Image Hiding In Spatial Domain Using An Image Differencing Approach

Da-Chun Wu[1] (吳大鈞), Wen-Hsiang Tsai[2] (蔡文祥)
Department of Computer and Information Science
National Chiao Tung University
Hsinchu, Taiwan 300, Republic of China
E-mail:whtsai@cis.nctu.edu.tw
Tel: +886-3-5712121 ext. 56621, Fax: +886-3-5721489

Abstract

A method to embed a secret image into a cover image with low degradation is proposed. The method is based on the characteristics of the similarity among the gray values of consecutive image pixels as well as the human visual system's sensitivity variation from smooth to contrastive. A stego-image is produced by replacing the gray values of a differencing result obtained from the cover image with those of a differencing result obtained from the secret image. Moreover, a pseudo-random mechanism is used to achieve cryptography. It is found the values of the peaks of the signal-to-noise ratios of the proposed method are high and the resulting stego-images are imperceptible, even when the size of the secret image is about a half of the cover image.

Keywords: image hiding, cover image, secret image, image differencing, stego-image, security, human visual system.

1. Introduction

In today's digital world, most text, image, audio, and video data can be represented in digital form. Due to the human visual system's low sensitivity to small changes and the high plasticity of digital media, we can easily make tiny changes in digital data with low perceptibility, or even produce new data by mixing different data sources. Consequently, many interesting applications of data hiding in digital media can be created, like caption data embedding, secret message delivering, temper proofing, copyright protection, watermarking, etc. The purpose of caption data embedding is to embed a large quantity of data into a certain host signal with low degradation. The purpose of secret message delivering is to embed data imperceptibly to avoid illicit access. The purpose of temper proofing is to assure that the received data is not forgery. And finally the purpose of copyright protection and watermarking is to produce imperceptible and undeletable results with robustness to common signal manipulations and lossy compression.

Several previous works about data hiding in text have been discussed in [1], such as line-shift coding, word-shift coding, feature coding, etc. The quantity of the embedded data is small and can be easily destroyed by reproduction. In the fields of image hiding, the method of changing least significant bits (LSBs) [2] [3], the patchwork method [4], and the texture block coding method [4] use the characteristic of the human visual system's low sensitivity to small changes in gray values to embed data in the host image. Image hiding can also be applied in frequency or other transform domains. Two of such methods are the use of randomly sequenced pulse position modulated codes [5], and the secure spread spectrum method [6].

In the image hiding process, the image that holds the hidden information will be called the cover image in this study. The output of the hiding process, which includes the hidden information, will be called the stego-image. The information to be hidden in the cover image will be called the secret message, which may be plain text, another image, or anything can be represented in a bit stream.

In this paper, we propose a new method to embed a secret image into a cover image. It can be used for secret image delivering, copyright protection, etc. The method is designed in such a way that the gray value of every pixel of the secret image is preserved, i.e., no distortion will be created in the secret image when it is extracted out from the stego-image. On the other hand, because the resulting stego-image usually contains a large quantity of embedded data, it may be degraded seriously. The proposed method, however, produces imperceptible changes in the resulting stego-image. Because the precision of pixel gray values may be destroyed when transforming them back and forth between the frequency and spatial domains, we adopt the way of embedding the pixel values of the secret image into the cover image directly in the spatial domain.

Liaw and Chen [7] proposed a method that embeds a secret image into a cover image byte after byte by finding for each pixel p in the secret image a pixel p' in the cover image whose gray value g' is closest to that (say, g) of p and then replacing g with g'. This gray-value replacement method suffers from a problem: in certain natural images, there might exist many pixels which do not have suitable corresponding pixels in the cover image with similar gray values to allow gray value replacement. If we replace gray values to be embedded with ones which are quite different, the resulting stego-image will have noticeable changes. The method proposed in this paper instead may be employed to embed an image into a cover image easily but cause less degradation in the cover image. Changes in the cover image after embedding are imperceptible to hackers. Moreover, we

[1] Also with the Department of Information Management, Ming Chuan University, Taipei, Taiwan 111, Republic of China.

