國 立 交 通 大 學

資訊學院

資訊科學與工程研究所

博 士 論 文

用於多媒體安全的影像驗證與秘密傳輸的新技術之研究

A Study on New Techniques for Image Authentication and Covert Communication for Multimedia Security Applications

研 究 生: 李 哲 瑋

指 導 教 授: 蔡 文 祥 博士

中華民國 一百零一 年 五 月

用於多媒體安全的影像驗證與秘密傳輸的新技術之研究

A Study on New Techniques for Image Authentication and Covert Communication for Multimedia Security Applications

研　究　生：李 哲 瑋　　Student: Che-Wei Lee

指 導 教 授：蔡 文 祥 博士　　Advisor: Dr. Wen-Hsiang Tsai

國 立 交 通 大 學 資 訊 學 院
資 訊 科 學 與 工 程 研 究 所
博 士 論 文

A Dissertation Submitted to
Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
In Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy
in Computer and Information Science

May 2012
Hsinchu, Taiwan, 300
Republic of China

中華民國 一百零一 年 五 月

用於多媒體安全的影像驗證與秘密傳輸的新技術之研究

研究生：李哲瑋　　　　　　　　　指導教授：蔡文祥博士

國立交通大學資訊學院

資訊科學與工程研究所

摘要

影像驗證是保護數位影像內容一種有效率的方法，此技術的目的在檢驗數位影像的完整性與真確性。當一張數位影像遭受竄改或破壞時，影像驗證技術可以偵測竄改並將影像遭竄改的部分定位出來。本論文中，我們針對灰階公文影像、黑白影像與灰階影像分別提出了可修復式之影像驗證方法。所提出的方法不只可以偵測並定位竄改，還能進一步地修復影像遭到竄改的部分。對於灰階公文影像與黑白影像，我們首先將輸入的影像轉換成含有透明層的PNG格式影像，接著利用Shamir的秘密分享技術將驗證訊號與修復訊號轉換成許多的秘密碎片，並以特定的方式將秘密碎片散佈在透明層中，以產生受保護的影像。如此一來，在一張遭受竄改的影像，只要能從透明層中蒐集回足夠數量並存活的秘密碎片，再經由Shamir秘密分享的反程序，便可進行影像修復。此外，本論文亦提出一個彩色影像驗證方法，該法是立基於我們所提出的一個利用影像透明層來隱蔽資料的新型資訊隱藏方法。而在灰階影像驗證方面，我們設計了一種盒子碼，它的概念是將像素中較重要的幾個位元壓縮成較短的位元。此盒子碼兼具驗證訊號與修復訊號的功能，因此我們利用此盒子碼來發展可修復式灰階影像驗證。

在多媒體安全方面，我們提出一個具自我驗證能力，並以常見的試算表作為機密訊息載體的新型機密傳輸方法。我們將機密訊息透過Shamir秘密分享技術轉換成特定數量的數字碎片，將這些數字碎片依據所設計的原則去替換掉試算表中某些數字；在擷取機密訊息時，只要藉由檢查多份數字碎片運算後之結果是否一致，便可達到自我驗證的功能。如此一來，使機密訊息接收者在不需任何輔助資訊的情況下，便可確認機密訊息的真確性與完整性。在可逆式資訊隱藏領域，我們提出一個兩階段式直方圖位移之可逆式資訊隱藏技術。藉由兩種直方圖位移方式的搭配，所提出的方法可以重複性地操作以達到更大的資訊隱藏量，並且在擷

取所藏入的資訊時，本法不需任何額外的輔助資訊即可進行。

以上所提出的方法之效能皆已經由理論分析與實驗做過評估，結果顯示了它們在實際應用上的可行性。

A Study on New Techniques for Image Authentication and Covert Communication for Multimedia Security Applications

Student: Che-Wei Lee **Advisor: Dr. Wen-Hsiang Tsai**

Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University

Abstract

Image authentication is an effective technique to protect digital image content and the goal of such techniques is to check the integrity and fidelity of a digital image. It means that image authentication techniques are capable of providing the function of tempering detection and tampering localization when an image has been tampered with. In this dissertation, repairable image authentication methods for grayscale document images, binary images and grayscale images are proposed, respectively. The proposed methods can not only detect and localize the tampered region but further repair the tampered region for an attacked image. For authentication of grayscale document images and binary images, an input cover image is first transformed into a stego-image in the PNG format with an alpha channel plane. Subsequently, the information used for authentication and image repairing are transformed into partial shares by the Shamir's secret sharing method, which are then distributed in a well-designed manner into an alpha channel plane to create a stego-image in the PNG format. In this way, data repairing can be applied to tampered region by a reverse Shamir scheme after collecting enough shares from the alpha channel plane. In addition, a color image authentication method which is based on a new data hiding method proposed is developed in this dissertation. The data hiding method conceals data by utilizing the character of the transparency of the alpha channel plane. For authentication of grayscale images, a method which is based on the concept of compressing a number of the most significant bits of a pixel’s gray value into a shorter “bin code” is proposed. The bin code is for use both as an authentication signal for the pixel and as an index for generating the data for repairing the pixel when it is authenticated to have been tampered with.

In the aspect of multimedia security, a new covert communication method which takes Spreadsheets as the cover medium and applies Secret Sharing method to yield a self-authentication capability is proposed. Through checking the consistency of the copies in value computed from secret shares, the proposed covert communication has the capability of self-authentication. In the field of reversible data hiding, a blind two-phase reversible data hiding method based on histogram shifting is proposed. Two types of histogram-shifting techniques are executed in two phases such that the proposed method can be performed iteratively during the data embedding process and blindly during the data extraction process.

The feasibility and effectiveness of all the above proposed method have been demonstrated by theoretical analyses and experiments.

Acknowledgement

I would like to express my sincere appreciation to my advisor, Professor Wen-Hsiang Tsai, for his guidance and invaluable training in every stage of the dissertation study. His knowledge, patience, and energy have urged me to become a mature researcher.

I am also grateful to the colleagues in the Computer Vision Laboratory at National Chiao Tung University for their valuable help and comments during this study. They certainly enriched my doctoral study life.

I would like to convey my gratitude to Miss Tzu-Ling Chen, my girlfriend, for her love, trust and encouragement. I would also like to thank my parents, sister, brother-in-law, and two lovely nephews for their love, support, and endurance. This dissertation is dedicated to them.

Thank Jesus Christ for giving me such a wonderful journey of the study life.

誌謝

整本論文我最想寫的就是這個部分。

首先我要感謝我的指導教授 蔡文祥教授，老師在研究上展現的創新精神與堅持到底的毅力，為我樹立了良好的榜樣。此外，老師在星巴克親自指導我改出一篇又一篇的論文，是我研究歷程中受益最深，也最難忘的一段時光。我要向老師致上我真摯的感謝，謝謝老師所給予的指導、磨練、與耐心。

我也要向實驗室全體成員致上謝意，特別是神恩、雅琳給予的協助，學弟致瑋、新翔的熱心幫忙，以及助理瑋馨、學弟為帥、彥成、張揚與冠霖的支援。我也要特別提及學弟楊國鋒，他在數學上的專業學識令人敬佩，而他健談的性格也豐富了我的研究生活。已畢業的學弟黃政揆與學長吳至仁，認識你們是一件很棒的事。

我的幾位老友，志光、哲毅、煒翰、周節與阿名，謝謝你們的陪伴與鼓勵，你們存在的價值無法衡量。

接著是我重要的家人。

我的父母，感謝你們的付出與關懷，使我可以選擇自己的道路，謝謝你們的信任與激勵。我的姊姊，還好這個世界上有妳，還好我的人生有妳，我才得以跨越一些難關，完成許多重要的事。我的姊夫，謝謝你把姊姊照顧得很好，我間接蒙受其利，這是一個食物鏈的問題，謝謝你。我的外甥子昕與子夫，你們的純真、擁抱與熱情，總是帶給我很多力量，我愛你們。

最後是我的女友，姿伶小姐，我要對她獻上我最深的感謝，她是這段歷程中最重要的人物，因為有她的陪伴、支持、信任與理解，這一切才會實現。我的感謝也包括她的家人，感謝他們的照顧與關心。

感謝主，榮耀歸給神。

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation of Study

With the era of cloud computing coming, data stored originally in personal computers mostly will eventually be moved to and processed in powerful servers at far ends. However, how can one be sure that personal data accessed from cloud servers are intact? Undoubtedly, this problem of data security has become a significant issue in the age of cloud computing. This study tries to explore this issue of keeping digital image data secure. Furthermore, with the fast advance of digital technologies, it is easy to make visually imperceptible modifications to the contents of digital images. To ensure the integrity and authenticity of a digital image is thus a challenge. In addition, it is hoped that if a critical image part is authenticated to have been altered illicitly, its original content can be recovered. The use of an *image authentication* technique provides a solution to this issue and this thesis study is also devoted to this second security issue of keeping digital image data.

On the other hand, covert communication is the art of concealing secret information into a *cover medium* in an undetectable way and only a sender and an intended receiver know the existence of the hidden data in the resulting *stego-medium*. However, if an adversary subtly makes modifications to the passing-by stego-medium for the purpose of misleading a receiver, or if the stego-medium undergoes some unintentional destruction during the transmission, a receiver might extract an incorrect secret message from the stego-medium. Therefore, how to provide a self-authentication capability for verifying the correctness of an extracted secret message is a challenge. This thesis study is devoted to this issue of multimedia security as well.

Also very important for the purpose of maintaining multimedia security is the development of reversible data hiding techniques. Such techniques embed message data into host images and are capable of restoring the resulting stego-images to their original states after the hidden data are extracted. In particular, histogram shifting is an efficient technique used in many reversible data hiding methods. However, before extracting the hidden data, certain information, called *overhead information*, needs to be known in advance. Therefore, a fourth goal of this research is to solve this common problem of how to share such overhead information between the sender and the recipient.

In summary, the goals of this thesis study include: 1) secure keeping of image data; 2) image authentication; 3) self-authentication of embedded messages; and 4) overhead sharing for reversible data hiding. Fulfillments of these goals together enhance the state-of-art studies on image authentication and covert communication for multimedia security applications.

1.2 Overview of Data Hiding and its Applications related to multimedia security

Data hiding is a process of embedding information imperceptibly into a certain digital file that acts as a host. Via the use of data hiding techniques, many applications related to multimedia security can be accomplished as shown in Fig. 1.1.

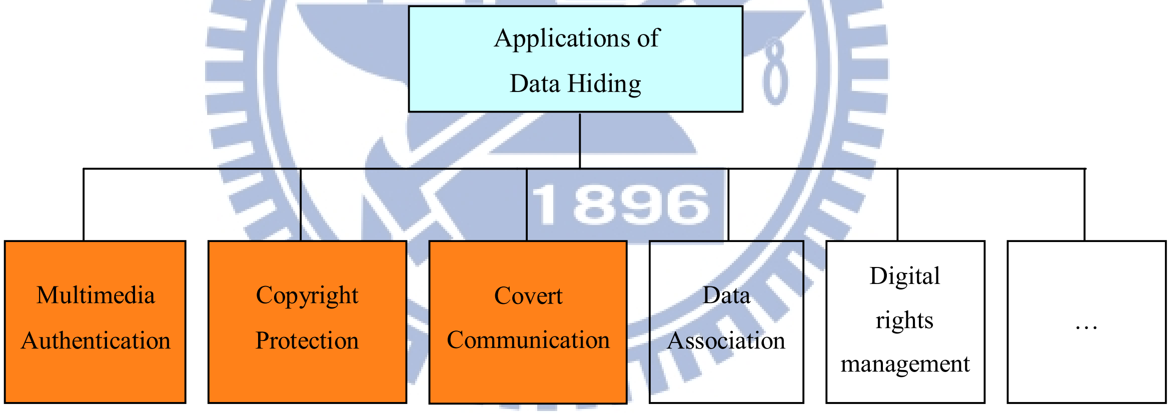

Applications of Data Hiding
Multimedia Authentication
Copyright Protection
Covert Communication
Data Association
Digital rights management
…

Fig. 1.1 Scope of data hiding applications.

Through data hiding techniques, multimedia authentication can be accomplished. Specifically, information such as authentication signals can be embedded by a data hiding technique into a cover image to be protected to create a stego-image, and illicit modifications made to the stego-image may then be localized such that the integrity and fidelity of the original image content can be checked. It is noted that the major concern in the application of image authentication is the fragility of the embedded signals. In practical uses, it is desirable that an image authentication method is capable of further recovering the localized tampered region.

On the other hand, in the application of copyright protection, an owner of a digital file can use data hiding techniques to embed a visible or invisible digital watermark into the file content to claim the ownership of the content. The major concern in such an application is the robustness of the embedded signals, which means that a surviving and recognizable watermark such as a logo has to be extracted from the stego-image even after attacks

A third application of data hiding is covert communication, by which people can hide a secret message into a cover file, resulting in a stego-file; and a receiver of the latter can extract the hidden message from it to complete the communication. In more detail, the major concern in the application of covert communication is the undetectability of the embedded information. By concealing secret information into a cover medium in an undetectable way, only a sender and an intended receiver know the existence of the hidden data in the resulting stego-medium.

