International Journal of Pattern Recognition and Artificial Intelligence
Vol. 16, No. 6 (2002) 681–696

OBJECT-BASED IMAGE STEGANOGRAPHY USING AFFINE TRANSFORMATION

YEUAN-KUEN LEE* and LING-HWEI CHEN†

Department of Computer and Information Science, National Chiao Tung University, 1001 Ta Hsueh Road, Hsinchu 30050, Taiwan, R.O.C.

** yklee@debut.cis.nctu.edu.tw*

† lhchen@cc.nctu.edu.tw

The typical model of steganography has led the prisoners' problem, in which two persons attempt to communicate covertly without alerting the warden. The general way to achieve this task is to embed the message in an innocuous-looking medium. In this paper, an object-based geometric embedding technique is proposed for solving the prisoners' problem. The main idea is to embed secret data through distorting a given object and the distorted object still looks natural. In the embedding process, the secret message is first converted into coefficients of an affine transformation. Then, the coordinates of each pixel of a selected object in the cover-image are recomputed by this affine transformation. Since these coefficients are restricted in a specific range, the transformed object looks natural. In the extracting process, a coarse-to-fine iterative search is proposed to accelerate the object location and the message extraction. Experimental results show that all transformed objects can be located precisely and the whole hidden message can be extracted correctly even if the stego-image is stored in various compression formats and rates. Furthermore, the embedded message is robust enough when the stego-image format is converted from GIF to JPEG, and vice versa.

Keywords: Information hiding; steganography; covert communication; security; affine transformation.

1. Introduction

Steganography is an ancient art of conveying messages in a secret way that only the receiver knows the existence of the message.[7] The most general way is to hide the message in another media such that the transmitted data will be meaningful and innocuous-looking to everyone. For all of the steganographic systems, the most important and fundamental requirement is undetectability. In addition, the difference between the media with hidden message, called the stego-media,[13] and their original ones, called the cover-media are imperceptible. Many factors must be involved in the design of a good steganographic system, such as payload, blindness and security.

†Author for correspondence.

A great deal of image steganographic techniques were introduced in Ref. 6. Many steganographic tools in the Internet are available for varied image formats.[1,5,10,11,14–16] Palette-based and JPEG are two commonly used formats. For palette-based images, directly embedding messages in those index will cause radical color changes, since two neighboring colors in the palette may look dissimilar. Many efforts are made to reduce the embedding distortion. S-Tools[1] is such a method, first, it reduces the number of colors to about 32 unique colors. Then, message-bits are embedded in the LSB of each RGB channel. There is a disadvantage that there are 32 color group in the modified palette. This specific pattern can be detected automatically and reveals the presence of the hidden message. Machado[10] proposed a steganographic method in which the palette is not modified. For each pixel, the message is embedded by replacing the index of a color with the one of the luminance-closed color. Since two colors with closed luminance may be radically different, the created stego-image may have perceptual distortion. To alleviate this problem, Fridrich[2] proposed a new steganographic scheme using the parity of palette colors. The parity of each color could be assigned randomly or simply by calculating $(R + G + B) \bmod 2$. The closest color with the same parity as the message-bit is used to replace the original color. Since parity bits are randomly distributed, the searched new color never departs too much from the original one. In Ref. 3, an optimal parity assignment was proposed in order to further improve the scheme and the energy of modification is decreased by 25–35% depending on the image. For JPEG images, Jsteg[16] embeds the hidden message by modulating the rounding choices either up or down in the quantized DCT coefficients. Note that the downgrade image fidelity caused by the embedding process depends not only on the amount of embedding messages but also on the quality factor setting in JPEG compression. If the quality factor is low, the embedding capacity should be limited in order to satisfy the imperceptible requirement. Note that the above-mentioned techniques strongly depend on the stego-image file format. The hidden message will disappear when the stego-image file format is converted from JPEG to GIF, and vice verse.

Johnson and Katzenbeisser[8] described an image steganographic system in the DCT domain, which is similar to the technique proposed by Zhao and Koch.[18] The cover-image is divided into 8×8 pixel blocks, and each block encodes exactly one secret message bit. Two DCT coefficients with the same quantization value in middle frequencies, such as coefficients in the locations (1, 2) and (3, 0) or in (4, 1) and (3, 2), are selected. Then the relative size of selected DCT coefficients is modulated for embedding the secret message bit. The drawback is that in specific image blocks the desired relation of DCT coefficients cannot be enforced without severely damaging the image data. Zhao and Koch[17] proposed an improved technique using three quantized DCT coefficients to avoid this drawback.

Marvel *et al.*[12] proposed a spread spectrum steganography, called SSIS. Because the original cover-image is needed while extracting the hidden message, an approximate estimation that was obtained by image restoration process is used. Since the

estimate of the embedded message is poor, SSIS should incorporate the use of error-control codes[9] to correct the large number of bit errors. Therefore, most of the payloads in SSIS are wasted on the redundancy for error correction.

