國立交通大學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

基於 PNG 影像之資訊隱藏技術及其研究

A Study on Information Hiding Techniques for PNG Images and Their Applications

研 究 生：楊芷婷

指導教授：蔡文祥　教授

中 華 民 國 九 十 八 年 六 月

基於 PNG 影像之資訊隱藏技術及其研究

A Study on Information Hiding Techniques for PNG Images and Their Applications

研 究 生：楊芷婷　　Student：Chih-Ting Yang

指導教授：蔡文祥　　Advisor：Wen-Hsiang Tsai

國 立 交 通 大 學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

A Thesis
Submitted to College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Industrial Technology R & D Master Program on
Computer Science and Engineering

June 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

基於 PNG 影像之資訊隱藏技術及其應用之研究

研究生：楊芷婷　　指導教授：蔡文祥 博士

國立交通大學資訊學院產業研發碩士班

摘要

由於網路的發達及其方便性，數位影像傳輸也隨之變得頻繁，其中 PNG 影像是當今應用相當廣泛的一種影像格式。在本論文中，我們針對 PNG 影像提出了幾種資訊隱藏的技術與應用，包括版權保護、秘密分享及秘密傳輸。在版權保護部份，我們提出一個基於固定大小區塊的影像驗證方法，讓我們調整影像中紅綠藍色及透明通道的像素值，使其符合某種限制條件，來作為驗證訊號，並用以驗證影像之真假性及完整性。在秘密分享方面，我們提出一種藉由解聯立方程式來達到秘密分享效果之方法，該法由 $n-1$ 張使用者自選的掩護影像(cover image)產生 n 份分享檔案，並且將這 n 份分享檔案藏進掩護影像的透明通道中，形成 n 份偽裝影像(stego-image)。我們可以藉由收集 n 張偽裝影像中的分享檔案來復原原本的秘密。在秘密傳輸部分，我們根據以上提到的驗證和秘密分享之技術提出了兩種資訊隱藏的方法。另外，我們亦結合驗證和資訊隱藏，提出一種可驗證秘密資訊之方法。最後，我們以實驗結果證明了所提方法之可行性。

A Study on Information Hiding Techniques for PNG Images and Their Applications

Student: Chih-Ting Yang Advisor: Prof. Wen-Hsiang Tsai

Industrial Technology R & D Master Program of
Computer Science College
National Chiao Tung University

ABSTRACT

With the convenience and fast speed on the Internet, exchanges of multimedia become more and more frequent nowadays. PNG images are used in a wide variety of applications. In this study, we propose several methods for data hiding applications via PNG images, including copyright protection, secret sharing, and covert communication. For copyright protection, we propose a block-based method for file authentication, which can authenticate the integrity and fidelity of PNG images by adjusting pixel values to meet some constraints in the three color channels (red, green, and blue) and the alpha channel. For secret sharing, we propose a method by solving three simultaneous equations to generate n shares from $n-1$ cover images, and embedding the shares in the alpha channels of the color images. We can recover the secret by collecting all the shares. For covert communication, we propose two data hiding methods which are based on the proposed authentication method and the proposed secret sharing method. We also apply the two methods to yield a combined data hiding and authentication method to authenticate hidden data. Good experimental results show the feasibility of the proposed methods.

ACKNOWLEDGEMENTS

The author is in hearty appreciation of the continuous guidance, discussions, support, and encouragement received from her advisor, Dr. Wen-Hsiang Tsai, not only in the development of this thesis, but also in every aspect of his personal growth.

Thanks are due to Mr. Chih-Jen Wu, Mr. Che-Wei Lee, Mr. Guo-Feng Yang, Mr. Chun-Pei Chang, Miss. Shu-Hung, Mrs. Chiao-Chun Huang, Miss Mei-Fen Chen, Mr. Jian-Yuan Wang and Mr. Yi-Chen Lai for their valuable discussions, suggestions, and encouragement.

Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during her thesis study.

Finally, the author also extends her profound thanks to her family for their lasting love, care, and encouragement. The author dedicates this dissertation to her beloved parents

CONTENTS

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation of Study

With the fast growth of the Internet, exchanges of multimedia become more and more frequent nowadays. Because people can get multimedia very easily through the Internet, duplicating and tampering with digital images without authorization becomes a serious problem today.

Many kinds of image formats are used on the Internet. PNG (portable network graphics) image is one of the commonly-used formats. In addition to the three channels of colors, namely, red (R), green (G), and blue (B), the PNG format includes an extra channel named alpha channel. The alpha channel is a kind of controller which can be used to adjust the weight of RGB channels that people can see.

Data hiding techniques can be used to hide data in a given image. Here the image may be regarded as a kind of camouflage covering the secret data. Additionally, the PNG image is a commonly-used multimedia format on the Internet, but data hiding applications involving the PNG image is rarely seen. It is desirable to develop new data hiding techniques for the PNG image in this study. After we hide secret in a cover image, except the owner, other people will not know the data is hidden in the cover image. And they will not try to extract it, apparently. So the data will be under a better protection by the data hiding technique.

Copyright protection is always a hot topic in the applications of data hiding. Lots of copyright and security protection techniques have been proposed. One of them is to

hide authentication signals in cover images with less distortion. It is desired also in this study to embed authentication signals in PNG images to authenticate the fidelity and integrity of the images to achieve the purpose of security protection.

Secret sharing is a third data hiding application, in which we convert a secret message into several shares by a certain data distribution technique, and each share is kept by a participant. Since these shares are created by certain secret sharing operations, each share may be created to contain a meaningless content. When a person who is not one of the participants is offered one of these meaningless shares, he/she might suspect the existence of secret messages in the share. Therefore, it is desirable also to develop secret sharing techniques with steganographic effects for PNG images in this study.

1.2 Overview of Proposed Methods

1.2.1 Definitions of Terms

Before describing the proposed method, some definitions of the terms used in this study are given first as follows.

1. *Secret*: a secret is a piece of information that is important and should be preserved properly and not revealed to unauthorized people.
2. *Stego-image*: a stego-image is one in which some digital information is embedded.
3. *Cover image*: A cover image is an image which is used for secret sharing and steganography.
4. *Protected image*: a protected image is an image in which authentication signals have been embedded.
5. *Image authentication*: image authentication is a process for verifying the

integrity and fidelity of a suspicious image.

6. *Share image*: a share image is one of the secret sharing results of a secret image.
7. *Embedding process*: an embedding process is a process to embed data into an image.
8. *Extraction process*: an extraction process is a process to extract hidden data in the stego-image.

1.2.2 Brief Descriptions of Proposed Methods

A. Proposed Image Authentication Method and Data Hiding Method by Adjusting Selected Values on Spatial Domain for PNG Images

In this study, we propose a method to verify the fidelity and integrity of PNG images by adding a kind of authentication signals into PNG images. We apply a division operation to each image block. In order to execute this operation, we have to have a sum, a divisor, and a remainder in each block at the beginning.

At first, we divide the cover image into non-overlap 3×3 blocks. Then we pick up several values from each 3×3 block and sum them up. Next, we use a user key to generate a series of numbers named N_{divisor}, and one block will correspond to the number N_{divisor} as a divisor. We then divide the sum by the divisor to get a remainder. After this, we adjust, by adding or subtracting, the previously-selected values in each block in order to change the remainder to be zero. The above-mentioned operation for each block then is taken as a kind of authentication signal. And the image after

embedding authentication signals in this way becomes a protected image.

After we get a protected image as just described, we can authenticate the image to ensure its fidelity. For this, we can extract the remainder in each block of a given image, and then check whether the remainder is zero or not. If not, the image will be decided to have been tampered with.

In addition, based on the authentication scheme, we also propose a data hiding method, in which we count the number of odd numbers and that of even number of the previously-selected values in each block. According to the data which are to be hidden, we adjust the numbers of odd numbers and even numbers of previously-selected values in each block to achieve the goal of hiding data.

When the user gets a stego-image, he/she may extract the hidden data by counting the numbers of odd numbers and even numbers of previously-selected values in each image block.

B. Proposed Secret Sharing Method for PNG Images

A method by secret sharing and steganography techniques for sharing secret images is proposed in this study. The technique of sharing secret is based on solving simultaneous equations with n unknown numbers.

More specifically, we generate n shares from $n-1$ images, and the n^{th} image is named the *residual image*. In these three equations, the number of unknown variables depends on the number of images which the user selected. We propose to hide the solutions of the $n-1$ unknown variables in the alpha channels of the selected $n-1$ images, respectively. And the solution of the n^{th} unknown variable, the residual value, is hidden in the previously-mentioned residual image. Additionally, in the residual image, we will hide the residual values in the red, green, and blue channels. The share

images then are distributed to the participants for custody by them.

On the other hand, after collecting all the sharing images from the participants, we can extract the hidden data and recover the secret image.

C. Proposed Information Hiding Method for PNG Images

A method using a data hiding technique for covert communication via PNG images is proposed in this study. We use again the above-mentioned simultaneous equations to hide secret information.

More specifically, we divide the cover image into at least four parts. Each part is taken to be a cover image or the residual image. Different from the last method, we also hide the residual values in the alpha channel. The residual values are much larger than the previously-mentioned solutions of $n - 1$ unknown variables. So when we hide the residual values in the alpha channel, the alpha value will be lower. And the situation of each alpha values differ from each other very extremely will become a kind of noise appearing in the cover image. In order to solve the problem, we propose to divide the residual value into two parts and hide them in two different places to make the alpha values higher. In this way, we can get the stego-image with a better quality.

When a user gets a stego-image, he/she can extract the hidden data by solving the three equations.

1.3 Contributions

The contributions made in this study are summarized in the following.

1. An authentication method for verification of the integrity and fidelity of

PNG images by adjusting some selected values in the R, G, B, and A (alpha) channels is proposed.

2. A data hiding method with capability of authentication of hidden data is proposed..
3. A new secret sharing method with steganographic effects is proposed for protecting secret images systematically and securely.
4. A data hiding method based on some properties of the PNG format is proposed for covert communication.

1.4 Thesis Organization

In the remainder of this thesis, a review of related works about techniques of image data hiding, file authentication, information sharing, and the PNG standard is given in Chapter 2. In Chapter 3, the proposed method for image authentication by adjusting some selected values of the red, green and blue channels is described. In Chapter 4, the proposed method for image sharing by solving three equations is described. In Chapter 5, the proposed method for data hiding is described. Finally, conclusions and some suggestions for future works are made in Chapter 6.

Chapter 2
Review of Related Works and Standards

2.1 Introduction

More and more data hiding techniques have been developed. Because of the differences between distinct document formats, different data hiding techniques have been developed to hide data in documents of different formats. So in this chapter we will make a review of data hiding techniques as well as some related techniques for image authentication and secret sharing, which have been developed in recent years. In addition, because the data hiding techniques and their applications proposed in this study are implemented on the PNG format, we will also make a review of the PNG image standard subsequently.

2.2 Previous of Studies on File Authentication Techniques via Digital Images

Many authentication methods have been proposed [1-6] in the past. Wang and Tsai [1] proposed a method to verify the fidelity and integrity of PDF files by embedding authentication signals in them. Weng and Tsai [2] proposed two methods for authentication of grayscale documents. One of them can authenticate grayscale document images by a semi-fragile watermarking technique against print-and-scan attacks. The other method can authenticate grayscale document images by using edge direction histograms with circular interpretation. Huang and Tsai [3] proposed a scheme for authentication by embedding visible watermarks and authentication

signals into grayscale images. Chiu and Tsai [4] proposed a method for image authentication in color images without the use of signatures.

2.3 Previous of Studies on Secret Sharing Techniques

Secret sharing is one of the data hiding techniques for use in covert communication. Many secret sharing techniques have been proposed [7-10]. When a pre-defined group of shares is collected, the secret data distributed in them can be recovered. Shamir [7] proposed first the concept of secret sharing in his (k, n)-threshold method. In this method, a secret is divided into n shares which are given to n participants for custody. At least k shares must be collected to recover the secret. However, if $k-1$ or fewer shares are collected, the secret cannot be recovered. On the contrary, if less than $n-k$ shares are stolen, the secret can still be recovered. Blakley [8] transformed the secret sharing problem into a geometric one by considering the secret as a point of the intersection of n hyperplanes in an n-dimensional space where $n > 2$. Lin and Tsai [9] proposed an effective scheme for secret multimedia information sharing with steganogrphic effects. They created n shares from $n-1$ cover images by using simple logic operations. Additionally, the operation can recover the secret from n shares very quickly.

2.4 Previous of Studies on Information Hiding Techniques via Digital Images

A lot of approaches to hiding data into images have been proposed [11-17]. Lee and Tsai [11] proposed a new method for data hiding in grayscale images based on a

human vision model with distortion-minimizing capabilities. Lai and Tsai [12] proposed a method to hide data in PNG images. They piled a PNG image on a gray-level one and hid secret data in its LSBs. And a calculation was made to determine how many bits in each byte of the gray-level image can be used for hiding for the sake of reducing the resulting distortion. Ni, et al. proposed [13] a novel reversible data hiding algorithm, which can recover the original image without any distortion from the marked image after the hidden data have been extracted.

2.5 Review of PNG Standard

In this study, all the proposed information hiding, authentication, and image sharing techniques employ PNG images as carrier media for hiding information. We give a brief review of the PNG standard in this section. In Section 2.5.1, the usages of PNG images will be described. And in Section 2.5.2, some properties of PNG images will be described.

2.5.1 Usages of PNG Images

In this section, we will describe the usages of PNG images. PNG is the abbreviation of Portable Network Graphics. The PNG format was created to replace and improve upon the GIF (Graphics Interchange Format) format at the beginning. It is a bitmapped image format that employs the LZW (Lempel-Ziv-Welch) lossless data compression technique. As the name suggests, it was designed for transferring images on the Internet, not for professional graphics. And except the RGB color space, the PNG format does not support other color spaces, such as CMYK.

PNGs have some advantages over GIFs: the alpha channel, the gamma correction option, and the two-dimensional interlacing progressive display capability. Though GIFs also have alpha channels, they only support alpha channels with single-level

transparency. In other words, the GIF image can only be fully transparent or opaque. Instead, PNGs that have alpha channels are of variable transparency.

