An Iterative Method for Minimum Palette-Based Image Steganograph

Jia-Hong Lee

Department of Information Management, Kung Shan Institute of Technology, Tainan, Taiwan
TEL: (06)2727175 ext. 290
email: jhlee@mis.ksit.edu.tw

Mei -Yi Wu

Department of Information Management, Chang Jung University, Tainan, Taiwan
TEL: (06)2780123 ext. 315
email: barbara@mail.cju.edu.t

Abstract

In this paper, an iterative met hod for palette-based image steganography is proposed for minimizing the RMS error between an original image and its stego -image. The method is based on a palette modification scheme which can iteratively embed one message bit into each pixel in a palette-based image. In each iteration, both the cost for removing a palette entry and the benefit for creating a ne one to replace it are calculated. If the maximal benefit is greater than the minimal cost, a palette entry replacement is performed. Experimental results indicate that the proposed method introduces remarkable improvement on reducing distortion to the carrier images (stego -images).

Keyword *: steganography, palette -based images, data hiding, EZ Stego method*

1. Introduction

Data hiding also frequent ly termed "*steganography*" is a method of embedding data into digital media for the purpose of identification, annotation, copyright protection, and secret data delivering, so that the embedded data can travel along with the host media. Data hiding is a hig hly multidisciplinary field that combines image and signal processing with cryptography, communication theory, coding theory, signal compression, and the theory of visual perception.

Depending on what information in which form is hidden in digital images, one can roughly divide data hiding schemes into two major types [3]: non -robust, undetectable data hiding, and robust image watermarking. In the first application, a digital image serves as a container for a secret message. For example, by replacing the l east significant bit of each pixel with an encrypted bit -stream, the changes to a typical image will be imperceptible and the encrypted message will be masked by some innocent looking image. The main applications of this kind of scheme are to transmit secret data [4][5]. In the second application, a short message (a watermark) is embedded in the image in a robust manner. There are many robust techniques such as statistics [6], signal transformation [7], and spread spectrum [8], which can be efficiently appl ied in doing watermarking into digital images. The stego-images generated by these methods own the abilit to survive common image processing operations, such as lossy compression, filtering, noise adding, geometrical transformations, etc.

In this paper, we focus on the research of data hiding in palette-based images such as GIF file format; since this kind of images are the most important container for secret messages on Internet. Palette -based images provide a hostile environment for the steganographer, since the limitation on the available colors imposed by the finite palette makes the process of data hiding a challenge. There are two approaches to embed messages in palette-based images: embedding messages into the palette; and embedding messages into th e image data. Easy to design is the main advantage of the first approach. However, it suffers that the capacity is limited by the palette size, it is impossible to use this way to embed a long message. The second approach has high capacity but it is genera lly difficult to avoid from distortion of the stego-images.

One of the most popular data hiding scheme for palette-based images is called EZ Stego method [1]. In this method, the palette is first sorted by luminance which is a linear combination of three colors R, G, B of palette entry, such as $R + G + B$. In the reordered palette, neighboring palette entries are typically near to each other in the color space, as well. The method then embeds the messages in a binary form into the LSB of indices (pixels) po inting to the palette colors. However, this method works not well in creating high quality stego-images. The main reason is that colors with similar luminance values may be relatively far from each other (e.g. colors [9, 28, 202] and [202,9,28] have the same luminance but represent two extremely different colors).

To avoid this problem, Jiri [2] presents a new method to hide message bits into the parity bit of close colors. The parity bit of the color R, G, B is calculated as $R + G + B$. *mod 2*. For each pi xel of an image, a message bit is embedded by searching the closest colors in the palette till a palette entry with the desired parity bit. Since the parit bits of palette entries corresponding to real images are more or less randomly distributed, this will guarantee that the original colors are not modified too much within the stego-image. However, there exists false contouring and noises in the output stego-images by using Jiri's method, especially to embed data for hand-drawing images (such as cartoon pictures). Since only few colors are used in this kind of images, the color difference between an entry and others will become larger in the embedding process.

In this paper, an iterative method is derived for minimizing the RMS error between the original image and the stego-image. The method will modify both the palette content and image data in the original palette -base image iteratively. Then, we adopt Jiri's method to embed messages into the modified image for achieving a less distorted stego-image. The motivation of the method is to replace the "*less important*" colors in the image with the neighboring color to spare palette entries for some " *more important*" colors. Thus, we can greatly reduce the root-mean-square (RMS) error during the embedding process of Jiri's method. Finally, we compare the original images and their stego-images produced by EZ Stego, Jiri's method and the proposed method using the RMS distance. Experimental results show that the proposed method produces significantly better results.