In the following sections, a survey of related works is given in Section 1.3, followed by the contributions of this study in Section 1.4. Finally, the organization of this thesis is described in Section 1.5.

1.3 Survey of Related Works

In this study, the concerned topics include Image authentication, covert communication, and data hiding. With the era of cloud computing coming, data stored originally in personal computers mostly will eventually be moved to and processed in powerful servers at far ends. A problem involved here is how to make sure that personal data accessed from cloud servers are intact. This problem of data security is a very significant issue. This study explores this issue of keeping digital image data secure, and proposes the use of an *image authentication* technique as a solution.

In the literature, several authentication methods for binary images have been proposed. Wu et al. [1] explored flippable image pixels in which data can be embedded without causing noticeable artifacts for the purpose of authentication and annotation of binary images. Yang et al. [2] proposed a pattern-based data hiding method for binary image authentication, which uses the flippability of pixels effectively by a connectivity-preserving criterion [3]. Kim et al. [4] chose a set of pseudo-random pixels which were later cleared for embedding authentication code in a binary or halftone image. Tzeng and Tsai [5] embedded authentication codes into blocks of a cover binary image for the later use of tampering detection in the verification process. In addition, to reduce the image distortion caused by the authentication codes embedding, a technique of using a code holder allowing multiple locations of pixels for embedding the codes was proposed. Lee et al. [6] embedded watermark data into one pixel in each binary image block by a Hamming-code-based data embedding method with small distortions and low false negative rates. Method proposed in [7] utilized an edge-line similarity measure to choose flippable pixels, yielding smaller distortion compared with that of method in [6].

As to the image authentication for grayscale images, several fragile watermarking techniques for image authentication have been proposed in the past and they may be categorized into two approaches: block-wise [8-14] and pixel-wise [15-18]. Methods of the former approach embed fragile watermarks as authentication signals into non-overlapping blocks of the cover image and identify possible tampered image parts in the unit of block. One weakness of such block-level authentication methods is that the detail of the tampered image part cannot be located precisely [16]. On the other hand, methods of the second approach [15-18] authenticate images at the pixel level such that tampered image parts can be identified pixel by pixel, yielding a detailed tampering localization result. Liu et al. [15] generated a binary image that is mapped from the difference image computed from the cover image and its so-called chaotic pattern. And the least-significant-bit (LSB) plane was used to accommodate the binary image as the fragile watermark for the use in later image authentication. Because of the binary nature of the embedded fragile watermark, the LSB of a tampered pixel value may coincide with the watermark bit, yielding a high erroneous pixel authentication rate up to 50%. To deal with this phenomenon, a statistical fragile watermarking method which utilizes probability distributions computed from the original pixels and the tampered ones to locate the tampered pixels was proposed in Zhang and Wang [16]. However, the method only works in the case that the tampering ratio is smaller than 1.1% [17]. As an improvement, Zhang and Wang [17] proposed later a fragile watermarking method for authenticating grayscale images using a hierarchical mechanism, which embeds watermark data derived from the pixels and blocks of the cover image into the LSBs of all the pixels. In the authentication process, tampered blocks are identified first, and tampered pixels within the identified blocks are located subsequently.

Covert communication is a technique of concealing secret information into a *cover medium* in an imperceptible way or with a camouflage effect such that only a sender and an intended receiver know the existence of the hidden data in the resulting *stego-medium*. In the literature, emphases were put on the use of multimedia like images, videos, and audios [19-22] because these media in general provide larger embeddable spaces and cause less suspicion due to their wide distributions. And weaknesses existing in human beings' visual capabilities are often exploited to design effective covert communication methods. For example, the methods proposed in [23-25] replace the least-significant bits of pixels in cover images to embed information, and that of [26] uses the parities of palette colors, composed by similar colors, to represent hidden message bits.

In addition to methods developed for multimedia, several others [27-30] used cover media of text, PDF, or Word documents for covert communication. In Brassil and Maxemchuk [27], data are embedded by slightly adjusting the lines, tabs, or characters in text files. Lee and Tsai [28] used special ASCII codes in PDF files to embed data between characters. Liu and Tsai [30] made use of the change tracking function in Microsoft Word to embed data imperceptibly by a document degeneration technique.

As to the topic of data hiding, many data hiding methods for use in the spatial domain of image files [31-35] have been proposed. Bender et al. [31] proposed the technique of least-significant-bit (LSB) replacement, in which a secret message is embedded in the least significant bits of image pixel values. The method yields high data hiding capacities and low computational complexities. Following Bender et al. [31], Wang et al. [32] proposed an optimal LSB-replacement-based method which is integrated with a genetic algorithm to reduce the computation time for finding the optimal hiding result. Mielikainen [33] proposed a modified LSB replacement method which embeds as many bits as the conventional method, but changes less pixel values. Wu and Tsai [34] proposed a steganographic method for images by pixel-value differencing, in which the difference values between pixel value pairs are classified into a number of ranges to decide the number of bits which can be embedded into the pixel pairs. Yang et al. [35] proposed an adaptive k-LSB substitution method in which larger values of k are adopted in the edge areas of the cover image and smaller ones are used for the smooth areas. Also, the range of value differences of consecutive pixel pairs is divided into different levels, and the value of k is adaptively decided according to the level into which the pixel difference value falls. In this way, larger data hiding capacities with higher stego-image qualities can be obtained.

Another data hiding approach is to utilize the frequency domain. Wang et al. [36] transformed image block contents into coefficients in the frequency domain by the discrete cosine transform (DCT), recomputed the AC values of the central block of every nine 8×8 image blocks, and embedded secret bits by modifying the magnitude relations between the new AC values and the original ones. Wu and Hsieh [37] embedded data in the frequency domain by using the so-called zerotree in the rearranged DCT coefficients of the cover image. Besides data embedding techniques using the DCT, the discrete wavelet transform (DWT) [38-40] and the discrete Fourier transform (DFT) [41-42] have also been used.

From another viewpoint, different types of image may be used as cover media such as JPEG [43-45] and palette-based images [46-49]. Fridrich and Du [46] explored the optimal parity of a color palette, and embedded message bits into the parities of palette colors. Tzeng, Yang, and Tsai [47] hid secret data into data-embeddable image pixels by using a color-mapping function, and the color of a data-embeddable pixel is replaced by an optimal one selected from the color palette. Zhang et al. [48] utilized the concept of gregarious color, meaning a color with some other colors similar to it, in the palette to hide at least one secret bit for any pixel with a gregarious color. In [49], Lee and Wu proposed a reversible data hiding method for palette-based images, which adjusts palette colors and image data to embed secret data and side information for reconstruction of the original image content.

Reversible data hiding methods embed message data into host images and are capable of restoring resulting stego-images to their original states after the hidden data are extracted. Histogram shifting is an efficient technique used in many reversible data hiding methods. Ni, et al. [50] shifted slightly part of a histogram between its peak point and the zero point by one pixel value to create an empty *bin* besides the peak point for accommodating message data. The knowledge of the locations of the peak point and the zero point of the histogram need be *memorized* in order to retrieve the hidden data and restore the original image losslessly, making the resulting method *non-blind*. Kuo et al. [51], Lee and Tsai [52], and Fallahpour et al. [53] proposed a similar idea of decomposing an entire cover image into blocks and using the peak point of the histogram of each block to hide data. The technique of block division successfully improves the data hiding capacity but the side information composed of peak points and zero points still should be kept. Tsai et al. [54] explored the similarity of neighboring pixels in the host image and employed a linear prediction technique to generate a residual histogram of the image for data embedding. The data hiding capacity of Tsai et al.'s method [54] is superior to the aforementioned histogram-based hiding methods and maintains simultaneously the high quality of the resulting stego-image; however, the side information of peak points and zero points is still needed. Later, Kim et al. [55] utilized the high spatial correlation between sub-sampled images of the cover image to obtain a high data embedding capacity while keeping the distortion in the stego-image at a low level.

1.4 Main Ideas and Contributions of This Study

In this study, the secret sharing proposed by Shamir in 1979 is utilized to develop a series of image authentication methods with data recovery capabilities. The original aim of the secret sharing method is to deal with the issue of key management in cryptographic systems. This technique transforms the value of a key into many data pieces, called *secret shares*, which are then distributed to some participants. At a later time, the value of the key can be reconstructed using a sufficient number, but *not necessarily all*, of the shares. This results in a property of loss-tolerant capability, which inspires us to apply such a secret sharing technique to develop a data recovery capability for the image authentication method. Accordingly, the content of the image to be authenticated may be transformed into many secret shares; and can be recovered, even if tampered with, when a sufficient number of shares are collected.

However, a problem one may encounter here is how to find enough space in an image to accommodate the secret shares. The embedding spaces in the spatial or frequency domain provided by conventional methods seem insufficient to satisfy the requirement of large-volume spaces for embedding the shares. To deal with this issue, the PNG image with an alpha channel plane is explored in this study to solve the problem. Meanwhile, the transparency of the alpha channel is also utilized to provide a camouflage effect to the embedded secret shares, resulting in a nearly transparent image.

In addition to the previously-mentioned inspiration of using the secret sharing technique, in the following we state further more benefits of using the secret sharing method for data recovery in the image authentication methods to be proposed later in this thesis. Moreover, some additional new ideas behind other methods proposed in this study are also described in this section. Finally, the main contributions made in this study are listed.

A. Why secret sharing

An intuitive strategy for image recovery in an image authentication method is *data duplication*, which means that multiple copies of image data are created and stored somewhere in the image itself. Accordingly, if the image is damaged to some extent, it may be recovered by utilizing the surviving copies. This strategy is reasonable but has two drawbacks. The first is that the data sequence in the stored image copies must be known, and the second is that if a part of the stored image data is destroyed, the image content will possibly not be recoverable even when other parts of the image data are retained.

In this study, we propose the use of the Shamir secret sharing method, also called the (k, n)-thresholding method, to overcome the two drawbacks mentioned above. By the method, the original data are transformed into n shares, and when k of the n shares are collected, the original data can be reconstructed without the prior knowledge of the data sequence. As a result, a higher capability of fault tolerance is yielded. This merit is not found in the approach of data duplication. An example for revealing this merit is shown in Fig. 1.2.

Specifically, in the upper rectangle shown in Fig. 1.2, the original data consisting of two parts, "10" and "20," are duplicated into three copies to become six digits as shown in the figure. Then, if two digits, both "20," survive attacks, the original data unfortunately cannot be reconstructed because both are the same value "20." On the other hand, the rectangle shown below in the figure illustrates the case of (2, 6)-thresholding. The original data 10 and 20 are transformed into 6 shares, and even if only two shares survive, the original data can still be reconstructed without errors. This reveals that in the same condition of collecting two surviving digits, the capability of fault tolerance provided by the (k, n)-thresholding method is better than that provided by the data duplication scheme.

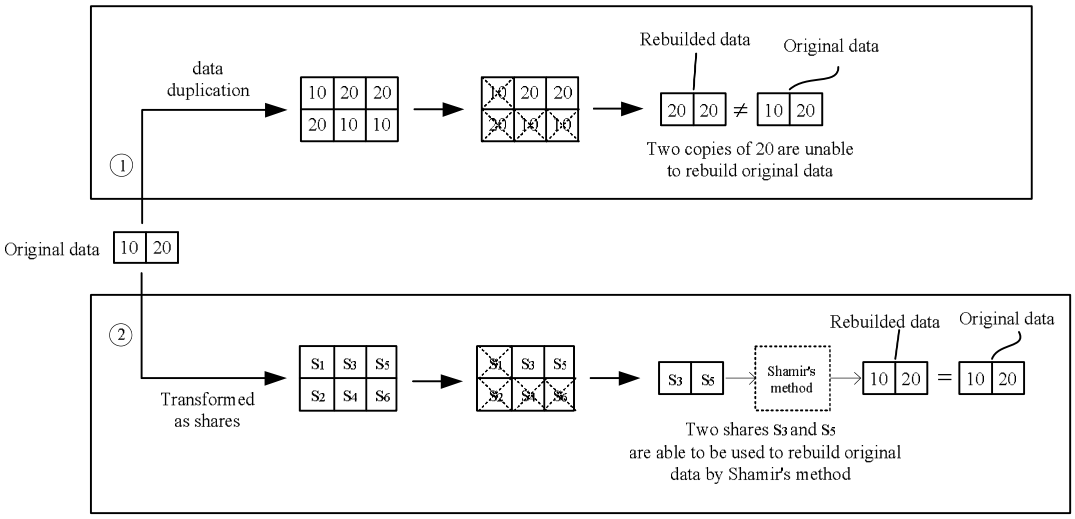


Fig. 1.2 Illustration of the better capability in fault tolerance of using secret sharing than using data duplication.

B. Other new ideas proposed in this study for enhancing multimedia security

In the field of steganography, the existing methods are mostly only concerned with statistical undetectability and none of them can promise the receiver that the message that he/she extracts from a stego-file is trusty and intact. To deal with this weakness, we propose in this study a new steganographic method with a *self-authentication* capability for secret data hiding in *spreadsheets* using the secret sharing technique. In the method, a scheme of (k, k+1)-threshold secret sharing is developed for the first time to provide a self-authentication capability by checking the *value-consistency* of $k + 1$ results coming from all $k + 1$ combinations to determine whether an extracted secret is intact or not.