All of the above-mentioned methods embed secret messages in the pixel values or the transformed coefficients of the cover-image directly or indirectly. In this paper, we will provide a new concept for information hiding. The secret message will be embedded in objects of a cover-image by distorting objects geometrically. The hidden message is converted into coefficients of an affine transformation first. Then, an object of the image is taken and the affine transformation with the converted coefficients is applied to the object. Two characteristics exist in an object. First, if the transform coefficients are restricted in a specific range, the transformed object will look natural by human eyes. Second, the image fidelity will not degrade by performing an affine transformation on an object. Hence the hidden message will not raise any alarm, and the warden will not detect the covert communication. The experimental results are also given to show that the hidden message can still be correctly extracted even if the stego-image is stored in various compression formats and rates, such as GIF, JPEG and J2K (JPEG 2000). Moreover, even if the format of the stego-image is converted from one compression format to another one, the secret message can be exactly extracted, vice versa.

The remainder of this paper is organized as follows. Section 2 will give a brief review of the affine transformation. Section 3 will describe the proposed image steganographic method. In Sec. 4, we will present some experimental results to show the effectiveness of the proposed method. The conclusions of the paper will be stated in Sec. 5.

2. Review of Affine Transformation

The affine transformation used in the proposed method is defined as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}, \tag{1}$$

where (x, y) are the coordinates of a pixel in an object O and (x', y') are the coordinates in the transformed object O'. The variables a, b, c and d are the transform coefficients. The matrix formed by transform coefficients is called transform matrix. The coefficients a and d produce scale and stretch transformations where $b = c = 0$. Rotation occurs where $|ad - bc| = 1$, and skew occurs where $|ad - bc| \neq 1$.

A feature of an object O is usually called affine invariant if it remains unchanged in the transformed object O'. The affine transformation changes the positions rather than pixel colors of an object. Hence, the color histogram of an object is an affine invariant feature.

Let $(\bar{x}, \bar{y})$ be the coordinates of the center of an object O, that is

$$\bar{x} = \frac{1}{P} \sum_{(x,y) \in O} x \,,$$

$$\bar{y} = \frac{1}{P} \sum_{(x,y)\in O} y\,,$$

where P is the number of pixels of the object O. Similarly, let $(\bar{x}', \bar{y}')$ denote the coordinates of the center of the affine transformed object O'. Then, the following equations can be easily derived.

$$\bar{x}' = a\bar{x} + b\bar{y}\,,$$

$$\bar{y}' = c\bar{x} + d\bar{y}\,.$$

If the coordinate system is centered at the center of the object O, i.e. $(\bar{x}, \bar{y}) = (0, 0)$. Then, $(\bar{x}', \bar{y}') = (0, 0)$. This means that the center of an object is an affine invariant feature. The above-mentioned affine invariant features will be used in the proposed method for locating the transformed objects, the details will be described in Sec. 3.2.

3. Proposed Image Steganographic Models

One assumption in our steganographic method is that both sender and receiver share an object database before conveying messages. Two image steganographic models are provided: the basic model and the advanced one. In these models, the message is converted into coefficients of an affine transformation. In the basic model, the stego-image contains only one transformed object. In order to embed more messages, the advanced model is provided to allow multiple transformed objects in the stego-image. In the following sections, we will describe the details of these models.

3.1. *Basic model*

The basic model consists of two major processes: embedding process and extracting one. In the embedding process, the message is first converted to the coefficients of the affine transformation. Then, the coordinates of each pixel in a selected object are recomputed according to Eq. (1). To extract the message, a searching process is provided for finding the transform matrix used in the embedding process. These processes will be described in the following sections.

3.1.1. *Embedding: message-coefficient conversion*

The messages may be in any digital form and be treated as a bit stream. They are converted into coefficients of an affine transformation. In order to make the transformed object look natural, the transform matrix is selected as close to the 2×2 identity matrix I as possible. Therefore, the coefficient a (or d) is restricted to two specific ranges, i.e. $[1-\epsilon-l, 1-\epsilon]$ and $[1+\epsilon, 1+\epsilon+l]$, and the coefficient b (or c) is restricted in $[-\epsilon-l, -\epsilon]$ and $[\epsilon, \epsilon+l]$, where l is the length of a selected range and ϵ is a constant for the range close to 1 and 0. The selected ranges are

quantized into several of equal-sized intervals, the message will be converted into the value of an interval center as a transform coefficient.

The number of bits embedded in a coefficient is called coefficient capacity. Let δ denote the quantization size, and k denote the coefficient capacity. Then,

$$k = \log_2 \left(\frac{2l}{\delta} \right) .$$

And the first interval center located in the range $[1-\epsilon-l, 1-\epsilon]$ will be $1-\epsilon-l+\frac{\delta}{2}$ and the first interval center in $[1+\epsilon, 1+\epsilon+l]$ will be $1+\epsilon+\frac{\delta}{2}$. Therefore, the conversion of message, m, to coefficient a (or d) can be expressed as

$$a = f_a(m) = \begin{cases} \left(1-\epsilon-l+\dfrac{\delta}{2}\right) + m \times \delta & \text{if } m < 2^{k-1} \\ \left(1+\epsilon+\dfrac{\delta}{2}\right) + (m-2^{k-1}) \times \delta & \text{otherwise}. \end{cases} \tag{2}$$

Similarly, the conversion of message, m, to coefficient b (or c) can be expressed as

$$b = f_b(m) = \begin{cases} \left(-\epsilon-l+\dfrac{\delta}{2}\right) + m \times \delta & \text{if } m < 2^{k-1} \\ \left(\epsilon+\dfrac{\delta}{2}\right) + (m-2^{k-1}) \times \delta & \text{otherwise}, \end{cases} \tag{3}$$

where m is an integer of k bits.