[2] To whom all correspondence should be sent.

walk through the cover image which hiding data in an order provided by a pseudo-random number generator to achieve cryptography, and so can prevent tempering access to the hidden data from illicit users.

The remainder part of this paper is organized as follows. In Section 2, the proposed image hiding method is presented. The process for extracting the hidden image is described in Section 3. And several experimental results are illustrated in Section 4. Finally, concluding remarks as well as some suggestions for future works are stated in Section 5.

2. Proposed Image Hiding Method

The proposed image hiding method utilizes the similarity property among adjacent pixels, which exists in most nature images. As an illustration, see the images of LENA and F-16 in Figs. 1(a) and (b), respectively. The histograms of these two images are shown in Figs. 1(c) and (d), respectively. They are quite different in shape. Now, perform the operation of image differencing to the two images in the following way: for every pixel value p_i in either image I, where i= 0, 1, ..., let $p'_i = p_i - p_{i-1} + 128$ where p_{i-1} is the gray value of a neighboring pixel to p_i, and create a new image I' with all p'_i as the new pixel values. Call I' as the difference image from I. For more details of image differencing in our proposed method, which are a little different from that of the above process, see the next section. The difference images from LENA and F-16 are shown in Figs. 1(e) and (f), respectively, and their histograms are shown in Fig. 1 (g) and (h), respectively. The histograms of the difference images are quite similar in shape, in contrast with those of their original images, which are quite different. We want to use this characteristic to embed the secret image into a cover image using gray value replacement.

Hiding data in the LSB's of a gray-valued image is a method that utilizes the characteristic of the human visual system's insensitivity to small changes in the image. Fig. 2 illustrates the resulting images that are created by randomly flipping the LSB's in Lena from 1 LSB to 7 LSB's. It can be seen that the more the LSB's are changed, the more distortion we get. Taking a close look at the effects in the resulting images of flipping 3 LSB's and 4 LSB's, we see that the changes in the former cannot be found by casual observation. The changes in smooth areas in the latter, like the shoulder and the background areas, are noticeable, but those on the edge areas or contrastive areas, like the hair and the hat acock, are not easy to find out. This shows that the human visual system's sensibility varies from smooth areas to contrastive areas. In the hiding process we propose, we use this characteristic to hide more data in contrastive areas and less in smooth areas. This way helps hiding more data in an image with imperception.

The proposed hiding method is sketched in Fig. 3, which consists of three major parts: the process of cover image differencing, that of secret image differencing, and the embedding process, which are described subsequently.

2.1 Cover and Secret Image Differencing

Cover and secret images used in the proposed method are gray-scale ones. To get the difference image from a given cover image, we calculate the difference value of the gray values in every non-overlapping two-pixel subimage of the cover image, in a zigzag order as shown in Fig. 4, and add an offset value 128 to the difference to produce a result. The resulting data stream is half of the cover image in size. For nature images, there results almost no difference value outside the range of [0, 255]. The resulting difference value of each subimage pair is stored as a byte, and treated as an unsigned integer in the range of 0 to 255 for later processing. Since the gray values of adjacent pixels are similar, the values of the resulting stream concentrate near the value 128. According to our experimental results from processing natural images, about 90% of the values are distributed in the range of [112, 143], and this 32-value range is only 12.5% of [0, 255]. Moreover, about 80% of the values are distributed in the range of [120, 135], and this 16-value range is only 6.25% of [0, 255]. This characteristic of natural images helps us to find similar values easily in doing gray value replacement during the image embedding process.

The process to get the difference image from the secret image is the same as that to process the cover image except that we calculate the difference value of the gray values of every two adjacent pixels, instead of every non-overlapping two-pixel subimage, in the zigzag order. So, the size of the resulting stream is the same as that of the secret image. In the calculation of the difference value of the top-leftmost pixel of the secret image, we assume that its previous pixel's gray value is 128. For convenience, in the sequel we will call the difference image from the secret image as the secret difference image, and that from the cover image as the cover difference image.

