full connection W_{jk}. Each output neuron O_k computes its activation to form the response of the net for the input pattern as in (12).

$$O_k = \Phi\left(\sum_{j=0}^{p} H_j W_{jk}\right), \quad H_0 = 1 \tag{12}$$

All activation functions that are used in our learning system is Bipolar sigmoid as in (13) with the range [1, -1] and the first derivative of this function is define at (14).

$$\Phi(\xi) = \frac{1 - e^{-\xi}}{1 + e^{-\xi}} \tag{13}$$

$$\Phi_x(\xi) = \frac{1 - \Phi^2(\xi)}{2} \tag{14}$$

During the training, the set of output neuron represents the {new free bits buffer, new probabilities buffer of stego image, and new Euclidian norm of stego image}. The new probabilities are used to check matching with probabilities of cover image t_k as in (15), using δ_k $k=s,..., n$ to compute the distribution error at the output neuron O_k where n-s+1 is equal to the size of the probabilities buffer of stego image.

$$\delta_k = (t_k - O_k)\ \Phi_O\left(\sum_{j=0}^{p} H_j W_{jk}\right) \tag{15}$$

where β is damping parameter at the interval [0,1] and in a same manner, the factor δ_j $(\forall j=1,..., p)$ has been computed for each hidden neuron H_j. The adjustment to the weight W_{jk} as in (16) is based on the factor δ_k and the activation h_j of the hidden neuron H_j.

$$\Delta W_{jk}^{new} = \beta\alpha_{jk}^{new}\delta_k H_j + (1-\beta).\Delta W_{jk}^{old} \tag{16}$$

The first derivative of the error factor the error factors are defined as in (17-18) respectively.

$$\delta_j' = \sum_{k=1}^{m} \delta_k W_{jk} \quad (17)$$

$$\delta_j = \delta_j' \ \Phi_h(\sum_{i=0}^{n} I_i V_{ij}) \quad (18)$$

The adjustment to the weight V_{ij} from input neuron I_i to hidden neuron H_j is based on the factor δ_j and the activation of the input neuron as in (19).

$$\Delta V_{ij}^{new} = \beta\ \alpha\delta_j I_i + (1-\beta)\Delta V_{ij}^{old} \quad (19)$$

The update processes on the weight function are defined in (20) and (21).

$$W_{jk}^{new} = W_{jk}^{old} + \Delta W_{jk} \quad (20)$$

$$V_{ij}^{new} = V_{ij}^{old} + \Delta V_{ij} \quad (21)$$

where β is a damping parameter in the interval $0 < \beta < 1$. In the present work, we select β=0.1 and using adaptive learning rate *AL* [12] to improve the speed of training by changing the rate of learning α during training process see (22).

$$\alpha_{jk}^{new} = \begin{cases} \alpha_{jk}^{old} + K & if \quad \Delta W_{jk}^{new}\Delta W_{jk}^{old} > 0 \\ (1-\gamma)\alpha_{jk}^{old} & if \quad \Delta W_{jk}^{new}\Delta W_{jk}^{old} < 0 \\ \alpha_{jk}^{old} & otherwise \end{cases} \quad (22)$$

We found that the suitable values of parameters K and γ that are used in this work are equal to 0.02 and 0.9 respectively. We repeat training process many times and update the old values of the two dimensional arrays V and W until satisfy the convergence criteria condition (satisfy matching) given in (23):

$$\max_k \left| t_k - O_k \right| < 10^{-6}, \qquad k = s,..., n \quad (23)$$

Finally, we introduce Adaptive Smoothing Error (ASE) to improve training process [12].

V. Implementation Approach of Steganography Algorithm

Assume that we have a cover-image which contains three types of *MC: MC1, MC2* and MC_6 and we have three types of pixels: MC_1 with SC_3,, MC_2 with SC_5 and MC_6 with SC_1. Now, we try to hide 2-bytes 01010101, 01010101. Before we perform hiding, we must compute the number of segments in the Cover-image through the following steps:

a) Let S be a number of characters of the input password (*PS*) and it represents the number of segments at each direction. That is

$$S = \left| PS \right| \quad (24)$$

b) Find the size of non-uniform segments for both vertical and horizontal directions *(VSeg ,HSeg)* by using the following formulas:

$$VSeg_i = \frac{Val(PS_i) * h(image)}{Total} \quad \forall i = 1,..., S \quad (25)$$

$$HSeg_{S-i+1} = \frac{Val(PS_i) * w(image)}{Total} \quad \forall i = 1, ..., S \quad (26)$$

where, $Val\ (PS_i)$ represents the decimal value of the i^{th} character at the *PS*. The *h(image), w(image)* are the height and the width of the cover image respectively and *Total* is define as the follows:

$$Total = \sum_{j=1}^{S} Val(PS_j) \quad (27)$$

c) Perform non-uniform segmentation on the cover-image into (*S* x *S)* segments, then using column wise searching on all segments. The present algorithm performs hiding into each segment separately according to row wise scanning of segment's pixels see Fig. 5.

