REVERSIBLE INFORMATION HIDING FOR BINARY IMAGES BASED ON SELECTING COMPRESSIVE PIXELS ON NOISY BLOCKS

Michiharu Niimi and Hideki Noda

Kyushu Institute of Technology
680–4 Kawazu, Iizuka, 820–8502 Japan
E-mail: {niimi,noda}@mip.ces.kyutech.ac.jp

ABSTRACT

This paper proposes a reversible information hiding method for binary images. A half of pixels in noisy blocks on cover images is candidate for embeddable pixels. Among the candidate pixels, we select compressive pixels by bit patterns of its neighborhood to compress the pixels effectively. Thus, embeddable pixels in the proposed method are compressive pixels in noisy blocks. We provide experimental results using several binary images binarized by the different methods.

Keywords: reversible informaiton hiding, binary image, compressive pixel

1. INTRODUCTION

The information hiding methods referred to as reversible, lossless, distortion free or invertible insert message by modifying the cover image, but enable the exact restoration of the original image after extraction the embedded message. Several reversible information hiding techniques have been developed [1][2][3][4][5], which uses gray scale images or full color images as cover media. These methods are not appropriate for binary images.

In this paper, we propose a reversible information hiding for binary images. A common approach of reversible information hiding is to select an embedding area in an image, and embed both the message and the original value in the embedding area into it. We employ the complexity measure[6] to select noisy block as embedding area, and select embeddable pixels from the noisy block by bit patterns of its neighborhood. The whole of embeddable pixels is compressed and embedded into the cover image to restore the original cover image. The embedding is performed by replacing the pixel value of embeddable pixels with the bit information to be hidden.

2. PROPOSED METHOD

The proposed method is a compression based reversible information hiding. We embed not only the message bit but also the compressed data of the whole of the original embeddable pixels. For the sort of the reversible information hiding, it is important to select pixel which does not affect visual perception when the pixel value is flipped, in addition, these pixels must be compressible effectively because the space made after the compression is for embedded message. We extract noisy block using complexity measure, and select embeddable pixel from the noisy block by bit patterns of its neighborhood.

2.1 Complexity measure for binary images

The total length of the black-and-white border equals to the summation of the number of color-changes along the rows and columns in the image when we use four connectivity for both objects and background. For example, a single black pixel surrounded by white background pixels has a border length of 4. We assume the image frame is always squared $m \times m$ pixels in size. Let us count the number of color-changes in the interior area of the image. Therefore, the minimum border length is 0 (in either black or white pattern), while the maximum one is $2 \times m \times (m-1)$ (in checkerboard patterns). Thus, the image complexity measure is defined by the following ratio.

$$\alpha = \frac{k}{2 \times m \times (m-1)} \tag{1}$$

Where, k is the total length of black-and-white border in the image. So, the value ranges over

$$0 \leq \alpha \leq 1. \tag{2}$$

2.2 Embedding

The message to be hidden is embedded into noisy blocks on cover images. Noisy blocks are extracted by the simple thresholding of complexity measure with α_{TH}. In other words, blocks whose complexity is greater than α_{TH} are used for the embedding.

Only half of the all pixels in noisy blocks is candidate for the embedding. Fig 1 shows the position of the candidate pixels with 8×8 in size. In Fig 1, doted small squares (or small white squares) represent the candidate pixels.

We select embeddable pixels from the candidate pixels by neighborhood bit patterns. Thus, one need to share both the complexity threshold and the neighborhood bit patterns to extract embedded information.

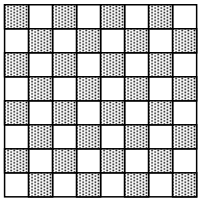


Fig. 1: Position of the candidate pixels for embeddable (8 $\times$ 8 noisy block)

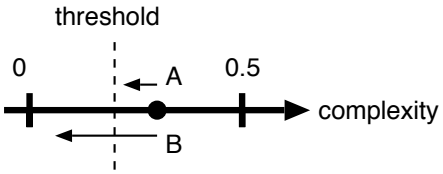


Fig. 2: Complexity changes of noisy blocks with replaceable pixels flipped

We can automatically select the embeddable pixels after we determine the two key value:the complexity threshold and the bit patterns of neighborhood. In the proposed method, however, we determine noisy blocks by simple thresholding of the complexity measure. When we flip the selected pixels as embeddable randomly, which is equals to that we embed random data into the selected pixels, the complexity of the noisy block may decrease below the threshold (case B in Fig2). In that case, we can not extract the noisy block into which we embed data because the complexity of the noisy block is smaller than the complexity threshold. Therefore, we use the noisy blocks whose complexity is always greater than the threshold when any bit patterns of the embeddable pixels are embedded (case A in Fig2).