Note that if the allowed error caused by the compressing distortion on the transform coefficient is ε, then the quantization size should be greater than 2ε, and the hidden message can be correctly extracted in a compressed stego-image. Therefore, if the quantization size increases, the robustness of the embedded message will increase as well. It is worth mentioning that whether the hidden message can be extracted correctly depends on not only the compressing distortion but also the object attributes, such as size, shape and color. Specifically, the bigger the object, the smaller the quantization size that can be used, i.e. the coefficient capacity is higher.

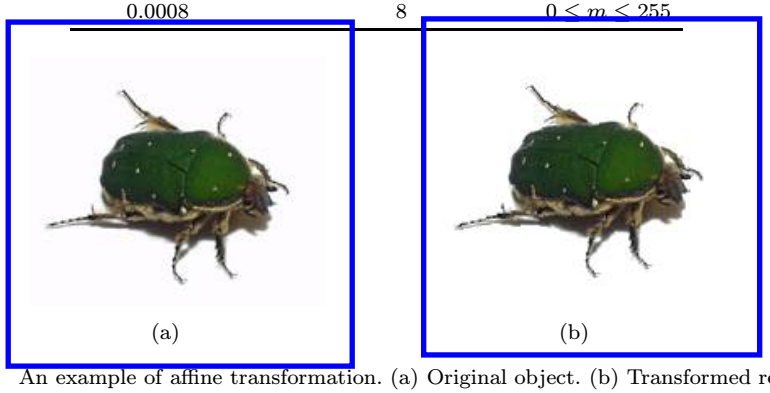
In our experiments, l is selected to be 0.1024, and ϵ is set as 0.01, therefore the ranges of the transform coefficient a (or d) are selected to be [0.8876, 0.99] and [1.01, 1.1124], and the ranges of b (or c) are $[-0.1124, -0.01]$ and [0.01, 0.1124]. Table 1 shows the relation between the quantization size, the coefficient capacity, and the range of messages for the selected coefficient range. As can be seen from Table 1, when the quantization size doubles, the coefficient capacity will be decreased by 1 bit. If the quantization size is selected to be 0.0001, the message $\langle 1302, 520, 1152, 1448 \rangle$ will be converted to $\langle 1.03785, -0.06045, 0.02285, 1.05245 \rangle$.

3.1.2. *Embedding: affine transformation*

After the messages have been converted to the transform matrix, the coordinates of each pixel in the object are recomputed according to Eq. (1). Figure 1 shows

Table 1. The relation between the quantization size and capacity.

Quantization Size	Coefficient Capacity (k)	Message (m)
0.0001	11	$0 \leq m \leq 2047$
0.0002	10	$0 \leq m \leq 1023$
0.0004	9	$0 \leq m \leq 511$
0.0008	8	$0 \leq m \leq 255$



(a) (b)

Fig. 1. An example of affine transformation. (a) Original object. (b) Transformed result.

an example of applying an affine transformation on a beetle object, the coefficient capacity is set as 11. Figure 1(a) is the original object, Fig. 1(b) shows the transformed result with hidden messages $\langle 1302, 520, 1152, 1448 \rangle$. The affine coefficients are set as 1.03785, -0.06045, 0.02285 and 1.05245 for a, b, c and d, respectively. Since this transform matrix is close to the 2×2 identity matrix I, the transformed beetle in Fig. 1(b) looks natural. Hence the hidden message will not raise any alarm.

3.1.3. *Extracting: coarse-to-fine iterative search*

The task of the extracting process is to search the transform matrix performed in the embedding process. To reach this aim, an iterative search method is provided. First, the template matching error between objects O_1 and O_2, is defined as

$$E(O_1, O_2) = \sum_{(i,j) \in O_1 \cup O_2} |O_1(i,j) - O_2(i,j)|\,, \tag{4}$$

where $O_k(i,j)$ is the color vector of a pixel of the object O_k at coordinates (i,j), $k = 1, 2$.

Let O be the original object and O_{stego} denote the received transformed object in the stego-image, and $O_{a,b,c,d}$ denote the object produced by performing the affine transformation on O with coefficients a, b, c and d. Then, the extracting problem is to find a, b, c and d, such that the template matching error $E(O_{\text{stego}}, O_{a,b,c,d})$ is minimum.

The simplest searching method is to apply all possible transform on the original object and take the one with the minimum template matching error. However, this is very time consuming. The reason is that if the capacity of each coefficient is

k, there are 2^k possible values for each coefficient. Consequently, a total of 2^{4k} combinations of coefficients a, b, c and d, should be tested to treat this problem, a coarse-to-fine iterative search method is provided to expedite the searching process.

