2011 2nd International Conference on Business, Economics and Tourism Management
IPEDR vol.24 (2011) © (2011) IACSIT Press, Singapore

A New Method for Image Security and Data Hiding in Image

Kaveh Ahmadi[1], Maral Mohamadi Zanjani[2]

[1] Young Research Club, Islamshahr Branch
[2]Young Research Club, Islamshahr Branch

Abstract. For image security the cryptography algorithm should be applied to every pixels of image and in the receiver side all of the pixels must be decoded. Coding and decoding image, therefore, takes a lot of time because of its huge number of pixels. A common and faster solution for image security is image-hiding instead of image coding. In fact in this method, pixels of the image will be hidden in another image called cover image and in the network cover image distribute, in the receiver side image will be extracted from cover image. Commonly image-hiding algorithms work in frequency-domain, make a lot of changes to the cover image and they are time consuming. In this paper an image-hiding algorithm is introduced that works in spatial-domain. Our algorithm is faster and makes a little change to the cover image indistinguishable by human eyes. Our method uses the difference of pixels instead of the color of pixels.

Keyword: Image Hiding, Image Coding, Secret Image, Cover Image.

1. Introduction

In the physical world when we want to hide an object, we wrap it in some unmarked paper. This work is similarly done in the digital world but here we have the pixels in the cover image instead of wrap paper. Data hiding is referred to as a process to hide data (representing some information) into cover media. That is, the data hiding process links two sets of data, a set of the embedded data and another set of the cover media data. The relationship between these two sets of data characterizes different applications[1].

Because of the human visual system's low sensitivity to small changes and the high flexibility of digital media, one can easily make small changes in digital data with low perceptibility. Consequently, many interesting applications of data hiding in digital media can be created, like watermarking, caption data embedding, secret message delivery, etc. The purposes and requirements of such applications vary and are illustrated in Table 1[2][3][4].

Drawing analogy between data hiding and communication, the embedding methods serve as a physical communication layer, on top of which other functionalities and features can be built. For instance, security issues may be handled by top layers in authentication applications. In these models, the major objective of an adversary is to forge authentication data so that an altered document can still pass authentication tests[5].

Most of these algorithms work in frequency-domain[6]. However working in frequency- domain is very complex and takes more time and makes more changes in the cover image. In this paper we introduce a new algorithm that works in spatial-domain instead of frequency domain. Finding same pixels in the secret image and the cover image is a difficult problem. Our method solves this problem by employing the difference of adjacent pixels as the main metric.

We propose a new method to embed a grey-valued image into a grey-valued cover image. It can be used for delivering secret images such as confidential images, military maps, etc. The method is designed in such a way that the grey 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 cover-image. On the other hand, because the resulting cover-image usually contains a large quantity of embedded data, it may seriously be degraded. The proposed method, however, produces indistinguishable changes in the resulting cover-image. Since the precision of pixel grey values may be destroyed when transforming them back and forth between the frequency and spatial domains, we adopt a way of embedding the secret image into the cover image directly in the spatial domain.

Table 1: Applications of data hiding in digital media[7][8]

Application	Purpose	Amount of data to be embedded	Difficulty of detection	Authentication
Watermarking	Copyright protection	Small	No	Yes
Caption embedding	Adding information into digital data	Large	No	No
Secret Message delivery	Sending a message without attention	Large	Yes	No
Key distribution	Distribute the session key with security	Small	Yes	No
Image hiding	Sending an image with security	Large	Yes	No

2. Methodology and Materials

2.1 Basic of method

Now, perform the operation of 'image difference' to the two images in the following way: for every pixel value g_i in either image, where i = 1, 2,. . . , let $g_i'=g_i - g_{i-1} + 128$ where g_{i-1}, is the grey value of a neighboring pixel to g_i , and create a new image with all g_i' as the new pixel values and call it the difference image. Adding 128 is for normalizing the value and reducing the number of negative value. The details of image differencing (ID) in our proposed method, which are a little different from what just mentioned, will be in the following section. The pixel values of the difference images are concentrated near 128 resulting from the grey-value similarity between adjacent pixels. The histograms of the difference images are quite similar in shape, in contrast with those of their original images, which are quite different. This similarity helps us exploit a way to embed a secret image easily by grey value replacement using the difference metric.

The amount of pixel-value change in a smooth image area is smaller than that in an edge or high-contrast area. Edge or high-contrast areas in general show greater changes. We use this characteristic to hide more data in high-contrast areas and less in smooth areas. This helps to hide more data indistinguishable in an image[9][10].

The proposed hiding method consists of three major parts: 1) cover ID processing 2) secret ID processing and 3) the embedding process, which are described next.