In addition, we quantize the difference values into eight ranges as illustrated in Fig. 5. These ranges are assigned by indices 0 though 7, respectively. A difference value which falls in a range with index k is said to have index k. All the values in a certain range (i.e., with an identical index) are considered as close enough and cannot be easily noticed to human perception after a value in the range is replaced by the other values in the same range. The range intervals we choose as shown in Fig. 5 are based on the human visual system's sensitivity variation from smooth to contrastive areas. The pixels in the contrastive area may suffer from larger value changes than those in the smooth area in the same sensitivity to the human perception. The difference value near 128 is considered as in a smooth area and that far from 128 is considered as in a contrastive area. So we create in Fig. 5 small ranges near 128 and large ranges far from 128 for the purpose of better replacement with less noticeable results.

2.2 Image Embedding Process

The proposed image embedding process consists of two steps: gray value replacement, and the inverse differencing of the stego-image.

A. Gray Value Replacement

An overview of the replacement process is illustrated in Fig. 6. For each pixel p_s in the secret difference image S, we find a pixel p_c in the cover difference image C, whose gray value has the same index as that of p_s, and hide p_s by replacing the value of p_c with that of p_s.

Although the gray value distributions of S and C are similar, there may exist insufficient pixels in C which have the same index as that of a certain pixel in S which we want to embed. So, the adjustment of the values of some pixels in C to the new values which are insufficient for embedding is needed before the whole replacement work begin. The adjustment work starts by counting the total number of pixels of every gray value from 0 to 255 in images S and C individually, and computing the total count for each index. The total counts of indices from 0 to 7 from both S and C then are checked to find out *insufficient indices*. We say that an index k is insufficient if the total count of the index k in C is smaller than that in S. The range of gray values that is represented by an insufficient index k is called an insufficient range. And we say that an index k is excessive if the total count of the index k in C is larger than that in S. For every insufficient range, we need to find enough pixels in C that are not with the index k and adjust their indices to k by changing the values of the pixels to a value in the insufficient range. In this process, we only record the number of pixels of each of the gray values that are supposed to be adjusted. The policy to decide how many pixels with a given gray value g can change their pixel values to a new value in an insufficient range R is determined by the quantity of the need in R and the number of pixels with g. We accumulate for R an enough number of pixels by collecting gray values outside R and spreading out of both ends of R. The accumulation work collects gray values from a range which is not processed before or which is processed but still is excessive. There may exist more than one insufficient range which need be processed. The order of processing the insufficient ranges in our method is decided by how large the distance of an insufficient range is to a nearest excessive range. We apply the above accumulation work to the insufficient range with the largest distance first.

After all of the insufficient ranges are checked and the numbers of pixels to be changed are accumulated, the work of changing values is performed by randomly traversing C using a pseudo-random generator, which visits each pixel in C only once. For every visited pixel in C with a gray value which need be changed, we change the value of the visited pixel to a value of the insufficient range. After the adjustment process, the adjusted cover difference image is ready for use in sequent replacement works.

For the purpose of easy finding of the desired p_c in C which has the same index as p_s, we rearrange all of the pixels in C into an eight-list structure as illustrated in Fig.7. Every list in the structure has an index which corresponds to the index of a gray value range mentioned previously. Every node in the lists is a record of the location in C. A process of traversing every pixel of C is applied, in which we put the visited pixel into the rear of the corresponding list according to the index of its gray value. For example, if the location of 4866 in C is visited and the its value is 165, since the gray value belongs to the range of [152, 183] whose index is 6, we add a node with the value 4866 to the rear of list 6. We use a pseudo-random generator to permute the traversing order of the pixels in C instead of scanning through C sequentially. We use the permuted order to visit each pixel in C only once. This pseudo-random mechanism aims to achieve cryptography. It means that if an illicit user does not have the seed of the pseudo-random generator that is used in the image hiding process, he or she cannot easily find out the correctly traversing order. So, the sequent steps of extracting the hidden image cannot be followed successfully.