The rest of the paper is organized as follows. In the next section, we briefly review the EZ Stego and Jiri's methods to data hiding. Section 3 details our new approach to minimum the RMS error between the original image and the stego-image. Section 4 presents experimental results that show that our method performs better than the other two methods. The final section summarizes our work.

2. Previous Work about Data Hiding for Palette-Based Images

2.1 EZ Stego method

EZ stego method is similar to the generally used LSB method for 24 bit color images (or 8 bit grayscale images). After sorting the palette colors by luminance, this method embeds the message in a binary form into the LSB of indices pointing to the palette colors. The detail steps are listed as follows

Step1: Reorder the palette color according to the luminance of each palette entry.
Step2: Find the index of the pixel's RGB color in the reordered palette.
Step3: Get one bit from the embedding binary message and
replace the LSB of the index.
Step4: Find the new RGB color that the index now points to the reordered palette.
Step5: Find the index of the new RGB color in the original
palette.
Step6: Replace the pixel with the index of the new RGB color.

The receiver can simply recover the message by collecting the LSBs of all indices in the image file.

2.2 Jiri's New Steganographic Method

Jiri proposes a steganographic method to hide message bit into the parity bit of close colors. First, the distance of two colors in a palette is given as follows :

Definition 1: Let *i*, *j* be two color entries in palette *P* with C_i=(r_i, g_i, b_i) and C_j=(r_j, g_j, b_j) respectively, the color distance between C_i and C_j (in Euclidean norm) is denoted as

$$d(i,j)=\sqrt{(r_i-r_j)^2+(g_i-g_j)^2+(b_i-b_j)^2} \quad (1)$$

Let *f* be a palette-based image of size *n* by *n*, there is totall *n*x *n* message bits can be embedded by using Jiri's method. For each pixel with color palette index *i* in *f*, one data bit *d* ("*0*" *or* "*1*") of embedded bit stream will be "coded" by keeping the following principle. If $r_i + g_i + b_i + d \bmod 2$ 7= *1*, a searching for other entries is performed to find a palette entry with the closest color distance and with the different parity ($r_i + g_i + b_i + d \bmod 2 = 0$). Once the entry is found, the index for the pixel is changed to point to the new color. Otherwise, (if $r_i + g_i + b_i + d \bmod 2 = 0$) nothing requires to do.

Message recovery is simply achieved by checking the color parity of the corresponding palette indices. For each pixel in a received image (stego-image) with palette index *i*, if $r_i + g_i + b_i \bmod 2 = 0$, an embedded message bit "0" will be decoded. Otherwise, message bit "1" is decoded. This method never replace a pixel color by a completely different color, which could occasionally happen in EZ Stego because ordering of th e palette by luminance may derive discontinuities in neighboring colors.

3. The Proposed Method

Our approach can be regarded as a preprocessing of Jiri's method. We minimum the RMS error before Jiri's embedding process by update a palette color content b iterations. It can be also regarded as a replacement operation to remove a specified entry of the palette and then to create a new color to occupy the removed entry. Within the iteration, the calculation of removing an entr and the benefit of creating a new one color is done. If the obtained benefit is larger than the loss of removal cost, we remove the selected entry to be an "*empty*" one by taking a replacement for the corresponding image data in the image. Then, a new color is created to occupy the *empty* position by which we can greatly reduce the RMS error in the embedding process. When the iteration algorithm is finished, Jiri's method is employed for data embedding and recovery.

Assuming that the embedded message is encrypted by a traditional encryption algorithm (such as 56-bit key DES) and output random pattern consisting of "0" and "1" with approximately the same size respectively. The following definitions are given before we introduce the details of the method:

Definition 2: Let *f* be a palette-based image with palette size *L* (with entries 0 to *L*-1), the occurrence frequency for each entry in *f* is denoted as *N*(*i*), where *i*=0 to *L*-1.

Definition 3: In a palette *P*, C_x is the closest color for C_y, if entry *x* satisfies the following equation

$d(x, y)$=Min{ $d(n, y)$: $n=0,1,\ldots,L-1$ and $n \neq y$ } where *L* is the size of *P*. (2)

We say that *x* is the first referenced entry for entry *y* in *P* and is denoted as $x= R^{firs}(y)$

Definition 4: In a palette *P*, C_x is the closest color for C_y with *different parity* (parity bit of the color C_x (r_x, g_x, b_x) is $r_x+g_x+b_x$ mod 2), if entry *x* satisfies the following equation

$d(x, y)$=Min{ $d(n, y)$: n=0,1,...,L-1 and $(r_y+r_n+g_y+g_n+b_y+b_n)$ mod 2=1} (3)