2.2 ID for Cover and Secret Images

Cover and secret images used in the proposed method are assumed to have 256 grey values. To get a difference image from a given cover image, we obtain the grey value G of a pixel in the resulting image from every non-overlapping two-pixel of the cover image, in a zigzag order,

$$\mathbf{d_i} = [\mathbf{g_{i+1}}-\mathbf{g_i}+128] \bmod 256 \qquad (1)$$

g_{i+1} and g_i are the grey values of pixels, respectively in zigzag order. Adding 128 is for normalizing the value and reduce the number of negative value, and MOD 256 is for normalizing the values to the range of [0 to 255]. This quality way the number of out of range pixels become very small and this normalization doesn't destroy the images. The resulting data stream is just a half of the cover image in size. The process of obtaining the difference image from the secret image is the same as that for the cover image except that we calculate the difference value of the grey values of every two adjacent pixels, instead of every non-overlapping two-pixel, in zigzag order. So the size of the resulting stream is the same as that of the secret image.

We save the gray value of top-leftmost pixel of the secret image. Other differencing methods assume that this value is 128[6][11]. In the sequel we call the difference image from the secret image as the secret difference image, and that from the cover image as the cover difference image.

We create smaller ranges near 128 and larger ones far from 128 for the purpose of better replacement with less noticeable results. This technique helps us to embed more information in high-contrast areas with less deterioration of the image.

2.3 Pixel Replacement

For each pixel P_s in the secret difference image S, we find a pixel P_c in the cover difference image C whose value has the same index as that of P_s, and hide P_s, by replacing the value of P_c, with P_s. Although the grey-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 it is necessary to adjust the values of some pixels in C into the new values which are insufficient for embedding before the whole replacement work begins. Replacement has three phases as follows:

Phase 1 (Cover Image Recovery): For the purpose of easily finding the desired P_c, in C which has the same index as P_s, we rearrange all the pixels in C into an n-list structure as illustrated in Figure 1. Every list in the structure has an index which corresponds to the index of a grey-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 grey value. For example, if pixel 4866 in C with a value of 165 is visited and if the grey value belongs to the range whose index is k, then we add a node with the value 4866 to the list k.

Phase 2 (Cryptography): We use a random key generator to permute the traversing order of the pixels in C instead of scanning through C sequentially. We use the permutation order to visit each pixel in C only once. This random key generator mechanism aims to achieve cryptography. This means that if an illicit user does not have the key used in the image hiding process, the user cannot easily find out the correct traversing order. So, the steps of extracting the hidden image cannot be followed successfully. This method guarantees that only the correct user that has the key can extract the image.

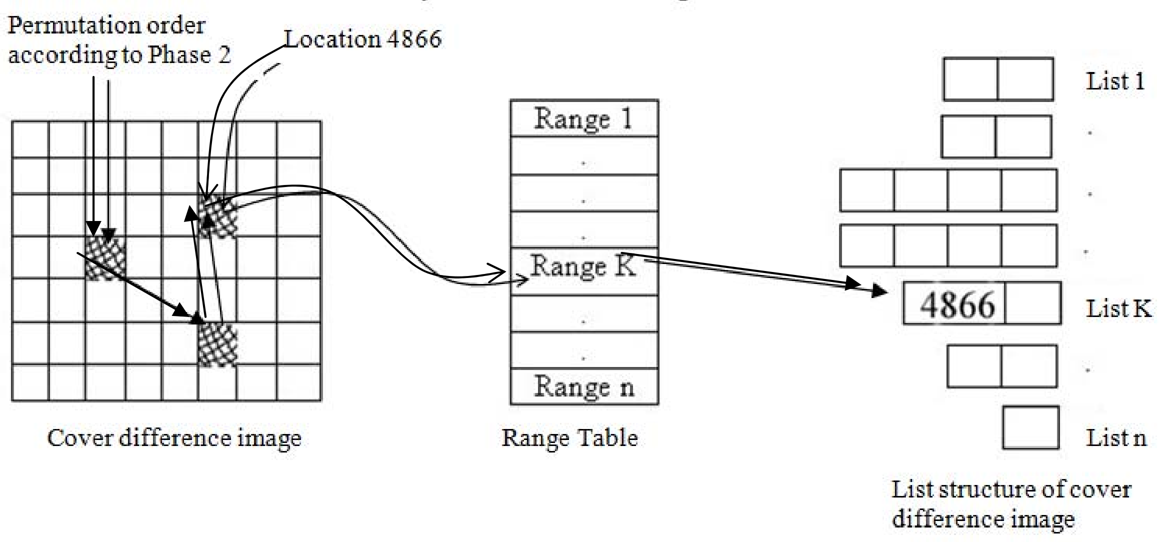

Figure 1 Assign Pixel value to the List

Phase 3 (Replacement): After constructing the list structure we find for every pixel in S the corresponding pixel in the list structure that has the same index to replace. 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 K_{th} list of the list structure which denotes a location in C, and replace the grey value of this location with the grey value of P_s. For example in Figure 2, if we visit a pixel in S with grey value 168 and index k , then we extract the head element from list k, whose value 4866 is a location in C where we want to do the replacement work. The grey value of the visited pixel is used to replace the grey value appearing in location 4866 of C, i.e. 165 is changed into 168. The replacement process is finished after all pixels in S are processed. The grey value replacement process described is efficient because the one-to-one mapping between the pixels of the secret difference image and those of the cover difference image in the process can be built by traversing both the cover difference image and the secret difference image only once.