Finally, in the field of reversible data hiding, histogram shifting is an efficient technique used in many related methods. However, an issue of how to share the information of the used peak points between the sender and the receiver remains to be solved. In this study, a new histogram-shifting-based blind reversible data hiding method composed of two-phase iterations for more effective data hiding is proposed. With the cooperation of the two data hiding phases based on histogram shifting, the proposed method skillfully stores the value of the peak point yielded in phase 1 via the use of phase 2, solving the common problem of sharing the peak information between the two parties.

C. Main contributions of this study

The main contributions of this thesis study are summarized in the following.

1. A new blind image authentication method with a data repair capability for binary-like grayscale document images based on secret sharing is proposed. The proposed method has higher possibility to survive image content attacks and provides pixel-level repairs of tampered image parts.
2. A new approach to binary image authentication based on the uses of PNG images and the secret sharing technique with a self-recovery capability for repairing tampered image data is proposed. The proposed method has the capabilities of tampering detection, tampering localization, and tampering recovery, and also has the capability of reversing the tampered image to its original content.
3. A new data hiding method via PNG images based on Shamir's (k, n)-threshold secret sharing scheme is proposed and can be applied to color image authentication. The proposed image authentication method possesses the merits of losslessness during image authentication, high sensitivity to image alterations, good tampering localization capability, and very low false acceptance and rejection ratios.
4. A new blind pixel-level self-repairing grayscale image authentication method, which is optimal under a minimax criterion of image distortion reduction, is proposed. The proposed method develops the "bin code" for use both as an authentication signal for the image pixel and as an index for generating the data for repairing the image pixel when it is authenticated to have been tampered with.
5. A new covert communication method with a self-authentication capability for secret data hiding in spreadsheets using the information sharing technique is proposed. In the proposed method, the presence of the steganographic content is statistically undetectable and a receiver can confirm the correctness of the extracted secret message.
6. A blind two-phase reversible data hiding method based on histogram shifting is proposed. The proposed method has the advantageous property of blindness and significantly improves the data hiding capacity while keeping low degradations of the image quality when compared with other methods.

In addition, a summarized table listing applications for which the aforementioned proposed methods are developed is given in the following.

Table 1.1 Proper applications of proposed methods.

	Image authentication	Covert Communication	Copyright protection	Data Association
1st method	▲			
2nd method	▲			
3rd method	▲			▲
4th method	▲			
5th method		▲		
6th method		▲		▲

1.5 Thesis Organization

The remainder of this thesis is organized as follows, In Chapter 2, an overview of the proposed techniques and the ideas associated with image authentication, covert communication, and reversible data hiding is given. In Chapter 3, the proposed method for authentication of grayscale document images with a data repairing capability is described. In Chapter 4, the proposed new approach to binary Image authentication is described. The proposed color image authentication method based on a new data hiding technique is presented in Chapter 5. In Chapter 6, the proposed pixel-level self-repairing authentication method for grayscale images is described. In Chapter 7, the proposed covert communication method based on information sharing is described. In Chapter 8, the proposed blind reversible data hiding method is described. Finally, in the last chapter, conclusions of this study and some suggestions for future research are included.

Chapter 2

Overview of Proposed Techniques and Ideas

In this chapter, we describe the main ideas and techniques of the proposed methods for image authentication, steganography and reversible data hiding.

2.1 Repairable Authentication Method for Grayscale Document Images

A method for authentication of document images with an additional self-repair capability for fixing tampered image data is proposed. The input cover image is assumed to be a binary-like grayscale image with two major gray values. After the proposed method is applied, the cover image is transformed into a stego-image in the PNG format with an additional alpha channel for transmission on networks or archiving in databases. The stego-image, when received or retrieved, may be verified by the proposed method for its authenticity. Integrity modifications of the stego-image can be detected by the method at the block level and repaired at the pixel level. In case that the alpha channel is totally removed from the stego-image, the entire resulting image is regarded as inauthentic, meaning that the fidelity check of the image fails. The proposed method is based on the so-called (k, n)-threshold secret sharing scheme proposed by Shamir [56] in which a secret message is transformed into n shares for keeping by n participants; and when k of the n shares, not necessarily all of them, are collected, the secret message can be recovered losslessly. Such a secret sharing scheme is useful for reducing the risk of incidental partial data loss.

To the best of our knowledge, this is the first secret-sharing-based authentication method for binary-like grayscale document images. It is also the first authentication method for such document images through the use of the PNG image. Note that this method is not a secret-sharing technique, but a document image authentication method.

2.2 Recoverable Authentication Method for Binary Images

A new method for binary image authentication with an additional self-recovery capability for repairing tampered image data is proposed. The method is based on the (k, n)-threshold scheme proposed by Shamir [56] for secret sharing and is developed via the use of PNG images with alpha channels. The secret sharing scheme is used in the proposed method to enhance the capability of self-recovery of lost data as well as to generate share data which carry information for image authentication and data recovery.

More specifically, the alpha channel in a cover PNG image is used in this study as a carrier for embedding secret shares, though it is defined originally for creating desired image transparencies. Since all channels in a PNG image should have the same sample depth according to the PNG standard [57], a PNG encoder will scale all the channels' samples up to the same depth. Therefore, a binary image used in this study whose sample depth is 1 will automatically be scaled up to have a depth of 8 but with only two pixel values 0 and 255, when an alpha channel with sample depth 8 is appended to the original binary image. Fig. 2.1 shows how samples of depth 1 are mapped to depth 8 in the scaling process.

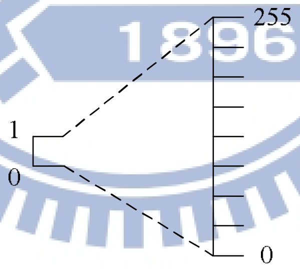


Fig. 2.1 Illustration of scaling sample depth of a binary image from 1 to 8 by a PNG encoder.

However, it is found that embedding data into the alpha channel will yield random transparency in the resulting stego-image, producing an undesired opaque effect with a lot of white noise. A solution to this phenomenon, as proposed in this study, is to map the resulting alpha channel values into a small range near their extreme value of 255, yielding instead uniformly distributed and nearly imperceptible noise in the alpha channel. This idea of creating the effect of imperceptibility by alpha-channel value mapping is proved feasible by the experiments conducted in this study.

Finally, to recover losslessly the original content of a tampered block, the technique adopted in the proposed method is to distribute the generated multiple shares randomly into the alpha channel. The resulting location randomness of the shares together with their multiplicity allows the share data to have more chances to survive attacks without being totally destroyed, thus promoting the capability of image recovery of the proposed method.

2.3 Color Image Authentication Based on an Information-sharing-based Data Hiding Method

In the *information-sharing-based* data hiding method proposed in this study, a PNG image is used as the cover image in which the alpha-channel value of each pixel is set to be 255 initially. That is, the cover image is a totally transparent color one at the beginning of the proposed data hiding process. A data string to be hidden is transformed into shares by the Shamir's secret sharing method, which is then embedded into the alpha-channel plane of the cover PNG image. Coefficient parameters involved in the Shamir method are used as *carriers* of the data to be hidden in the proposed method. A prime number used in the method, which is found to dominate the resulting visual quality and data hiding capacity of the stego-image, is properly selected. Also, a mapping function is designed for adjusting the alpha-channel values to create uniform transparency in the alpha-channel plane, resulting in an imperceptible effect in the stego-image. The original R, G, and B channels are untouched so that the original image appearance revealed by the color information of these three channels is kept.

The proposed data hiding method is suitable for applications of *image authentication* and metadata hiding. In particular, the application of the proposed method to color image authentication is investigated and relevant algorithms are proposed in the subsequent chapter.

2.4 Self-repairing Authentication Method for Grayscale Images

A method for pixel-level grayscale image authentication using fragile authentication signals with an additional capability for repairing attacked image parts automatically is proposed. The method is based on the concept of compressing a number of the most significant bits (MSBs) of a pixel's gray value into a shorter "bin code" for use both as an authentication signal for the pixel and as an index for generating the data for repairing the pixel when it is authenticated to have been tampered with. The bin code is generated from a bin-mapping scheme which transforms each pixel's gray value into one of eight "bins," coded by three bits. It is proved that the choice of using three bits out of eight ones in a pixel as the bin code is *optimal* under a *minimax* criterion of reducing the total maximum pixel-level gray-value distortion resulting both from authentication signal embedding and from tampered pixel repairing.

2.5 Covert Communication Method via Spreadsheets with an Authentication Capability

We propose a new covert communication method which applies Shamir's (k, n)-threshold secret sharing scheme with $n = k + 1$ to a given secret item to yield k+1 shares, and the generated $k + 1$ shares are embedded into the number items in a spreadsheet as if they are part of the spreadsheet content. The purpose of transforming the secret data into secret shares by the (k, k+1)-threshold secret sharing scheme is *not* to enforce robustness, but to yield a blind self-authentication capability for the embedded secret. Conventionally, the concept of (k, n)-threshold secret sharing is applied to provide destruction-tolerant capabilities. That is, any k shares collected from n ones may be processed to reveal the shared secret even though up to ($n - k$) shares are destroyed. But in the proposed method, the scheme of (k, $k + 1$)-threshold secret sharing is developed for the first time to provide instead a self-authentication capability by checking the *value-consistency* of $k + 1$ results coming from all $k + 1$ combinations to determine whether the extracted secret is intact or not. That is, only when the results computed from any k shares collected from $k + 1$ shares are *all identical* in value can the extracted secret be decided to be intact. Moreover, to conceal the presence of hidden data, secret shares are spread throughout the cover spreadsheet in a *sparsely* fashion. And a spreadsheet containing numeral items with a *high scatter level* is more suitable to be used as a cover spreadsheet for better concealment.

2.6 Reversible Data Hiding Method

An iterative reversible data hiding method which is composed of two phases and yields high data embedding rates is proposed. The method utilizes the spatial similarity of neighboring pixels to create a difference histogram. In the first phase of histogram shifting, the peak point of the difference histogram is used to accommodate some message data, and in the second phase the value of the peak point, combined with the remaining message data, is embedded into the difference histogram again using another histogram-shifting scheme.

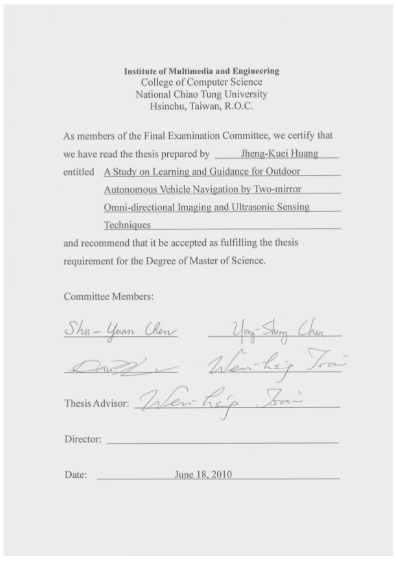
Chapter 3

A Secret-Sharing-Based Method for Authentication of Grayscale Document Images via the Use of the PNG Image with a Data Repairing Capability

3.1 Introduction

Document images, which include texts, tables, line arts, etc. as main contents, are often digitized into grayscale images with two major gray values. One of the two values represents the background (including mainly blank spaces) and the other represents the foreground (including mainly texts). It is noted that such images, though gray-valued in nature, look like binary. For example, the two major gray values in the document image shown in Fig. 3.1 are 174 and 236, respectively. It seems that such binary-like grayscale document images may be thresholded into binary ones for later processing, but such a thresholding operation often destructs the smoothness of the boundaries of text characters, resulting in visually unpleasant stroke appearances with zigzag contours. Therefore, in practical applications text documents are often digitized and kept as grayscale images for later visual inspection.

In general, the image authentication problem is difficult for a binary document image because of its simple binary nature. This nature leads to creation of perceptible changes after authentication signals are embedded in the image pixels. Such changes will arouse possible suspicions from attackers. A good solution to such binary image authentication thus should take into account not only the security issue of preventing image tampering, but also the necessity of keeping the visual quality of the resulting image. In this study, we propose an authentication method which deals with binary-like grayscale document images instead of pure binary ones, and solves simultaneously the problems of image tampering detection and visual quality keeping.



Institute of Multimedia and Engineering
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

As members of the Final Examination Committee, we certify that we have read the thesis prepared by Jheng-Kuei Huang entitled A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques and recommend that it be accepted as fulfilling the thesis requirement for the Degree of Master of Science.

Committee Members:

Shu-Yuan Chen   Yong-Sheng Chen

[illegible]   [illegible]

Thesis Advisor: [illegible]

Director: ____________

Date: June 18, 2010

Fig. 3.1 A binary-like grayscale document image with two major gray values.