As we mentioned previously, the PNG was designed for transferring images on the Internet. If a user downloads a GIF image and a PNG image through the Internet simultaneously, he/she will get the complete PNG image earlier than the GIF image.

In the next section, we will describe the properties of PNG images.

2.5.2 Properties of PNG Images

Each PNG image consists of several ‘chunks’ substantially. In other words, with the exception of the first 8 bytes in each file, a PNG image comprises nothing but chunks. In the PNG format, two categories of chunks are included. One is named *critical chunk* which is a group of chunks that a decoder must be able to interpret in order to read and render a PNG file. This group consists of 4 chunks which are IHDR, IDAT, IEND, and PLTE. After the introduction to these *critical chunks*, we will introduce the other group of chunks, named *ancillary chunk*.

In general, a PNG image is at least composed with three types of chunks: IHDR, IDAT, and IEND. The IHDR is the image header chunk; the IDAT is the image data chunk; the IEND is the end-of-image chunk. An IHDR must be the first chunk in a PNG image, and it contains all of the detail about the type of image: its width and height, bit depth, color type, compression, filter, and interlacing methods. It can record whether the image is a truecolor, grayscale, or palette image and how many dimensions in pixels. An IDAT contains the actual data of an image, which are the output stream of the compression algorithm. It is allowed to have multiple IDAT chunks; if so, they all appear consecutively with no intermediate chunks. Each IDAT contains at most two gigabytes of compressed data. An IEND is the simplest chunk. It points out the end of a PNG image, and it contains no data. The previously-mentioned

three chunks can make up a PNG image, with or without an alpha channel. But if it is an image with a palette, it should contain another type of chunk, PLTE, the palette chunk. A PLTE has 256 palette entries and each entry is a 24-bits color in the RGB color space.

As we mentioned previously, the IHDR consists of the color information of the image which is named ColorType. If the ColorType equals 0, the image type is gray-level; if the ColorType equals 2, the image type is truecolor without an alpha channel; if the ColorType equals 3, the image type is indexed-color, with a palette; if the ColorType equals 4, the image type is gray-level with an alpha channel; if ColorType equals 6, the image type is truecolor with an alpha channel.

The *ancillary chunks* are also defined in the International Standard. They are not always necessary for constructing a PNG image. Therefore, these chunks may be ignored by decoders. There are 14 ancillary chunks, and we can classify them into 5 types according to their actions. The first type is about “transparency information” which includes tRNS chunk; the second type is about “color space information” which includes cHRM, gAMA, iCCP, sBIT, and sRGB chunks; the third type is about “textual information” which includes iTXt, tEXt, and zTXt chunks; the forth type is about “miscellaneous information” which includes bKGD, hIST, pHYs chunks, and the last type is about “time information” which only includes the tIME chunk.

Chapter 3
Authentication of PNG Images by Adjusting Selected Pixel Values in Spatial Domain and Its Application to Data Hiding

3.1 Introduction

With more convenience and faster speed on the Internet, exchanges of images are more frequent in recent years. Especially, after the type of PNG image was created for data transferring on the Internet, its uses also become common. Development of the data hiding techniques for PNG images becomes necessary and very important.

In this chapter, we describe the proposed method for authentication of PNG images and its application to data hiding. In Section 3.1.1, several related definitions will be given. In Section 3.1.2, the rough idea of the proposed authentication method for PNG images will be described. The process of embedding authentication signals and the extraction process will be given in Section 3.2. In additional, the proposed data hiding technique which is based on the above-mentioned authentication method is described in Section 3.3. Finally, some experimental results are shown in Section 3.4, and some discussions and summaries are given in Section 3.5.

3.1.1 Problem Definition

Although copyright protection becomes a hot topic recently, most of the existing techniques for this purpose were developed only for some very common file formats which have been used for a long time, such as BMP and JPEG. Due to the

characteristics of the alpha channel, the PNG becomes a popular format in the Internet recently. However proposed data hiding methods for PNG images so far are very rare.

Most of the authentication methods embed authentication signals as a kind of secret data in cover images. The authentication signals and the secret data usually cannot be hidden at the same place. In other words, if we want to hide secret data and authentication signals in the same cover image, the data hiding capacity of the cover image must be bigger than the sum of the capacity of authentication signals and the capacity of secret data. In this study, we will propose a method which can combine the authentication signals and the secret data.

3.1.2 Proposed Ideas

In this section, we briefly describe the proposed method. At first, we divide the cover image into non-overlapping 3×3 blocks, and we select randomly a fixed number of values from the data of the four channels (red, green, blue, and alpha) of each block. In each block, we add the selected pixel values up to get a sum. Also, we select a random number for use as a divisor. So, in each block, we have a sum and a divisor. Then, we divide the sum by the divisor to get a remainder. Afterward, we adjust, by adding or subtracting, the previously-selected pixel values in each block to change the remainder to be 0. In this way, the final content of the block is regarded as a kind of authentication signal for use in image authentication.

In addition, we propose a data hiding method which is based on the above-mentioned authentication method. In the previous discussion, we mentioned that we pick up several values in a block. We use these selected pixel values to hide data. First, we count the number of odd numbers and that of even numbers in the selected pixel values in each block. And according to the data which are to be hidden, we adjust these numbers of odd numbers and even numbers to achieve the goal of

hiding data. More details are described in the following sections.

3.2 Proposed Image Authentication And Data Hiding Methods

In this section, the processes for the proposed image authentication method will be described. In Section 3.2.1, the process of embedding authentication signals will be described. In Section 3.2.2, the process of image authentication will be given. And in Section 3.3, the proposed data hiding scheme which is based on the proposed image authentication method will be described.

3.2.1 Authentication Signal Embedding Process

In order to protect the copyright of a PNG image, we propose an image authentication method to achieve the goal. The PNG format includes four channels which are red, green, blue, and alpha channels. In the proposed method, first we divide the cover image into non-overlapping 3×3 blocks. Also, because conceptually we may regard each of the four channels of a PNG image block as a 3×3 *gray-level* image, we combine the four channels to form a 6×6 gray-level image with 36 pixels, as shown in Fig. 3.1.

Next, we choose two keys to generate two series of distinct random numbers, denoted as N_p and N_d, respectively, for each block. The series N_p includes distinct numbers in the range of 0 to 35, representing the positions of the pixels in the 6×6 gray-level image mentioned above as shown in Figure 3.1. Also, each number in the series N_d is regarded as a divisor limited to be within the range from 3 to 18.

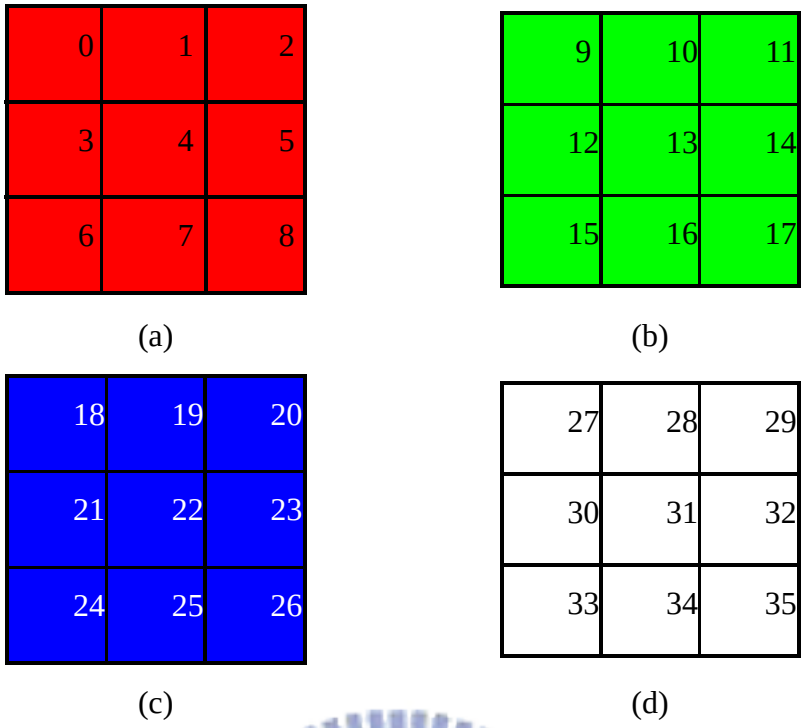


Fig. 3.1 Combination of four channels of an image block into a larger 6×6 gray-level image and the assigned positions of the 36 pixels. (a) Red channel (b) Green channel (c) Blue channel (d) Alpha channel

Then, for each 3×3 image block, we select randomly 18 pixel values according to N_p from the corresponding 6×6 gray-level image and add them up to get a sum, and we divide it by a divisor selected randomly according to N_d to get a remainder. Then we apply a value adjustment process to adjust the selected pixel values to make the corresponding remainder to become 0. In this way, we can get a protected image with authentication signals embedded in the final adjusted image block values.

The details of such authentication signal creation and embedding is described as an algorithm below.

Algorithm 3.1. Authentication signal creation and embedding for a PNG image.

Input: a cover image *I* in PNG format, two secret keys *p* and *d*, and a random number generator *f*.

Output: a protected image I_p with authentication signals.

Steps.

1. Divide I into non-overlapping 3×3 blocks.
2. Use the two keys, g and h, as seeds for f to generate two series of distinct random numbers, N_p and N_d, where the range of N_p is limited to be between 0 and 35, and that of N_d is limited to be between 3 and 18.
3. For each 3×3 image block B, perform the following steps.
   - 3.1. Compose a 6×6 gray-level image B' with 36 pixels using the gray-level image planes of the three color channels and the alpha channel of B (as illustrated by Fig. 3.1).
   - 3.2. Pick up in order 18 values from N_p and regard them to represent the positions of 18 randomly selected pixels of B'.
   - 3.3. Add up the gray values of the pixels of B' located at the 18 positions to get a sum S.
   - 3.4. Divide S by a divisor d selected in order from N_d to get a remainder r.
   - 3.5. Apply the following value adjustment process to change the values of the 18 selected pixels of B'.
     - a. When $r \geq d/2$:
       - i. if the values of the selected pixels are not all equal to 255, then increase the values of those selected pixels whose values are not equal to 255, each time by one, until the re-computed remainder r becomes equal to d; or
       - ii. if the values of the selected pixels are all equal to 255, then decrease the values of those selected pixels, each time by one, until the re-computed remainder r, becomes equal to 0.
     - b. When $r < d/2$ and $r \neq 0$:

i. if the values of the selected pixels are not all equal to 0, then decrease the values of those selected pixels whose values are not equal to 0, each time by one, until the re-computed remainder r becomes equal to 0; or

ii. if the values of the selected pixels are all equal to 0, then increase the values of those selected pixels, each time by one, until the re-computed remainder r becomes equal to d.

c. When $r = 0$:

do nothing.

4. Take the resulting image as the desire protected image I_p with the adjusted pixel values in B' being regarded as authentication signals.

In Step 3.5 above, we increase or decrease in order the 18 selected pixel values of B', each time by one. Sometimes, the selected pixel value in each block can not be increased or decreased, because we can not decrease 0 to be negative and we can not increase 255 to be larger, either. Therefore, if the first round of value increasing or decreasing can not make the re-computed remainder r' to be 0, we will start the second or more rounds until r' becomes so.

3.2.2 Image Authentication Process

In this section we will authenticate protected images to see whether they are tampered with or not. We apply the same steps, from Step 1 to Step 3.4, in Algorithm 3.1. That is, we select randomly 18 pixel values according to N_p from the corresponding 6×6 gray-level image and add them up; and in each block, we pick up the same divisor as we mentioned in the preceding discussion from the series N_d. Then we divide the sum by a divisor to get a remainder r. After these steps, we extract

the authentication signals to authenticate the image. If r is equal to 0, the corresponding block is decided to be correct, that is, not being tampered with. But if r is not equal to 0, we decide that the block might be modified, and mark it so in the image authentication report.

The detailed algorithm for image authentication is described in Algorithm 3.2 below.

Algorithm 3.2. Authentication of PNG images.

Input: a protected image I_p in PNG format, two secret keys p and d, and a random number generator f.

Output: An authentication report P for I_p.

Steps.

1. Divide I_p into non-overlapping 3×3 blocks.
2. Use the two keys, g and h, as seeds for f to generate two series of distinct random numbers, N_p and N_d, where the range of N_p is limited to be between 0 and 35, and that of N_d is limited to be between 3 and 18.
3. For each 3×3 image block B, perform the following steps.
   - 3.1. Compose a 6×6 gray-level image B' with 36 pixels using the gray-level image planes of the three color channels and the alpha channel of B (as illustrated by Fig. 3.1).
   - 3.2. Pick up in order 18 values from N_p and regard them to represent the positions of 18 randomly selected pixels of B'.
   - 3.3. Add up the gray values of the pixels of B' located at the 18 positions to get a sum S.
   - 3.4. Divide S by a divisor d selected in order from N_d to get a remainder r.
   - 3.1 Consider the following cases to authenticate I_p.

a. When $r = 0$:

decide that image block B has been modified and mark it so in P.

b. When $r \neq 0$:

decide that image block B has not been modified.

4. Take the final authentication report P as output.

3.3 Proposed Image Hiding Technique Based on Proposed Authentication Method

In this section, we will introduce a data hiding method which is based on the previously-mentioned authentication method. In Section 3.3.1, the secret data embedding process will be given. We will describe the data extraction process in Section 3.3.2.

As mentioned previously, we can apply the two methods to yield a *combined* image authentication method and data hiding method. The combination process will be described in Section 3.4.

3.3.1 Data Hiding Process

In this study, we propose a data hiding method which is based on the above-mentioned authentication method. And the detailed algorithm will be given.

Because this data hiding method is based the above-mentioned authentication method, the steps are almost the same as those of Algorithm 3.1. In this method, we use a key to generate randomly a series of distinct numbers, denoted as N_p. The use of N_p is the same as that in the proposed authentication method, representing the positions of a 6×6 gray-level image. We count the number of the odd gray values and that of even gray values located at the pixels at these selected positions in each block.

As to the secret data, we have to transfer every byte of them into binary. We hide secret data, each time by one bit, in each block by adjusting the numbers of odd gray values and even gray values, and then we can get a stego-image.