After constructing the list structure, we find out for every pixel in S the corresponding pixel in the list structure which have the same index to accomplish the replacement work. We process every pixel of S in the zigzag order mentioned previously. For each visited pixel p_s in S with index k, we extract the head element in the kth list of the list structure which denotes a location in C, and then replace the gray value of this location with the gray value of p_s. For example (see Fig. 8), if we visit a pixel in S with gray value 160 and index 6, then we exact the head element from list 6, whose value 4866 is a location in C where we want to do the replacement work. The gray value of the visited pixel then is used to replace the gray value appearing in location 4866 of C, i.e., 165 is changed into 160. The replacement process is finished after all pixels in S are processed.

B. Inverse Image Differencing

In the inverse image differencing process which produces a stego-image, for each difference values d' in the processed cover difference image, an inverse difference calculation of this value is applied to find the gray values (g'_{i-1}, g'_i) of the corresponding two-pixel subimage (p'_{i-1}, p'_i) of the stego-image whose difference value is d'. Because we want to cause less distortion to the human perception, the information about the gray values (g_{i-1}, g_i) of the corresponding two-pixel subimage (p_{i-1}, p_i) in the original cover image is needed. Assume the difference value of (g_{i-1}, g_i) is d. We produce (g'_{i-1}, g'_i) according to the equations

$$g'_{i-1} = g_{i-1} - \left\lfloor \frac{d'-d}{2} \right\rfloor$$

$$g'_i = g_i + \left\lceil \frac{d'-d}{2} \right\rceil.$$

The above equations together satisfy the requirement that the difference of g'_{i-1} and g'_i is d'. The equations cause changes in g_{i-1} and g_i nearly equally to produce g'_{i-1} and g'_i, so the distortion caused by changing g_{i-1} and g_i is averaged over this two-pixel pair, resulting in less perception. Some of the calculations may cause g'_i or g'_{i-1} to fall off the boundary of the range [0, 255] of a pixel value. In such cases, we set the falling off pixel value to the boundary value, i.e., 0 or 255, and readjust the other pixel value to a new value to preserve the difference value of

g'_{i-1} and g'_i to be d'.

C. Embedding of Leading Information

Since an extra table is constructed to record the index of the gray value of each pixel during the replacement process, the table must be used in the process of extracting the secret image from the stego-image. We can leave the table in a separate place or just hide it into the cover image. We adopt the latter choice in this study and embed it in the cover image. For this, we use the Huffman coding method to reduce the size of the table, since the indices in our proposed method, within 0 to 7, are mostly 3 or 4. The bits required to represent an index can thus be reduced from 3 bits to about 1.9 on the average after using the Huffman coding method. Firstly, we embed into the cover image some extra information, i.e., (1) the width and height of the secret image, (2) the number of quantized ranges, the boundaries of each range, the number of elements in each quantized range, which are needed to construct the Huffman tree, and finally (3) the bit stream of the indices which are generated by the Huffman coding method. The leading formation above provides a way to use different number of ranges and different range boundaries in the embedding process. All of the leading information are considered as a bit stream. The bit stream is embedded into the LSB's of the cover image randomly. We walk through the cover difference image using a pseudo-random number generator and embed every six bits of the bit stream into a pixel-pair of the original cover image, i.e. each pixel in this pixel-pair takes three bits in the rightmost bits. After embedding all of the bit stream, we use the rest of the cover difference image to hide the secret difference image.

3. Process of Extracting Hidden Image

The process of extracting the hidden image is proceeded by using first the seed of the pseudo-random number generation to extract out the data that are embedded in the stego-image. A stego-difference image S_d is produced from to the stego-image S using a method similar to that for producing the cover difference image in the hiding process, and the seed of pseudo-random number generation is used to produce the same traversing order for visiting S_d as in the embedding process.

Firstly, we extract the leading information embedded in the stego-image. The leading information contains the width and the height of the secret image and the information for building the Huffman tree of the range indices. And following the leading information a bit stream of the range indices which are represented by the Human coding. All the leading information are extracted directly out from the three LSB's of the gray values of each pixel-pair in S that corresponds to the visited pixel of S_d.