In this study, a method for authentication of document images with an additional self-repair capability for fixing tampered image data is proposed. The input cover image is assumed to be a binary-like grayscale image with two major gray values like the one shown in Fig. 3.1. After the proposed method is applied, the cover image is transformed into a stego-image in the PNG format with an additional alpha channel for transmission on networks or archiving in databases. The stego-image, when received or retrieved, may be verified by the proposed method for its authenticity. Integrity modifications of the stego-image can be detected by the method at the block level and repaired at the pixel level. In case that the alpha channel is totally removed from the stego-image, the entire resulting image is regarded as inauthentic, meaning that the fidelity check of the image fails. The proposed method is based on the so-called (k, n)-threshold secret sharing scheme proposed by Shamir [56] in which a secret message is transformed into n shares for keeping by n participants; and when k of the n shares, not necessarily all of them, are collected, the secret message can be recovered losslessly. Such a secret sharing scheme is useful for reducing the risk of incidental partial data loss.

Conventionally, the concepts of “secret sharing” and “data hiding for image authentication” are two irrelevant issues in the domain of information security. But in the proposed method, we combine them together to develop a new image authentication technique. The secret sharing scheme is used in the developed technique not only to carry authentication signals and image content data, but also to help repair tampered data through the use of shares.

An issue in self-repairing of tampered data at attacked image parts is that after the original data of the cover image are embedded into the image itself for use in later data repairing, the cover image is destructed in the first place and the original data are no longer available for data repairing, resulting in a contradiction. A solution to this problem is to embed the original image data somewhere else without altering the cover image itself. The way proposed in this study to implement this solution is to utilize the extra alpha channel in a PNG image to embed the original image data. However, the alpha channel of the PNG image is used originally for creating a desired degree of transparency for the image. Embedding of data into the alpha channel will so create random transparency in the resulting PNG image and produce an undesirable opaque effect. One way out, as proposed in this study, is to map the resulting alpha channel values into a small range near their extreme value of 255, yielding a nearly imperceptible transparency effect on the alpha channel plane.

Another problem encountered in self-repairing of the original image data is that the data to be embedded in the carrier are often large-sized. For our case here with the alpha channel as the carrier, this is not a problem because we may just transform the binary-like cover image into a binary version and embed the small-sized result into the carrier. Furthermore, through a careful design of authentication signals, a proper choice of the basic authentication unit (i.e., the unit of 2×3 image block), and a good adjustment of the parameters in Shamir's scheme, we can reduce the data volume of the generated shares effectively so that more shares can be embedded into the alpha channel plane. It is noted that by the proposed method, the larger the number of shares is, the higher the resulting data repair capability becomes, as can be seen in the subsequent sections. Finally, we distribute the multiple shares randomly into the alpha channel to allow the share data to have large chances to survive attacks and so to promote the data repair capability. To the best of our knowledge, this is the first secret-sharing-based authentication method for binary-like grayscale document images. It is also the first authentication method for such document images through the use of the PNG image. Note that this method is not a secret-sharing technique, but a document image authentication method.

3.2 Review of Shamir's Method for Secret Sharing

In the (k, n)-threshold secret sharing method proposed by Shamir [56], a secret d in the form of an integer is transformed into shares which then are distributed to n participants to keep; and as long as k of the n shares are collected, the original secret can be recovered accordingly, where $k \leq n$. The detail of the method is reviewed in the following.

Algorithm 1: (k, n)-threshold secret sharing.

Input: a secret d in the form of an integer, the number n of participants, and a threshold $k \leq n$.

Output: n shares in the form of integers for the n participants to keep.

Steps.

Step 1. Choose randomly a prime number p which is larger than d.

Step 2. Select $k - 1$ integer values $c_1, c_2, \ldots, c_{k-1}$ within the range of 0 through $p - 1$.

Step 3. Select n distinct real values $x_1, x_2, \ldots, x_n$.

Step 4. Use the following $(k - 1)$-degree polynomial to compute n function values $F(x_i)$, called *partial shares* for $i = 1, 2, \ldots, n$:

$$F(x_i) = (d + c_1x_i + c_2x_i^2 + \ldots + c_{k-1}x_i^{k-1})_{\mathrm{mod}\,p}. \qquad (1)$$

Step 5. Deliver the 2-tuple $(x_i, F(x_i))$ as a *share* to the ith participant where $i = 1, 2, \ldots, n$.

Since there are k coefficients, namely, d and c_1 through c_{k-1} in (1) above, it is necessary to collect at least k shares from the n participants to form k equations of the form of Eqs. (1) to solve these k coefficients in order to recover the secret d. This explains the term, *threshold*, for k and the name, (k, n)*-threshold*, for the Shamir method [56]. Below is a description of the just-mentioned equation-solving process for secret recovery.

Algorithm 2: secret recovery.

Input: k shares collected from the n participants and the prime number p with both k and p being those used in Algorithm 1.

Output: the secret d hidden in the shares and the coefficients c_i used in Eqs. (1) in Algorithm 1, where $i = 1, 2, \ldots, k - 1$.

Steps.

Step 1. Use the k shares $(x_1, F(x_1))$, $(x_2, F(x_2))$, …, $(x_k, F(x_k))$ to set up the following equations:

$$F(x_j) = (d + c_1x_j + c_2x_j^2 + \ldots + c_{k-1}x_j^{k-1})_{\bmod p}, \tag{2}$$

where $j = 1, 2, ..., k$.

Step 2. Solve the k equations above by Lagrange's interpolation to obtain d as follows [58]:

$$d = (-1)^{k-1}[F(x_1)\frac{x_2x_3\ldots x_k}{(x_1-x_2)(x_1-x_3)\ldots(x_1-x_k)} + F(x_2)\frac{x_1x_3\ldots x_k}{(x_2-x_1)(x_2-x_3)\ldots(x_2-x_k)}$$
$$+\ldots+F(x_k)\frac{x_1x_2\ldots x_{k-1}}{(x_k-x_1)(x_k-x_2)\ldots(x_k-x_{k-1})}]_{\bmod p}.$$

Step 3. Compute c_1 through c_{k-1} by expanding the following equality and comparing the result with (2) in Step 1 while regarding the variable x in the equality below to be x_j in (2):

$$F(x) = [F(x_1)\frac{(x-x_2)(x-x_3)\ldots(x-x_k)}{(x_1-x_2)(x_1-x_3)\ldots(x_1-x_k)} + F(x_2)\frac{(x-x_1)(x-x_3)\ldots(x-x_k)}{(x_2-x_1)(x_2-x_3)\ldots(x_2-x_k)}$$
$$+\ldots+F(x_k)\frac{(x-x_1)(x-x_2)\ldots(x-x_{k-1})}{(x_k-x_1)(x_k-x_2)\ldots(x_k-x_{k-1})}]_{\bmod p}.$$

Step 3 in the above algorithm is included additionally for the purpose of computing the values of the parameters c_i in the proposed method. In other applications, if only the secret value d need be recovered, this step may be eliminated.

3.3 Merits of Proposed Method

In addition to being capable of data repairing and being *blind* in nature (requiring no overhead other than the stego-image), the proposed method has several other merits, which are described in the following.

(1) *Providing pixel-level repairs of tampered image parts* — As long as two untampered partial shares can be collected, a tampered block can be repaired *at the pixel level* by the proposed method. This yields a better repair effect for texts in images because text characters or letters are smaller in size with many curved strokes and need finer pixel-level repairs when tampered with.

(2) *Having higher possibility to survive image content attacks* — By combining skillfully the Shamir scheme, authentication signal generation, and random embedding of multiple shares, the proposed method can survive malicious attacks

of common content modifications, such as superimposition, painting, etc., as will be demonstrated by experimental results described subsequently.

(3) *Making use of a new type of image channel for data hiding* — Different from common types of images, a PNG image has the extra alpha channel plane which normally is used to produce transparency to the image. It is utilized differently by the proposed method for the first time as a carrier *with a large space* for hiding share data. As a comparison, many other methods use LSBs as carriers of hidden data.

(4) *Causing no distortion to the input image* — Conventional image authentication methods which usually embed authentication signals into the cover image itself will unavoidably cause destruction to the image content to a certain extent. Different from such methods, the proposed method utilizes the pixels' values of the alpha channel for the purpose of image authentication and data repairing, leaving the original image (i.e., the grayscale channel) *untouched* and so causing *no distortion* to it. The alpha channel plane may be removed after the authentication process to get the original image. Fig. 3.2 shows the framework of the proposed method in this aspect; and Fig. 3.3, shown for comparison, illustrates a conventional image authentication method.

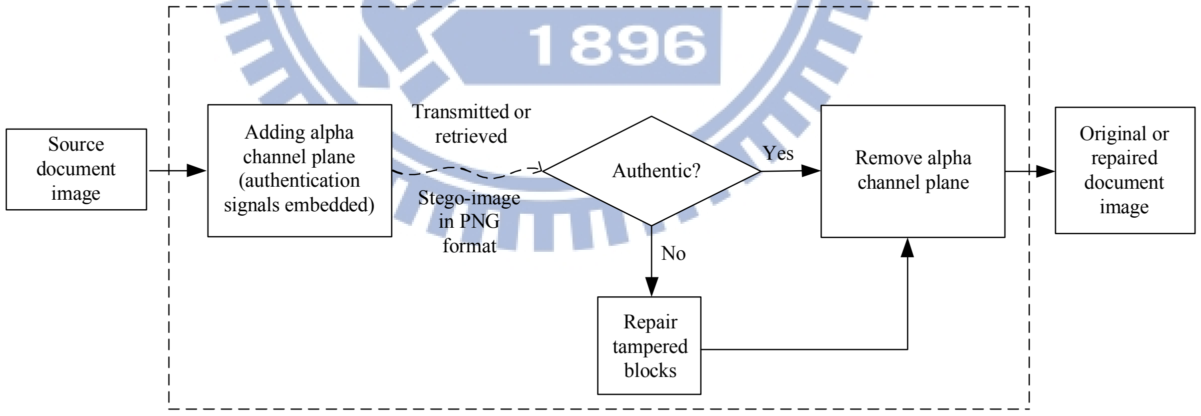

Source document image
Adding alpha channel plane (authentication signals embedded)
Transmitted or retrieved
Stego-image in PNG format
Authentic?
Yes
No
Remove alpha channel plane
Repair tampered blocks
Original or repaired document image

Fig. 3.2 Framework of proposed document image authentication method.

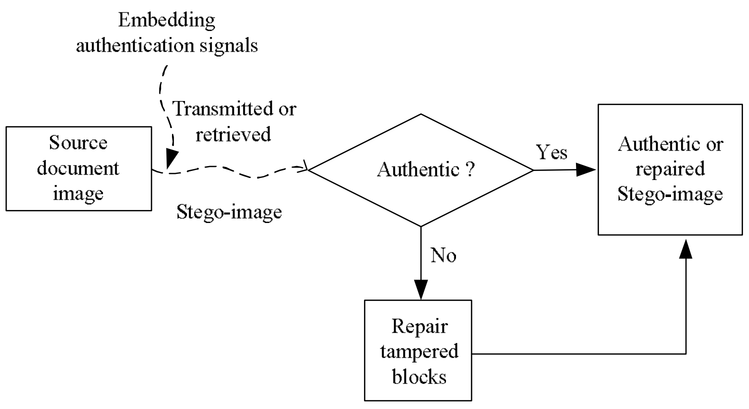


Fig. 3.3 Framework of a conventional image authentication method.

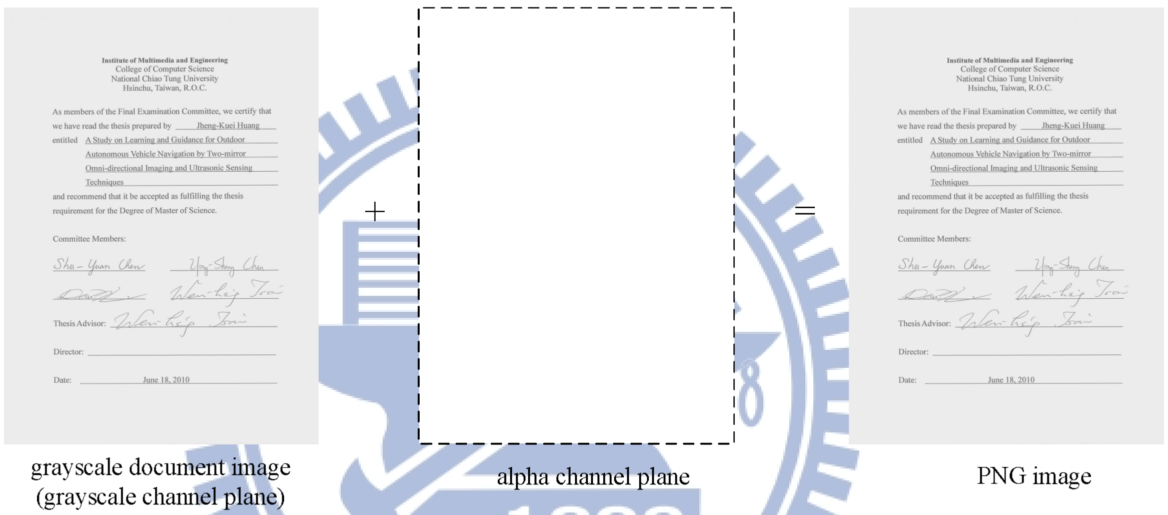
(5) Enhancing data security by secret sharing — Instead of hiding data directly into document image pixels, the proposed method embeds data in the form of shares into the alpha channel of the PNG image. The effect of this may be regarded as double-fold security protection, one fold contributed by the shares as a form of disguise of the original image data and the authentication signals, and the other fold contributed by the use of the alpha channel plane which is created to be nearly transparent, as mentioned previously.