The detailed algorithm of secret data embedding is described in Algorithm 3.3 below.

Algorithm 3.3. Secret data embedding process for PNG images.

Input: a cover image C, a secret key g, a random number generator f, and a secret message D.

Output: a stego-image S.

Steps.

1. Divide C into non-overlapping 3×3 blocks.
2. Use the key g as a seed for f to generate a series of distinct random numbers, N_p, where the range of N_p is limited to be between 0 and 35.
3. Transform D into a binary string D'.
4. For each 3×3 image block B, perform the following steps.
   - 4.1. Compose a 6×6 gray-level image B' with 36 pixels using the gray-level image planes of the three color channels and the alpha channel of B (as illustrated in Fig. 3.1).
   - 4.2. Pick up in order 18 values from N_p and regard them to represent the positions of 18 randomly selected pixels of B'.
   - 4.3. Count the number of odd gray values, O, and that of even gray values, E, of the pixels of B' located at the 18 positions
   - 4.4. Apply the following steps to hide D', once a bit.
     - a. When $O \geq E$:
       - i. when the next bit in D' to be hidden is 0:

decrease the values of those selected pixels whose values are odd, each time by one, until $O < E$;

ii. when the next bit in D' to be hidden is 1:

do nothing.

b. when $O < E$:

i. when the next bit in D' to be hidden is 0:

do nothing;

ii. when the next bit in D' to be hidden is 1:

increase the values of those selected pixels whose values are even, each time by one, until $O > E$.

5. Take the resulting image as the desire stego-image S with secret data embedded.

3.3.2 Data Extraction Process

To extract the secret data, we will do several steps as follows. Because we hide secret data in PNG images by a block-based method, we should also divide the image into blocks to extract the data. At first, we divide the protected image into non-overlapping 3×3 blocks. Then we use a user key to generate a series of random numbers, denoted as N_p, which represents the positions of 18 selected pixel values in the block. We count the number of odd gray values O and that of even gray values E and compare them. If $O > E$, we decide that the hidden bit in this block is "1". On the other hand, if $O < E$, we decide that the hidden bit in this block is "0." After performing these steps, we can extract the secret data successfully.

The detailed algorithm of the proposed secret data extraction process is described in Algorithm 3.4 as follows.

Algorithm 3.4. Secret data extraction process for PNG images.

Input: a stego-image *S*., a secret key *g*, and a random number generator *f*.

Output: a secret message *P*.

Steps.

1. Divide *C* into non-overlapping 3×3 blocks.
2. Use the key *g* as seed for *f* to generate two series of distinct random numbers, N_p, where the range of N_p is limited to be between 0 and 35.
3. For each 3×3 image block *B*, perform the following steps.
   3.1. Compose a 6×6 gray-level image *B*′ with 36 pixels using the gray-level image planes of the three color channels and the alpha channel of *B* (as shown in Fig. 3.1).
   3.2. Pick up in order 18 values from N_p and regard them to represent the positions of the 36 pixels of *B*′.
   3.3. Count the number of odd gray values *O* and that of even gray values *E* of the pixels located at these 18 positions
   3.4. Apply the following steps to extract data, once a bit:
      a. when $O \geq E$:
         extract a bit of the secret data *D* to be 1;
      b. when $O < E$:
         extract a bit of the secret data *D* to be 1.
4. Combine the extracted bits of the secret data *D* together in order as *D*′.
5. Transform every 8 bits of *D*′ into a decimal number *D*″ expressed as an ASCII code, and then transform *D*″ into characters as the desired secret message *P*.

3.4 Combination of Proposed Authentication Method and Data Hiding Method

As mentioned previously, we propose a method which combines the previously-proposed authentication method and data hiding method. The overall effect is a method for data hiding with the capability of authenticating the hidden data. We call it a *combined method for data hiding and authentication*. In this combined method, we hide data first, and then adjust the selected pixel values to hide authentication signals. Different from the above-proposed *simple* authentication method, this combined algorithm adjusts the values in a different way. The detailed algorithm of the method is given below.

Algorithm 3.5. Combined method of data hiding and authentication for PNG images.

Input: an image C, two secret keys g and h, a secret message D, and a random number generator f.

Output: a stego-image S with authentication signals and secret data embedded.

Steps.

1. Divide C into non-overlapping 3×3 blocks.
2. Use the two keys, g and h, as seeds for f to generate two series of distinct random numbers, N_p and N_d, where the range of N_p is limited to be between 0 and 35, and that of N_d is limited to be between 3 and 18.
3. Transform D into a bit sequence D'.
4. For each 3×3 image block B, perform the following steps.
   3.1. Compose a 6×6 gray-level image B' with 36 pixels using the gray-level image planes of the three color channels and the alpha channel of B (as

shown in Fig. 3.1).

3.2. Pick up in order 18 values from N_p and regard them to represent the positions of the 36 pixels of B'.

3.3. Count the number of odd gray values O and that of even gray values E of the pixels located at the 18 positions

3.4. (*Data hiding*) Apply Step 4.4 in Algorithm 3.3 to hide the data of D', once a bit.

3.5. (*Authentication signal embedding*) Perform the following steps to embed authentication signals.

3.1.1 Add up the gray values of the pixels of B' located at the 18 positions to get a sum S.

3.1.2 Divide S by a divisor d selected in order from N_d to get a remainder r.

3.1.3 Apply the following value adjustment process to change the values of the 18 selected pixels of B'.

a. If the hidden bit of D' is "1," that is, if $O \geq E$:

I. when $r \geq d/2$, $d - r$ is odd, and the values of selected pixels are not all equal to 255:

i. pick up an even number, and increase it by one;

ii. increase the values of those selected pixels whose values are not equal to 255, each time by 2, until the re-computed remainder r becomes d;

II. when $r \geq d/2$, $d-r$ is odd, and the values of the selected pixels are all equal to 255:

i. pick up an even number, and decrease it by one;

ii. decrease the values of those selected pixels, each

time by 2, until the re-computed remainder r becomes 0;

III. when $r \geq d/2$, d-r is even, and the values of selected pixels are not all equal to 255:

i. increase the values of those selected pixels whose values are not equal to 255, each time by 2, until the re-computed remainder r becomes d;

IV. when $r \geq d/2$, d-r is even, and the values of selected pixels are all equal to 255:

i. decrease the values of those selected pixels, each time by 2, until the re-computed remainder r becomes 0;

V. when $r < d/2$, $r \neq 0$, and r is odd:

i. pick up an even number, and decrease it by one;

ii. decrease the values of those selected pixels whose values are not equal to 0, each time by 2, until the re-computed remainder r becomes 0;

VI. when $r < d/2$, $r \neq 0$, and r is even:

i. decrease the values of those selected pixels whose values are not equal to 0, each time by 2, until the re-computed remainder r becomes 0.

b. If the hidden bit in D' is "0", that is, if $O < E$:

I. when $r \geq d/2$ and $d - r$ is odd:

i. pick up an odd number, and increase it by one;

ii. increase the values of those selected pixels whose values are not equal to 255, each time by 2, until the

re-computed remainder r becomes d;

II. when $r \geq d/2$ and $d-r$ is even:

i. increase the values of those selected pixels whose values are not equal to 255, each time by 2, until the re-computed remainder r becomes d;

III. when $r < d/2$, $r \neq 0$, r is odd, and the values of selected pixels are not all equal to 0:

i. pick up an odd number, and decrease it by one;

ii. decrease the values of those selected pixels whose values are not equal to 0, each time by 2, until the re-computed remainder r becomes 0;

IV. when $r < d/2$, $r \neq 0$, r is odd, and the values of selected pixels are all equal to 0:

i. pick up an odd number, and increase it by one;

ii. increase the values of the selected pixels, each time by 2, until the re-computed remainder r becomes d;

V. when $r < d/2$, $r \neq 0$, r is even, and the values of selected pixels are not all equal to 0:

i. decrease the values of those selected pixels whose values are not equal to 0, each time by 2, until the re-computed remainder r becomes 0;

VI. when $r < d/2$, $r \neq 0$, r is even, and the values of selected pixels are all equal to 0:

i. increase the values of the selected pixels, each time by 2, until the re-computed remainder r becomes d.

c. When $r = 0$,

do nothing.

5. Take the resulting image as the desire stego-image.

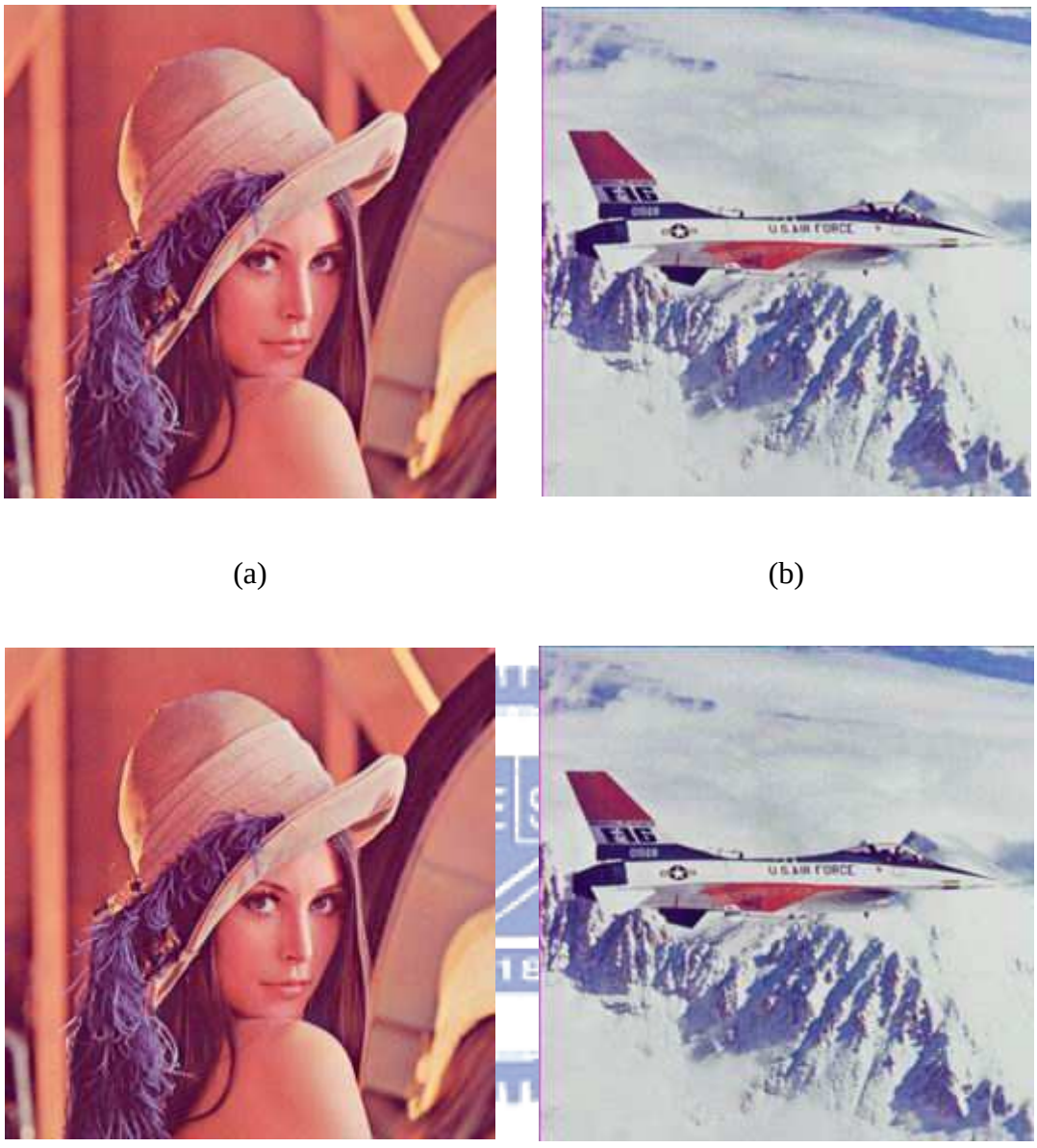
After these steps, both authentication signals and message data are embedded in each block. We can authenticate the protected image as well as extract the hidden data in the same image using Algorithms 3.2 and 3.4. If the authentication report of this image indicates that some blocks have been tampered with, then the hidden data in this block are also incredible.

3.5 Experimental Results

Some experimental results of applying the proposed method are shown here. Figures 3.2(a) and 3.2(b) show two color images both with size 256×256. And the stego-images resulting from embedding authentication signals into them are shown in Figures 3.2(c) and 3.2(d), respectively.

Four images tampered with are shown in Figures 3.3(a) and 3.3(b), and the authentication results are shown in Figures 3.3(c) and 3.3(d). If the area is not tampered with, it will be white in the result; otherwise, red.