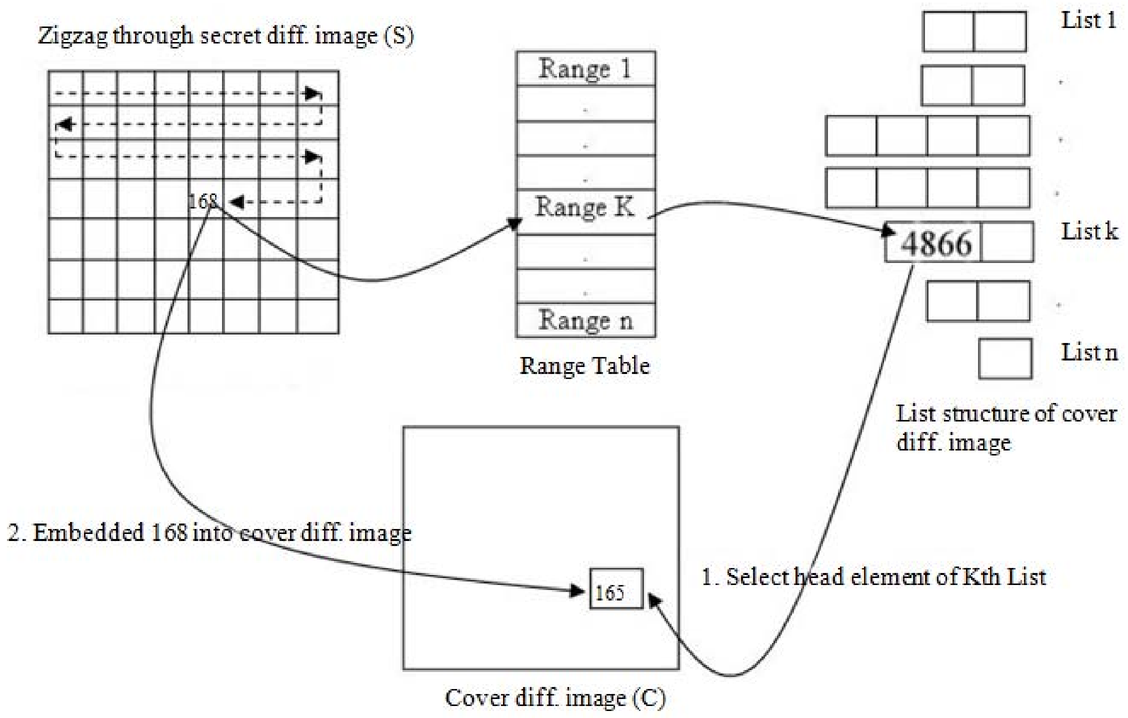
Zigzag through secret diff. image (S)
168
Range 1
Range K
Range n
Range Table
List 1
List k
4866
List n
List structure of cover diff. image
2. Embedded 168 into cover diff. image
1. Select head element of Kth List
165
Cover diff. image (C)

Figure 2 The process of hidding

2.4 Inverse ID

In the inverse ID process which produces a stego-image (Cover + Secret images), for each difference values d' in the processed cover difference image, an inverse calculation for this value is performed to find the grey values (g'_{i-1},g_i') of the corresponding two-pixel sub image (p'_{i-1},p_i') of the stego-image whose difference value is d'. Because we want to cause minimum perceptive distortion, the information about the grey values (g_{i-1},g_i) of the corresponding two-pixel sub image (p_{i-1}, p_i) in the original cover image is needed. Assume the difference value of (g_{i-1} ,g_i) is d[1]. We produce (g'_{i-1},g_i') according to the following equations:

$g'_{i-1}=g_{i-1}-((d'-d)/2)$

$g_i'=g_i+((d'-d)/2)$ *(2)*

The two equations together satisfy the requirement that $d'=|g'_{i-1}-g_i'|$. Because the equations cause changes in g_{i-1} and g_i, nearly equally to produce g'_{i-1} and g_i', the distortion caused by changing g_{i-1}, and g_i, is averaged over this two-pixel pair and so is less. 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 pixel value to the boundary value, i.e. to 0 or 255, and re-adjust the other pixel value to a new value to preserve the difference value of g'_{i-1} and g_i' to be d'.