The proposed method consists of two successive passes. In Pass 1, the coefficients a and b are fixed, only the combinations of coefficients c and d are searched to get the one (c_0, d_0) with $E(O_{\text{stego}}, O_{a,b,c_0,d_0}) = \min_{c,d} E(O_{\text{stego}}, O_{a,b,c,d})$. In Pass 2, the coefficients c and d are fixed to search the combinations of coefficients a and b, to get the one (a_0, b_0) with $E(O_{\text{stego}}, O_{a_0,b_0,c,d}) = \min_{a,b} E(O_{\text{stego}}, O_{a,b,c,d})$. Note that the fixed values of c and d used in Pass 2 are those obtained from Pass 1, and the fixed values of a and b used in Pass 1 are those calculated from Pass 2. These two passes are performed iteratively until the result does not change anymore.

In order to speed up the searching process, a coarse-to-fine scheme is provided. The idea behind the scheme is that $E(O_{\text{stego}}, O_{a,b,c,d})$ is dependent on the similarity between two transform matrices performed for O_{stego} and $O_{a,b,c,d}$. If the associate coefficients of transform matrix performed on O_{stego} are closer to a, b, c and d, then $E(O_{\text{stego}}, O_{a,b,c,d})$ will be smaller. Based on this idea, the above iterative search can be coarsely performed to reduce the searching range. Then a finer search is performed in the reduced range. The details are described as follows.

Let m denote the hidden message, m is an integer consisting of k message-bits. That is, the coefficient capacity is k, the range of m will be $[0, 2^k-1]$. In the proposed method, the coarse-to-fine iterative search is implemented using $\lfloor \log_{10}(2^k) \rfloor$ levels. Let M_i^j denote the set of integers, which will be searched in the level-i iterative search for coefficient j. Thus, M_i^j is defined as

$$M_i^j = \left\{ m' | m' = m_{i-1}^j + s_i * t, \text{ for } -\frac{s_{i-1}}{s_i} < t < \frac{s_{i-1}}{s_i} \right\}$$

where t is an integer, s_i is the step size used in the level-i (Li) search, m_i^j is the estimated message obtained from the Li search for coefficient j, and $m_0^j = 2^{k-1}$ is the initial center. Table 2 lists the step sizes used in the proposed coarse-to-fine search under the coefficient capacity k from 8 to 11.

Note that if the coefficient capacity k is selected to be 11, then M_1^j, M_2^j and M_3^j contain 21, 19 and 19 elements, respectively for each coefficient j. The search numbers are 21^2, 19^2 and 19^2 in each pass of L1, L2 and L3 iterative search, respectively. Thus, the total search number is $p_1 \times 21^2 + p_2 \times 19^2 + p_3 \times 19^2$, where

Table 2. The step sizes used in the proposed coarse-to-fine search.

k	s_0	s_1	s_2	s_3
11	1100	100	10	1
10	550	50	10	1
9	275	25	1	–
8	132	12	1	–

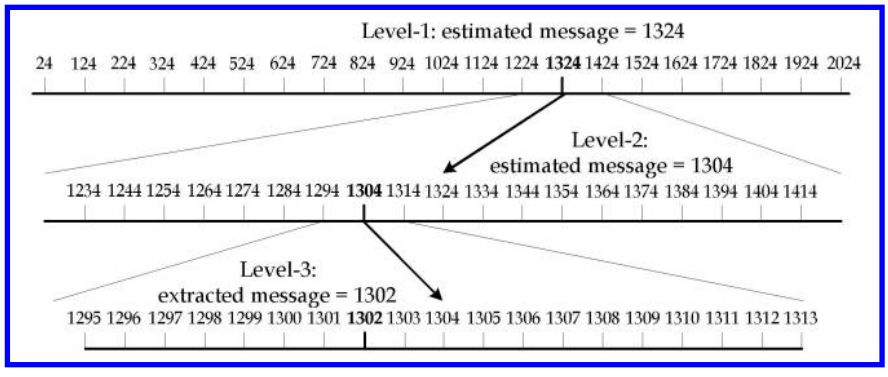


Fig. 2. An example of the coarse-to-fine search when the coefficient capacity $k = 11$.

p_1, p_2 and p_3 are the numbers of passes performed in the L1, L2 and L3 search, respectively. In our experiments, p_1, p_2 and p_3 are less than 7. However, for an exhausted search, the original number is 2048^2 in each pass of the iterative search. Thus, the searching speed will be highly raised.

Figure 2 shows the result of applying the three-level coarse-to-fine search to Fig. 1(b) for coefficient a, and k is set as 11. In L1 search, the step size $s_1 = 100$, and the searching result $m_1^a = 1324$. In L2 search, based on the estimated message m_1^a the searching range becomes [1234, 1414] and $s_2 = 10$. The searching result $m_2^a = 1304$. Based on m_2^a, the searching range is reduced to [1295, 1313] and $s_3 = 1$ in L3 search. The final searching result, 1302, is treated as the extracted message.

The details of the provided coarse-to-fine iterative search are described in *Algorithm* 1.