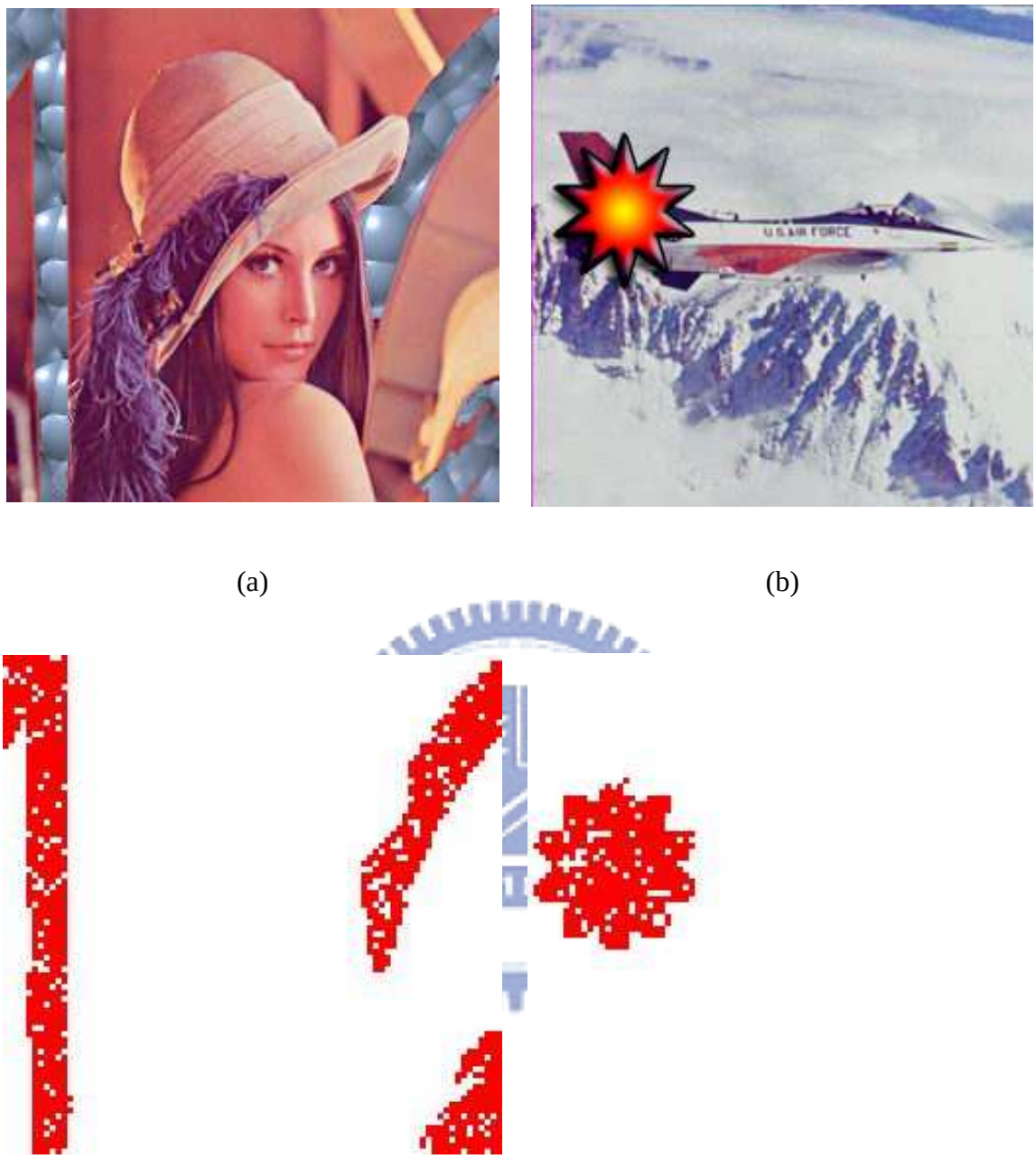
A protected image and an image tampered with in the alpha channel are shown in Figures 3.4(a) and 3.4(b), respectively. The authentication result is shown in Figures 3.4(c). The secret message and extracted hidden data are shown in Figures 3.5(a) and (b).



(a) (b)

(c) (d)

Fig. 3.2 Input color PNG images, output stego-images with authentication signals and secret data, and the differences. (a) Color cover image “Lena.” (b) Color cover image “Jet.” (c) and (d) Stego-images after embedding authentication signals, respectively.



(a) (b)

(c) (d)

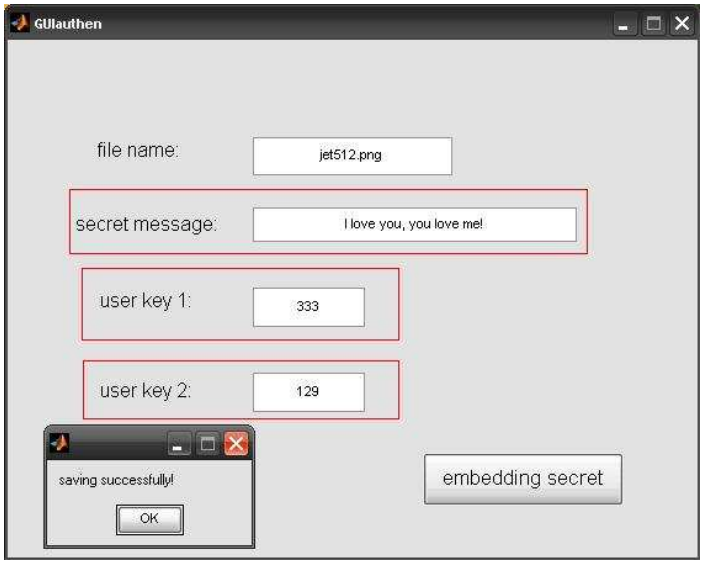
Fig. 3.3 Some tampered images and authentication results. (a) and (b) Tampered images. (c) and (d) Authentication results, respectively.

(a)

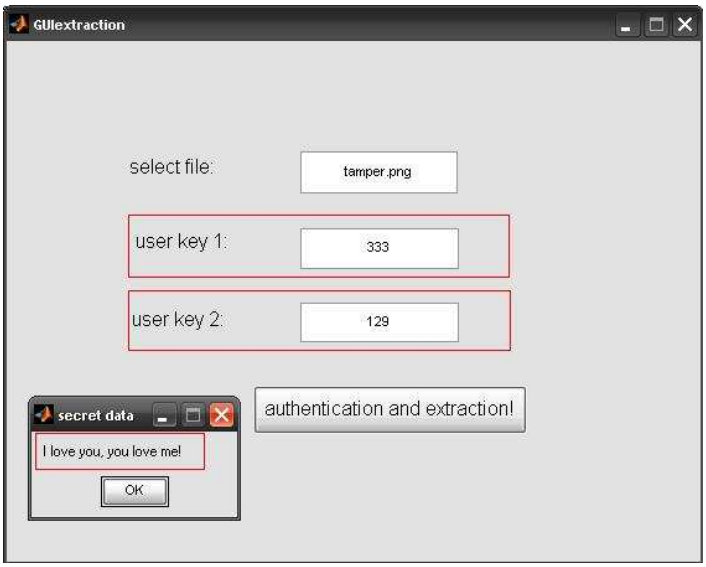
(b)

(c)

Fig. 3.4 Experimental results of the proposed method. (a) The original image. (b) Image be tampered with in alpha channel, but we can not tell where is different. (c) The result of authentication.



(a)



(b)

Fig. 3.5 Experimental results of the proposed data hiding method. (a) The secret message and two user keys. (b) The extracted hidden data with two correct user keys.

3.6 Discussions and Summary

In this chapter, we have presented a novel authentication method and a data hiding method. In the authentication method, we allow a user to use two keys to

generate two series of random numbers. Each block will get a random number from the second series, which is used as a divisor. We add up the values of 18 pixel values selected randomly using the first random number series, and then divide the sum by the divisor to get a remainder. We adjust the selected pixel values to make the remainder to be 0, with the resulting content of the block as a kind of authentication signal.

In the proposed data hiding method, first we count the number of odd numbers and that of even numbers in the selected pixel values in each block, and then adjust the two numbers so obtained to hide data.

We also apply the two methods to yield a combined data hiding and authentication method. We can authenticate the image with hidden data to see whether is tampered with or not. And we can also decide whether the hidden data in the image is correct or not.

However, in the proposed method of data hiding, the data hiding capacity may not be large enough. In the future work, we may divide the selected 18 values into three groups, with each group containing 6 values. We can do the same operation in these three groups in each block to hide secret data, thus enlarging the data hiding capacity of the method.

Chapter 4
Secret Sharing of Multimedia Information with Steganographic Effect via PNG Images by Solving Simultaneous Equations

4.1 Introduction

With the growth of the Internet, exchanges of files have become more frequent. People sometimes may get files easily without authorization. For this reason, we have to find some ways to solve this problem. Information sharing is a good solution. In a method using this technique, a given secret message is divided into several shares, and the shares are distributed to the secret sharing participants for custody. In this way, nobody can recover the secret without collecting all the shares.

In this section, we describe briefly the idea of the proposed secret sharing method. We generate n shares from a secret image and hide them into n PNG images in which the first $n - 1$ ones are user-selected and the n^{th} one is artificially created, as illustrated in Fig. 4.1. The proposed method is based on a novel scheme of solving three linear equations with a number of unknown variables, say, n. The formulation of the three equations is described in Section 4.1. We hide the solutions to the first $n - 1$ unknown variables into the alpha channels of the $n - 1$ user-selected images, respectively. And we hide into the n^{th} image, named the *residual image*, the solutions to the n^{th} variable of the equations, called *residual values*. The n images, called shares, are finally distributed to the participants for custody by them. We extract the hidden data and recover the secret by collecting all the n shares. The details will be described

later in Section 4.3.

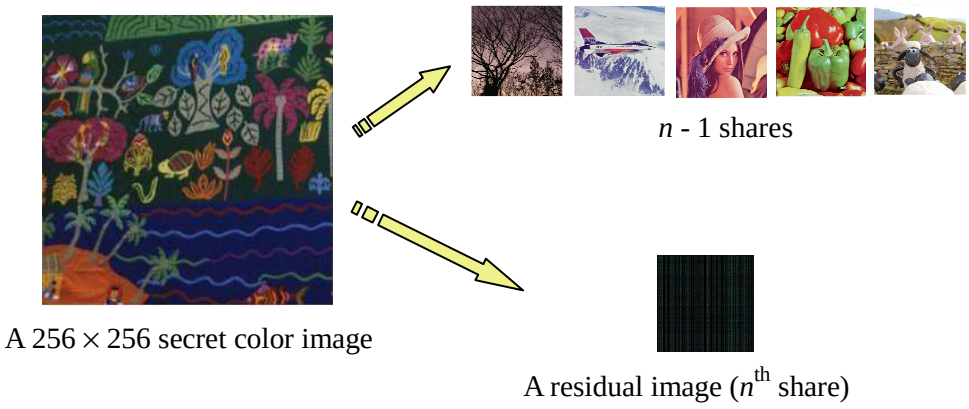


Fig. 4.1 Concept of proposed secret sharing process.

In the remainder of this chapter, a detailed algorithm describing the proposed secret sharing method will be given in Section 4.2, where an algorithm of the proposed corresponding secret recovery process will also be described. Some experimental results will be shown in Section 4.3, and some discussions and a summary will be given in Section 4.4.

4.2 Proposed Secret Sharing Method

In this section, we describe the proposed secret sharing method with steganographic effects in detail. The processes of share creation and secret recovery will be described in Sections 4.2.1 and 4.2.2, respectively, as algorithms.

4.2.1 Proposed Secret Sharing Process

The proposed secret sharing technique essentially divides a secret image S into several parts to generate shares by solving three equations using $n - 1$ user-selected cover images in PNG format, as mentioned previously. The equations are described as follows:

$$R_{sec} = \alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-1} \times R_{n-1} + R_{res};$$
$$G_{sec} = \alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-1} \times G_{n-1} + G_{res}; \quad (4.1)$$
$$B_{sec} = \alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-1} \times B_{n-1} + B_{res},$$

where R, G, and B represent the red, green, and blue channel values of each corresponding pixel of the involved images. Specifically, R_{sec}, G_{sec}, and B_{sec} represent the red, green, and blue channel values, respectively, of a pixel in the secret image S; R_k, G_k, and B_k represent the red, green, and blue channel values, respectively, of a pixel in the k^{th} cover image where $k = 1, 2, ..., n - 1$. Additionally α_1, α_2…, α_{n-1} and R_{res}, G_{res}, and B_{res} are unknown variables in the three equations.

For the purpose of making the recovered secret image lossless, we also propose a mapping technique which transforms the solution into a mapped value by applying a simple process. The two detailed algorithms for such value mapping and the proposed secret sharing based data hiding process are described as follows.

Algorithm 4.1. The process of value mapping.

Input: a sequence V of binary values (bits) of the length of $6k$ where k is a known integer.

Output: a sequence V' of decimal values.

Steps.

1. Divide every 6 bits of V into a *group of k segments* $f_1, f_2 \ldots, f_k$.
2. Transform f_1 through f_k into decimal numbers (digits) $d_1, d_2, \ldots, d_k$, respectively.
3. Compute

$$n_p = d_p \times 3 + 1 \quad (4.2)$$

where $p = 1, 2 \ldots, k$.

4. Take the sequence of n_1, n_2…, n_k as the desired output V'.

Algorithm 4.2. The process of secret sharing

Input: $n - 1$ cover images, I_1 through I_{n-1}, in PNG format which are selected by the users, a secret image I_{sec}, a secret key K, and a random number generator f.

Output: $n - 1$ shares, I_1' through I_{n-1}', in the forms of PNG images and a residual PNG image I_n'.

Steps.

1. Combine the red, green, and blue channel values of all pixels in I_{sec} in a raster-scan order and transform the result into an 8-bit binary string D.
2. Transform D into a mapped value D' by Algorithm 4.1.
3. For each pixel P located at the same positions in I_1 through I_{n-1}, perform the following steps.
   3.1. Set in order the mapped values, D's, of the three color channels (red, green, and blue), as the constant values R_{sec}, G_{sec}, and B_{sec}, respectively, of the left-hand sides of Eqs. (4.1).
   3.2. Apply the following steps to find the solutions to the $n - 1$ unknown variables α_1 through α_{n-1} and the residual values R_{res}, G_{res}, and B_{res} in Eqs. (4.1).
      3.2.1. Limit each of α_1, α_2…, and α_{n-1} to be one of 12 pre-selected values in H = {0.0, 0.1, 0.2, 0.3, …, 0.9, and 1.0} and generate all possible combinations of the $n - 1$ variables α_1 through α_{n-3}.
      3.2.2. For each possible combination C_j of α_1 through α_{n-1}, perform the following steps.
         (i) Compute the residual values R_{res}, G_{res}, and B_{res}

according to Eqs. (4.1) in the following way:

$$R_{res} = R_{sec} - (\alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-1} \times R_{n-1});$$
$$G_{res} = G_{sec} - (\alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-1} \times G_{n-1}); \quad (4.3)$$
$$B_{res} = B_{sec} - (\alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-1} \times B_{n-1}).$$

(ii) Compute the sum $M_{res} = R_{res} + G_{res} + B_{res}$.

3.2.3. Find the combination C_{opt} among all possible ones for which all the residual values R_{res}, G_{res}, and B_{res} computed above are non-negative and the sum M_{res} is the minimum.