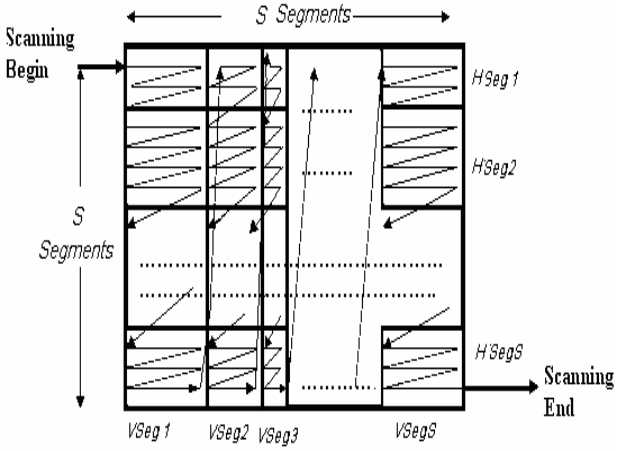

Fig. 5 Scanning pixels on the adaptive image's segments

The algorithm starts hiding information inside the highest priority $Pr(MC_1)$, assume that *CP* includes the following values of colors (*R*=15,*G*=0,*B*=13) and the *MC* of *CP* is MC_1 with SC_3. In the present Steganography algorithm, we can hide the information "01010101", as follows:

We assign the left most bit of the input information into the right most bit of the *R*-byte, the bit before the left most bit is assigned to the right most bit of the *G*-byte and we assign the

rest two bits (the right nibble of input information) into the right most two bits of the *B*-byte as it is shown in the Fig. 6a.

After hiding information in all the MC_1, it will start hiding inside the next highest priority which is MC_2 in this example. Let us assume that the first pixel from MC_2 was in SC_5 and its value was of colors (*R=16, G=20, B=90*), then it will hide the information as follows. The least 2 bit will be modified in the current selected color and in the color which its *MC* equal to the current selected color *MC* as shown in the Fig. 6b.

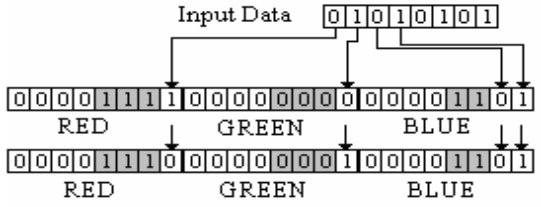

Fig. 6a Information Hiding Using MC 1 with SC 3

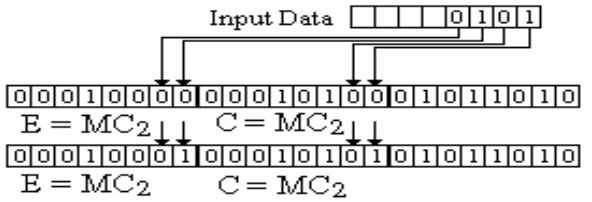

Fig. 6b Information Hiding Using MC_2 with SC_5

After hiding the information in all, the MC_2 will start hiding inside the next high priority, in this example it is MC_6. Let us say that the first pixel from MC_6 was in SC_1 and its value was (17, 21, 90), it will hide the information as follows. The least 2 bit will be modified in the current selected color for the current selected pixel and in the current selected color for the next pixel as it is shown in Fig. 6c.

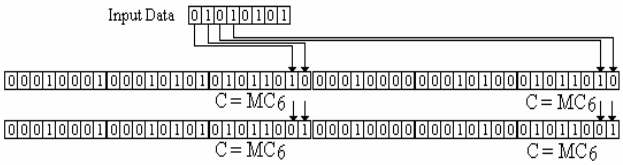

Fig. 6c Information Hiding Using MC_6 with SC_1

VI. Results and Discussion

Confidence in the present results is gained by comparising of the results obtained from the present Steganography algorithm with those previously published in the literatures [1]-[3], [5], [6], [9], [11].