In order to recover the original cover images, we embed the compressed data of the original pixel value of the embeddable pixels using the arithmetic coding. The space made after the compression is for the message. Therefore, embedding capacity becomes larger when the embeddable pixels are compressed effectively.

2.3 Extraction and recovering the cover image

At the receiver side, one can extract the bit-stream from complex blocks whose complexity is greater than the threshold α_{TH} used in the embedding by scanning the image in the same order in the embedding. Because not only the complexity threshold but also the bit patterns of neighborhood are shared between the sender and the receiver, the receiver can extract embedded pixels from noisy blocks. We can determine which noisy blocks are used for the embedding in the same manner in the embedding. The extracted bit-stream is separated into the message bits and the compressed bit-stream. The compressed bit-stream is decompressed to reveal the original embeddable pixel. The image is then processed and the original pixel is adjusted. Thus, the original image can be restored completely.

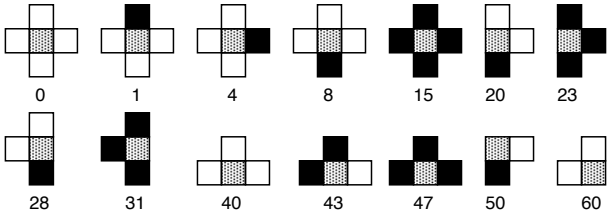


Fig. 3: Examples of neighborhood bit patterns

3. EXPERIMENTAL RESULTS

Two binarization methods were used to make binary images from gray scale images. One is the Otsu's method[7] and another is the matrix dither method. We call the binary image produced by Otsu's method and a binarized image by the matrix dither method *bi-level image* and *dither image* in the following.

Three pictures, 816 $\times$ 612 in size, taken from a digital camera, namely flowers, students and characters image, with full-color mode were converted to gray-scale images and then two kinds of binary images: bi-level image and dither image were produced from the gray scale images.

In the blocks we use on binary images, there are 64 neighborhood bit patterns for the candidate of embeddable pixels including 4 neighborhood, 3 neighborhood and 2 neighborhood. We assign a unique number, which is one of from 0 to 63, to a neighborhood bit pattern. Examples of the number assignment is illustrated in Fig.3.

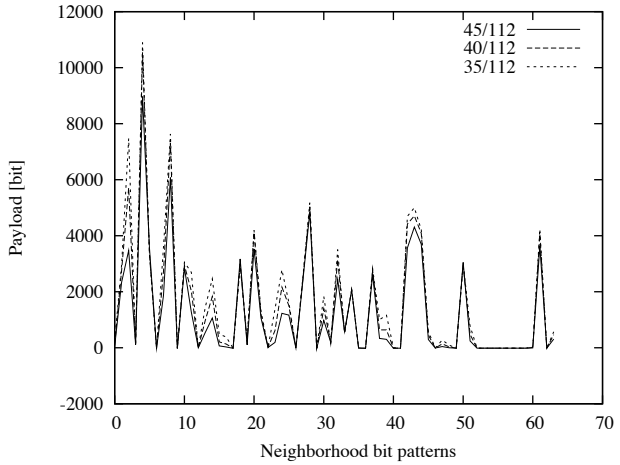
Fig. 4 shows the payload representing the actual space for message to be hidden for the students image with changing neighborhood bit patterns and complexity threshold. The selection of neighborhood bit patterns affects the payload clearly. We also found that the adequate bit patterns of neighborhood are different by image types. In the dither image, we embed larger information when we use the pattern number of 4, 8, 28, 43, 50 or 61, on the other hand, that of the 0, 8, 15, 23, 32, 40 or 47 provide better performance in the bi-level image. In addition, the result indicates that the change of the complexity threshold is not affected to the payload in the dither image.

