Abstract—Steganography is a technique that conceals secret data into a cover medium for delivering secret data over public computer networks. Reversible data hiding schemes not only can achieve secret data delivery, but also can restore the cover medium. Histogram shifting is one of the most popular reversible data hiding techniques. Luo et al. presented a reversible data hiding technique that shifts the histogram of prediction error. But the embedding payload of Luo et al.’s method can further be improved. The proposed method uses a difference segmentation strategy and pseudo pixel generation to increase the height of peak in the prediction error histogram. The experimental results show that the embedding payload of the proposed method is higher than that of Luo et al.’s method.

Index Terms—Data hiding, histogram shifting, secret data delivery, steganography.

1. Introduction

Traditionally, private data can be securely delivered to the receiver by adopting a cryptosystem. The cryptosystem generates keys to encrypt/decrypt private data. Using an encrypting procedure makes encrypted private data look like random noise. Encrypted data make it hard for an unintended user to know the information without right keys. Therefore, only the intended receiver with the correct decryption key can fully decrypt the cipher data into plaintext.

However, using a cryptosystem to assist secret data delivery has two drawbacks. First, the encryption process may lead to a high computation cost due to a large size of secret information. Second, the secret data deliveries may fail if an unexpected user intercepts the cipher data transmission. Because the unexpected user is interested in the ciphertext even it looks like meaningless random noise.

Steganography is another method of secret data delivery. The main concept of steganography is to use a cover medium to cover the cipher data and deliver the stego medium to the receiver via public computer networks. The benefit of steganography is that an unexpected user will see a meaningful stego medium with imperceptible distortion. Thus, the unexpected user will be deceived. From this point of view, the inconspicuous differences are one of the most important factors for designing a data hiding scheme. Several different media can play the cover medium in the steganography technique. Generally, a cover medium can be the text, audio, HTML file, video, image, or program code. Here, we use an image as the cover medium in the proposed data hiding scheme.

Steganography techniques can be briefly classified into spatial inconspicuous differences, frequency domain, and compression domain. The spatial domain method tries to conceal secret data into a digital image by modifying pixels values in a cover image for implying secret messages[1]. Spatial domain data embedding is easy to implement and carries a low computation cost. The frequency domain data hiding scheme focuses on modifying coefficients for implying secret data. Generate the image’s coefficients by applying transformation functions (e.g., discrete cosine transform, discrete wavelet transform, etc.) to transform pixel values into coefficients. The frequency domain is time consuming and the embedding payload is limited. But one of the benefits of the frequency domain data hiding technique is that the stego images have better visual quality than the spatial domain data hiding method. To save the storage cost and transmission bandwidth, data compression can significantly reduce the size of digital images. The compression domain data hiding technique modifies the compression procedure for implying the secret data in the compression code[2][3].

Reversibility is an important aspect of secret data delivery in recent years. In some institutions, such as military and distance medical treatment, applications require restoring the original cover image. Thus, reversible data hiding means a cover image carrying some secret message via a reversibility scheme. Then the receiver not only can extract the secret message, but also can fully restore the cover image back to the original one[4][5].
Several strategies can achieve the goal of data hiding reversibility. These strategies can be classified into frequency transformation, differences expansion, and histogram shifting. Vleeschouwer et al. used histogram shifting to conceal secret data into a cover image[3]. Vleeschouwer et al.’s method classifies pixels in a cover image into two sets of circularly interpolating histograms, and then shifts the histograms to conceal secret data. Ni et al. presented a reversible data hiding method that analyzes the pixel distribution in a cover image to generate the pixel histogram[8]. And, peak and zero points were found by scanning the histogram then shifting the histogram for concealing secret data.

Hwang et al. proposed a data hiding technique with reversibility by extending Ni et al.’s method[9]. Hwang et al.’s method utilizes extra information to remember the restoring information, thus the cover image still can be restored. Lin and Hsueh proposed a reversible data hiding technique by using bin exchange to modify histograms[10]. Lin and Hsueh’s successful method increased the pure embedding payload and achieved lower distortion of stego images. Further, Kim et al. presented a reversible data hiding scheme by selecting two peak points and two zero points to improve the embedding payload performance[8]. Then, Luo et al. utilized the pixel prediction strategy to generate predicted pixels to increase the height of peak in prediction error histograms[11]. Because the prediction error is significantly narrowed down in a limited range, the height of peak point has been increased and the embedding payload has been improved. However, the embedding payload of Luo et al.’s method can still be improved.