3.2.4. Take the $n - 1$ variables and the residual values corresponding to C_{opt}, denoted as α_1' through α_{n-1}' and R_{res}', G_{res}', and B_{res}', respectively, as the desired solutions to the $n - 1$ unknown variables α_1 through α_{n-1} and the residual values R_{res}, G_{res}, and B_{res} in Eqs. (4.1).

3.3. Map α_1' through α_{n-1}' to new values α_1'' through α_{n-1}'', respectively, according to the following rule:

$$\alpha_i'' = (10 - (10 \times \alpha_i')) + 245 \quad (4.4)$$

where $i = 1, 2, \ldots, n - 1$.

3.4. Hide α_1'' through α_{n-1}'' into the alpha channels of the corresponding $n - 1$ cover images I_1 through I_{n-1}, respectively, to obtain $n - 1$ stego-images I_1' through I_{n-1}', respectively.

3.5. Assign respectively the residual values, R_{res}', G_{res}', and B_{res}', to the red, green, and blue channel values of the pixel in the residual image I_n' corresponding to P.

4. Use K as a seed for f to randomize the positions of the pixel values in the red,

green, and blue channels of the residual image I_n'.

5. Take the n resulting images I_1' through I_n' as the desired n shares.

In Step 3.3 above, we map the original solutions to α_1' through α_{n-1}' to α_1'' through α_{n-1}'', respectively, by $\alpha_i'' = (10 - (10 \times \alpha_i')) + 245$ before embedding them into the alpha channel. Because the solutions α_1' through α_{n-1}' are limited to be within the range of $H = \{0.0, 0.1, 0.2, 0.3, \ldots, 0.9, 1.0\}$, the values of α_1' through α_{n-1}' will be limited to be within the range of $H' = \{255, 254, \ldots., 245\}$, with 0.0 mapped to 255, 0.1 mapped to 254, …, and 1.0 mapped t0 245. The reason behind this mapping is that we want to avoid embedding *small* values into the alpha channel because small alpha channel values will cause the resulting PNG image to become white-noised which is not desired from the viewpoint of steganography.

4.2.2 Proposed Secret Multimedia Recover Process

In the proposed secret recovery process, we collect the n shares to recover the secret by using Eqs. (4.1). We first extract the hidden solutions in the $n - 1$ shares and the residual image, and then bring them back into Eqs. (4.1) to get the values of R_{sec}, G_{sec}, and B_{sec} for each pixel of the secret image. Afterward, we can recover the secret image. The detailed algorithm is described in the following.

Algorithm 4.3. The process of recovery of secret image.

Input: $n - 1$ shares I_1' through I_{n-1}' in PNG format, a residual image I_n', a secret key K, and a random number generator f.

Output: a secret image I_{sec}'

Steps.

1. Use K as a seed for f to recover the randomized positions of the pixel values

in the red, green, and blue channels of I_n'.

2. For each pixel P' located at the same positions in I_1' through I_{n-1}' and I_n', perform the following steps.

   2.1. Extract the alpha values A_i from I_i' where $i = 1, 2\ldots, n-1$ and apply the following rule to get the hidden solutions to the unknown variables, α_1 through α_{n-1}:

$$\alpha_i = (10 - (A_i - 245)) / 10. \tag{4.5}$$

   2.2. Extract residual values, R_{res}, G_{res}, and B_{res}, in the three color channels (red, green, and blue) from I_n'.

   2.3. Use the follow formulas to compute the *mapped* secret data, R_{sec}, G_{sec}, and B_{sec}:

$$\begin{aligned} R_{sec} &= \alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-1} \times R_{n-1} + R_{res}; \\ G_{sec} &= \alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-1} \times G_{n-1} + G_{res}; \\ B_{sec} &= \alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-1} \times B_{n-1} + B_{res}, \end{aligned} \tag{4.6}$$

   where R_k, G_k, and B_k ($k = 1, 2, \ldots, n-1$) represent the red, green, and blue channel values of I_k', respectively.

   2.4. From R_{sec}, G_{sec}, and B_{sec}, use the following formulas to compute corresponding *non-mapped* values R_{sec}', G_{sec}', and B_{sec}':

$$\begin{aligned} R_{sec}' &= (R_{sec} - 1) / 3; \\ G_{sec}' &= (G_{sec} - 1) / 3; \\ B_{sec}' &= (B_{sec} - 1) / 3. \end{aligned} \tag{4.7}$$

   2.5. Combine the values of R_{sec}', G_{sec}' and B_{sec}' in order, and transform the result into a 6-bit binary string D.

3. Combine the D's of all pixels in order into a sequence D'.

4. Apply the following steps to D':

4.1. divide every 8 bits of D' into a *group of segments* $f_1, f_2 \ldots, f_m$;

4.2. transform the sequence of f_1 through f_m into a sequence of decimal numbers, d_1 through d_m, respectively.

5. Take in order every three numbers in the sequence $d_1, d_2 \ldots, d_{\mathrm{m}}$ to represent the red, green, and blue channels values of a pixel in the secret image I_{sec}' and repeat this until all d_i are processed.
6. Take the resulting image I_{sec}' as the desired secret image.

In Step 4.1 above, if the number of bits in the last *group of segments* is smaller than eight, we will append several bits of 0 to the original bits until the number of bits in the last segment becomes eight.

4.3 Experimental Results

Some experimental results of applying the proposed method are shown here. First, Figures 4.2 and 4.3 show the two user interfaces for the proposed secret sharing process and the proposed secret recovery process. Next, the secret image is shown in Figures 4.4(a), and the $n - 1$ shares are shown in Figures 4.4(b) through 4.4(f). Figures 4.4(g) shows the resulting residual image. Note that, the size of the secret image is 256×256, and those of the shares are 512×512, all in PNG format.

The experimental results of applying the proposed secret image recovery method with correct shares are shown in Figures 4.5. Figures 4.5(a) through 4.5(f) shows the correct shares, and the correct of applying the proposed secret sharing algorithm with correct shares as input is shown in Figures 4.5(g).

The experimental results of applying the proposed secret image recovery method with incorrect shares are shown in Figures 4.6. Figures 4.6(b), 4.6(d), and 4.6(e) show incorrect shares, and Figures 4.6(c), 4.6(f), and 4.6(g) show correct shares.

Figures 4.6(a) shows the incorrect result of applying the secret recovery algorithm with incorrect shares as input. The result is noisy.

4.4 Discussions and Summary

In this chapter, we have presented a new type of secret sharing method. In this method, we proposed a scheme to share secrets by solving simultaneous equations. Each equation has n unknown variables, and we hide the solutions to the n variables in different cover images. Additionally, we allow a user to select $n - 1$ PNG images as shares, and generate the n^{th} share after applying the three equations. We can recover the secret by collecting all the shares, and bring the hidden solutions to these shares back to achieve the goal.

The secret sharing method is designed to be used in PNG images, and the generated shares have good steganographic effects.

Fig. 4.2 The user interface for proposed secret sharing method.

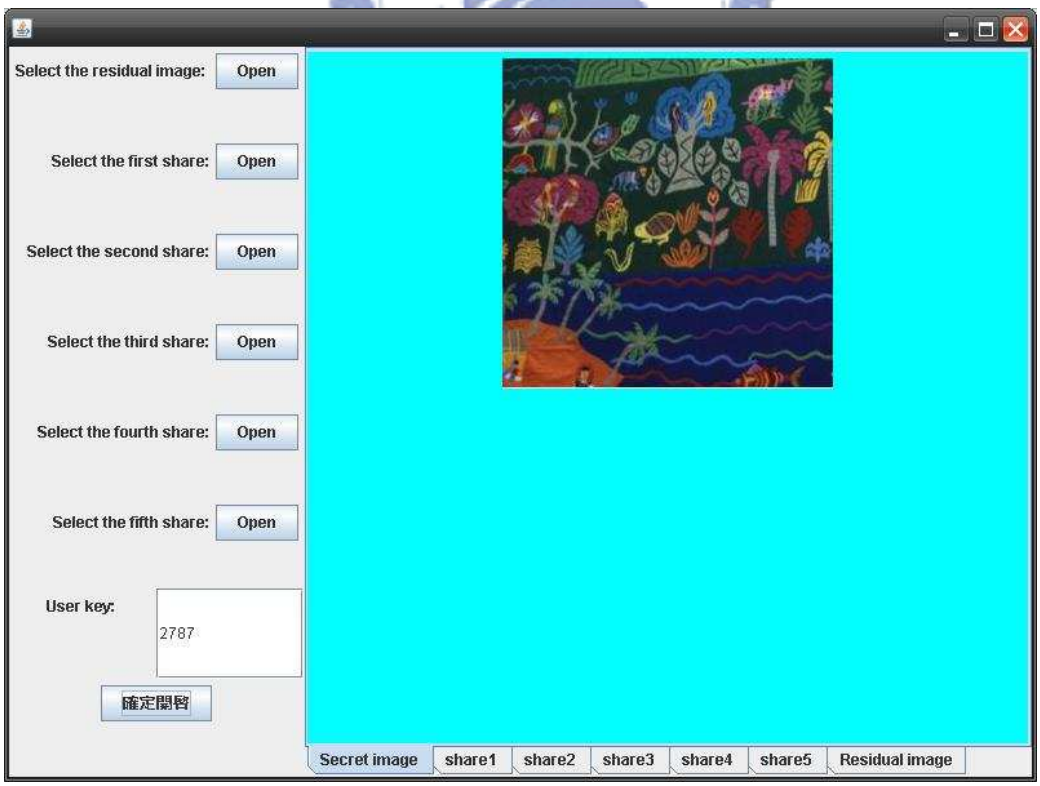
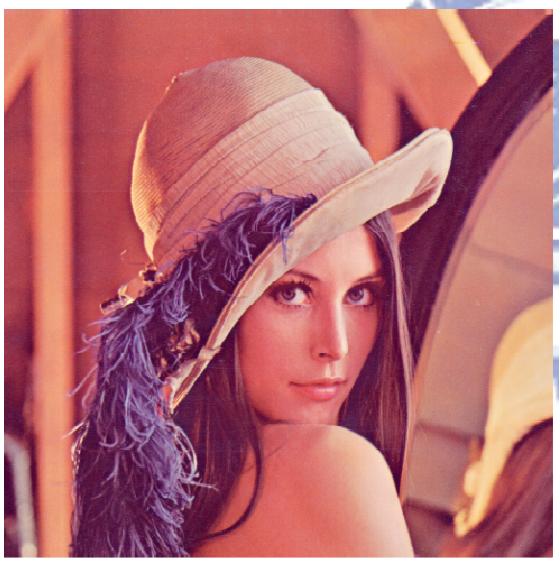


Fig. 4.3 The user interface for proposed secret recovery method.

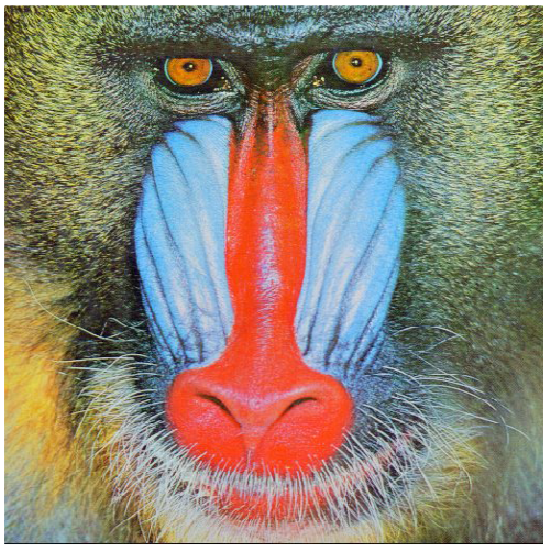
(a)



(b) (c)

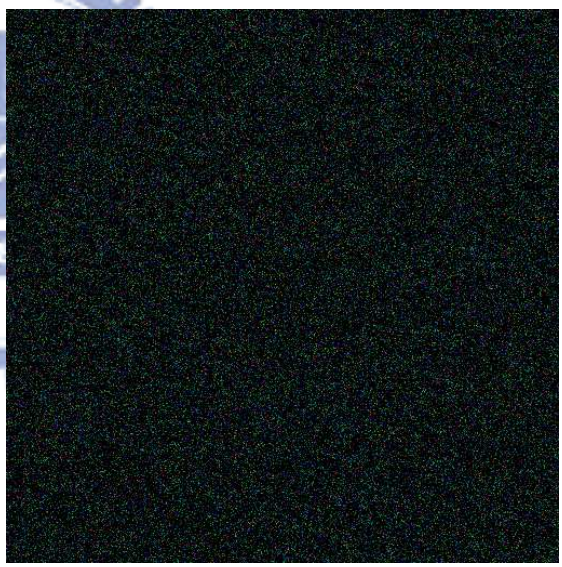
Fig. 4.4 The secret image and its shares after applying the proposed method. (a) The secret image with size 256×256; (b)-(f) the cover images which were selected by users keeping the $n - 1$ shares; (g) the residual image.

(d)



(e)

(f)



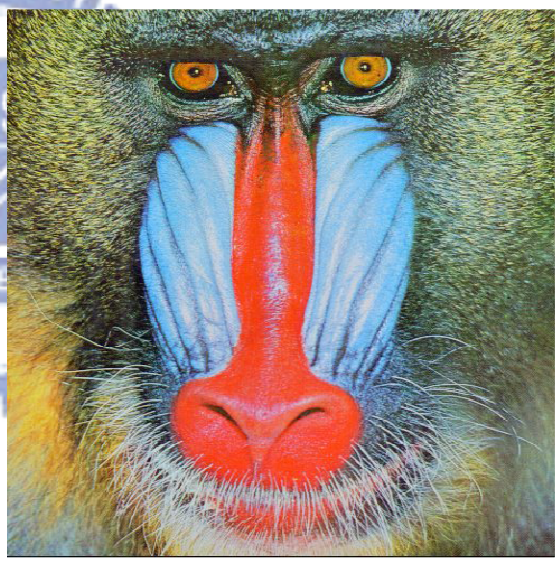
(g)

Fig. 4.4 The secret image and its shares after applying the proposed method. (a) The secret image with size 256 × 256; (b)-(f) the cover images which were selected by users keeping the $n - 1$ shares; (g) the residual image (continued).