As we mentioned before, specifications of a good Steganography algorithms on a bitmap image depend on the following points:

a) Comparison between the present work and the previous work.
b) A Large amount of hiding information (Avoid Statistical attack): using one cover-image as a carrier to hold a maximum number of bytes instead of a sequence of carriers.
c) Human vision scale (Avoid visual attack): Steganography message can be embedded into digital images in ways that are imperceptible to the human eye. In other words, a stego-image that is generated by the present algorithm has to be normal for human vision and cannot be detected.

A. Comparison with Previous Work

We use more than 50 BMP images to test the present algorithm and to be sure that the aim of information embedding is satisfied. In this work, it appears that the efficiency of embedding information is very high when we perform comparison with previous works. Fig. 7 and Fig. 8 show the comparison results with S-Tools algorithm [1]. The results illustrate the level of efficiency according to the following concepts:

1) The amount of hidden information.
2) The amount of noise detections.

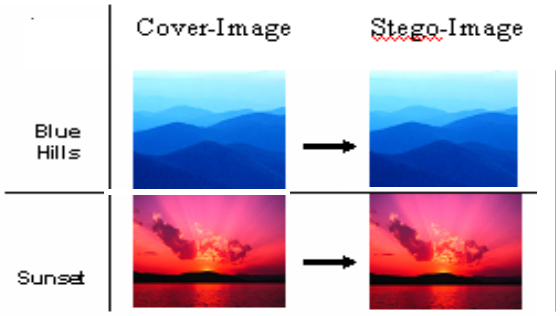

Fig. 7 *Blue hills* and *Sunset* bitmap images before and after information hiding

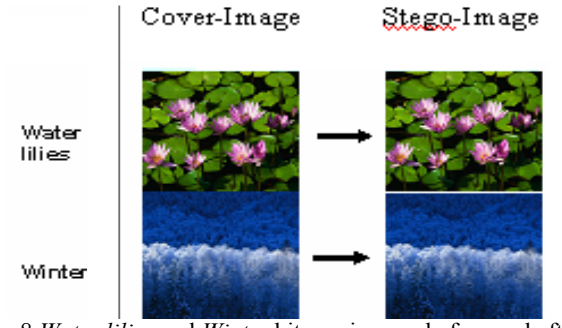

Fig. 8 *Water lilies* and *Winter* bitmap images before and after information hiding

Fig. 9 illustrates the amount of the hidden information on selected images from Fig. 7-8 with fix size (400x400) pixels.

It appears that the present work can hide a large amount of information and it exceeds the capability of the S-Tools. We should emphasize that the "*Blue Hills*" image in Fig. 7 can hide the maximum amount of information while the "*Water* lilies" image in Fig. 8 is in the minimum. This variety of image's capabilities to hide bytes depends on the color distribution and the sequence of the input characters.

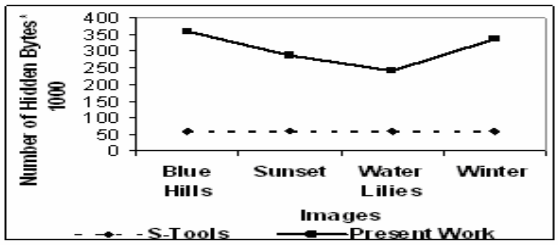

Fig. 9 The amount of hidden byte vs four different bitmap carriers

Fig. 10a and Fig. 10b illustrate the amount of the noise on 4 images by using two kinds of noise detections, the first one is used to find the average of 4 neighbor pixels in 3x3 pixels Fig. 11-left around the pixel P_i as in (28). The second formula is used to find the average of 12 neighbor pixels in the 5x5 pixels Fig. 11-right around the pixel P_i as in (29).

$$Noise_{3x3} = \sum_{i=1}^{mxn} \left(\left| \frac{\left(P_i^S + \sum_{j=1}^{4} P_j^S \right)}{5} - \frac{\left(P_i^C + \sum_{j=1}^{4} P_j^C \right)}{5} \right| \right) \quad (28)$$

$$Noise_{5x5} = \sum_{i=1}^{mxn} \left(\left| \frac{\left(P_i^S + \sum_{j=1}^{12} P_j^S \right)}{13} - \frac{\left(P_i^C + \sum_{j=1}^{12} P_j^C \right)}{13} \right| \right) \quad (29)$$

where P_i^S and P_i^C are the color values of the pixel i in both stego-image and cover-image respectively, and *(mxn)* is a number of pixels in a bitmap image. We find that using neural based stego in the present steganography algorithm can reduce the noise into more than 50% for any type of image.