3.4 Proposed Method for generation of a stego-image

In the proposed method, a PNG image is created from a binary-type grayscale document image I with an alpha channel plane. The original image I may be thought as a *grayscale channel plane* of the PNG image. An illustration of this process of PNG image creation is shown in Fig. 3.4. Next, I is binarized by moment-preserving thresholding [59], yielding a binary version of I, which we denote as I_b. Data for authentication and repairing then are computed from I_b and taken as input to Shamir's secret sharing scheme to generate n secret shares. The share values are mapped subsequently into a small range of alpha channel values near the maximum transparency value to create an imperceptibility effect. Finally, the mapped secret shares are randomly embedded into the alpha channel for the purpose of promoting the security protection and data repair capabilities. Two block diagrams describing the proposed method are shown in Figs. 3.5 and 3.6.

Since the alpha channel plane is used for carrying data for authentication and repairing, no destruction will occur to the input image in the process of authentication.

In contrast, conventional image authentication methods often sacrifice part of image contents, such as LSBs or flippable pixels, to accommodate data used for authentication. In addition, once a stego-image generated from a conventional authentication method like an LSB-based one is unintentionally compressed by a lossy compression method, the stego-image might cause false positive alarms in the authentication process. In contrast, the proposed method yields a stego-image in the PNG format which in normal cases will not be compressed further, reducing the possibility of erroneous authentication caused by imposing undesired compression operations on the stego-image.


+
=
grayscale document image
(grayscale channel plane)
alpha channel plane
PNG image

Institute of Multimedia and Engineering
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

As members of the Final Examination Committee, we certify that we have read the thesis prepared by Jheng-Kuei Huang entitled A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques and recommend that it be accepted as fulfilling the thesis requirement for the Degree of Master of Science.

Committee Members:

Shu-Yuan Chen Yong-Sheng Chen
[illegible] [illegible]

Thesis Advisor: [illegible]

Director: ______________________

Date: June 18, 2010

Institute of Multimedia and Engineering
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

As members of the Final Examination Committee, we certify that we have read the thesis prepared by Jheng-Kuei Huang entitled A Study on Learning and Guidance for Outdoor Autonomous Vehicle Navigation by Two-mirror Omni-directional Imaging and Ultrasonic Sensing Techniques and recommend that it be accepted as fulfilling the thesis requirement for the Degree of Master of Science.

Committee Members:

Shu-Yuan Chen Yong-Sheng Chen
[illegible] [illegible]

Thesis Advisor: [illegible]

Director: ______________________

Date: June 18, 2010

Fig. 3.4 Illustration of creating a PNG image from a grayscale document image and an alpha channel.

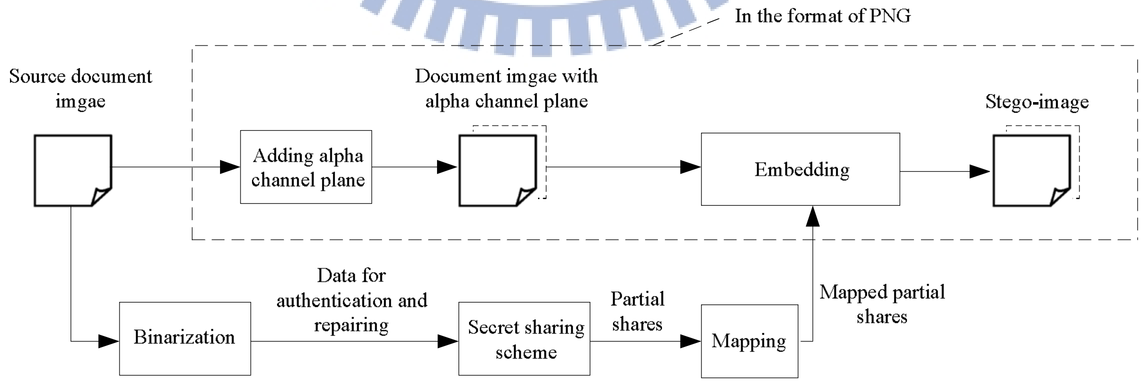
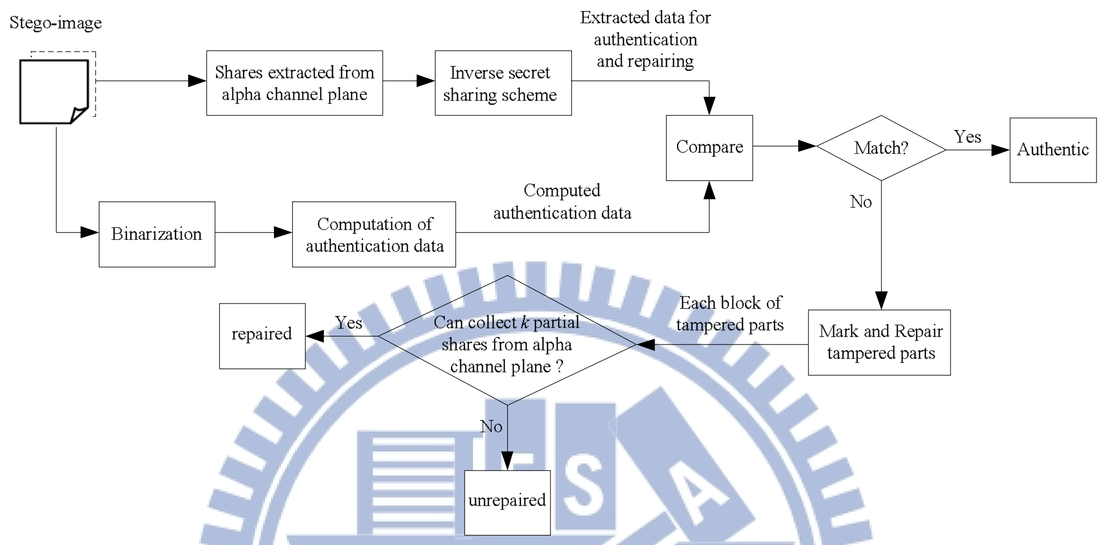


Fig. 3.5 Generation process of a stego-image in PNG format from a grayscale document image.

A. Algorithm for generation of a stego-image

A detailed algorithm for describing the generation of a stego-image in the PNG format of the proposed method is presented in the following.



Stego-image
Shares extracted from alpha channel plane
Inverse secret sharing scheme
Extracted data for authentication and repairing
Compare
Match?
Yes
Authentic
No
Binarization
Computation of authentication data
Computed authentication data
Mark and Repair tampered parts
Each block of tampered parts
Can collect *k* partial shares from alpha channel plane ?
Yes
repaired
No
unrepaired

Fig. 3.6 Authentication process including verification and self-repairing of a stego-image in PNG format.

Algorithm 3: generation of a stego-image in the PNG format from a given grayscale image.

Input: a grayscale document image I with two major gray values, and a secret key K.

Output: a stego-image I' in the PNG format with relevant data embedded, including the authentication signals and the data used for repairing.

Steps.

Stage I — generation of authentication signals.

Step 1. (*Input image binarization*) Apply moment-preserving thresholding [59] to I to obtain two representative gray values g_1 and g_2; compute $T = (g_1 + g_2)/2$; and use T as a threshold to binarize I, yielding a binary version I_b with “0” representing g_1 and “1” representing g_2.

Step 2. (*Transforming the cover image into the PNG format*) Transform I into a PNG image with an alpha channel plane I_α by creating a new image layer with 100% opacity and no color as I_α and combining it with I using an image processing software package.

Step 3. (*Beginning of looping*) Take in a raster-scan order an unprocessed 2×3 block B_b of I_b with pixels $p_1, p_2, \ldots, p_6$.

Step 4. (*Creation of authentication signals*) Generate a 2-bit authentication signal $s = a_1a_2$ with $a_1 = p_1 \oplus p_2 \oplus p_3$ and $a_2 = p_4 \oplus p_5 \oplus p_6$ where $\oplus$ denotes the exclusive-OR operation.

Stage II — creation and embedding of shares.

Step 5. (*Creation of data for secret sharing*) Concatenate the eight bits of a_1, a_2, and p_1 through p_6 to form an 8-bit string; divide the string into two 4-bit segments; and transform the segments into two decimal numbers m_1 and m_2, respectively.

Step 6. (*Partial share generation*) Set p, c_i, and x_i in Eqs. (1) of Algorithm 1 to be the following values: (a) $p = 17$ (the smallest prime number larger than 15); (b) $d = m_1$, $c_1 = m_2$; (c) $x_1 = 1, x_2 = 2, \ldots, x_6 = 6$; and perform Algorithm 1 as a (2, 6)-threshold secret sharing scheme to generate six partial shares q_1 through q_6 using the following equations:

$$q_i = F(x_i) = (d + c_1x_i)_{\text{mod } p}, \tag{3}$$

where $i = 1, 2, \ldots, 6$.

Step 7. (*Mapping of the partial shares*) Add 238 to each of q_1 through q_6, resulting in the new values of q_1' through q_6', respectively, which fall in the nearly total transparency range of 238 through 254 in the alpha channel plane I_α.

Step 8. (*Embedding two partial shares in the current block*) Take the block B_α in I_α corresponding to B_b in I_b, select the first two pixels in B_α in the raster-scan order, and replace their values by q_1' and q_2', respectively.

Step 9. (*Embedding remaining partial shares at random pixels*) Use the key K to select randomly four pixels in I_α but outside B_α, which are *unselected yet* in this step and *not* the first two pixels of any block; and in the raster scan order replace the four pixels' values by the remaining four partial shares q_3' through q_6' generated above, respectively.

Step 10. (*End of looping*) If there exists any *unprocessed* block in I_b, then go to Step 3; otherwise, take the final I in the PNG format as the desired stego-image I'.

The possible values of q_1 through q_6 yielded by Eqs. (3) above are between 0 and

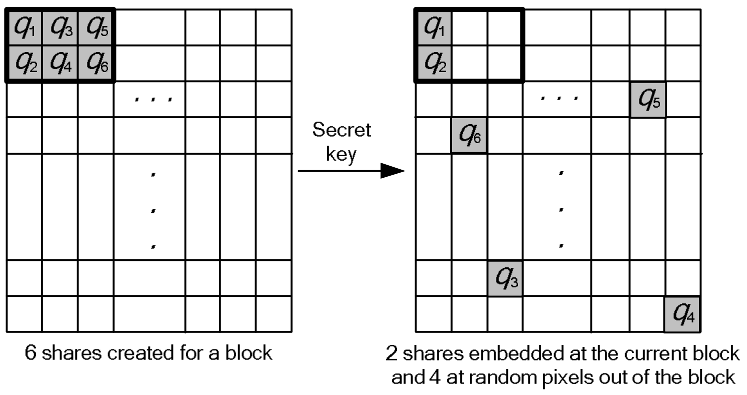
16 because the prime number p used there is 17. After performing Step 7 of the above algorithm, they become q_1' through q_6', respectively, which all fall into a small interval of integers ranging from 238 to 254 with a width of 17 (the value of the prime number). Subsequent embedding of q_1' through q_6' in such a narrow interval into the alpha channel plane means that *very similar* values will appear everywhere in the plane, resulting in a *nearly uniform transparency effect*, which will not arouse notice from an attacker.

The reason why we choose the prime number to be 17 in the above algorithm is explained here. If it was chosen instead to be larger than 17, then the above-mentioned interval will be enlarged and the values of q_1' through q_6' will become possibly smaller than 238, creating an undesired *less transparent* but *visually whiter* stego-image. On the other hand, the 8 bits mentioned in Steps 5 and 6 above are transformed into two decimal numbers m_1 and m_2 with their maximum values being 15 (see Step 5 above), which are constrained to lie in the range of 0 through $p-1$ (see Step 2 in Algorithm 1). Therefore, p should *not* be chosen to be smaller than 16. In short, $p = 17$ is an *optimal* choice.

As to the choice of the block size, the use of a larger block size, like 2×4 or 3×3, will reduce the *precision* of the resulting integrity authentication (i.e., the stego-image will be verified *in a spatially-coarser manner*). On the other hand, it seems that a smaller block size such as 2×2 instead 2×3 may be tried to increase the authentication precision. However, a block in the alpha channel with a size of 2×2 can be used to embed *only four* partial shares instead of six (see Steps 6 through 9 of Algorithm 3). This decreases share multiplicity and so reduces the data repair capability of the method. In short, there is a tradeoff between the authentication precision and the data repair capability, and our choice of the block size 2×3 is a *balance* in this aspect. Finally, we use Fig. 3.7 to illustrate Steps 8 and 9 of Algorithm 3 where a core idea of the proposed method is presented: two shares of the generated six are embedded at the current block and the other four embedded at four randomly-selected pixels outside the block, with each selected pixel not being the first two ones in any block.