Algorithm 1: **Coarse-to-fine iterative search**

Input: received stego-object O_{stego}, original object O, coefficient capacity k

Output: extracted message $m = \langle m_a, m_b, m_c, m_d \rangle$

Step 1 (Initialization) $i = 0$, $a = 1$, $b = 0$, $m_a = 0$, $m_b = 0$.
$\vec{m_1} = \langle 0, 0, 0, 0 \rangle, \vec{m_2} = \langle 0, 0, 0, 0 \rangle$.

Step 2 i =i +1

Step 3 if $(i > \lfloor \log_{10}(2^k) \rfloor)$
goto *Step* 10.

Step 4 Perform an exhaust search to find a pair of $\langle m_c, m_d \rangle$, $m_c \in M_i^c$, $m_d \in M_i^c$, such that $E(O_{\text{stego}}, O_{a,b,f_c(m_c),f_d(m_d)})$ is minimum.

Step 5 $c = f_c(m_c)$, $d = f_d(m_d)$, $\vec{m_1} = \langle m_a, m_b, m_c, m_d \rangle$.

Step 6 if $(\vec{m_1} = \vec{m_2})$
goto *Step* 2.

Step 7 Perform an exhaust search to find a pair of $\langle m_a, m_b \rangle$, $m_a \in M_i^a$, $m_b \in M_i^b$, such that $E(O_{\text{stego}}, O_{f_a(m_a),f_b(m_b),c,d})$ is minimum.

Step 8 $a = f_a(m_a)$, $b = f_b(m_b)$, $\vec{m_2} = \langle m_a, m_b, m_c, m_d \rangle$.

Step 9 if $(\vec{m_1} = \vec{m_2})$

goto *Step* 2.
else
goto *Step* 4.
Step 10 Output $m = \langle m_a, m_b, m_c, m_d \rangle$.

3.2. *Advanced model*

In the advanced model, each created stego-image contains multiple objects. Hence, the amount of embedded messages in a single stego-image is increased. Figure 3 shows the block diagrams of the embedding process and the extracting process.

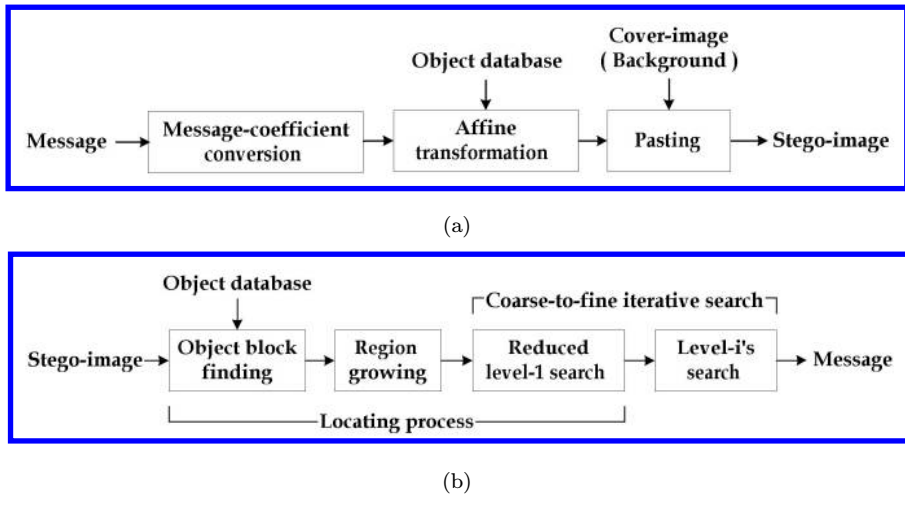


Fig. 3. The block diagrams of the advanced model. (a) Embedding process. (b) Extracting process.

In the embedding process, the center pixel of the original object is set as the origin. The embedding process is the same as that in the basic model, except that all transformed objects are pasted into a cover-image (white background) to form a stego-image. Figure 4 shows an example of a stego-image that contains eight objects.

Note that the stego-image may be stored in various compressed formats before transmission on the Internet. Therefore, the compressing distortion will be imposed on the transformed object, this may make the locating process more difficult. Only if these object in the stego-image can be located exactly, that is, its center pixel can be found precisely, can the hidden message be extracted correctly in the extracting process.

The locating process consists of three steps: object block finding, region growing and reduced level-1 (RL1) search. Given an original object, the first step is to find a block that contains the transformed object in the stego-image. Then, region growing is used to find the maximum connected component in this block, and the center