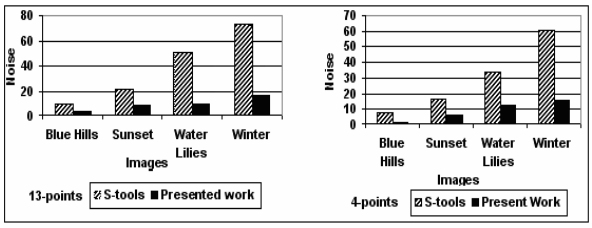

Fig. 10a Noise detection of the present work with fourth layer and previous works vs four deference bitmap carriers

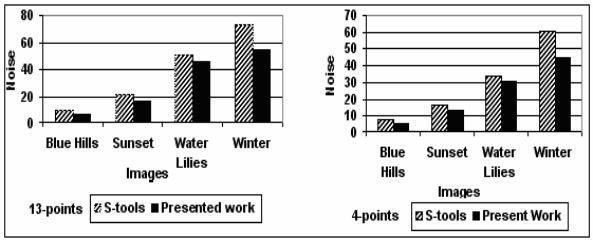

Fig. 10b Noise detection of the present work without fourth layer and previous works vs four deference bitmap carriers

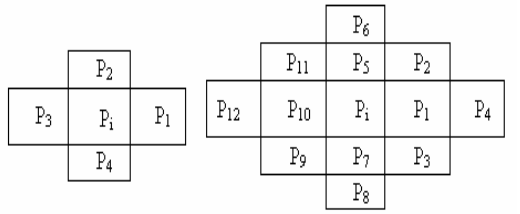

Fig. 11 Noise detections using 4 and 12 neighbor pixels

In Fig. 12a and Fig. 12c we divide a noise frequency into 21 classes, started from the minimum value (-10) to the maximum value (+10) in the selected bitmaps. We calculate the frequency of damage pixels and then is determined the corresponding class for each pixel. We perform sector comparison instead of pixel comparison to show the smoothing of color around a specific pixel; this comparison has been implemented by using (29). It appears that the present work is less noise frequency than S-Tools algorithm for all images. In addition we found that using fourth layer of security is more efficient to damp the frequency of the noise from (-5,+5) into (-2,+2) for both *Sunset* and *Winter* images .

B. Avoids Statistical Attack

We use three kinds of testing in this section as in the following:

a) Challenge χ^2 attacks.

$$\chi^2_{k-1} = \sum_{i=1}^{k} \frac{(n_i - n_i')^2}{n_i'} \quad (30)$$

where, k-1 is degree of freedom and n_i' is calculated according to (18).

$$n_i' = \frac{\left| Freq \ \ of \ \ color \ \ in \ the \ set \ \{C_{2i}^{\{R,G,B\}}, C_{2i+1}^{\{R,G,B\}}\} \right|}{2} .... \quad (31)$$

where, $\{C_{2i}^{\{R,G,B\}}, C_{2i+1}^{\{R,G,B\}}\}$ is the i-th pair of the palette colors $C_0^{\{R,G,B\}}, C_1^{\{R,G,B\}}, ..., C_{255}^{\{R,G,B\}}$ for each color *{R,G,B}*.

We use mathematical formula as in (32) to express the probability that distributed of n_i' , n_i are equal.

$$n_i = \left| Freq \ \ of \ \ color \ \ accors \ \ at \ \ the \ color \ C_{2i} \right| ..... \quad (32)$$

The probability *P* that a χ^2 value calculated for an experiment with *d* degrees of freedom (where *d=k*-1, one less the number of possible outcomes) is due to chance is:

$$P_{\chi^2, d} = \left(2^{\frac{d}{2}} \Gamma \left(\frac{d}{2} \right) \right)^{-1} \int_{\chi^2}^{\infty} (t)^{\frac{d}{2}-1} e^{-\frac{t}{2}} \, dt \quad (33a)$$

where Gamma Γ is the generalization of the factorial function to real and complex arguments:

$$\Gamma_x = \int_0^{\infty} t^{x-1} e^{-t}\, dt \qquad (33b)$$

The message-carrying pixels in the image are selected randomly rather than sequentially according to *MCs* with their *SCs* concepts. It appears that chi-square attack on stego images with randomly scattered messages produces fluctuating *P* values in the beginning and then, as the sample size increases, the *P* value eventually drops to zero due to the sensitivity of the test. Figs 13a-13h show the comparison between Cover and Stego images, it appears that the present embedded algorithm produce the same behavior of the probability *P* with respect to the length of secret message (40% , 60%, 50%, 70% and 75%) of the image size and any window size using four types of images. In addition, we see that the *p* value eventually drops to zero at the window sizes 20 or 35 depending on the message size. We conclude that stegoanalysis can not detectean embedded information due to the matching of Stego and Cover images curves.