The embedding payload of histogram shifting data hiding schemes is highly related to the height of peak point and the number of peak points they use. On the other hand, for achieving reversibility, some extra information (i.e., peak points and zero points) needs to be stored. The proposed method uses a difference segmentation strategy and prediction error to increase the height of peak point in order to obtain a higher embedding payload. Also, the proposed method does not need to remember any peak point or zero point information.

2. Related Works

Luo et al. presented a data hiding technique using histogram shifting, in which the secret data is embedded into a cover image by applying a histogram modification strategy[11]. Luo et al.’s main idea is using predefined sample pixels to predict pixels for generating prediction error. The key steps of Luo et al.’s method are described as follows. For simplicity, a cover image is denoted as \( I = \{ p_{i,j} \} \) \( i = 1, 2, \ldots, H \), \( j = 1, 2, \ldots, W \) and \( p_{i,j} \in \{0, 1, \ldots, 255\} \), where \( H \) and \( W \) represent the height and width of the cover image \( I \), respectively. First, the marginal pixels in the cover image are reserved for concealing the extra data used for secret data extraction and image restoration. The extra data is concealed into the marginal pixels by using least significant bits (LSB) replacement. Because the marginal pixels will be modified, thus the original LSB data of marginal pixels are collected for concealing into sample pixels. Second, Luo et al.’s method generates prediction three times, namely Level 1 prediction, Level 2 prediction, and Sample pixel prediction. Note that all of the prediction processes are just for the remaining pixels (i.e., all pixels except marginal pixels).

The first prediction focuses on Level 1 pixels, which are predicted by using its neighboring sample pixels. The sample pixels come from its 45° and 135° directions. For example, let the current Level 1 pixel be \( p_{i,j} \) and predicted value \( p'_{i,j} \) is generated by adopting (1)–(5):

\[
p'_{i,j} = w_{45}H_{45} + w_{35}H_{35} \quad (1)
\]

\[
\mu_{45} = \frac{(p_{i-1,j} + p_{i+1,j})}{2} \quad (2)
\]

\[
\mu_{35} = \frac{(p_{i-1,j} + p_{i+1,j})}{2} \quad (3)
\]

\[
\mu = \frac{(\mu_{45} + \mu_{35})}{2} \quad (4)
\]

\[
\sigma_{45} = \left( (p_{i-1,j} - \mu)^2 + (p_{i+1,j} - \mu)^2 \right)^{1/2} \quad (5)
\]

\[
\sigma_{35} = \left( (p_{i-1,j} - \mu)^2 + (p_{i+1,j} - \mu)^2 \right)^{1/2} \quad (6)
\]

where \( \mu_{45} \) and \( \mu_{35} \) represent the mean values of sample pixels in 90° and 180° directions corresponding to \( p_{i,j} \), respectively.

Then, after the Level 1 pixels prediction has been done, the Level 2 pixels can be predicted by referring to its neighboring sample pixels and Level 1 predicted pixels using (6)–(10):

\[
p'_{i,j} = w_{90}H_{90} + w_{180}H_{180} \quad (6)
\]

\[
\mu_{90} = \frac{(p_{i-1,j} + p_{i+1,j})}{2} \quad (7)
\]

\[
\mu_{180} = \frac{(p_{i-1,j} + p_{i+1,j})}{2} \quad (8)
\]

\[
\mu = \frac{(\mu_{90} + \mu_{180})}{2} \quad (9)
\]

\[
\sigma_{90} = \left( (p_{i-1,j} - \mu)^2 + (p_{i+1,j} - \mu)^2 \right)^{1/2} \quad (10)
\]

\[
\sigma_{180} = \left( (p_{i-1,j} - \mu)^2 + (p_{i+1,j} - \mu)^2 \right)^{1/2} \quad (10)
\]

where \( \mu_{90} \) and \( \mu_{180} \) represent the mean values pixels of 90° and 180° directions corresponding to \( p_{i,j} \), respectively.

Then, the prediction error is the difference between the original pixel and the predicted pixel. When the prediction errors have been calculated, the next step is analyzing the differences to generate a histogram. After that, secret data
can be embedded by applying the histogram modification strategy. Further, the predicted values of sample pixels are predicted by referring to stego Level 1 and Level 2 pixels. Here, sample pixels are used to conceal the information about reserved LSB bits of the marginal pixels. Again, the prediction works by using (6)–(10). The embedding procedure for sample pixels still adopts histogram modification. Because the histogram shifting strategy requires remembering the peak and the zero points, the extra information is embedded into the reserved marginal pixels by using LSB replacement.