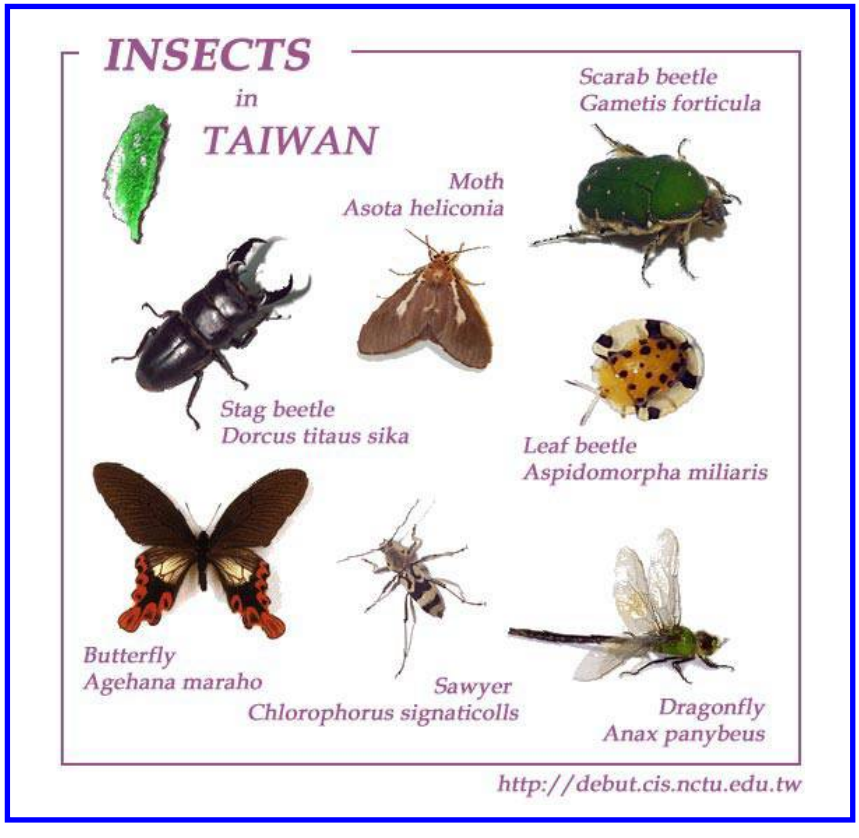


Fig. 4. An example of the created stego-images.

pixel of the located object is considered as a candidate center. Since the compressing distortion may affect the shape of the transformed object, the x- or y-coordinate of the center pixel may be shifted by one pixel. That is, the center will reside in a 3×3 window at the candidate center. To get the original center, the RL1 search is then performed on these nine possible pixels to precisely locate the center pixel. Finally, as in the basic model, the L*i's* searches are performed to extract the hidden message.

In the following sections, we will describe the details of the three locating steps in sequence.

3.2.1. *Object block finding*

The goal of the object block finding step is to use affine invariant features of the original object to find a block that contains the corresponding transformed object. Note that the transformed object has two types of distortion: affine-transformed distortion and compressing distortion. For speeding up the object block finding process and decreasing the effect from the compressing distortion, the scales of the received stego-image and the original object are reduced to a quarter of the original

size. Let O_s denote the size-reduced version of the original object, (x_o, y_o) represent the coordinates of the center of O_s.

As mentioned in Sec. 2, the color histogram is an affine invariant feature. We first construct the histogram $h(c)$ of O_s and define $CS = \{c|h(c) > 0\}$, where c is a color vector. Next, the reduced stego-image is first split into blocks of size O_s, two steps are performed on each block: the pixel marking and the block alignment. In the pixel marking step, for a given block with histogram $g(c)$, each pixel satisfies the following two conditions marked: (1) the color, c, of the pixel belongs to CS, (2) $h(c) > 0$. To avoid locating another object with a dominant color c belonging to CS, when a pixel with c is marked, $h(c)$ will be decreased by 1. If the block has the number of marked pixels greater than a quarter of the size of O_s, it is considered as an object block. Then, the block alignment step is performed. Otherwise, the block is skipped. Let (x_c, y_c) be the center of all marked pixels in the object block, then the object block is moved such that its center is at (x_c, y_c). These two steps are repeated until the object block cannot be moved anymore.

Figure 5 shows an example for the object block finding process. Figure 5(a) shows a block considered as a beetle object block, and the center of marked pixels is indicated by the cross sign. Based on this center pixel, the aligned block is found and shown in Fig. 5(b). The object block shown in Fig. 5(c) is the final result of the iterative process.

(a) (b) (c)

Fig. 5. An example for the object block finding process. (a) A beetle object block. (b) The aligned object block. (c) Final result.

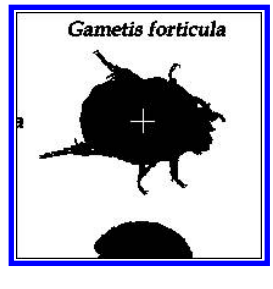


Fig. 6. Connected components of stego-object block shown in Fig. 5(c).

3.2.2. *Region growing*

The final found object block is called stego-object block in our paper. A region growing process is then performed on the stego-object block. Using the background information of the stego-image and the technique of region growing,[4] the maximum connected component in the stego-object block can be found and considered as the transformed object. The center pixel of the extracted object is calculated and treated as a candidate center. Figure 6 shows connected components of the stego-object block shown in Fig. 5(c), and the center of the maximum connected component is marked by the cross sign.

3.2.3. *RL1 search*

If the center of the transformed object is located correctly, then the hidden message can be extracted by using the same process in the basic model. However, since the compressing distortion may affect the shape of the object, and the x- or y-coordinate of the center may be shifted by one pixel. Thus, the original center will reside in the 3×3 window centered at the candidate center.