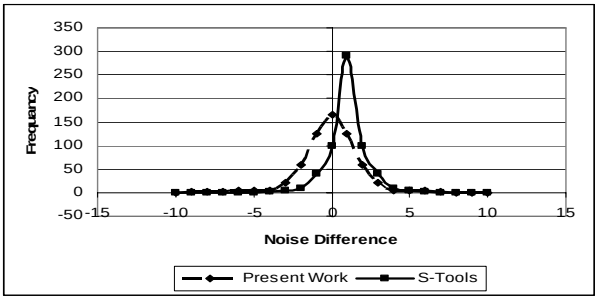

Fig. 12a Frequency of the sector noise for *Sunset* image without fourth layer

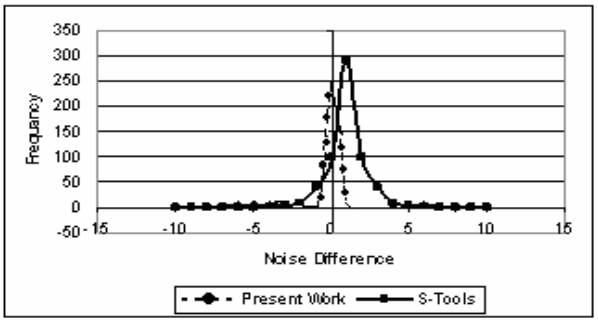

Fig. 12b Frequency of the sector noise for *Sunset* image with fourth layer

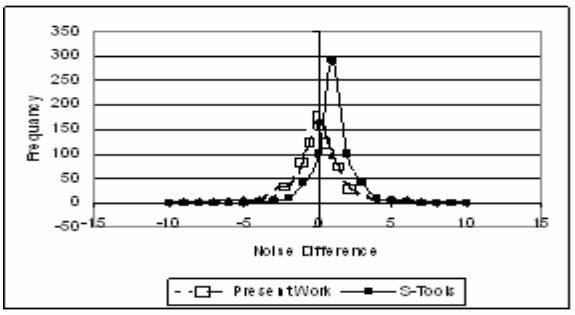

Fig. 12c Frequency of the sector noise for *Winter* image with out fourth layer

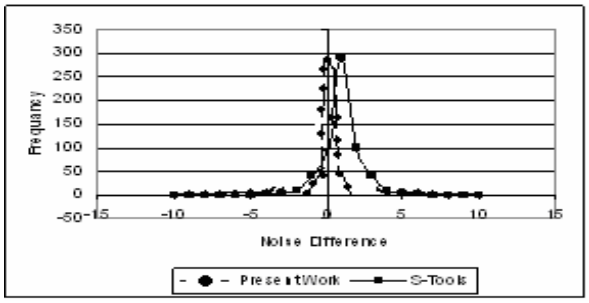

Fig. 12d Frequency of the sector noise for *Winter* image with fourth layer

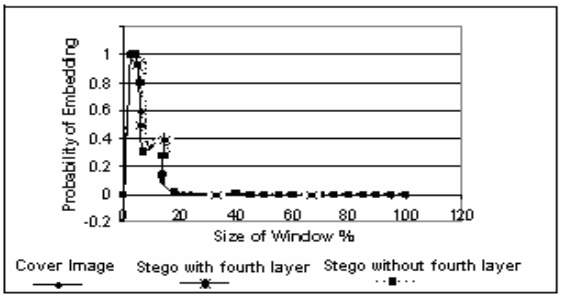

Fig. 13a Probability of embedding for a secret message of length of 40% of *Blue Hills* image size

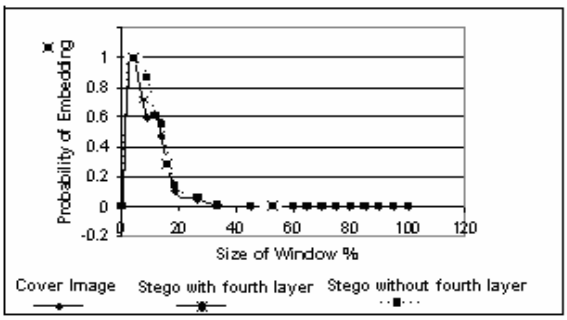

Fig. 13b Probability of embedding for a secret message of length of 40% of *Sunset* image size