For data extraction on the receiver side, the right receiver can extract the secret data from the stego image with the secret data extracting procedure. First, the extra information is extracted by directly taking LSB from the marginal pixels. Then, the same prediction method is adopted to generate the prediction values for sample pixels. After the sample pixels’ prediction, the error histogram has been generated, and then the original marginal LSB data can be extracted by referring to the extra information. According to the property of histogram modification, the sample pixels can be fully reconstructed. Then, the Level 1 and Level 2 pixels prediction procedures are adopted to generate the prediction pixels. Again, the prediction error histogram of Level 1 and Level 2 pixels with the extra information is used to extract secret data. The Level 1 and Level 2 pixels can be fully restored, too.

3. Proposed Method

3.1 Data Embedding

The property of a natural image contains a local area with similar pixels distribution, so the prediction error will be narrowed down to a small range, which will help increase the height of the histogram. In other words, the height of the peak point will be increased. The proposed method contains three main phases, namely: pixel adjustment, prediction error generation, and data embedding. Pixel adjustment is needed to prevent ambiguous results on the data extracting side. Then, pseudo pixels are generated by calculating the mean values of segments in a block. Here, the segments will be generated by segmenting the difference between the maximum value and minimum value in a block. Then, the prediction error is generated by calculating the difference between the pixel value and pseudo pixel value. After the prediction errors have been generated, secret data is embedded into the cover image by applying the histogram shifting strategy.

Because the local area has similar pixels distribution, the proposed method divides a cover image into non-overlapping blocks with the size of \( n \times n \). For simple description, let the cover image \( I = \{ p_{i,j} | i,j = 1, 2, \ldots, H \times W \} \), where \( H \) and \( W \) are the height and width of the image and \( p_{i,j} \in \{0, 1, \ldots, 255\} \). After division, the image is also represented as \( I = \{ B_{i,j} | i,j = 1, 2, \ldots, n \times n \} \), where \( B_{i,j} \) refers to the total number of image blocks and \( B_{ik} = \{ x_{i,j} | i,j = 1, 2, \ldots, n \times n \} \).

For achieving security, the secret data will be encrypted as cipher text by using any crypto system. Thus, the unexpected user can not read real secret messages even if the user has extracted secret information from the stego image. The cipher text is denoted as \( M = \{ b_{i,j} | i,j = 1, 2, \ldots, NM \} \), where \( NM \) is the total bits of cipher text and \( b_{i,j} \in \{0, 1\} \). Actually, the final embedded data \( S \) concatenate \( NE, M, \) and \( RB \), together denoted as \( S = NE || M || RB \), where “||” is the concatenate operation. Here, \( NE \) and \( RB \) are the non-embeddable blocks’ information and first block’s LSB data.

To avoid the underflow and overflow problems, some blocks will be defined as non-embeddable blocks. Here, the underflow problem occurred when the pixel value is close to 0, and the pixel value might become a negative value due to secret data embedding. The overflow problem occurs when the pixel value is closest to the maximum gray value (i.e., the maximum value of a 8-bits gray pixel is 255) and the pixel value might become greater than the maximum gray value due to secret data embedding. If a block satisfies one of following cases, then the block is a non-embeddable block that will not be used for concealing secret data. Let the block’s maximum pixel value and minimum pixel value be generated by \( B_{i}^{\max} = \max \{ x_{i,j} \} \) and \( B_{i}^{\min} = \min \{ x_{i,j} \} \), respectively. The difference between the maximum value and minimum value is generated by \( D_{i} = B_{i}^{\max} - B_{i}^{\min} \). \( N_{\text{seg}} \) is the number of segments.

Case 1: if \( D_{i} \leq N_{\text{seg}} \), then do nothing.

Case 2: if \( B_{i}^{\min} \leq \left[ \frac{3N_{\text{seg}}}{2} \right] \), then add \( B_{i}^{\min} \) into extra information and set \( B_{i}^{\min} \) as 0.

Case 3: if \( B_{i}^{\max} \geq 255 - \left[ \frac{3N_{\text{seg}}}{2} \right] \), then add \( (255 - B_{i}^{\max}) \) into extra information and set \( B_{i}^{\max} \) as 255.