3.5 Proposed Image Authentication and Repairing Process

A detailed algorithm describing the proposed stego-image authentication process, including both verification and self-repairing of the original image content, is presented in the following.



q_1 q_3 q_5
q_2 q_4 q_6
Secret key
q_1 q_2 q_5 q_6 q_3 q_4
6 shares created for a block
2 shares embedded at the current block and 4 at random pixels out of the block

Fig. 3.7 Illustration of embedding six shares created for a block — two shares embedded at the current block and the other four in four randomly-selected pixels outside the block, with each selected pixel not being the first two ones in any block.

Algorithm* 4: *authentication of a given stego-image in the PNG format.

***Input*:** a stego-image I'; the representative gray values g_1 and g_2, and the secret key K used in Algorithm 3.

***Output*:** an image I_r with tampered blocks marked, and their data repaired if possible.

Stage I — extraction of the embedded two representative gray values.

Step 1. (*Binarization of the stego-image*) Compute $T = (g_1 + g_2)/2$ and use it as a threshold to binarize I', yielding a binary version I_b' of I' with "0" representing g_1 and "1" representing g_2.

Stage II — verification of the stego-image.

Step 2. (*Beginning of looping*) Take in a raster-scan order an unprocessed block B_b' from I_b' with pixel values p_1 through p_6, and find the six pixels' values q_1' through q_6' of the corresponding block B_α' in the alpha channel plane I_α' of I'.

Step 3. (*Extraction of the hidden authentication signal*) perform the following steps to extract the hidden 2-bit authentication signal $s = a_1a_2$ from B_α'.

(1) Subtract 238 from each of q_1' and q_2' to obtain two partial shares q_1 and q_2 of B_b', respectively.

(2) With the shares $(1, q_1)$ and $(2, q_2)$ as input, perform Algorithm 2 to extract the two values d and c_1 (the secret and the first coefficient value, respectively) as output.

(3) Transform d and c_1 into two 4-bit binary values, concatenate them to form

an 8-bit string S, and take the first two bits a_1 and a_2 of S to compose the hidden authentication signal $s = a_1a_2$.

Step 4. (*Computation of the authentication signal from the current block content*) Compute a two-bit authentication signal $s' = a_1'a_2'$ from the values p_1 through p_6 of the six pixels of B_b' by $a_1' = p_1 \oplus p_2 \oplus p_3$ and $a_2' = p_4 \oplus p_5 \oplus p_6$.

Step 5. (*Matching of the hidden and computed authentication signals and marking of tampered blocks*) Match s and s' by checking if $a_1 = a_1'$ and $a_2 = a_2'$; and if any mismatch occurs, mark B_b', the corresponding block B' in I', and all the partial shares embedded in B_α' as *tampered*.

Step 6. (*End of looping*) If there exists any *unprocessed* block in I_b', then go to Step 2; otherwise, continue.

Stage III — self-repairing of the original image content.

Step 7. (*Extraction of the remaining partial shares*) For each block B_α' in I_α', perform the following steps to extract the remaining four partial shares q_3 through q_6 of the corresponding block B_b' in I_b' from blocks in I_α' other than B_α'.

(1) Use the key K to collect the four pixels in I_α' in the same order as they were randomly selected for B_b' in Step 9 of Algorithm 3, and take out the respective data q_3', q_4', q_5', and q_6' embedded in them.

(2) Subtract 238 from each of q_3' through q_6' to obtain q_3 through q_6, respectively.

Step 8. (*Repairing the tampered regions*) For each block B' in I' marked as *tampered* previously, perform the following steps to repair it if possible.

(1) From the six partial shares q_1 through q_6 of the block B_b' in I_b' corresponding to B' (two computed in Step 3(1) and four in Step 7(2) above), choose two of them, say q_k and q_l, which are *not* marked as tampered, if possible.

(2) With the shares (k, q_k) and (l, q_l) as input, perform Algorithm 2 to extract the values of d and c_1 (the secret and the first coefficient value) as output.

(3) Transform d and c_1 into two 4-bit binary values and concatenate them to form an 8-bit string S'.

(4) Take the last six bits b_1', b_2', …, b_6' from S' and check their binary values to repair the corresponding tampered pixel values y_1', y_2', …, y_6' of block

B' by the following way:

if $b_i' = 0$, set $y_i' = g_1$; otherwise, set $y_i' = g_2$

where $i = 1, 2, \ldots, 6$.

Take the final I' as the desired self-repaired image I_r.

3.6 Security Enhancement

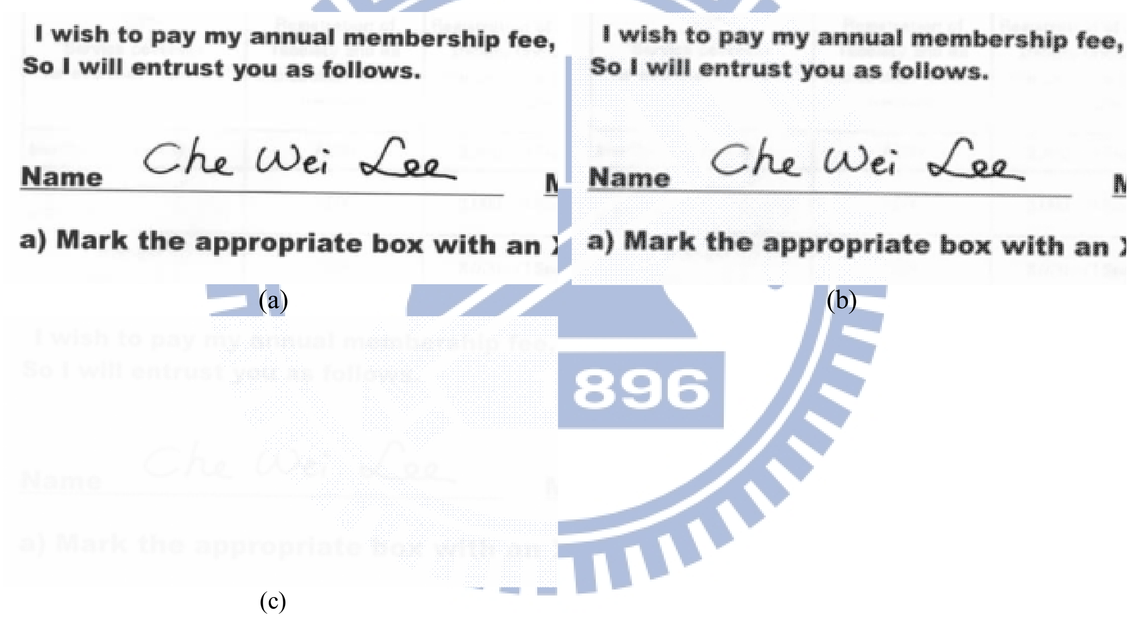
The secret key K, which is used to randomize the pixel positions for embedding the mapped partial shares q_3' through q_6' mentioned in Step 9 of Algorithm 3, provides a measure to protect the shares. More specifically, as described in Algorithm 3, each block in the alpha channel plane may be regarded to consist of two parts, *the first part* including the first two pixels and *the second* including the remaining four. The first part of each block is used for keeping the first two partial shares q_1' and q_2', and the second part for keeping the last four partial shares q_3' through q_6' of other blocks located at random positions. Therefore, the probability of correctly guessing the locations of all the embedded partial shares in a stego-image is $1/[(m\times n) - (m\times n/6)\times 2]!$ where $m\times n$ is the size of the cover image; $m\times n/6$ is the total number of blocks, each with six pixels; and $(m\times n) - (m\times n/6)\times 2$ is the total number of pixels in the blocks other than those in the first parts of all the blocks. This probability obviously is very small for common image sizes, meaning that a correct guess of the embedded partial shares is nearly impossible.

To enhance further the security of the data embedded in the stego-image, one additional measure is adopted in the proposed method (but not included in the previously-proposed algorithms for clarity of algorithm descriptions). It is *randomization* of the constant values of x_1 through x_6 used in Step 6 of Algorithm 3 and Step 3(2) in Algorithm 4. Specifically, in Step 3(2) in Algorithm 4 we can see that the input shares into Algorithm 2, $(1, q_1)$ and $(2, q_2)$, can be forged easily, leading to the possibility of creating fake authentication signals. To remedy this weakness, with the help of another secret key we may choose these values of x_1 through x_6 for each block to be *random* within the allowed integer range of $0 \leq x_i < p$ (= 17) [56]. Then, the probability of guessing correctly all these values for all the $m\times n/6$ blocks in a stego-image can be figured out to be $1/[(17\times16\times15\times14\times13\times12)]^{m\times n/6} \approx 1/(8.911\times10^6)^{m\times n/6}$ which is also very small for common image sizes $m\times n$.

3.7 Experimental Results

A. Experimental Results Using a Document Image of a Signed Paper

The first results we show here come from our experiments using a document image of a signed paper shown in Fig. 3.8(a). The result of applying Algorithm 3 to embed share data into Fig. 3.8(a) is shown in Fig. 3.8(b). As can be seen, the stego-image shown in the latter is visually almost identical to the cover image shown in the former although the alpha channel content of the latter image includes the embedded data. As a comparison, Fig. 3.8(c) shows a result created similarly but without conducting Step 7 of Algorithm 3 which maps the original partial share values into the small interval of alpha channel values ranging from 238 through 254. An obvious opaque effect (nearly white) appears in Fig. 3.8(c).


I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee
a) Mark the appropriate box with an

(a)

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee
a) Mark the appropriate box with an

(b)

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee
a) Mark the appropriate box with an

(c)

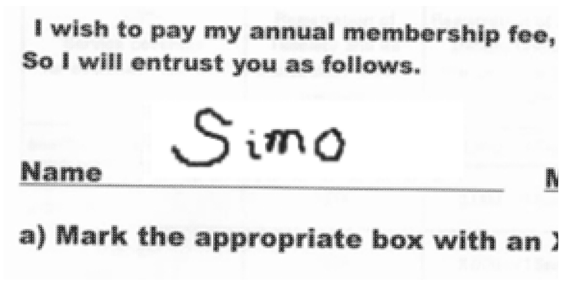
Fig. 3.8 Experimental result of a document image of a signed paper. (a) Original cover image. (b) A stego-image with embedded data. (c) Another stego-image created without conducting partial share value mapping.

We have also conducted image-modification attacks to the stego-images using two common image editing operations, namely, *superimposing* and *painting*. Tampered images yielded by the superimposing operation are presented in Figs. 3.8 through 3.10. It was observed from these experimental results that the superimposing operation, like that provided by the image editing software Adobe Photoshop or Corel PhotoImpact, destroys the content of the alpha channel values by replacing all the original alpha channel values at the attacked part with the new values of 255. Since the largest alpha channel values created by the proposed method is 254 (see Step 7 of Algorithm 3), all pixels with the unique values of 255 in the alpha channel plane may be easily detected as tampered by a modified version of Step 3 of Algorithm 4, which we describe as follows:

Step 3′ (*Check of superimposing attacks and extraction of the hidden authentication signal*) Check if both q_1' and q_2' are 255; if so, *then* (1) regard the corresponding block B' in I' as attacked by superimposing; (2) mark B', B_b', and all the partial shares embedded in B_α' as tampered; and (3) go to Step 6; *otherwise*, perform the original operations of Step 3 of Algorithm 4.

Fig. 3.9(a) shows the result of superimposing a white rectangular shape with a fake signature "Simo" on the genuine signature "C. W. Lee" in the stego-image of Fig. 3.8(b). Fig. 3.9(b) shows the authentication result yielded by Algorithm 4, with the gray blocks indicating the detected tampered image parts. As can be seen, the superimposing rectangular part on the signature C. W. Lee has been detected completely. For each of the detected tampered blocks, if at least two untampered shares of it can be collected, its original content can be repaired, yielding the result shown in Fig. 3.9(c); otherwise, the tampered block is left *unrepaired* as shown by the red dots in Fig. 3.9(d). Additionally, we show the data repair result obtained with a wrong key in Fig. 3.9(e). As can be seen, the repair work cannot be accomplished correctly; the result is just noise.

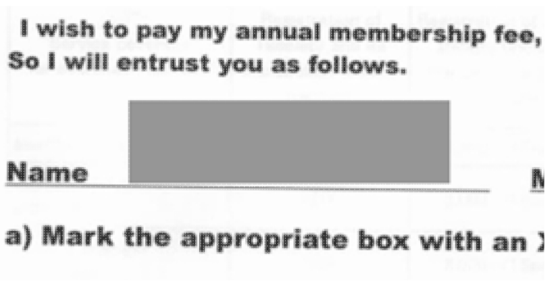
In Fig. 3.10(a), a text line under the signature in the signed paper disappeared after a white rectangular band was superimposed on it. The results of image authentication and repairing are shown in Figs. 3.10(b) and 3.10(c), respectively. Though some blocks are not repaired as indicated by the red dots in Fig. 3.10(d), the repair result of the text line is visually well recognizable. To test further the performance of the proposed method, we enlarged tampered areas during attacks and a result is shown in Fig. 3.11. It can be seen from the figure that the data repair result becomes worse when the tampered area grows. This is reasonable because when the tampered area becomes larger, fewer partial shares for data repairing will survive.