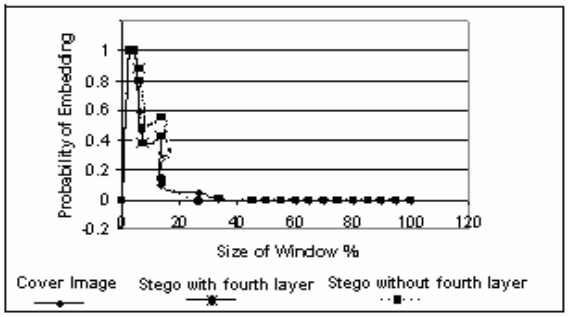

Fig. 13c Probability of embedding for a secret message of length of 75% of *Blue Hills* size

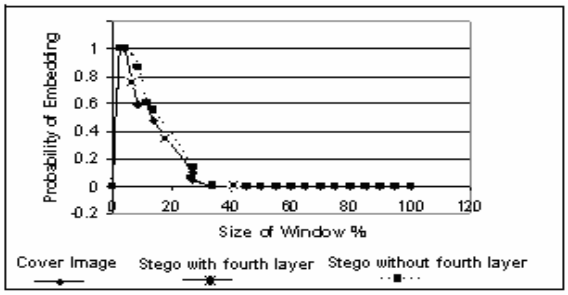

Fig. 13d Probability of embedding for a secret message of length of 60% of *Sunset* size

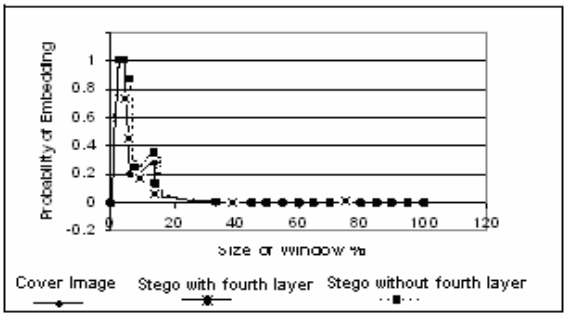

Fig. 13e Probability of embedding for a secret message of length of 40% of *Water lilies* image size

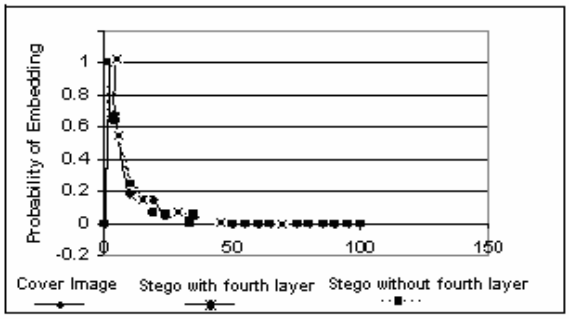

Fig. 13f Probability of embedding for a secret message of length of 40% of *Winter* image size

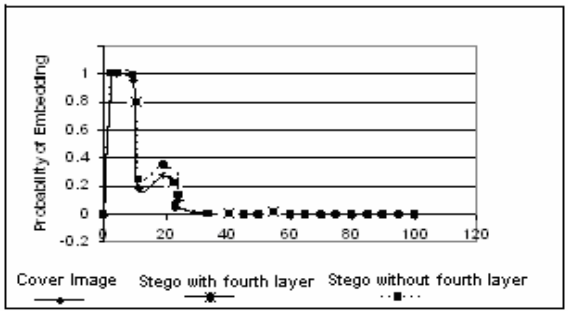

Fig. 13g Probability of embedding for a secret message of length of 50% of *Water lilies* image size

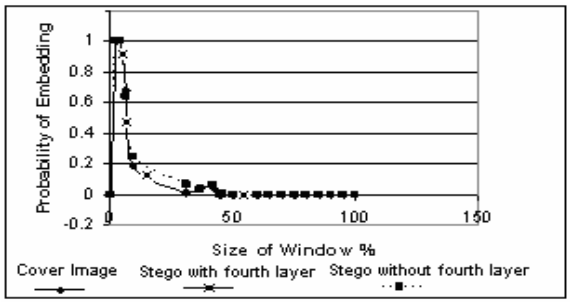

Fig. 13h Probability of embedding for a secret message of length of 70% of *Winter* image size

b) Statistics of the value difference between neighboring pixels.

In this work we calculate the difference of the horizontally neighboring pairs of the image Fig. 14 and we perform the comparison between Cover and Stego images on four types of images according to the following formula:

$$Val_{i,j}^{C} = \left| P_{i,j}^{C} - P_{i,j+1}^{C} \right|$$
$$Val_{i,j}^{S} = \left| P_{i,j}^{S} - P_{i,j+1}^{S} \right| \quad (34)$$
$$\forall\, i = 1,.., (nxm) - 1$$

It appears that all differences values come near to zero for both Cover and Stego images and comparison between them is indeed remarkably similar. We conclude that the smoothing of the color is equivalent for both Stego and Cover images.