Fig. 5 shows the compression efficiency of embeddable pixels defined by

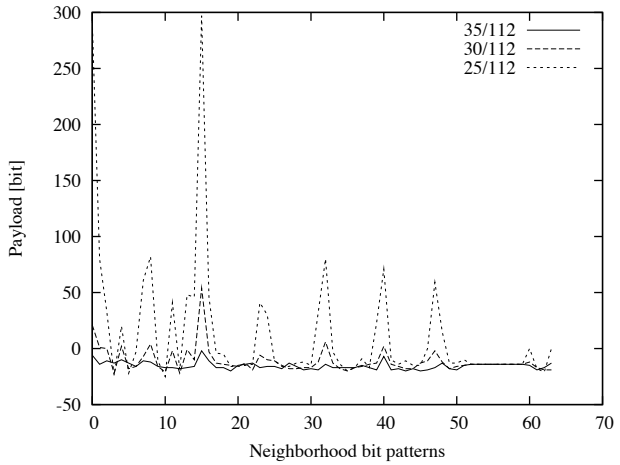
$$\frac{a-b}{a},$$

where a and b represents the number of pixels of the whole embeddable pixels and the number of bits of compressed data of the embeddable pixels. We found that embeddable pixels in the dither image are more compressible than that in the bi-level image.

Fig. 6 shows the payload of three cover images. Adequate bit patterns of neighborhood for the three images are almost same.

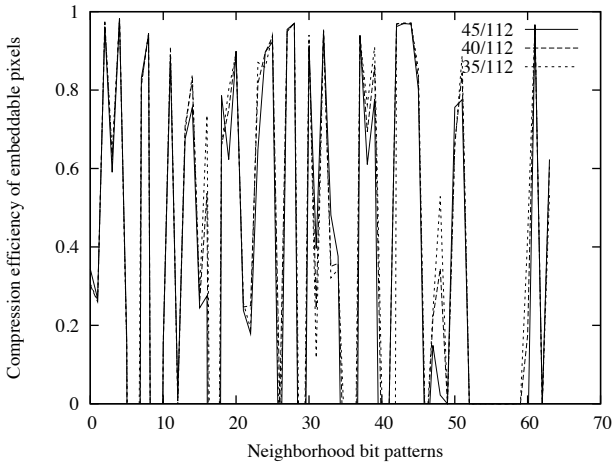


(a) Dither image

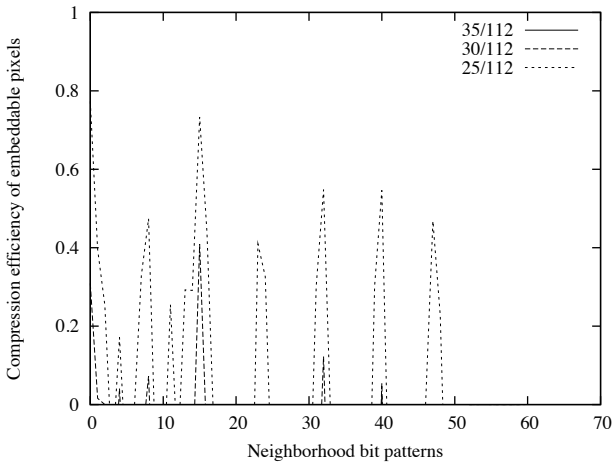


(b) Bi-level image

Fig. 4: Payload with changing neighborhood bit patterns and complexity threshold (students image)



(a) Dither image



(b) Bi-level image

Fig. 5: Compression efficiency of embeddable pixels with changing neighborhood bit patterns and complexity threshold (students image)

Lossless image authentication is one of applications of reversible information hiding. The hash value of the whole cover image is embedded using reversible information hiding. The verification starts with extracting the embedded hash value and the compressed embeddable pixels. The compressed bit-stream is used to obtain the original cover image. After this step, we can recalculate the hash value of the reconstructed image and compare with the extract hash value. We here assume the hash value consists of 128 bits, we apply the proposed method to the lossless image authentication system. The result of the bi-level image and the dither image for the students image is shown in Fig.7. We can not find any image distortion from Fig.7. Only 186 bits are flipped to embed the 128 bits hash value.