We say that x is the first referenced entry for entry y in P and is denoted as $x = R_{DP}^{first}(y)$

Definition 5: In a palette P, if C_x is the first closest color for C_y, then the second closest color C_z for color C_y is defined as follows

$d(z, y)$=Min{ $d(n, y)$: n=0,1,...,L-1. $n \neq x$ and $(r_y+r_n+g_y+g_n+b_y+b_n)$ mod 2=1} (4)

We say that z is the second referenced entry for entry y in P, and is denoted as $z = R_{DP}^{second}(y)$

3.1 Cost of Removing an Entry

According to the above definitions, we can estimate the removal cost with three items (Jiri's error cost, removal of one color from the palette, and the reference error b removing the palette entry) for each entr i in a palette. Assume that the embedded message is encrypted and consisting of the same size of "0" and "1", this means that there is about half of the number of pixel with color entry i will be replaced by other color entry in the embedding process. The first item denoted as $COST_{\text{jiri}}(i)$ is the embedding error by employing Jiri's method.

$$COST_{Jiri}(i) = \frac{1}{2} N(i) \times d(i, R_{DP}^{first}(i))$$ (5)

Where $N(i)$.is the totally number of entr i distributed in the original image f.

The second item denoted as $COST_{\text{selt}}(i)$, is the cost derived from the replacement error for updating the entry to the closest color one and the embedding error in embedding process for the updated entry.

$$COST_{self}(i) = N(i) \times d(i, R^{first}(i)) + \frac{1}{2} N(i) \times d(k, R_{DP}^{first}(k))$$ where

$$k = R^{first}(i)$$ (6)

The third item denoted as $COST_{\text{ref}}(i)$ is the reference error derived by other entries in the palette which select entry i as their closest color during the embedding process.

$$COST_{ref}(i) = \frac{1}{2} \sum_{\substack{k=0 \cdots L-1 \\ R_{DP}^{first}(k)=i}} \left\{ N(k) \times \left[d(k, R_{DP}^{second}(k)) - d(k, R_{DP}^{first}(k)) \right] \right\}$$ (7)

Finally, the total error cost is given b

$$\Delta COST(i) = COST_{self}(i) + COST_{ref}(i) - COST_{Jiri}(i)$$ (8)

3.2 Benifit of Creating a New Entry

For an entry i of the palette with index color $C_i(r_i, g_i, b_i)$, a new entry j is created to make a *referenced pair* which will reduce the embedding error into only 1 by using the following equation.

$r_j = r_i$

$g_j = g_q$

$$b_j = \begin{cases} 1 & if \quad b_j = 0 \\ b_i - 1 & otherwise \end{cases}$$ (9)

Therefore, the benefit of creating a new entry with minimum color distance for entry i is given as follows

$$Benifit(i) = \frac{1}{2} N(i) \times \left[d(i, R_{DP}^{first}(i)) - 1 \right]$$ (10)

Where $d(i, R_{DP}^{first}(i))$ is the embedding error by Jiri's method.

To implement the proposed method, the following algorithm is given

Algorithm: An Iterative Method for Palette-Based Image Steganography

Input: a palette -based image f , the occurrence frequency $N(i)$ of each color entry in f

Output: a palette-based image g

Step1: For each palette entr i , calculate $\Delta COST(i)$ and $Benifit(i)$, respectivel . Assuming thatentr p owns the minimum cost (denoted as $COST_{min} = \Delta COST(p)$) and entr q owns the maximum benifit (denoted as $Benifit_{max} = Benifit(q)$);
If $Benifit_{max} - COST_{min} > 0$ Then
An entry replacement is performed (from step 2 to step 4)
Else
Stop this procedure.

Step2: Replace palette color of entr p in f with $C_p(r_p, g_p, b_p)$ and then copy this modified palette to image g, where
/* palette color with index p will be updated by index q */

$r_p = r_q$

$g_p = g_q$

$$b_p = \begin{cases} 1 & if \quad b_j = 0 \\ b_q - 1 & otherwise \end{cases}$$

Step3: Sequentially scan the image data of $f(i, j)$ and output a new image $g(i, j)$ according to the following condition
/* $f(i, j)$, $g(i, j)$ denote the corresponding palette indices on pixel location (i, j), respectively */

Case $f(i, j) = p$: $g(i, j) = R^{firs}(p)$
Case $f(i, j) = q$: if cnt < $N(q)$ / 2 then
$g(i, j) = p$;
cnt=cnt+1 ;
end if
Otherwise:
$g(i, j) = f(i, j)$

Step4: Update the occurrence frequencies of each color entry in the modified image for next iteration.