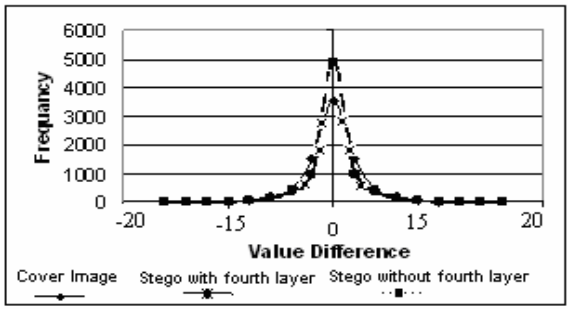

Fig. 14a Difference value on neighboring pixels for both cover and stego *Sunset* images

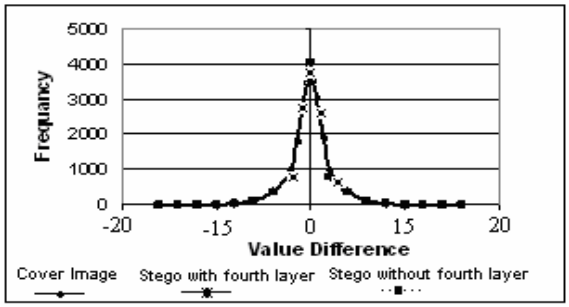

Fig. 14b Difference value on neighboring pixels for both cover and stego *Blue Hills* images

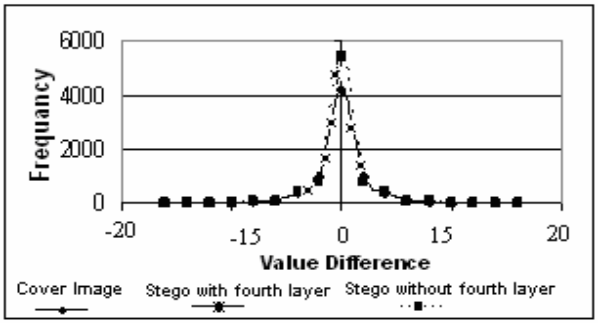

Fig. 14c Difference value on neighboring pixels for both cover and stego *Water lilies* images

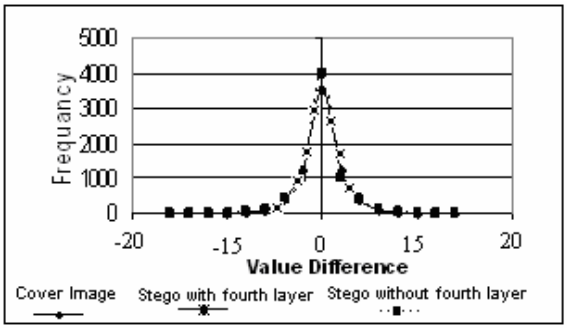

Fig. 14d Difference value on neighboring pixels for both cover and stego *Winter* images

C. Avoids Visual Attack

This section presents two kinds of testing as follows:

a) The amount of Euclidean norm

Fig. 15 shows the amount of the Euclidean norm to compute the distance between the Cover-images and Stego-image present in at Fig. 7a and Fig. 7b. This measure is used to find the set of the closest palette color (in Euclidean norm) as in (35).

$$d=\sqrt{(R_c-R_s)^2+(G_c-G_s)^2+(B_c-B_s)^2} \qquad (35)$$

Where the subscript c means the cover-image and the subscript s means the stego-image.

It appears that *Sunset* image has a maximum norm due to the large area of uniform palette color (Red, Pink), while *Winter* image has a minimum norm due to the small area of the uniform palette colors (Blue or White or Black).

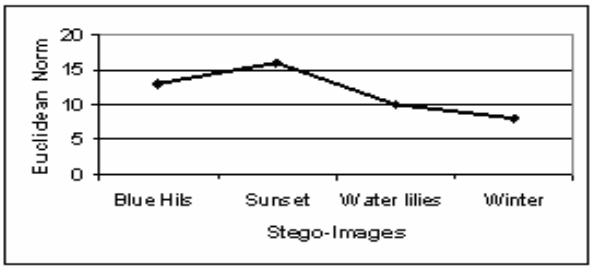

Fig. 15 Euclidean norm evaluations on four stego images

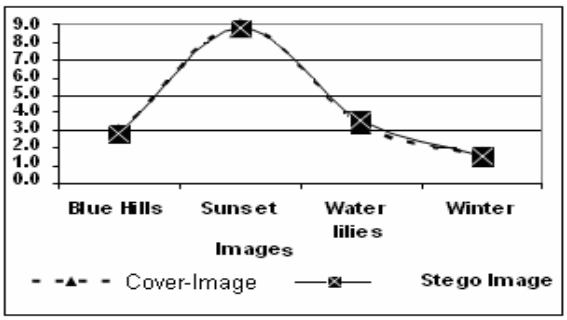

Fig. 16 The length of cover and stego images

b) The amount of brightness information

Fig. 16 shows the amount of the length L, see (23) of the RGB vector for both cover and stego images shown in Fig. 6 and Fig. 7. This measure is necessary to use against visual attack to find the brightness information.