(a)

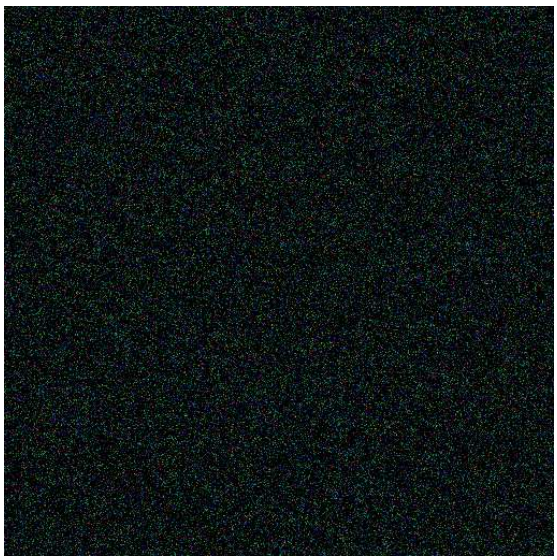
(b)

(c)

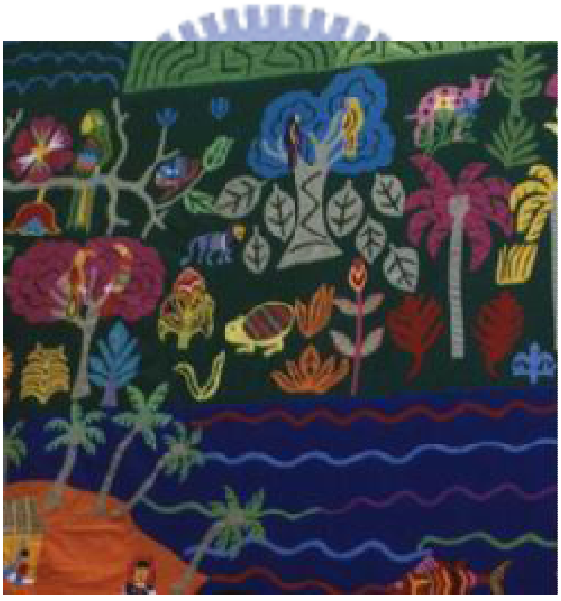


(d)

Fig. 4.5. The results of applying proposed secret image recovery method with correct shares, (a)-(e) the correct shares; (f) the correct residual image; (g) the recovered secret.

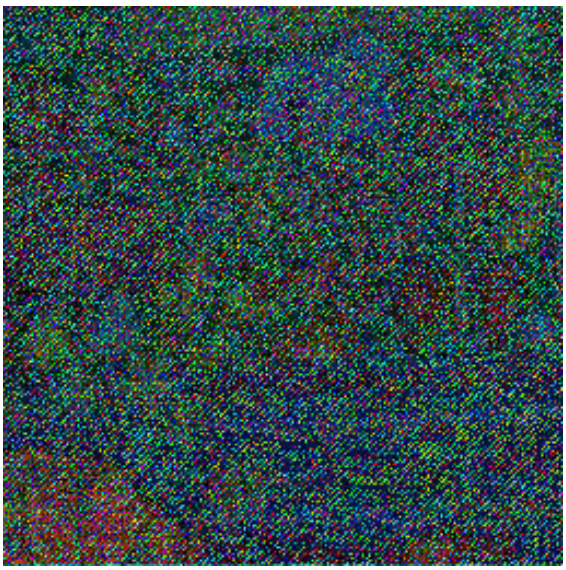


(e) (f)



(g)

Fig. 4.5. The results of applying proposed secret image recovery method with correct shares, (a) an incorrect share; (b)-(e) the correct shares; (f) the correct residual image; (g) the recovered secret which is noisy (continued).



(a)

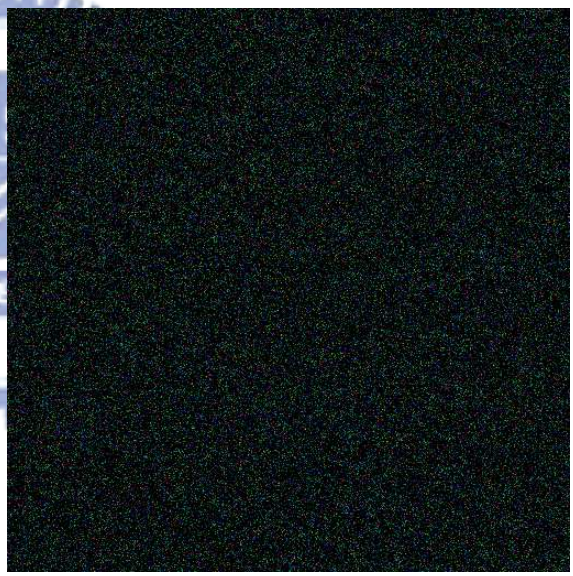
(b) (c)

Fig. 4.6. The results of applying proposed secret image recovery method with incorrect shares, (b), (d), and (e) the wrong shares; (c), (f), and (g) the correct shares; (a) the result recovery of secret sharing.

(d)

(e)

(f)



(g)

Fig. 4.6. The results of applying proposed secret image recovery method with incorrect shares, (b), (d), and (e) the wrong shares; (c), (f), and (g) the correct shares; (a) the result recovery of secret sharing (continued).

Chapter 5
Data Hiding in PNG Images for Covert Communication

5.1 Introduction

Because of the convenience of the Internet, the spreads of secrets on the Internet are safe no more. To solve the problem, we can hide secret data in a given multimedia file, and regard the stego-file as a kind of camouflage covering the secret data. Therefore, if users want to get the hidden secret data, he/she have to apply certain data extraction techniques, and that can make secret data more secure.

In this chapter, we will describe the proposed data hiding technique in PNG images. The related definitions will be given in Section 5.1.1, and in Section 5.1.2, the proposed data hiding method will be described roughly. The detailed data hiding method and the data extraction method will be given in Section 5.2.1 and Section 5.2.2, respectively. Some experimental results will shown in Section 5.3, and a summary and some discussions will be given in Section 5.4.

5.2 Problem Definition

Although the PNG format has been developed for years, its applications on data hiding are still rare. PNG images are commonly-used multimedia on the Internet, because its characteristic of having the alpha channel can make web page creation easier. Therefore, it is desired to develop a data hiding technique in PNG format.

5.2.1 Proposed Idea

In this section, we will briefly describe the proposed data hiding method. We use

again the previously-mentioned three equations, Eqs. (4.1), to hide secret information.

More specifically, we divide the cover image into at least six parts, and regard each part as a cover image or residual image. The cover image division process in concept is shown in Figure 5.1. We have to allot three divided parts to the residual values for hiding because Eqs. (4.1) have three different residual values.

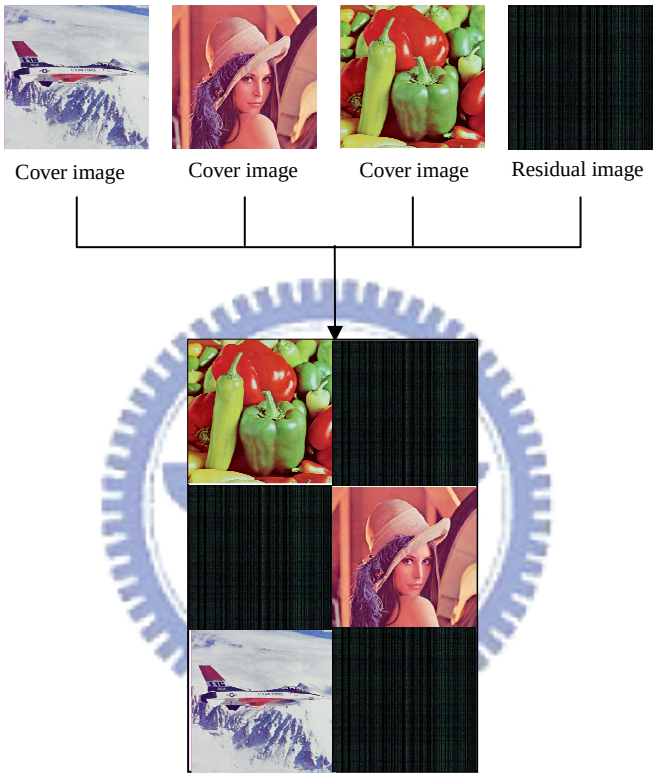


Fig. 5.1 Illustration of process of image division.

Different from the proposed secret sharing method, we hide the solution to the n^{th} unknown variable, the residual value, in the alpha channel instead of in the red, green, and blue channels. Residual values are usually larger than the solutions of the n – 1 unknown variables in value magnitude. If we hide it in the alpha channel instead, it will be displayed very differently from other pixels, and the result will become a kind of noise appearing in the stego-image. Therefore, we divide the larger residual

value into two parts and hide them in different places. That can make the stego-image have a better quality. The value division process is shown in Figure 5.2. More details are described in the following sections.

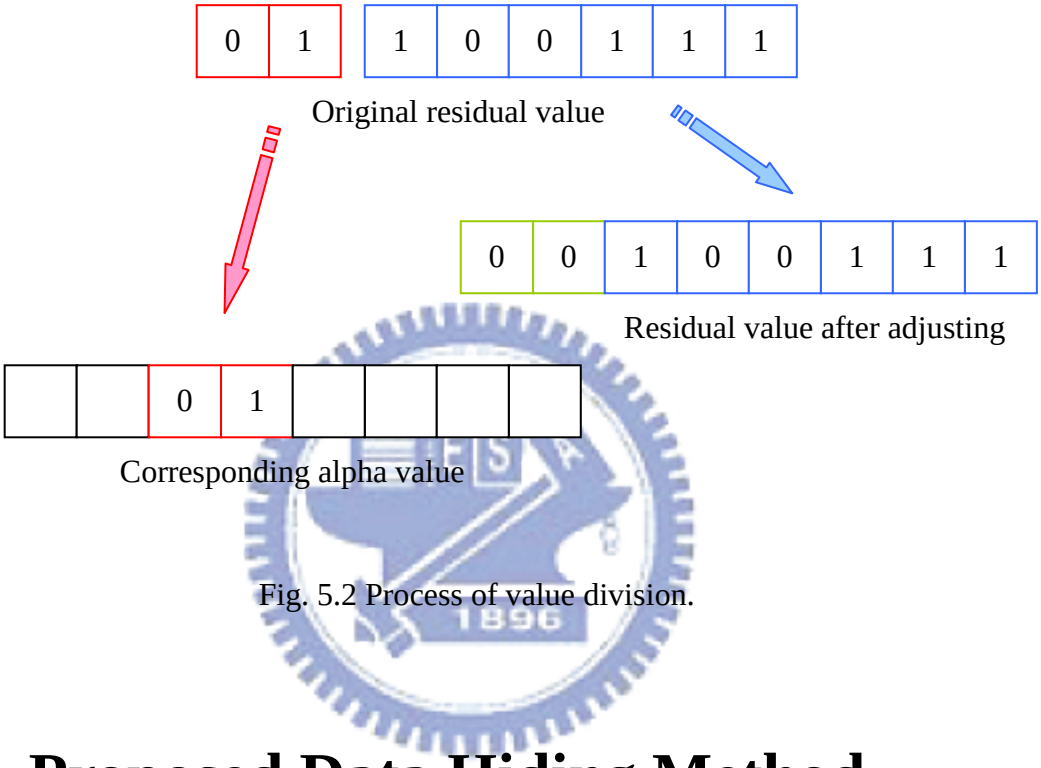


Fig. 5.2 Process of value division.

5.3 Proposed Data Hiding Method

In this section, the processes of data hiding and data extraction will be described in detail in Section 5.2.1 and Section 5.2.2, respectively.

5.3.1 Proposed Embedding Process

In this section, we describe the proposed data hiding method which is based on the proposed secret sharing method described in the last chapter. We divide a cover image into *n* parts and regard each part as an independent image to apply Eqs. (4.1). The detailed algorithms of value division and data hiding are given below.

Algorithm 5.1. Process of residual value division.

Input: a residual values V and an alpha value α.

Output: a adjusted residual value V and a adjusted alpha value α.

Steps.

1. Transform α and V into 8-bits binary strings, denoted as a_1, a_2…, a_8, and v_1, v_2…, v_8, respectively, and perform following steps.
   a. If $V \geq 64$, then
      i、 if $v_1 = 0$ and $v_2 = 1$,
         set $a_3 = 0$ and $a_4 = 1$;
         set $v_1 = 0$ and $v_2 = 0$; or
      ii、 if $v_1 = 1$ and $v_2 = 0$,
         set $a_3 = 1$ and $a_4 = 0$;
         set $v_1 = 0$ and $v_2 = 0$; or
      iii、 if $v_1 = 1$ and $v_2 = 1$,
         set $a_3 = 0$ and $a_4 = 0$;
         set $v_1 = 0$ and $v_2 = 0$;
   b. else,
      set $a_3 = 1$; $a_4 = 1$.
2. Transform the two binary strings, a_1, a_2…, a_8 and v_1, v_2…, v_8, into decimal numbers α' and V'.
3. Take the results, α' and V', as the output.

Algorithm 5.2. Process of data hiding based on secret sharing.

Input: a secret message D, a cover image I in PNG format, a secret key K, and a random number generator f.

Output: a stego-image I'.

Steps.