$N(R^{firs}(p)) = N(R^{firs}(p)) + N(p)$

$N(p) = N(q)/2$

$N(q) = N(q) - N(q)/2$

4. Results

To evaluate the performance of different steganographic methods, two images (Fruit and Swimmer) with 256 color palette entries are employed. Both these images of size 256×256 are displayed in Figure 1(a) and Figure 2(a), respectively. The embedded message is simulated using random bit stream with length 256×256 bits. Figure 1(b) and Figure 2(b) are the output stego-images by using EZ

Stego method. Figure 1(c) and Figure 2(c) show the stego-images by using Jiri's method. Figure 1(d) and Figure 2(d) show the stego-images by using the proposed method. The corresponding root-mean-square (RMS) errors between the original images and the stego -images are listed in TABLE 1. Figure 3(a) and Figure 3(b) show the RMS errors with different iterations when the proposed method is employed.

The obvious distortion distributed in Figure 1(b) and Figure 2(b) illustrates that the EZ Stego method generated bad stego-images. The RMS errors in TABLE 1 shown 21.970 and 36.882 also support the bad quality resul ts. Jiri's method performs better than EZ Stego, but there is some false contouring happen (see the apple located in the image of Figure 1(c)). In addition, Jiri's method ma work badly for artificial images (such as cartoon pictures) since the color distance between palette entries are larger in artificial images than in the real images. Therefore, Jiri's method may derive noises in the stego-images when it is employed in not real images.

5. Conclusion

We have described a novel steganographic method for palette-based images that dynamically and iterativel modify the palette colors to minimize RMS error between the stego-image and the original image. Our experimental results reveal the practicability and superiority of the new technique.

Acknowlegement

This work was supported in part by National Science Council of the R.O.C. under contact NSC89-2213-E-168-001.

References

[1] R. Machado, EZ Stego, [http://www.stego.com]
[2] Jiri Fridrich," A New Steganographic Method for Palette-Based Images", [http://ssie.binghamton.edu/~jirif]
[3] Jiri Fridrich, "Applications of Data Hiding in Digital Images", Tutorial for the ISPACS'98 Conference, 1998
[4] Tung-Shou Chen, Chin-Chen Chang, and Min-Shiang Hwang, "A Virtual Image Cryptosystem Based upon Vector Quantization", *IEEE Trans. Image Processing*, Vol. 7, No.10, pp. 1485-1488,1998
[5] M. T. Sandford II , J. N. Bradley, and T. G. Handel," The data embedding method", SPIE Photonics East Conference ,1995
[6] W. Bender, D.Gruhl, N.Morimoto, and A. Lu, "Techniques for Data Hiding", IBM System Journal, Vol. 35, Nos 3&4, 1996, pp.313-336
[7] Chiou-Ting Hsu, Ja-Lling Wu, "Hidden Signatures in Images", Proceedings of ICIP, 1996, pp. 223 -226
[8] L. M Marvel, C.G. Boncelet, and C.T.Retter, " Spread Spectrum Image Steganography", IEEE Trans. On Image Processing, Vol. 8, No. 8, 1999, pp. 1075-1083

Figure 1(a)

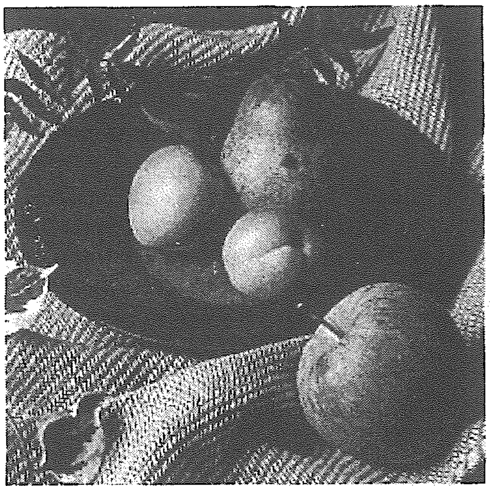

Figure 1(b)

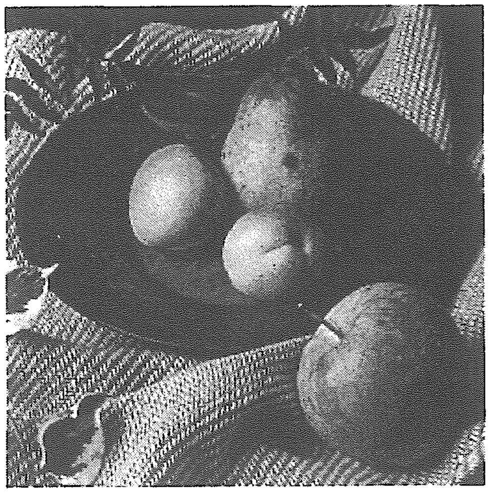

Figure 1(c)