$$L=\sqrt{R^2+G^2+B^2}. \qquad (36)$$

VII. Conclusion

In this work, we satisfy the aim that says Steganography is an effective way to obscure information and hide sensitive information. The present algorithm allows an individual to hide information inside other information with hopes that the transfer medium will be so obscure that no one would ever think to examine the contents of the file. The algorithm which is described by pseudo-code is presented and it is possible to implement a Steganography algorithm to hide a large amount of information into carrier bitmap image.

We used four layers of security by obscuring the context in which it was transferred. With continued research and an improvement in algorithms design, neural based steganography can be taken as a serious way to hide information and the present work appears that it was more efficient than the most familiar algorithm like (S-Tools) [1]. Working against visual and statistical attacks need adaptive algorithm on each step of embedded information. It is found that the present algorithm is attractive and results reached by this algorithm are efficient in the field of embedding information (Steganography). We performed three types of

comparison; the first one was used to compare the present algorithm with S-Tools algorithm through the amount of noise and the amount of size. It appears that the present work was less effective of the noise at the pixels and lager amount of embedded information. The second comparison was made upon the statistical attack; it shows that it was difficult to distinguish between Cover and Stego image when chi square and the difference between neighboring pixels were implemented.

The last comparison was found that visual attack results indicate that using non uniform color was extremely powerful when we have large amount of embedded information.

REFERENCES

[1] S-Tools (http://digitalforensics. champlain. edu/ download/ s-tools 4.zip).
[2] Chandramouli, R. and N. Memon,. "Analysis of LSB based image steganography techniques," Proc. of ICIP, Thessaloniki, Greece, pp. 7-10, Oct. 2001.
[3] Dumitrescu, S., W. Xiaolin and Z. Wang, "Detection of LSB steganography via sample pair analysis," In: LNCS, Springer-Verlag, New York, Vol. 2578/2003, pp: 355–372, 2003.
[4] Ahn, L.V. and N.J. Hopper, "Public-key steganography. In Lecture Notes in Computer Science of Advances in Cryptology," EUROCRYPT 2004, Vol. 3027 / 2004, Springer-Verlag Heidelberg, pp: 323–341, 2004.
[5] Pang, H.H., K.L. Tan and X. Zhou, "Steganographic schemes for file system and b-tree," IEEE Trans. on Knowledge and Data Engineering, Vol. 16, pp.701–713, 2004.
[6] Dobsicek, M., "Extended steganographic system," In: 8th Intl. Student Conf. on Electrical Engineering. FEE CTU. 2004
[7] Mittal, U. and N. Phamdo, "Hybrid digital-analog joint source-channel codes for broadcasting and robust communications," IEEE Trans. on Info. Theory, vol. 48, pp. 1082 –1102, 2002.
[8] Pavan, S., S. Gangadharpalli and V. Sridhar, "Multivariate entropy detector based hybrid image registration algorithm," IEEE Intl. Conf. on Acoustics, Speech and Signal Processing, pp: 18-23, 2005.
[9] Moulin, P. and J.A. O'Sullivan, "Information-theoretic analysis of information hiding," IEEE Trans. on Info. Theory, vol. 49, pp. 563–593, 2003.
[10] Amin, P., N. Liu and K. Subbalakshmi, "Statistically secure digital image data hiding," IEEE Multimedia Signal Processing MMSP05, China, 2005.
[11] Jackson, J., G. Gunsch, R. Claypoole and G. Lamont,. "Detecting novel steganography with an anomaly- based strategy," J. Electr. Imag., Vol. 13, 860– 870, 2004.
[12] Nameer N. EL-Emam, "Reallocation of mesh points in fluid problems using back-propagation algorithm," Information Journal, Vol 9, No. 1, pp 175-184. January 2006.
[13] C. Zhang, H.W. Guesgen, W.K. Yeap "Neural Based Steganography, Lecture note in computer science Computational Intelligence. Neural Networks," LNAI 3157, pp. 429–435, Springer-Verlag Berlin Heidelberg 2004.