The gray values of the embedded secret difference image are extracted by visiting the rest of the pixels of S_d. We need to rearrange S_d into an eight-list structure as in the embedding process. The same list which was used in the embedding process is reconstructed by the same traversing order. And then the list of the range indices and the eight-list structure are used in extracting the gray values from S_d to build the secret difference image. In short, the secret image can be recovered by applying an inverse differencing process to the secret difference image.

4. Experimental Results

In our experiments, we use four cover images shown in Fig. 9, each with size 512*512, and two secret images shown in Fig. 10, each with two sizes of 256*512 and 256*256. Some embedding results without the leading information are shown in Fig. 11. The resulting stego-images that include the leading information are shown in Fig. 12. It is found that the embedding results are similar to the cover images, and the peaks of the signal-to-noise (PSNR) values are high (See Table 1). The quality is good even in a stego-image with a poor PSNR. This demonstrates that changes in edge areas will not be noticed by the human visual system.

5. Conclusion

We have proposed a novel method for embedding a gray-valued image into another without producing noticeable changes. The method includes image differencing operation to create difference images from which similar values can be found easily in doing gray value replacement. The method utilizes the characteristic of the human visual system's sensitivity to embed more secret data. It preserves the values of the embedded image precisely and causes less degradation in the stego-image. The method not only provides a way for embedding large quantities of data into cover images with imperception, but also offers an easy way to accomplish cryptography. In the future, we will study the problems of embedding data without extra tables and embedding data in color images.

References

[1] J. T. Brassil, S. Low, N.F. Maxemchuk, and L. O'Gorman, "Electronic Marking and Identification Technique to Discourage Document Copying," *IEEE J. Selected Areas Commum.*, vol. 13, no. 8, pp.1495-1503, Oct .1995.

[2] L.F. Turner, "Digital data security system," Patent IPN WO 89/08915, 1989.

[3] S. Walton, "Image Authentication for a Slippery New Age," *Dr. Dobb's Journal*, pp. 18-26, April 1995.

[4] W. Bender, D. Gruhl and N. Morimoto, "Techniques for Data Hiding," *Proc. Of SPIE*, vol. 2420, Feb. 1995.

[5] E. Koch and J. Zhao, "Towards Robust and Hidden Image Copyright Labeling," *Proc. IEEE Nonlinear signal and image processing*, pp. 452-455, June 1995.

[6] Ingemar J. Cox, Joe Kilian, F. Thomson Leighton and Talal Shamoon, "Secure Spread Spectrum Watermarking for Multimedia," *IEEE Transacations on Image Processing*, vol. 6, no. 12, Dec. 1997.

[7] Miin-Shenq Liaw and Ling-Hwei Chen, "An Effective Data Hiding Method," *Proc. IPPR Conf. CVGIP*, pp.146-153, Taiwan, R.O.C., 1997

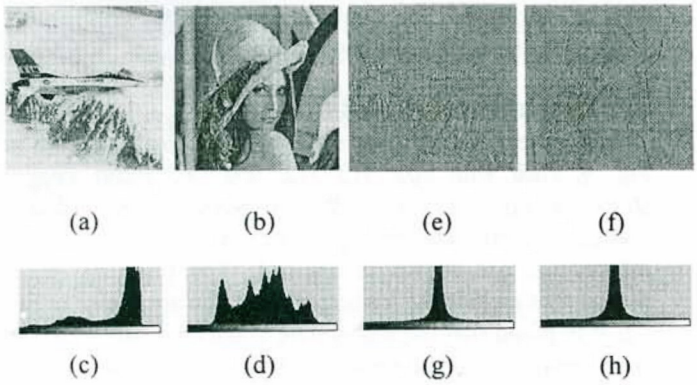


Figure 1: (a) The image of "F-16". (b) The image of "Lena". (c) The histogram of "F-16". (d) The histogram of "Lena". (e) The difference image of "F-16". (f) The difference image of "Lena". (g) The histogram of the difference image of "F-16". (h) The histogram of the difference image of "Lena".