If a block satisfies Case 1, then the block is a smooth block with similar pixels distribution and the block will do nothing. Case 2 and Case 3 are the cases of underflow and overflow, thus the block will not be used for concealing secret data.

For an embeddable block, calculate the difference \( D_{i} \) and segment \( D_{i} \) into \( N_{\text{seg}} \) segments. In order to prevent ambiguous results in the data extracting phase, pixel adjustment is needed. First, a new minimum \( B_{i}^{\min'} \) and new maximum \( B_{i}^{\max'} \) are calculated by using (11) and (12), respectively.

\[
B_{i}^{\min'} = B_{i}^{\min} - \left\lfloor \frac{3N_{\text{seg}}}{2} \right\rfloor \quad (11)
\]
\[
B_{i}^{\max'} = \begin{cases} 
B_{i}^{\max} - \left\lfloor \frac{3N_{\text{seg}}}{2} \right\rfloor, & \text{if } \text{mod}(N_{\text{seg}}, 2) = 1 \\
B_{i}^{\min} - \left\lfloor \frac{3N_{\text{seg}}}{2} \right\rfloor, & \text{otherwise} 
\end{cases} \quad (12)
\]


The original pixels can also be adjusted into new pixel values. The start pixel value of the first segment is $B_i^{min} + 3$. The end pixel value of last segment is $B_i^{max} - 2$. Otherwise, create a gap (i.e., the gap is 4) between two segments. Then, the length of the segment is determined by $\left(\frac{B_i^{max} - B_i^{min}}{N_{seg}}\right)$. The pseudo pixel $y_k$ of the kth segment is determined by $y_k = \left(\frac{\text{seg}_{k}^{\text{max}} + \text{seg}_{k}^{\text{min}}}{2}\right)$, where $\text{seg}_{k}^{\text{max}}$ and $\text{seg}_{k}^{\text{min}}$ are the maximum and minimum values in the segment, respectively. Next, generate the prediction error by calculating the difference between the pixel and corresponding pseudo pixel values.

After all of the prediction errors have been generated, create the prediction error histogram by counting the difference values from the whole image. To observe the result of difference histogram, we found that the three highest peaks are −1, 0, and 1. Thus, −1, 0, and 1 are set as the default peaks. Then, the prediction error is less than or equal to −2 is decreased by two. The prediction error that is equal to −1 is shifted to −2. The prediction error greater than or equal to 2 is increased by one. Fig. 1 illustrates the histogram adjustment concept.

After that, if the prediction error is equal to −2, 0, or 1 and secret bit $b_{i} = 0$, then do nothing. If $b_{i} = 1$, and the prediction error is equal to −2, 0, or 1, then adjust the prediction error to −3, −1, and 2, respectively. If the prediction error is not equal to −2, 0, or 1, then keep the prediction error without any change. And generate the stego pixel by using a pixel’s corresponding pseudo pixel plus the prediction error. Finally, the information regarding the number of non-embeddable blocks is remembered in the first block by using LSB (least significant bits) replacement.

### 3.2 Data Extracting

Data extracting is a reverse process of data embedding. First, extract the non-embeddable block numbers from the first block’s LSB. Then, extract data from the remaining blocks. Every remaining block must belong to one of the following cases.

**Case I.** If $D_{i}^{1} = (B_{i}^{max} - B_{i}^{min}) ≤ N_{seg}$, then no secret data can be extracted.

**Case II.** If $B_{i}^{min} = 0$, then no secret data can be extracted and restore $B_{i}^{min}$ by taking $\left\lfloor \log_2 \left(3N_{seg}/2\right) \right\rfloor$ bits from $S$.

**Case III.** If $B_{i}^{max} = 255$, then no secret data can be extracted and restore $B_{i}^{max}$ by taking $\left\lfloor \log_2 \left(3N_{seg}/2\right) \right\rfloor$ bits from $S$.

**Case IV.** The block does not belong to any of the above three cases.

For a block belongs to Case IV (i.e., an embeddable block), calculate $D_{i}$ and segment $D_{i}$ into $N_{seg}$ segments. Then, calculate each segment’s pseudo pixel and generate the prediction error by calculating the difference between the stego pixel and pseudo pixel. After that, secret data can be extracted by using following rules.