1. Divide I into n parts, p_1 through p_n, with $n \geq 6$ conceptually.
2. Transform all characters in D into an 8-bit binary string D'.
3. Transform D' into a mapped value D'' by Algorithm 4.1.
4. For each pixel located at the same positions in parts p_1 through p_{n-3}, perform the following steps.
   - 4.1 Set in order the mapped values, D''s, of the three color channels (red, green, and blue), as the constant values R_{sec}, G_{sec}, and B_{sec}, respectively, of the left-hand sides of Eqs. (4.1).
   - 4.2 Apply the following steps to find the solutions to the n-3 unknown variables α_1 through α_{n-3} and the residual values R_{res}, G_{res}, and B_{res} in Eqs. (4.1).
     - 4.2.1 Limit each of α_1, α_2…, and α_{n-3} to be one of 12 pre-selected values in H = {0.0, 0.1, 0.2, 0.3, …, 0.9, and 1.0} and generate all possible combinations of the $\tilde{n} - 3$ variables α_1 through α_{n-3}
     - 4.2.2 For each possible combination C_j of α_1 through α_{n-3}, perform the following steps.
       - (i)、 Compute the residual values R_{res}, G_{res}, and B_{res} according to Eqs. (4.1) in the following way:

$$\begin{aligned} R_{res} &= \tilde{R_{sec}} - (\alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-3} \times R_{n-3}); \\ G_{res} &= \tilde{G_{sec}} - (\alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-3} \times G_{n-3}); \\ B_{res} &= \tilde{B_{sec}} - (\alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-3} \times B_{n-3}). \end{aligned} \quad (5.1)$$

       - (ii)、 Compute the sum $M_{res} = R_{res} + G_{res} + B_{res}$.
     - 4.2.3 Find the combination C_{opt} among all possible ones for which all

the residual values R_{res}, G_{res}, and B_{res} computed above are non-negative and the sum M_{res} is the minimum.

4.2.4 Take the n ~ – 3 variables and the residual values corresponding to C_{opt}, denoted as α_1' through α_{n-3}' and R_{res}', G_{res}', and B_{res}', respectively, as the desired solutions to the n–3 unknown variables α_1 through α_{n-3} and the residual values R_{res}, G_{res}, and B_{res} in Eqs. (4.1).

5. Map α_1' through α_{n-3}' to new values α_1'' through α_{n-3}'' according to Eqs. (4.4), respectively.

6. For each residual value, R_{res}', G_{res}', and B_{res}', perform following steps:

   a. if $R_{res}' \geq 64$, then

      bring α_1'' and R_{res}' into Algorithm 5.1 to get α_1''' and R_{res}'' as the new residual value and alpha value; or

   b. if $G_{res}' \geq 64$, then

      bring α_2'' and G_{res}' into Algorithm 5.1 to get α_2''' and G_{res}'' as the new residual value and alpha value.; or

   c. if $B_{res}' \geq 64$, then

      bring α_3'' and B_{res}' into Algorithm 5.1 to get α_3''' and B_{res}'' as the new residual value and alpha value; or

   d. else,

      do nothing.

7. Map the residual values R_{res}'', G_{res}'', and B_{res}'' to R_{res}''', G_{res}''', and B_{res}''' by the following rules:

$$R_{res}''' = 255 - R_{res}''$$
$$G_{res}''' = 255 - G_{res}'' \qquad (5.2)$$

$$B_{res}''' = 255 - B_{res}''$$

8. Assign α_1''', α_2''', α_3''', and α_4'' through α_{n-3}'', and R_{res}''', G_{res}''' and B_{res}''' to the alpha channel values of the pixel in the corresponding n parts p_1 through p_n, respectively, to obtain n stego-parts p_1' through p_n', respectively.
9. Take the resulting image as the desire stego-image I' with secret data embedded.

In the Step 1 above, we divide the cover image by using the user key K to select n ($n \geq 6$) positions of pixels randomly each time until the number of non-selected pixels which is smaller than n.

In Steps 6 and Step 7 above, we map the original residual values R_{res}', G_{res}', and B_{res}' to R_{res}''', G_{res}''', and B_{res}''', respectively. In Step 6, we divide the original residual values into two parts to limit R_{res}'', G_{res}'', and B_{res}'' to be between 0 and 63. Then in Step 7, we map R_{res}'', G_{res}'', and B_{res}'' to R_{res}''', G_{res}''', and B_{res}''', respectively, by $X_{res}''' = 255 - X_{res}''$ where $X = R, G, B,$ to get new mapped values X_{res}'''. The reason behind this mapping is to avoid embedding *small* values into the alpha channel. In Step 6, we lower the residual values, and then compute $X_{res}''' = 255 - X_{res}''$ to get X_{res}''' which is larger than its original values, X_{res}'. In this way, the resulting PNG image will not become white-noised, and that is desired from the viewpoint of steganography.

5.3.2 Proposed Data Extraction Process

To extract the hidden data, we proposed a data extraction method. The detail algorithm will be given as follows.

Algorithm 5.3. Process of the recovery of residual values and alpha values.

Input: a residual values V and an alpha value α.

Output: an original residual value V' and an alpha value α' .

Steps.

1. Transform α into an 8-bit binary string $a_1,a_2\ldots,a_8$, and perform following steps:
   a. if $a_3 = 0$ and $a_4 = 0$,
      compute $V' = V + 192$; or
   b. if $a_3 = 0$ and $a_4 = 1$,
      compute $V' = V + 64$; or
   c. if $a_3 = 1$ and $a_4 = 0$,
      compute $V' = V + 128$; or
   d. if $a_3 = 1$ and $a_4 = 1$,
      do nothing.
2. Set $a_3 = 1$ and $a_4 = 1$.
3. Transform a_1 through a_8 into a decimal number α'.
4. Take V' and α' as the desired digits.

Algorithm 5.4. Process of data extraction.

Input: a stego-image I', a secret key K, and a random number generator f.

Output: a secret message S.

Steps.

1. Divide I' into n parts, p_1' through p_n' conceptually.
2. For each pixel P' located at the same positions in the p_1' through p_n', perform the following steps.
   2.1 Extract the alpha values, $\alpha_1\ldots$, α_{n-3}, R_{res}, G_{res}, and B_{res} from p_1' through

p_n'.

2.2 Get the non-mapped residual values R_{res}', G_{res}' and $B_{res}'$from R_{res}, G_{res} and B_{res} by Eqs. (5.2).

2.3 Pair up the first three alpha values and non-mapped residual values to get (α_1, R_{res}'), (α_2, G_{res}') and (α_3, B_{res}').

2.4 For each pair, perform Algorithm 5.3 to get the original residual value R_{res}'', G_{res}'' and B_{res}'' and alpha value α_1', α_2', and α_3'.

2.5 Bring α_1' through α_{n-3}' into Eqs. (4.5) to get the hidden solutions to the unknown variables, α_1'' through α_{n-3}''.

2.6 Use Eqs. (4.6) to compute the *mapped* secret data, R_{sec}, G_{sec}, and B_{sec}:

$$\begin{aligned} R_{sec} &= \alpha_1'' \times R_1 + \alpha_2'' \times R_2 + \ldots + \alpha_{n-3}'' \times R_{n-3} + R_{res}''; \\ G_{sec} &= \alpha_1'' \times G_1 + \alpha_2'' \times G_2 + \ldots + \alpha_{n-3}'' \times G_{n-3} + G_{res}''; \\ B_{sec} &= \alpha_1'' \times B_1 + \alpha_2'' \times B_2 + \ldots + \alpha_{n-3}'' \times B_{n-3} + B_{res}'', \end{aligned} \quad (4.6)$$

where R_k, G_k, and B_k (k = 1, 2, …, $n-3$) represent the red, green, and blue channel values of p_k', respectively.

2.7 From R_{sec}, G_{sec}, and B_{sec}, use Eqs. (4.7) to compute corresponding *non-mapped* values R_{sec}', G_{sec}', and B_{sec}':

$$\begin{aligned} R_{sec}' &= (R_{sec} - 1) / 3; \\ G_{sec}' &= (G_{sec} - 1) / 3; \\ B_{sec}' &= (B_{sec} - 1) / 3. \end{aligned} \quad (4.7)$$

2.8 Combine the values of R_{sec}', G_{sec}' and B_{sec}' in order, and transform the result into a 6-bit binary string D.

3. Combine the D's of all pixels in order into a sequence D'

4. Apply the following steps to D':

4.1 divide every 8 bits of D' into a *group of segments* f_1, f_2…, f_m;

4.2 transform the sequence of f_1 through f_m into a sequence of decimal

numbers, d_1 through d_m, which is expressed as an ASCII code respectively.

5. Transform all d_1 through d_m into a sequence of characters C.
6. Take C as the desired secret data.

In Step 1 above, the way we divide the cover image is the same as in Algorithm 5.2. In Step 4.1 above, if the number of bits in the last *group of segments* is smaller than eight, we will append several bits of 0 to the original bits until the number of bits in the last segment becomes eight.

5.4 Experimental Results

In this section, we will show some experimental results of applying the proposed data hiding method. The user interfaces for the proposed data hiding and data extraction methods will be shown in Figures 5.3 and 5.4. The experimental results are shown in Figures 5.5 and 5.6. The two original cover images are shown in Figures 5.5(a) and 5.6(a). The stego-images are shown in Figures 5.5(b) and 5.6(b). Note that, the size of cover images which are shown in Figures 5.5 and Figures 5.6 are 512 × 512, all in PNG format. Figure 5.7 shows the secret file, and the extracted file is shown in Figure. 5.8.

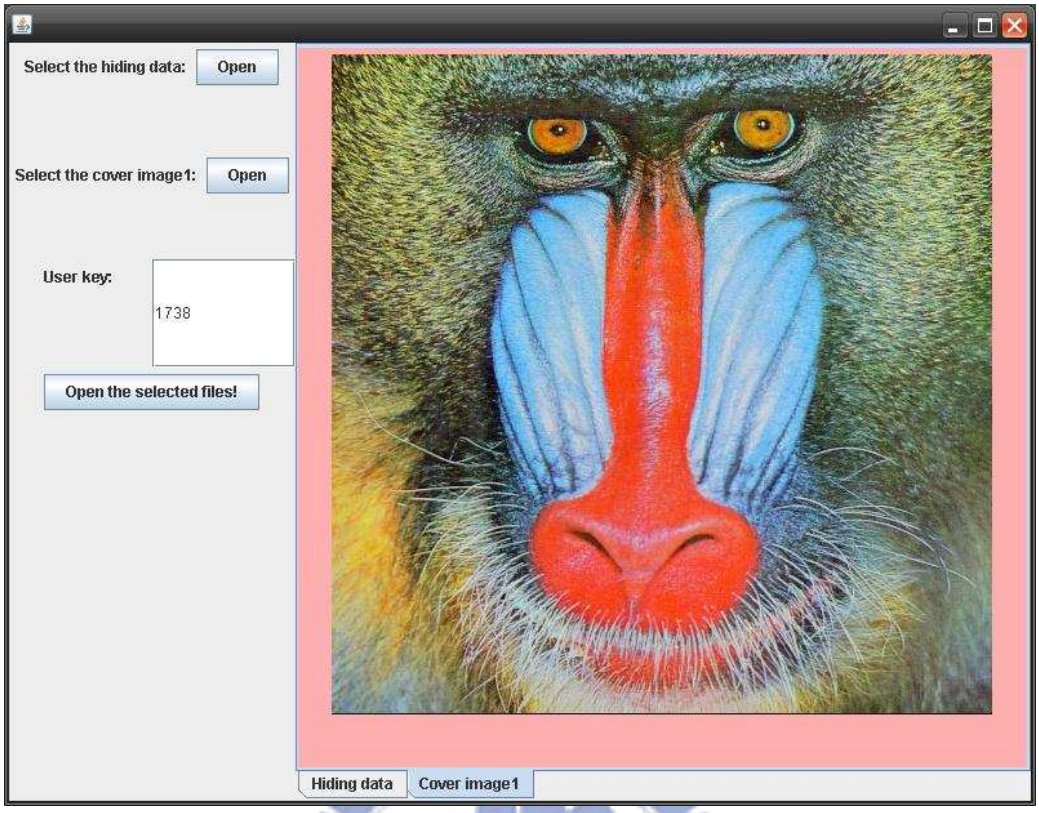


Fig. 5.3 The user interface for data hiding.

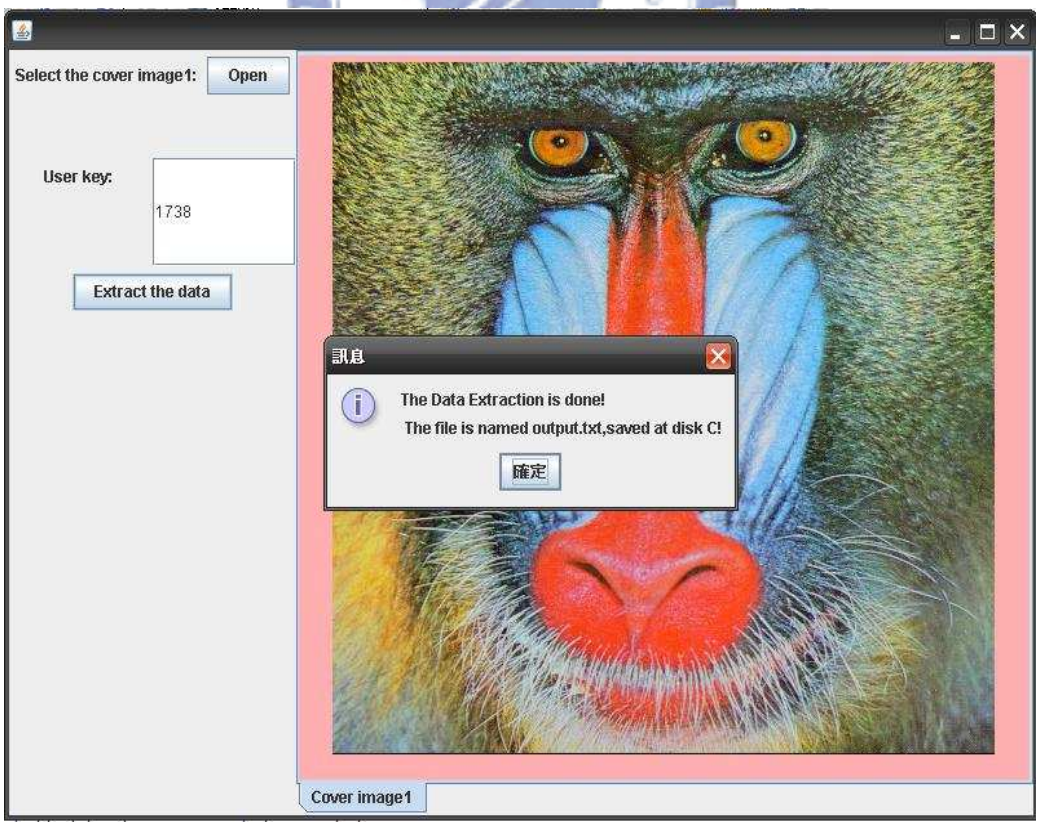
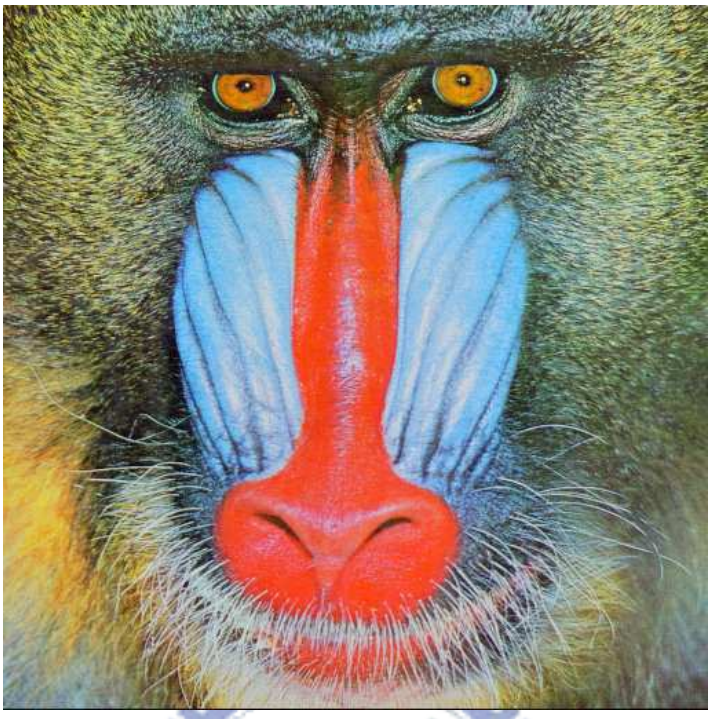


Fig. 5.4 The user interface for data extraction.