Figure 2: Images of original "Lena" (the top leftmost) and results after inserting noise into the least significant bits (from one bit to seven bits).

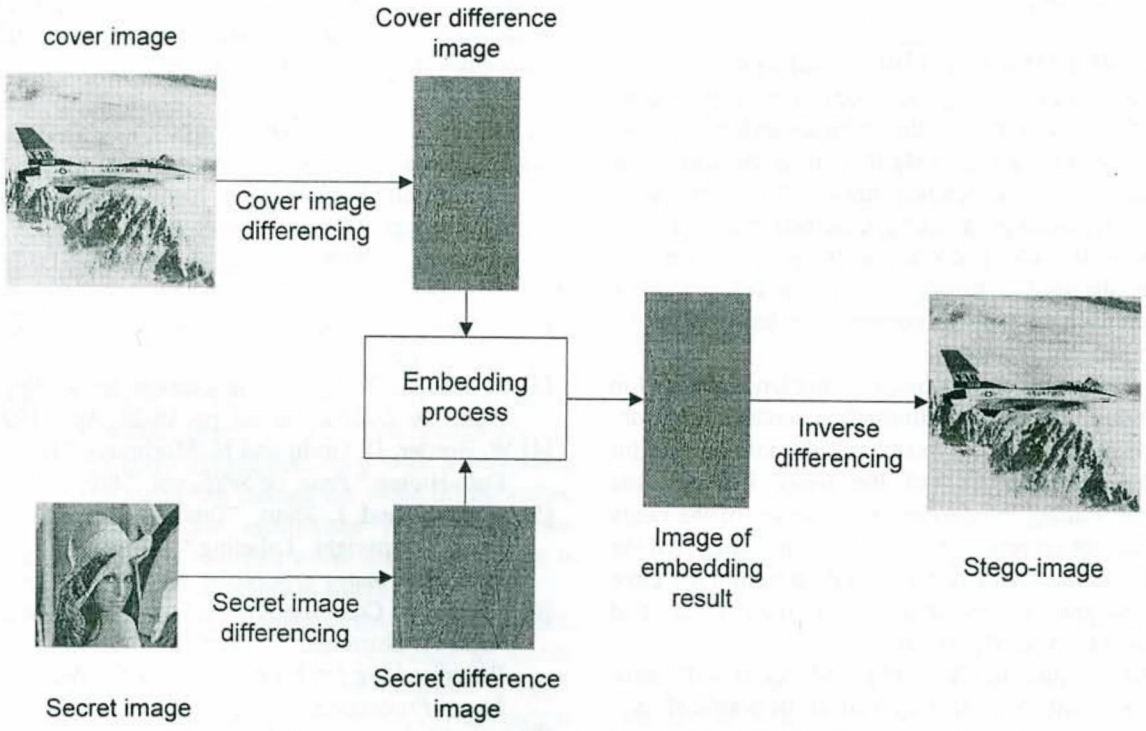


Figure 3: Illustration of proposed image hiding method.

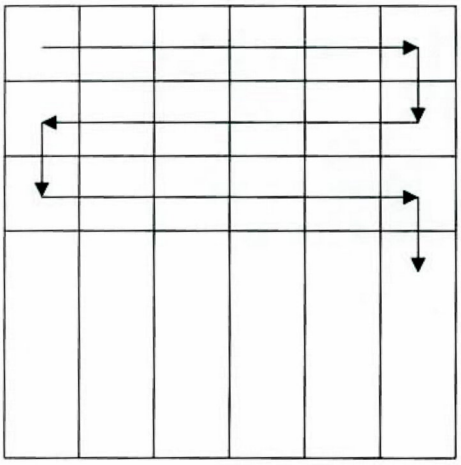


Figure 4: The zig-zag order used for calculating difference values of both cover image and secret image.