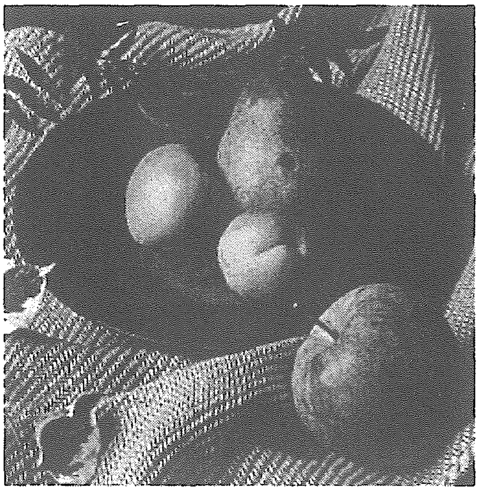

Figure 1(d)

Figure 1: An example to illustrate the steganograph methods. (a) The original image "Fruit" of size 256×256 (b) The stego-image by using EZ Stego method (c) The stego-image by using Jiri's method (d) The stego -image by using the proposed method with 129 iterations.

Figure 2(a)

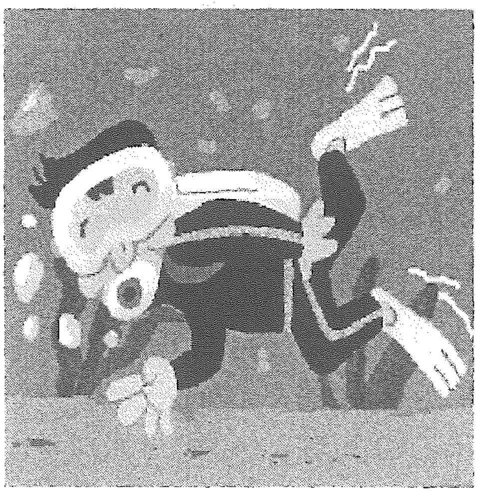

Figure 2(b)

Figure 2(c)

Figure 2(d)

Figure 2: An example to illustrate the steganograph methods. (a) The original image "Swimmer" of size 256×256 (b) The stego-image by using EZ Stego method (c) The stego-image by using Jiri's method (d) The stego-image by using the proposed method with 115 iterations.

TABLE 1: The RMS errors between the original image and the stego-image by employing different steganographic methods.

	Fruit	Swimmer
EZ Ste	21.970	36.882
Jiri's Method	7.782	20.407
The proposed method	1.696	0.502

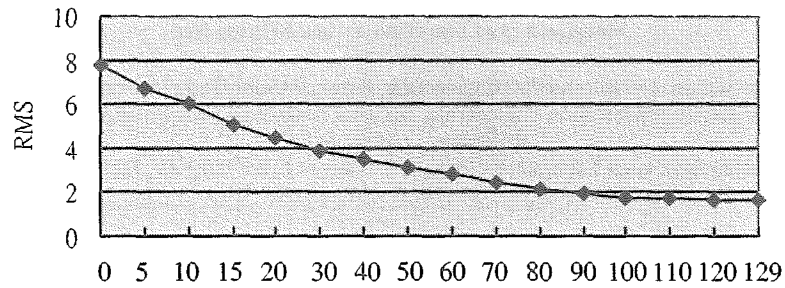

Figure 3(a)

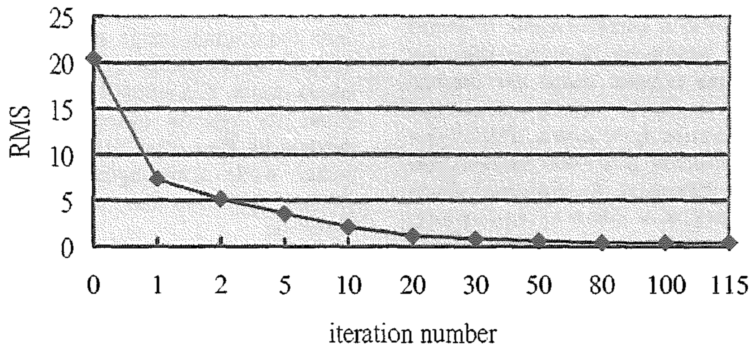

Figure 3(b)

Figure 3: The RMS errors with different iterations by employing the proposed method in Figure 3(a) the Fruit image, and Figure 3(b) the Swimmer image, respectively.