2.5 Embedding the Secret Key

Since an extra table is constructed to record the index of the grey value of each pixel during the replacement process, the table must be available and used in the process of extracting the secret image from the stego-image[12]. One can send the table to receiver using a separate mechanism or just hide it into the cover image. The latter way needs a bit extra processing. More specifically, we embed into the cover image the following extra information: (a) the width and height of the secret image, (b) the number of quantized ranges, the boundaries of each range, the number of elements in each quantized range, (c) the gray value of top-leftmost pixel of the secret image, and finally (d) the table of indices organized as a bit stream. This extra information provides a way to use different numbers of ranges and different range boundaries in the embedding process. We call this extra information **secret key**, because without this information the secret image can't be recovered.

The secret key, taken as a bit streams, is embedded into the LSBs of the cover image randomly. We walk through the cover difference image using a pseudorandom number generator and embed every 8 bits of the bit

[1] d calculated in the ID for secret and cover image

stream into a pixel-pair of the original cover image, i.e. each pixel in this pixel-pair takes 4 bits as the rightmost bits. The pseudorandom number generator and the random key generator that describe in section 2-2, phase (2) must exist in the two communication sides (sender and receiver).

2.6 Extracting the Hidden Image

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

More specifically, we extract the leading information embedded in the stego-image directly from the four LSBs of the grey values of each pixel-pair in stego-image that corresponds to the visited pixel of secret image. The grey values of the embedded secret difference image are extracted then by visiting the rest of the pixels. We rearrange pixels, into an n-list structure as in the embedding process. The same list used in the embedding process is reconstructed by the same traversing order. And then the list of the range indices and the n-list structure are used in extracting the grey values and building the secret difference image. In short, the secret image can be recovered by applying an inverse Difference process to the secret difference image.

3. Conclusion

We have proposed a novel method for embedding a grey data. The method not only provides a way for embedding valued secret image into a cover but also the preserves the quality of cover images and secret image with no loss. The method produces the stego-image but also offers an easy way to accomplish image cryptography with random key generator and pseudorandom number generator. In near future we intend to extend our method to color image. Color images have more problems for finding the best pixel in the cover image for replacement[13], because color images have three ranges from 0 to 255 and color image hiding usually involves quantizing a full color image. Then the embedding and extraction procedures of this related scheme would be difficult [4][6].

4. References

[1] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, Reversible Data Hiding, *IEEE Transactions on circuits and systems for video technology,* 2006, **10** (16): 354-362.

[2] B. J.T. Low, S. Maxlmchuk, K.F and O. Gorman, Electronic Marking and Identification Technique to Discourage Document Copying, *IEEE J. Sel. Areas Common* 2005, **20** (13): 1495-1503.

[3] W. Bender, D. Gruill, N. Moimo and A. Lu, Techniques for Data Hiding, *IBM System Journal* 1996, **35** (3): 311-336.

[4] Y. Hui Yu, C. Chen Chang, I. Chang Lin, A new steganographic method for color and grayscale image hiding, *Computer Vision and Image Understanding* 2009, **14**(1): 45-58.

[5] M. Wu, Data Hiding in Binary Image for Authentication and Annotation, *IEEE Transactions on multimedia* 2004, **12** (6): 140-152.

[6] W. H. Tasi, Spatial-Domain Image Hiding, *Image Signal Processing IEEE* 2000, **147** (7): 29-37.

[7] C.I.J. Kilian, J. Lighton, ET, and Shamoon T., Secure Spread Spectrum Watermarking for Multimedia, *IEEE Trans. Image Process* 1997, **6** (12): 1673-1687.

[8] F. Ge, Linfei Chen, D. Zhao, A half-blind color image hiding and encryption method in fractional Fourier Domains, *Optics Communications* 2008, **281** (2): 4254-4260.

[9] Cheddad, J. Condell, Kevin Curran, Paul Mc Kevitt, Digital image steganography: Survey and analysis of current methods, *Journal of Signal Processing* 2010, **90** (9): 727-752.

[10] C. J.J, and Manjunath, U.S., A Robust Embedded Data From Wavelet coefficients, *Proc. SPIE-Int. Soc. Opt. Eng.* , USA, 1998: 308-317.

[11] B. M. Benartolini, F., Cappillinl. V, and Piva, A., A DCT- Domain System for Robust Image Watermarking, *Journal of Signal Possessing* 1998, **44** (10): 357-372.

[12] R. Zan Wang, Image hiding by optimal LSB substitution and genetic algorithm, *Pattern Recognition* 2001, **12**

[13] R. Zan Wang, Y. De Tsai, An image-hiding method with high hiding capacity based on best-block matching and k-means clustering, *Pattern Recognition* 2007, **40** (4): 398-409.