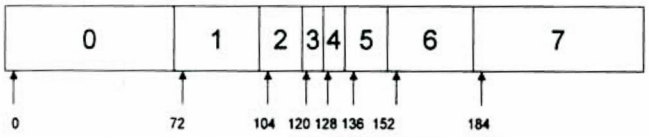


Figure 5: Quantization of gray values 0 through 255 into eight unequal ranges. An index 0 to 7 is assigned to each of the ranges.

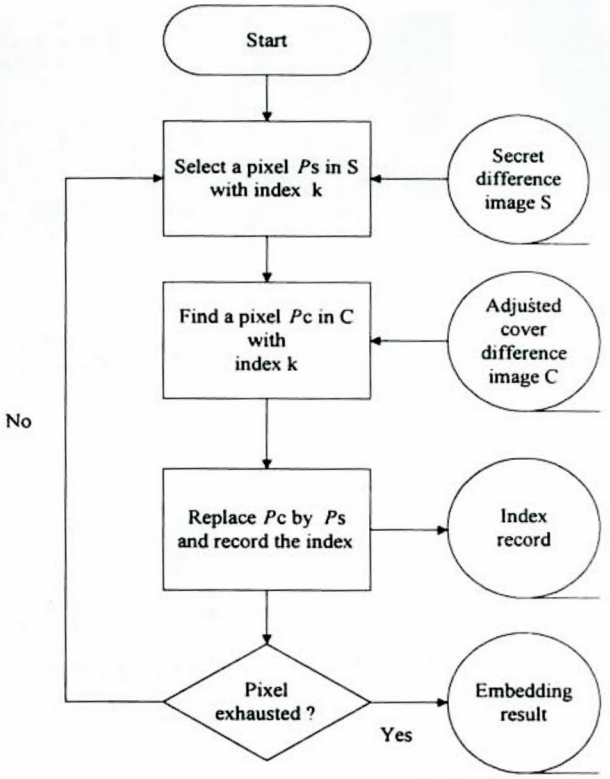


Figure 6: The system overview of the gray value replacement process

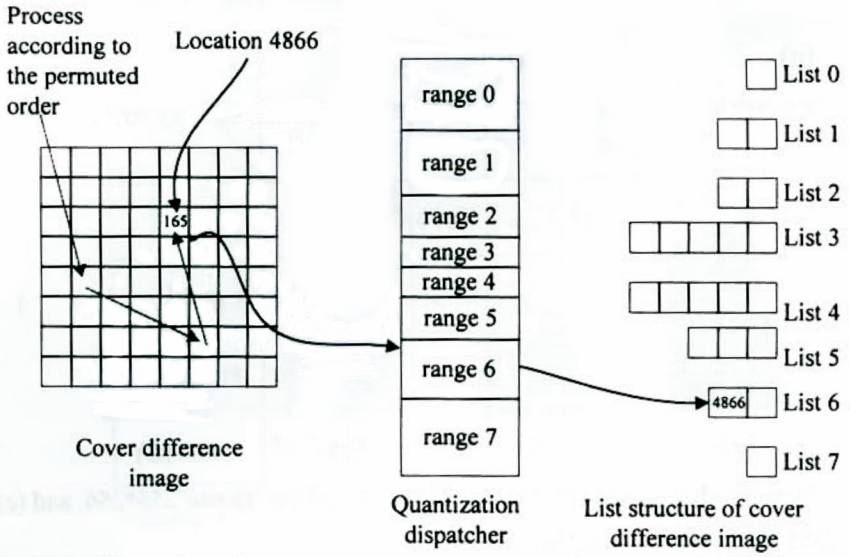


Figure 7: An illustration of rearranging cover difference image data into a list structure