**Rule 1.** If the prediction error is equal to −3, then output a secret bit ‘1’ and restore the prediction error as −1.

**Rule 2.** If the prediction error is equal to −2, then output a secret bit ‘0’ and restore the prediction error as −1.

**Rule 3.** If the prediction error is equal to −1, then output a secret bit ‘1’ and restore the prediction error as 0.

**Rule 4.** If the prediction error is equal to 0, then output a secret bit ‘0’.

**Rule 5.** If the prediction error is equal to 2, then output a secret bit ‘1’ and restore the prediction error as 1.

**Rule 6.** If the prediction error is equal to 1, then output a secret bit ‘0’.

**Rule 7.** If the prediction error is equal to 0, then output a secret bit ‘0’.

**Rule 8.** If the prediction error is equal to 3, then no secret data can be extracted, so restore the prediction error by adding 2 to the prediction error.

Further, the first block’s LSB is restored by referring to the least $\left\lfloor \log_2 (n \times n) \right\rfloor$ bits of the extracted secret data. Finally, the right receiver with a decryption key will gain the real secret data by decrypting the extract message. Also, the stego image can be fully restored.

### 4. Experimental Results

To evaluate the performance of the proposed method, the proposed method and Luo et al.’s method are implemented using Octave v3.24 software works on a Linux platform. In order to test the effect of the proposed method for different images, nine commonly tested images are used in our simulations. Fig. 2 shows the test gray-scale images sized 512×512 pixels.

The visual quality of stego image and embedding payload are the two most important factors for evaluating the performance of a data hiding technique. As we know, evaluating the visual quality through humans’ eyes is effective, but it is very subjective. To avoid observation bias, we use the peak signal-to-noise ratio (PSNR) for evaluation, which is an objective visual quality measurement. Normally, a high PSNR indicates that the
stego image is the most similar to its original image. Contrarily, a small PSNR value indicates that the stego image contains large distortions that are dissimilar to its original cover image. Generally, it is hard for a user to identify the distortion from the stego image by eyes when the PSNR value is greater than 30 dB.

Further, the embedding payload is used to evaluate the total number of secret bits embedded in a cover image. A data hiding scheme with a small embedding payload needs to send more different stego images than a high embedding payload scheme for the same secret data size. Sending too many different stego images will attract unexpected users to pay more attention on transmission. Here, the embedding payload is used to count the total bits embedded into a cover image, defined as $e_p = ||S||$. Because the proposed method needs to remember the information of non-embeddable blocks, the pure embedding payload is calculated by $e_p = e_p - ||NE||$, where the $||\cdot||$ operation is calculating the total bits and NE represents the extra information of non-embeddable blocks.

Fig. 3 shows the PSNR comparison for testing different segment numbers. From the experimental result, Luo et al.’s method has better PSNR outcomes, which is reasonable because it conceals more secret data than the proposed method. Also, all of PSNR values of stego images are higher than 30 dB. Further, using lesser segments will get better PSNR outcomes.

Fig. 4 shows the embedding payload comparison results for testing different segment numbers. From experimental results, complex content images (e.g., Baboon) achieve a higher embedding payload when the segment number is increased. Contrary, smooth content images will get a lower embedding payload when the segment number is increased. This situation is reasonable, because a smooth content image contains more smooth blocks, thus the number of non-embeddable blocks will be increased when increasing the segment number. For the experimental results, we can conclude that 4 or 5 are the suitable segment number.
5. Conclusions

Data hiding techniques make it easier to deceive the unexpected user for achieving the goal of secret data delivery. The proposed reversible data embedding technique utilizes difference segmentation to divide the difference between the maximum pixel and minimum pixel into several segments. Thus, the histogram of prediction error will be significantly increased, which helps increase the embedding payload performance. Also, the proposed method does not need to remember the information about the peak and zero points because the peak points are fixed as $-1$, $0$, and $1$. The experimental results show that the proposed method has higher embedding payload performance than Luo et al.'s method.

References


Yung-Chen Chou received the B.S. degree in management information systems from National Pingtung University of Science & Technology, Pingtung, Taiwan in 1998, and the M.S. degree in information management from Chaoyang University of Technology, Taichung, Taiwan in 2002. He received the Ph.D. degree in computer science and information engineering from the National Chung Cheng University, Chiayi, Taiwan in 2008. Since February 2009, he has been an assistant professor with Asia University, Taichung, Taiwan. His current research interests include steganography, watermarking, and image processing.

Huang-Ching Li was born in Taiwan. He is currently pursuing the B.S. degree with the Department of Computer Science and Information Engineering, Asia University, Taichung, Taiwan. His research interests include image processing and steganography.