I wish to pay my annual membership fee,
So I will entrust you as follows.

Name Simo

a) Mark the appropriate box with an

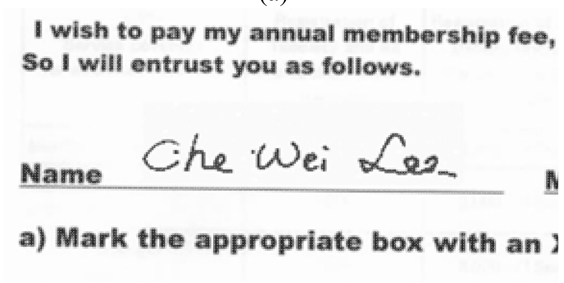
(a)


I wish to pay my annual membership fee,
So I will entrust you as follows.

Name

a) Mark the appropriate box with an

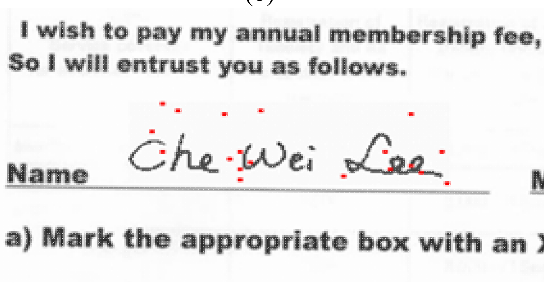
(b)


I wish to pay my annual membership fee,
So I will entrust you as follows.

Name Che Wei Lee

a) Mark the appropriate box with an

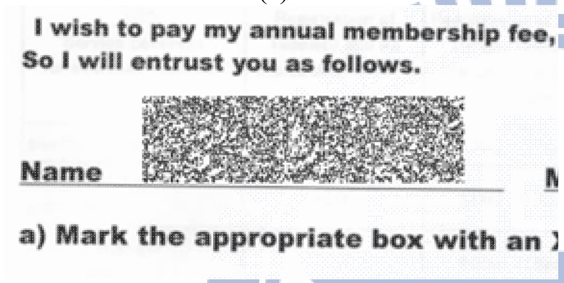
(c)


I wish to pay my annual membership fee,
So I will entrust you as follows.

Name Che Wei Lee

a) Mark the appropriate box with an

(d)


I wish to pay my annual membership fee,
So I will entrust you as follows.

Name

a) Mark the appropriate box with an

(e)

Fig. 3.9 Authentication result of a PNG document image of a signed paper attacked by superimposing a white rectangular shape on the signature in Fig. 3.8(b). (a) Tampered image yielded by the superimposing operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result. (d) Data repair result with red dots indicating unrepaired tampered blocks. (e) Erroneous data repair result obtained with a wrong key.

Table 3.1 includes the statistics of the performance of the proposed method shown by the above experimental results in terms of the five parameters: *tampering ratio*, *detection ratio*, *repair ratio*, *false acceptance ratio*, and *false rejection ratio*, which are defined in the following:

(1) *tampering ratio* = (the number of tampered blocks)/(the total number of blocks);

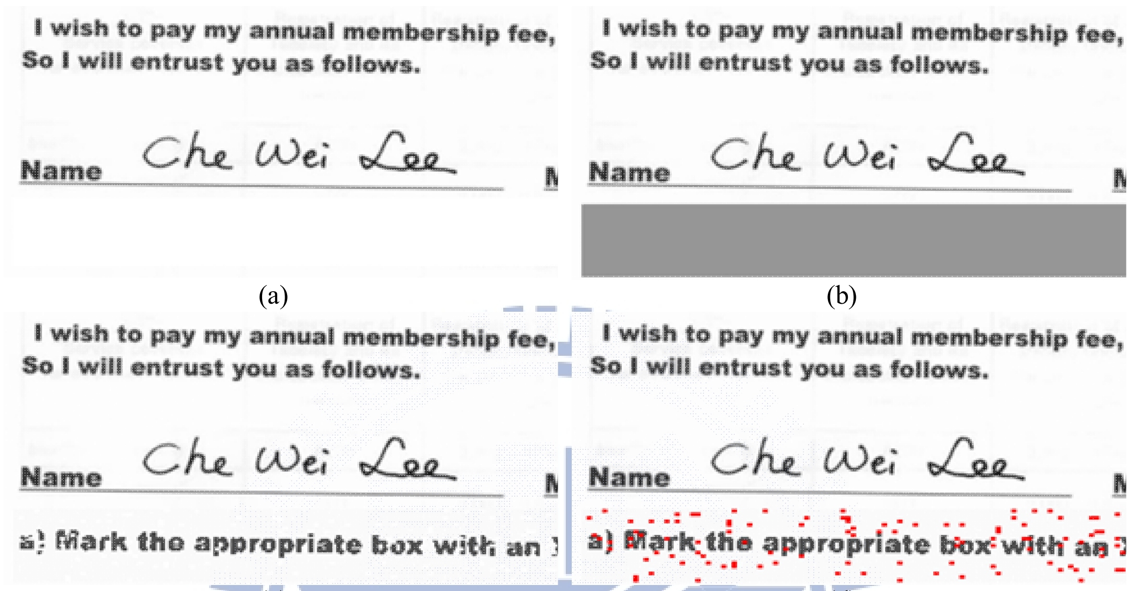
(2) *detection ratio* = (the number of detected blocks)/(the number of tampered blocks);

(3) *repair ratio* = (the number of repaired blocks)/(the number of detected blocks);

(4) *false acceptance ratio* = (the number of tampered blocks marked as

untampered)/(the total number of tampered blocks);

(5) *false rejection ratio* = (the number of untampered blocks marked as tampered)/(the total number of untampered blocks).


I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee

(a)

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee

(b)

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee
a) Mark the appropriate box with an

(c)

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name Che Wei Lee
a) Mark the appropriate box with an

(d)

Fig. 3.10 Authentication result of the document image of a signed paper attacked by superimposing a white rectangular shape on a piece of text in Fig. 3.8(b). (a) Tampered image yielded by the superimposing operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result. (d) Data repair result with red dots indicating unrepaired tampered blocks.

Note that the detection ratios are all 100% due to the ease in detection of the alpha channel values of 255 (using Step 3′ described above) at image parts attacked by superimposing, as mentioned previously. Likewise, the alpha channel value corresponding to an intact block will not be 255 and can be easily checked to be so, yielding a false rejection rate of 0%. On the contrary, the alpha channel value corresponding to a tampered block is 255 which is easy to check as well, yielding a false acceptance rate of 0%.

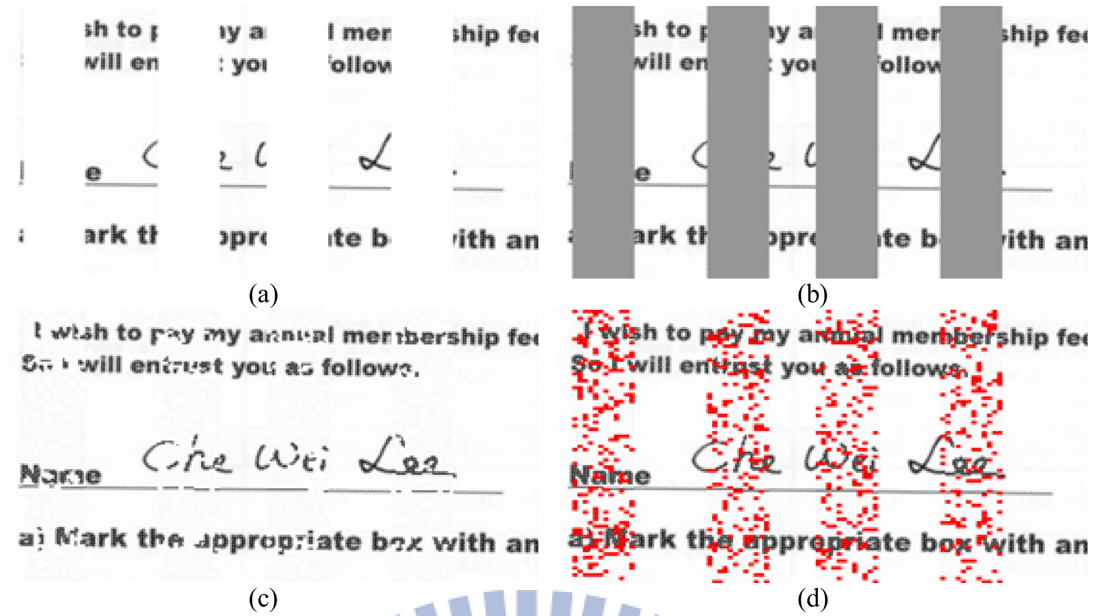


Fig. 3.11Authentication result of the document image of a signed paper attacked by superimposing white raster rectangular shapes on the content in Fig. 3.8(b). (a) Tampered image yielded by the superimposing operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result. (d) Data repair result with red dots indicating unrepaired tampered blocks.

Table 3.1 Statistics of experimental results of attacks using superimposing operations.

Experimental result (image size = 340×158)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of repaired blocks (repair ratio)	false acceptance ratio	false rejection ratio
Exp. 1 shown in Fig. 3.9	8927	1488 (16.67%)	1488 (100%)	1469 (98.72%)	0%	0%
Exp. 2 shown in Fig. 3.10	8927	2200 (24.64%)	2200 (100%)	2087 (94.86%)	0%	0%
Exp. 3 shown in Fig. 3.11	8927	3792 (42.48%)	3792 (100%)	3011 (79.40%)	0%	0%

The content of a stego-image may be modified as well by the common operation of painting provided by well-known image editing software. Again, painting using Adobe Photoshop will replace the alpha channel values by 255, just like the superimposing operation mentioned previously. However, it was found in this study that the painting operation provided by Corel PhotoImpact does *not* change the alpha channel values. Therefore, we conducted experiments of stego-image attacks using this type of painting. Some results are given in Figs. 3.12 through 3.14. In Fig. 3.12, the painting operation was used to smear background gray values on the original signature “C. W. Lee” and write a fake signature “Simo” on it as shown in Fig. 3.12(a). Fig. 3.12(b) shows the authentication result in which gray blocks were used again to indicate image parts where mismatching authentication signals were detected. Note that the smeared part (the signature C. W. Lee) and the added part (the signature “Simo”) have both been revealed by the authentication process. It is can be seen that some black parts exist within the gray region, meaning that these parts, though tampered, were not detected by the proposed method. This is due to the fact that there is actually a probability of 1/4 for an erroneous block authentication to occur because only two bits are created as the signal for block authentication (see Step 8 of Algorithm 3 or Step 5 of Algorithm 4). Though this probability seems large, yet due to the use of the secret sharing technique for generating multiple shares which are embedded randomly to survive attacks, correctly authenticated and repaired blocks still yield results with contents *visually recognizable* for image integrity judgment, as shown by Figs. 3.12(c) and 3.12(d).

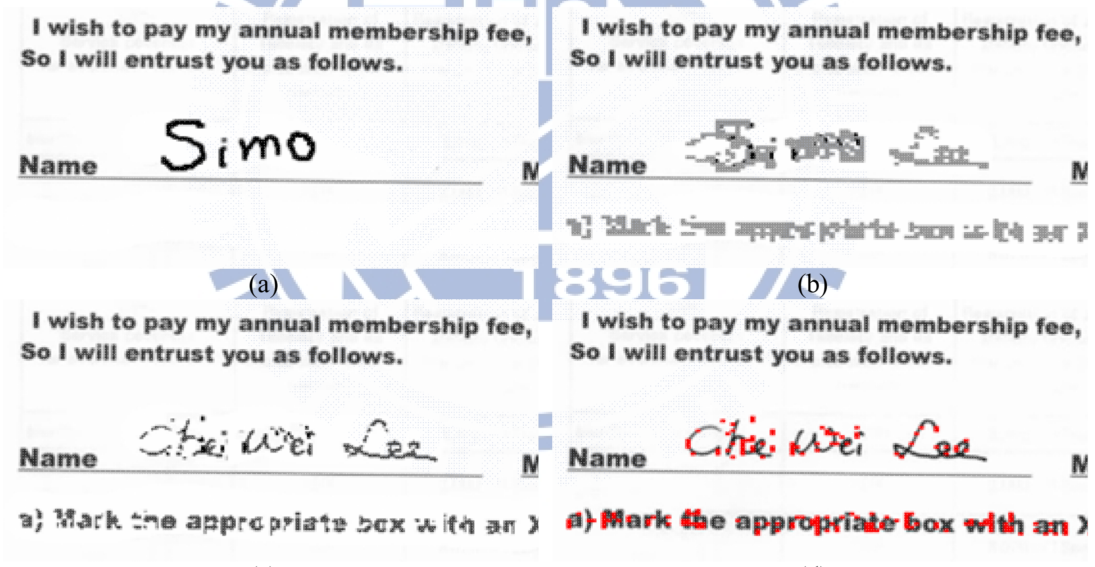
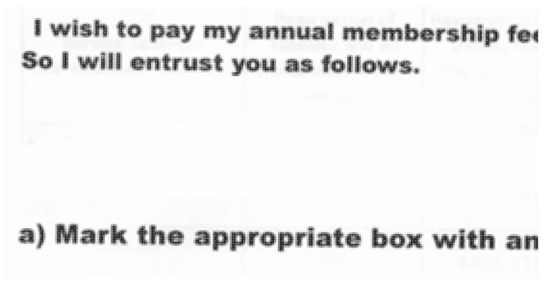

I wish to pay my annual membership fee,
So I will entrust you as follows.
Name
Simo
(a)
(b)
a) Mark the appropriate box with an
(c)
(d)

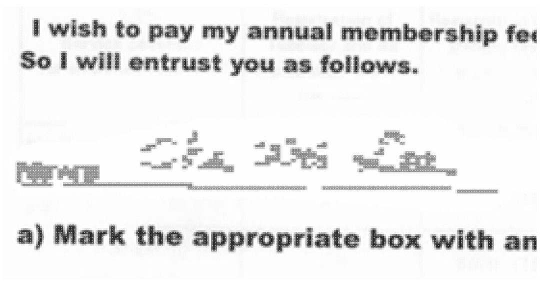
Fig. 3.12 Authentication result of document image of a signed paper attacked by painting white color on the original signature and texts, and replacing the signature by a fake one in Fig. 3.8(b). (a) Tampered image yielded by the painting operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result (d) Data repair result with red dots indicating unrepaired blocks.