The simplest way to extract the hidden message is to perform the proposed coarse-to-fine iterative search on these nine possible positions. However, this approach is quite time-consuming, it is necessary to expedite the searching process. By observing nine template matching errors obtained by performing L1 search on these nine positions, we find that the correct center pixel will be the position with the minimum template matching error being the minimum of the nine local minimum template matching errors. To further accelerate the process, we also find that the above-mentioned property is still valid when the template matching error is only evaluated on those pixels with both x-coordinate and y-coordinate being even. The above procedure is called reduced level-1 (RL1) search. Finally, the Li*'s* searches are performed to extract the hidden message. Note that the other Li*'s* searches in the advanced model are identical to those in the basic model.

4. Experimental Results

In our experiments, several objects are used and kept on both sides of sender and receiver. Using the shared object database, the hidden message can be extracted automatically. Note that the database includes both objects and their corresponding coefficient capacities. And all created stego-images are stored in three most commonly used compression formats, i.e. GIF, JPEG and J2K. Three quality settings in JPEG compression are chosen: 75 (high quality, JPG_{hq}), 50 (middle quality, JPG_{mq}) and 25 (low quality, JPG_{lq}), and two quality setting in J2K are used: 75 (high quality, $J2K_{hq}$) and 50 (middle quality, $J2K_{mq}$). To demonstrate the effectiveness of the proposed models, two kinds of experiments are provided: (1) locating the object in the stego-image with the low quality JPEG compression setting; (2) exploring the object capacity under various compression formats and rates.

For the first experiment, eight objects shown in Fig. 4 are used. Table 3 presents

Table 3. The locating results for storing the stego-image in JPEG low quality setting.

Object	Original Center	Region Growing	RL1 Search
1. Taiwan	(128, 88)	$\surd$(128, 88)	$\surd$(128, 88)
2. Stagbeetle	(238, 140)	(239, 140)	$\surd$(238, 140)
3. Moth	(233, 298)	(232, 298)	$\surd$(233, 298)
4. Scarab	(153, 455)	$\surd$(153, 455)	$\surd$(153, 455)
5. Leafbeetle	(272, 451)	$\surd$(272, 451)	$\surd$(272, 451)
6. Butterfly	(397, 133)	(397, 132)	$\surd$(397, 133)
7. Sawyer	(425, 287)	$\surd$(425, 287)	$\surd$(425, 287)
8. Dragonfly	(457, 447)	(458, 447)	$\surd$(457, 447)

Table 4. Coefficient capacity under GIF and various quality settings in JPEG and $J2K$.

Object	Size (pixels)	GIF	JPG_{hq}	JPG_{mq}	JPG_{lq}	$J2K_{hq}$	$J2K_{mq}$
7. Sawyer	1933	10	11	10	10	11	11
1. Taiwan	2556	11	10	9	8	11	11
8. Dragonfly	5066	11	11	11	10	11	10
5. Leafbeetle	5483	11	11	10	9	11	11
3. Moth	5681	11	11	10	10	11	11
2. Stagbeetle	6440	11	11	10	10	11	11
4. Scarab	6593	11	11	11	10	11	11
6. Butterfly	9737	11	11	11	11	11	11

the original centers of these objects in the stego-image, the located center pixels after performing the region growing process, and the refined center after applying the RL1 search. Note that those locations with check marks are correct locations. The correct location of the center is within the 3×3 neighborhood of the pixel located after the region growing process. As shown in the last column, all center pixels can be precisely located in the RL1 search.

For the second experiment, we aimed to explore the object capacity under various compression formats and rates. The object capacity means the maximum number of message-bits which can not only be embedded in an object but can also be correctly extracted from the stego-image. The coefficient capacity of each object in Fig. 4 is presented in Table 4. The objects are presented in the order of object size. Tho coefficient capacity of 11 mean that the quantization size is set as small as 0.0001 (see Table 1), and the hidden message can still be extracted correctly.

As mentioned in Sec. 3.1.1, the capacity depends on object attributes (size, shape, color), and the compression quality setting. As can be seen from Table 4, when the quality setting becomes lower, the coefficient capacity will be smaller. Under the same quality setting, when the object size (the number of pixels) is bigger, the coefficient capacity will be higher. Note that the number of pixels of object 5, *Leafbeetle*, is larger than that of object 8, *Dragonfly*, but the coefficient capacity is lower. This is due to that its shape is near a circle. Note that since the circular shape makes all of pixels as close to the center as possible, the effect

of linear transformation is smaller, thus a larger quantization size should be used. Similarly, the size of object 1, *Taiwan*, is bigger than object 7, *Sawyer*, but color differences among neighboring pixels are smoother. Thus, the coefficient capacity of object 1 is lower.

It is worth mentioning that our proposed models are also effective when the file format of the stego-image is converted from one compression format to another one. The final coefficient capacity, denoted D, can be evaluated using the following expression.

$$D = \min(C_1, C_2)$$

where C_1 and C_2 are the coefficient capacities under the original format and the converted one, respectively.