(a)

(b)

Fig. 5.5 An experimental result of proposed method. (a) The original cover image. (b) The stego-image.

(a)

(b)

Fig. 5.6 Another experimental result of proposed method. (a) The original cover image. (b) The stego-image.

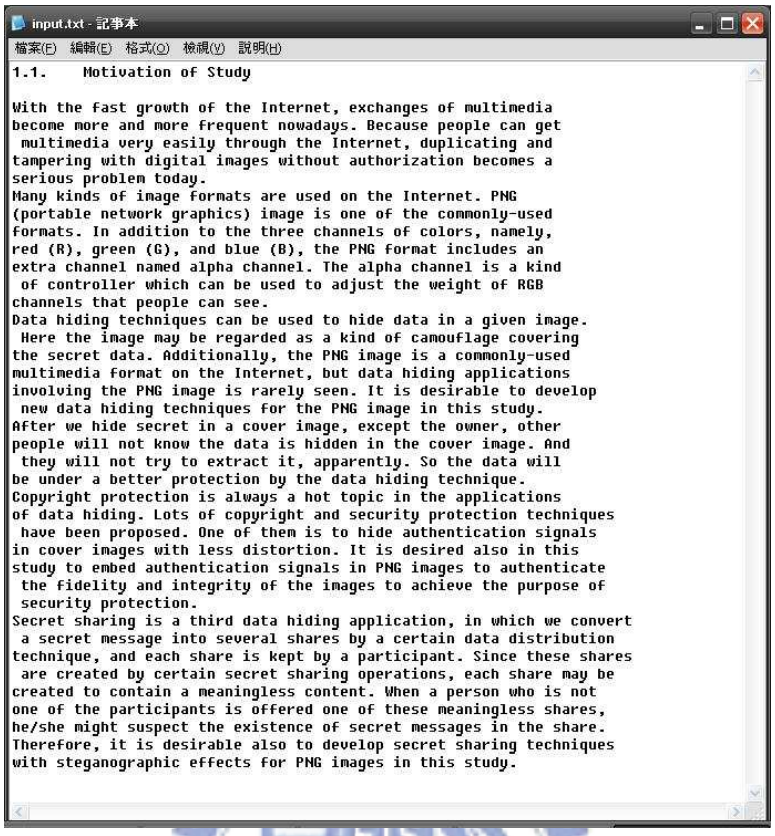


Fig. 5.7 The secret file.

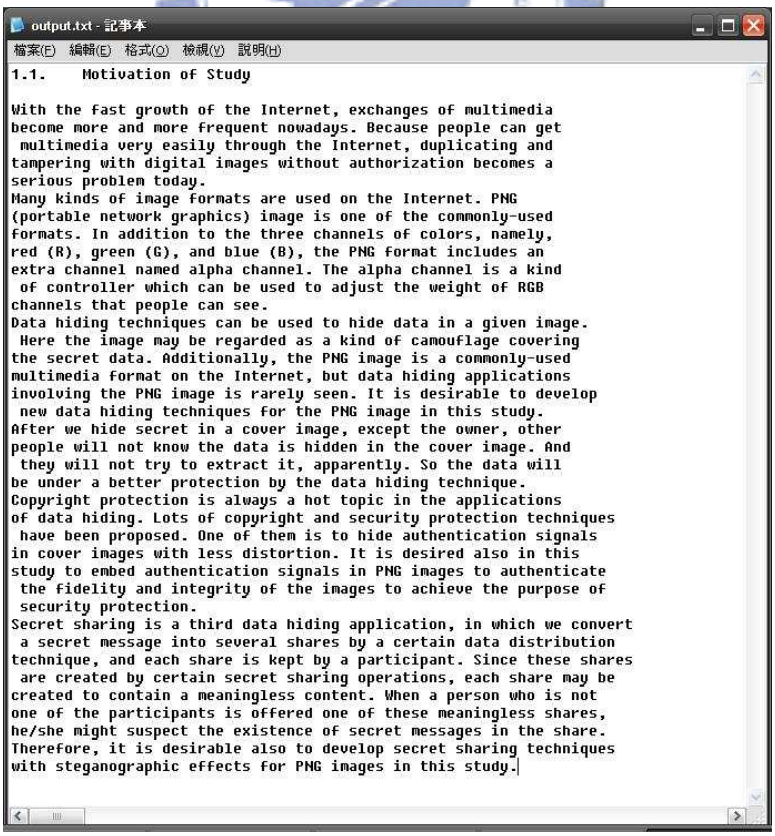
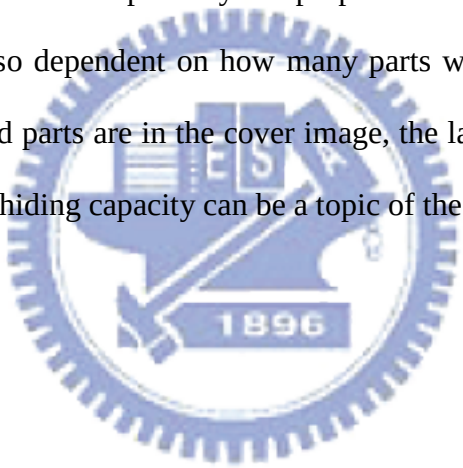


Fig. 5.8 The extracted secret file.

5.5 Discussions and Summary

In this chapter, we have proposed a data hiding method which is based on the proposed secret sharing method described in Chapter 4. We divide a cover image into n parts and regard each of them as an independent image to apply the data hiding method. In this method, we hide all the solutions in the alpha channels. But the solution of the n^{th} unknown variable, the residual value, always differ from each other hidden data, which may make the resulting PNG image become white-noised. We therefore also proposed a value division method to solve such a problem.

We can hide three bits in a pixel by the proposed data hiding method, and the data hiding capacity is so dependent on how many parts we divide the cover image into. The less the divided parts are in the cover image, the larger capacity we can get. How to enlarge the data hiding capacity can be a topic of the future works.



Chapter 6
Conclusions and Suggestions for Future Works

6.1 Conclusions

In this study, we have proposed several methods for data hiding purposes, such as file authentication, covert communication, and secret sharing using PNG images as cover data.

For file authentication, we have proposed a block-based method which can hide authentication signals in the spatial domain. We divide a PNG cover image into 3×3 blocks with three color channels (red, green, and blue) and an alpha channel. We regard each block in the four channels of the PNG image as a 3×3 *gray-level* image, and we combine the four channels to form a 6×6 gray-level image with 36 pixels. In each 6×6 block, we select several pixel values and add up them to get a sum, and we divide the sum by a divisor, which is generated by a user key, to get a remainder. We adjust the selected pixel value to make the re-computed remainder becomes 0. That is a kind of authentication signals.

For secret sharing, we have proposed a method that can share the secret of multimedia by solving simultaneous equations. We generate n shares from $n - 1$ cover images, and we allow the user to select the $n - 1$ cover images. In this study, we hide the solutions to the $n - 1$ unknown variables in the alpha channels of user-selected cover images. After solving each equation, a residual value will be generated. We hide the residual value, the solution to the n^{th} unknown variables, in the three color channels (red, green, blue) of a residual image.

For covert communication, we have proposed two methods which are based on

the proposed authentication method and the proposed secret sharing method. In the first data hiding method which is based on the proposed authentication method, we count the number of odd numbers and that of the even numbers of the selected pixel values in each 6×6 block. We adjust the numbers of odd numbers and even numbers to hide data. We also apply the two methods to yield a combined data hiding and authentication method. We can authenticate the image with hidden data to see whether it is tampered with or not. And we can also decide whether the hidden data in the image is correct or not. In the second data hiding method which is based on the secret sharing method, we divide a cover image into n parts and regard them as a cover image or a residual image to apply the proposed method. Because of the difference of the solution hidden, it may make the resulting image white-noised. To solve this problem, we divide each larger solution into two parts and hide them in different places. The approach can make the stego-image to possess better quality.

The experimental results show the feasibility and practically of the proposed methods for file authentication, covert communication, and secret sharing.

6.2 Suggestions for Future Works

Several suggestions for future research works are listed as follows:

1. It is interesting to hide user information when we embed the authentication signals in a cover image.
2. In the first data hiding method which is based on the authentication method, we can divide the 18 selected values into more groups to enlarge the data hiding capacity.
3. For the authentication method, it is interesting to divide the cover image into $n \times n$ blocks where $n > 3$.
4. In the secret sharing method, it is interesting to hide the residual image into

others shares by some reversible data hiding method.

5. In the second data hiding method, the more parts are divided in the cover image, the less data hiding capacity can be used to hide data. Therefore, it is interesting to find an approach that can have more data hiding capacity and better quality.
6. It is interesting to the hide features of a cover image when embedding authentication signals, so that we can recover the cover image if it is tampered with.

References

[1] C. T. Wang and W. H. Tsai, “Data hiding in PDF files and applications by Imperceivable modifications of PDF object parameters,” *Proceedings of 2008 Conference on Computer Vision,* Ilan, Taiwan, Aug. 2008.

[2] L. Y. Weng and W. H. Tsai, “Integrity authentication of grayscale document images surviving print-and-scan attacks,” *Proceedings of 2005 Conference on Computer Vision, Graphics and Image Processing*, Taipei, Taiwan, June 2005.

[3] P. M. Huang and W. H. Tsai, “Copyright protection and authentication of grayscale images by removable visible watermarking and invisible signal embedding techniques: a new approach,” *Proceedings of 2003 Conference on Computer Vision, Graphics and Image Processing*, Kinmen, Taiwan, pp. 276-283, 2003.

[4] Y. C. Chiu and W. H. Tsai, “Copyright protection by watermarking for Color Images against Rotation and Scaling Attacks Using Peak Detection and Synchronization in Discrete Fourier Transform Domain,” *Proceedings of Third Workshop on Digital Archives Technologies,* Taipei, Taiwan, pp. 207-213, Aug. 2004.

[5] I. S. Lee and W. H. Tsai "Security protection of software programs by information sharing and authentication techniques using invisible ASCII control codes," *International Journal of Network Security*, vol. 10, no. 1, pp. 1-10, Jan. 2010.

[6] C. C. Lin and W. H. Tsai “Secret image sharing with steganography and authentication,” *Journal of Systems & Software*, vol. 73, no. 3, pp. 405-414, Nov.-Dec. 2004.

[7] A. Shamir, “How to share a secret,” *Communications of Association for*

Computing Machinery , vol. 22, no. 11, pp. 612- 613, Nov. 1979.

[8] G. R. Blakley, “Safeguarding cryptographic keys,” *Proceedings of the National Computer Conference*, New York, U. S. A., pp. 313–317, 1979.

[9] C. C. Lin and W. H. Tsai, “Secret multimedia information sharing with data hiding capacity by simple logic operations,” *Proceedings of 5th World Multiconference on Systemics, Cybernetics, and Informatics, Vol. I: Information Systems Development*, Orlando, Florida, U. S. A., pp. 50-55, July 2004.

[10] Cramer, et al., “On codes, matroids, and secure multiparty computation from linear secret-sharing schemes”, *IEEE Transactions on Information Theory*, vol. 54, no. 6, pp. 2644-2657, June 2008.

[11] I. S. Lee and W. H. Tsai, "Data hiding in grayscale images by dynamic programming based on a human visual model," *Pattern Recognition*, vol. 42, no. 7, pp. 1604-1611, July 2009.

[12] C. H. Lai and W. H. Tsai, “Data hiding by stacking up two PNG images,” *Technical Report*, Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, June 2007.

[13] Ni, et al., “Reversible data hiding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354-362, Mar. 2006

[14] T. Y. Liu and W. H. Tsai “Robust watermarking in slides of presentations by blank space coloring: a new approach,” accepted and to appear in *LNCS Transactions on Data Hiding and Multimedia Security*, 2009.

[15] T. Y. Liu and W. H. Tsai "A new steganographic method for data hiding in Microsoft Word documents by a change tracking technique," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 1, pp. 24-30, Mar. 2007.

[16] I. S. Lee and W. H. Tsai "Data hiding in binary images with distortion-minimizing capabilities by optimal block pattern coding and dynamic

programming techniques," *IEICE Transactions on Information and Systems*, vol. E90-D, no. 8, pp. 1142-1150., July 2007.

[17] J. Jiang and A. Armstrong “Data hiding approach for efficient image indexing,” *IEE Electronics Letters*, vol. 38, no. 23, pp.1424-1425, Nov. 2002.