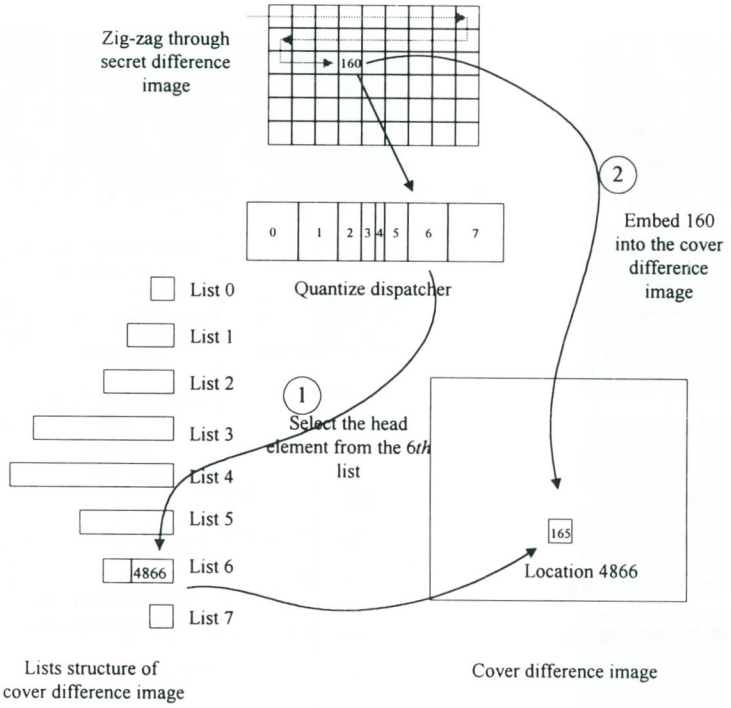
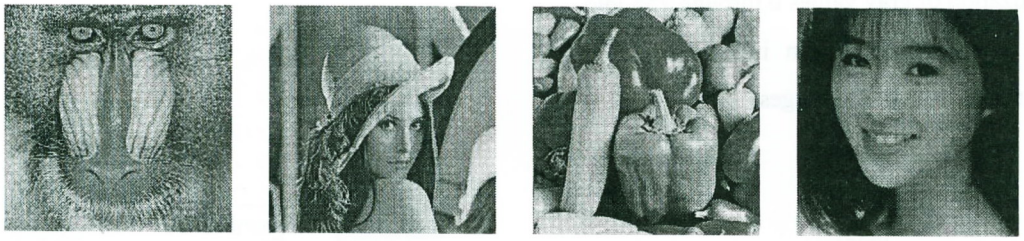
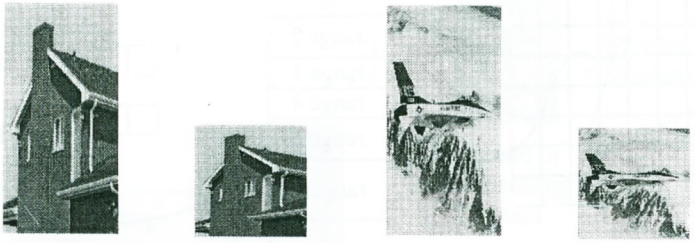


Figure 8: An illustration of the proposed process for embedding data into a cover difference image.



(a) (b) (c) (d)

Figure 9: The cover images (512*512). (a) "Baboon", (b) "Lena", (c) "Peppers", and (d) "Norica".



(a) (b) (c) (d)

Figure 10: The embedded images.(a) "House" 256*512, and (b) "House"256*256, and (c) "F-16" 256*512, and (d) "F-16 256*256.

(a) (b) (c) (d)

Figure 11: The stego-image results of embedding (a) "House" 256*512, (b) "House" 256*256, (c) "F-16" 256*512, and (d) "F-16" 256*256.

(a) (b)

Figure 12: The stego-image results of embedding (a) "House" 256*256, and (b) "F-16" 256*256 and all of the leading information.

Table 1. The PSNR values of the stego-images.

PSNR (Secert images / Cover images)	House			F-16		
	256*512	256*256	256*256@	256*512	256*256	256*256@
Baboon	30.33	44.29	41.77	32.24	41.71	40.22
Lena	39.63	44.90	42.58	33.96	40.65	38.99
Peppers	39.53	44.16	42.18	33.82	40.50	38.81
Norica	36.36	41.39	39.93	30.01	34.72	34.15

@ denotes that embeds the leading information also.