5. Conclusions

In this paper, we have proposed an object-based image steganographic technique for covert communication. In our method, the message is converted into coefficients of an affine transformation. Since these coefficients are restricted in a specific range, the difference between the original object and the transformed one is indistinguishable by the human eyes, i.e. the transformed object still looks natural. Therefore, the hidden message does not alarm the warden when the stego-image is transmitted on the Internet. In addition, the hidden message is embedded in the geometric distortion and the proposed embedding technique is independent of the stego-image format. Hence, the hidden message can be exactly extracted, nevertheless the stego-image is stored in various formats.

Two models are provided: the basic model and the advanced one. In the basic model, a coarse-to-fine iterative algorithm is presented to accelerate the message extracting process, and make the proposed scheme more serviceable. For increasing the amount of embedding message, the stego-image created in the advanced model contains multiple objects. Experimental results show that all objects can be located precisely, and the embedded message is robust enough when the stego-image format is converted from GIF to JPEG, and vice versa.

Acknowledgments

This research was supported in part by the National Science Council, R.O.C., under contract NSC90-2213-E-009-127.

References

1. A. Brown, "S-Tools V4," ftp://idea.sec.dsi.unimi.it/pub/security/crypt/code/s-tools4.zip.
2. J. Fridrich, "A new steganographic method for palette-based images," *Proc. IS&T PICS* 1998, pp. 285–289.
3. J. Fridrich and R. Du, "Secure steganographic methods for palette-based images," *Proc. 3rd Int. Workshop Information Hiding*, 1999, pp. 47–60.

4. R. G. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, NY, 1992.
5. H. Hastur, "Mandelsteg," ftp://idea.sec.dsi.unimi.it/pub/security/crypt/code/.
6. N. F. Johnson and S. Jajodia, "Steganography: seeing the unseen," *IEEE Comput.* **31**, 2 (1998) 26–34.
7. D. Kahn, *The Codebreakers — the Comprehensive History of Secret Communication from Ancient Times to the Internet*, Scribner, NY, 1996.
8. S. Katzenbeisser and F. A. P. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking*, Artech House, London, 2000.
9. S. Lin and D. J. Costello, Jr., *Error Control Code: Fundamentals and Applications*, Prentice Hall, NJ, 1983.
10. R. Machado, "Stego," http://www.stego.com.
11. C. Maroney, "Hide & seek," http://www.rugeley.demon.co.uk/security/hdsk50.zip.
12. L. M. Marvel, C. G. Boncelet, Jr. and C. T. Retter, "Spread spectrum image steganography," *IEEE Trans. Imag. Process.* **8**, 8 (1999) 1075–1083.
13. B. Pfitzmann, "Information hiding terminology," *Proc. 1st Int. Workshop Information Hiding*, 1996, pp. 347–350.
14. H. Repp, "Hide4PGP," http://www.rugeley.demon.co.uk/security/hide4pgp.zip.
15. STEGANOS company, "Steganos 3," http://www.demcom.com/en/index.htm, 2001.
16. D. Upham, "Jsteg," ftp://ftp.funet.fi/pub/crypt/steganography.
17. J. Zhao and E. Koch, "Embedding robust labels into images for copyright protection," *Proc. Int. Conf. Intellectual Property Rights for Information, Knowledge and New Techniques*, 1995, pp. 242–251.
18. J. Zhao and E. Koch, "Towards robust and hidden image copyright labeling," *Proc. IEEE Workshop Nonlinear Signal and Image Processing*, 1995, pp. 452–455.

Yeuan-Kuen Lee received his B.S., M.S. and Ph.D. degrees in computer and information science from the National Chiao-Tung University, Hsinchu, Taiwan, in 1989, 1991 and 2002, respectively. From 1993 to 1995, he was a Lecturer at the Aletheia University, Taipei, Taiwan, from 2001 to 2002, he worked at the Chung Hua University, Hsinchu, Taiwan. He is currently an Assistant Professor of computer science at Ming Chuan University, Taoyuan, Taiwan.

His research interests are in the areas of image and signal processing, digital steganography and computer security.

Ling-Hwei Chen received the B.S. degree in mathematics and the M.S. degree in applied mathematics from the National Tsing Hua University, Hsinchu, Taiwan in 1975 and 1977, respectively, and the Ph.D. in computer engineering from National Chiao Tung University, Hsinchu, Taiwan in 1987.

From August 1977 to April 1979, she worked as a research assistant in the Chung-Shan Institute of Science and Technology, Taoyan, Taiwan, after which she worked as a research associate in the Electronic Research and Service Organization, Industry Technology Research Institute, Hsinchu, Taiwan. From March 1981 to August 1983, she worked as an engineer in the Institute of Information Industry, Taipei, Taiwan and is now a Professor at the Department of Computer and Information Science at the National Chiao Tung University.

Her current research interests include image processing, pattern recognition, document processing, image compression and image cryptography.

This article has been cited by:

1. Wen-Chao Yang, Ling-Hwei Chen. 2013. A steganographic method via various animations in PowerPoint files. *Multimedia Tools and Applications* . [CrossRef]