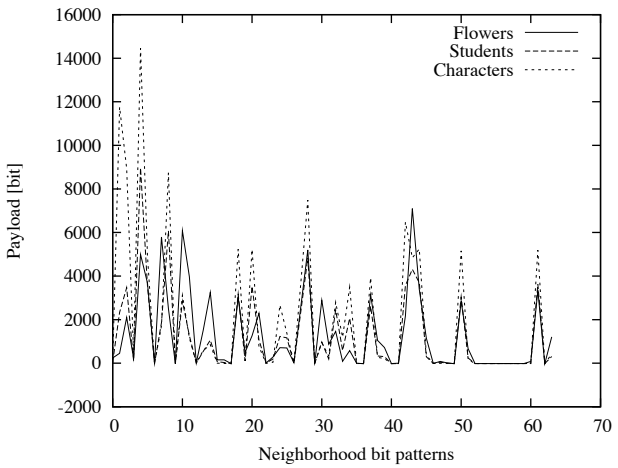
4. CONCLUSION

We proposed a technique for reversible information hiding using complexity measure of binary images. The proposed method selects compressive pixels from noisy blocks as embeddable pixels. To recover the original cover images, we embed the compressed data of the whole of embeddable pixels. From the several experiments, we have found that Adequate bit patterns of neighborhood for the embedding are different by image types. We have shown that only 186 bits are needed to flip the original pixel value to build the lossless image authentication system which embeds 128 bits hash value for the bi-level image of the stdudents image. In addition, this image degradation is not perceptible.

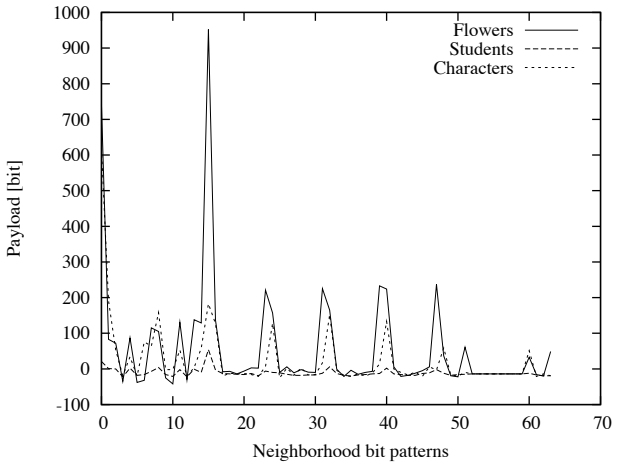
5. REFERENCES

[1] M.Goljan, J. J. Fridrich, R. Du "Distortion-Free Data Embedding for Images" I. S. Moskowitz (Ed.) : LNCS 2137, pp.27-41, 2001.

[2] Celik, M.U., Sharma, G., Tekalp, A.M., Saber, E.,



(a) Dither image (threshold=45/112)



(b) Bi-level image (threshold=25/112)

Fig. 6: Payload of each cover images

(a) cover image

(b) stego image

Fig. 7: Bi-level image of cover and stego (students image)

"Reversible data hiding", Proc.of 2002 Int. Conf. on Image Processing, Vol.2, pp.II-157 - II-160, 2002.

[3] Jun Tian,"Reversible Data Embedding Using a Difference Expansion", IEEE Trans. on Circuits and Systems for Video Technology, Vol.13, No.8, pp.890-896, 2003.

[4] Adnan M. Alattar, "Reversible Watermark Using the Difference Expansion of a Genelized Integer Transform", IEEE Trans. on Image Processing, Vol.13, No.8, pp.1147-1156, 2004.

[5] Zhicheng Ni, Yuu-Qing Shi, Nirwan Ansari, and Wei Su, "Reversible Data Hiding", IEEE Trans. on Circuits and Systems for Video Technology, Vol.16, No.3, pp.354-362, 2006.

[6] M. Niimi, H. Noda and E. Kawaguchi, "Steganography based on region segmentation with a complexity measure," Systems and Computers in Japan, Vol.30, No.3, pp.1–9 (1999).

[7] N.Otsu, "A Threshold Selection Method from Gray-Level Histograms," IEEE Trans. Sys., Man, and Cybernetics, SMC-9, No.1, pp.62-66, 1979.