In Fig. 3.13, the signature was removed by replacing it with the background gray value using painting. The results of image authentication and data repairing are shown in Figs. 3.13(b) through 3.13(d). Fig. 3.14 shows the results of removing the entire image content by painting. In both cases, the untouched content of the alpha channel values still yields repair results with their contents recognizable to a certain degree.


I wish to pay my annual membership fe
So I will entrust you as follows.

a) Mark the appropriate box with an

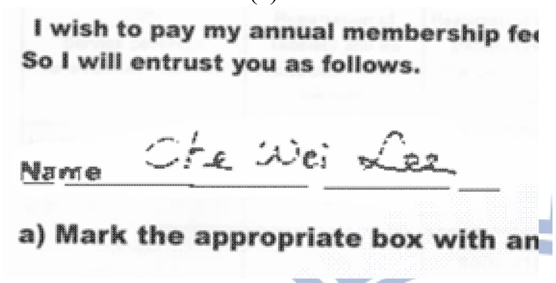
(a)


I wish to pay my annual membership fe
So I will entrust you as follows.

Name [illegible]

a) Mark the appropriate box with an

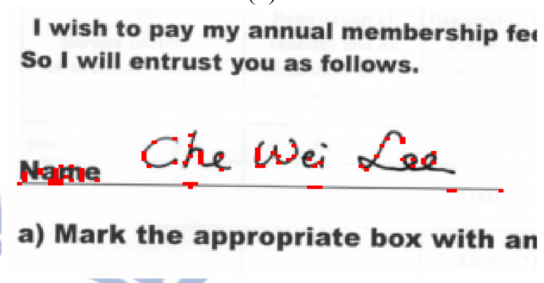
(b)


I wish to pay my annual membership fe
So I will entrust you as follows.

Name Che Wei Lee

a) Mark the appropriate box with an

(c)


I wish to pay my annual membership fe
So I will entrust you as follows.

Name Che Wei Lee

a) Mark the appropriate box with an

(d)

Fig. 3.13 Authentication result of document image of a signed paper attacked by painting white color on signature in Fig. 3.8(b). (a) Tampered image yielded by the painting operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result. (d) Data repair result with red dots indicating unrepaired tampered blocks.

It is noted here that, when a stego-image is tampered with by painting which does *not* changed the content of the alpha channel plane, the hidden authentication signals and data for repairing are not destructed. Therefore, the computed authentication signals from the alpha channel values are always true, and as long as the computed authentication signal is not identical to the extracted authentication signal for a block, the block will be marked as having been tampered with. This explains why the false rejection rate is 0%. However, as mentioned previously there is a probability of 1/4 for an erroneous block authentication to occur because only two bits are created as the signal for block authentication, and this leads to a false acceptance ratio of at most 25%. These reasonings are verified by the performance statistics of Figs. 3.12 through 3.14 listed in Table 3.2. The table also shows that the detection ratios are roughly around 75% as the attacked part becomes large (like the case of Fig. 3.14), meeting the probabilistic expectation of 1/4 block authentication misses just mentioned. Also, the false acceptance ratios are smaller than 25%, as expected.

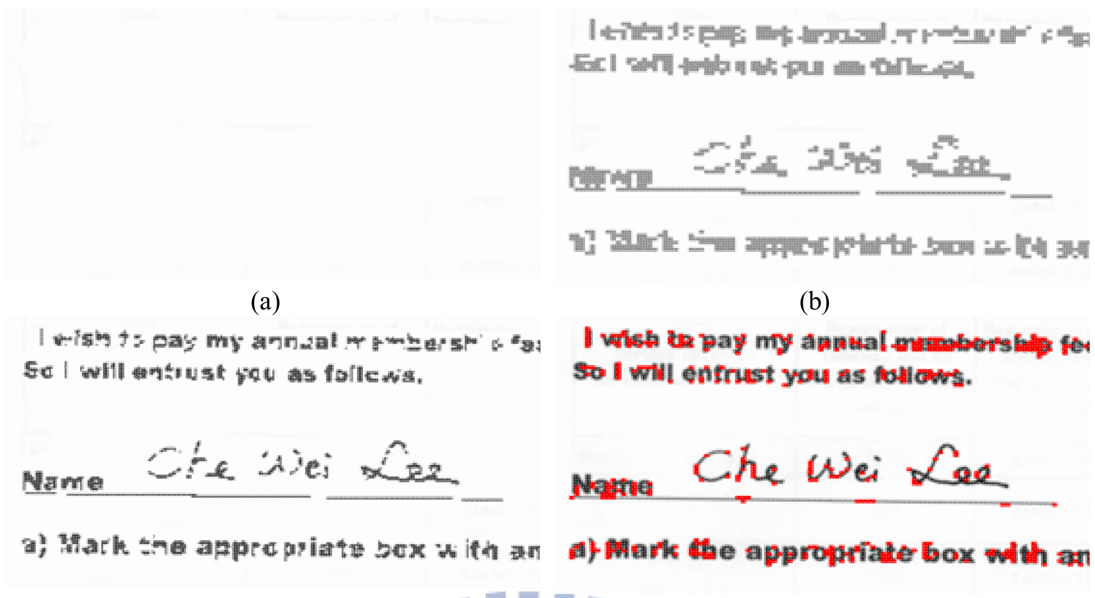


Fig. 3.14 Authentication result of the document image of a signed paper attacked by painting white color on entire content of Fig. 3.8(b). (a) Tampered image yielded by the painting operation. (b) Result with tampered blocks detected and marked as gray. (c) Data repair result. (d) Data repair result with red dots indicating unrepaired tampered blocks.

Table 3.2 Statistics of experimental results of attacks using painting operations.

Experimental result (image size = 340×158)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of repaired blocks (repair ratio)	false acceptance ratio	false rejection ratio
Exp. 4 shown in Fig. 3.12	8927	442 (5%)	396 (89.59%)	396 (100%)	10.41%	0%
Exp. 5 shown in Fig. 3.13	8927	781 (8.75%)	635 (81.31%)	635 (100%)	18.69%	0%
Exp. 6 shown in Fig.3.14	8927	1591 (17.82%)	1221 (76.74%)	1221 (100%)	23.26%	0%

B. Experimental Results Using a Document Image of a Check

Experimental results yielded by the use of a document image of a check are shown in Figs. 3.15(a) through 3.15(f) where Figs. 3.15(a) and 3.15(b) are respectively the cover document image and the stego-image generated by the proposed method. Both the amount-related text and numerals in the check image were modified as shown in Fig. 3.15(c). Fig. 3.15(d) shows that the tampering was successfully detected and marked as gray, and the result of data repairing is shown in

Fig. 3.15(e). Finally, Fig. 3.15(f) shows the result of data repairing in which two unrepaired tampered blocks are shown in red. Also, we show the statistics of this experiment in Table 3.3.

At last, we put some noise onto the stego-image of Fig. 3.15(b) as an intended attack to the image, yielding the noisy image of Fig. 3.16(c) which then was authenticated by Algorithm 3 to get the result of Fig. 3.16(d) with detected tampered blocks marked in gray. The data repair result is shown in Fig. 3.16(e). As can be seen, some noise was not detected. The reason, as mentioned previously, is again that there is a probability of 1/4 for an erroneous block authentication to occur. Fig. 3.16(f) shows the unrepaired pixels in red, and the statistics of this experiment is also included into Table 3.3, where we can see that the false acceptance rate is about 17.12% resulting from the above-mentioned undetected noise.

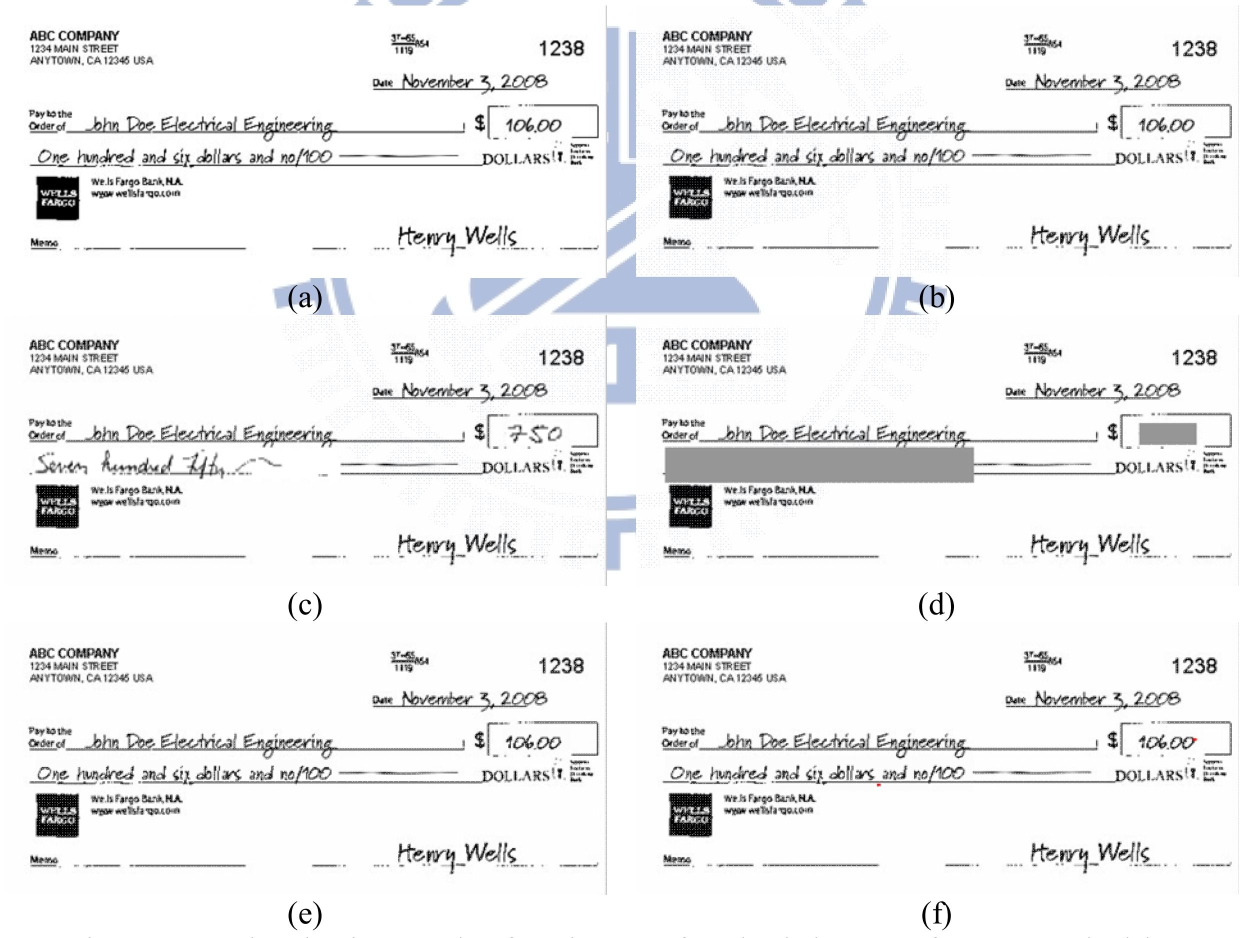


Fig. 3.15 Authentication result of an image of a check in PNG format attacked by superimposing counterfeit number “750” located at right side and text “Seven hundred fifty” located at left side. (a) Original cover image. (b) A stego-image with embedded data. (c) Tampered image by the superimposing operation. (d) Result with tampered blocks detected and marked as gray. (e) Result of data repair. (f) Data repair result with red dots indicating unrepaired tampered blocks.

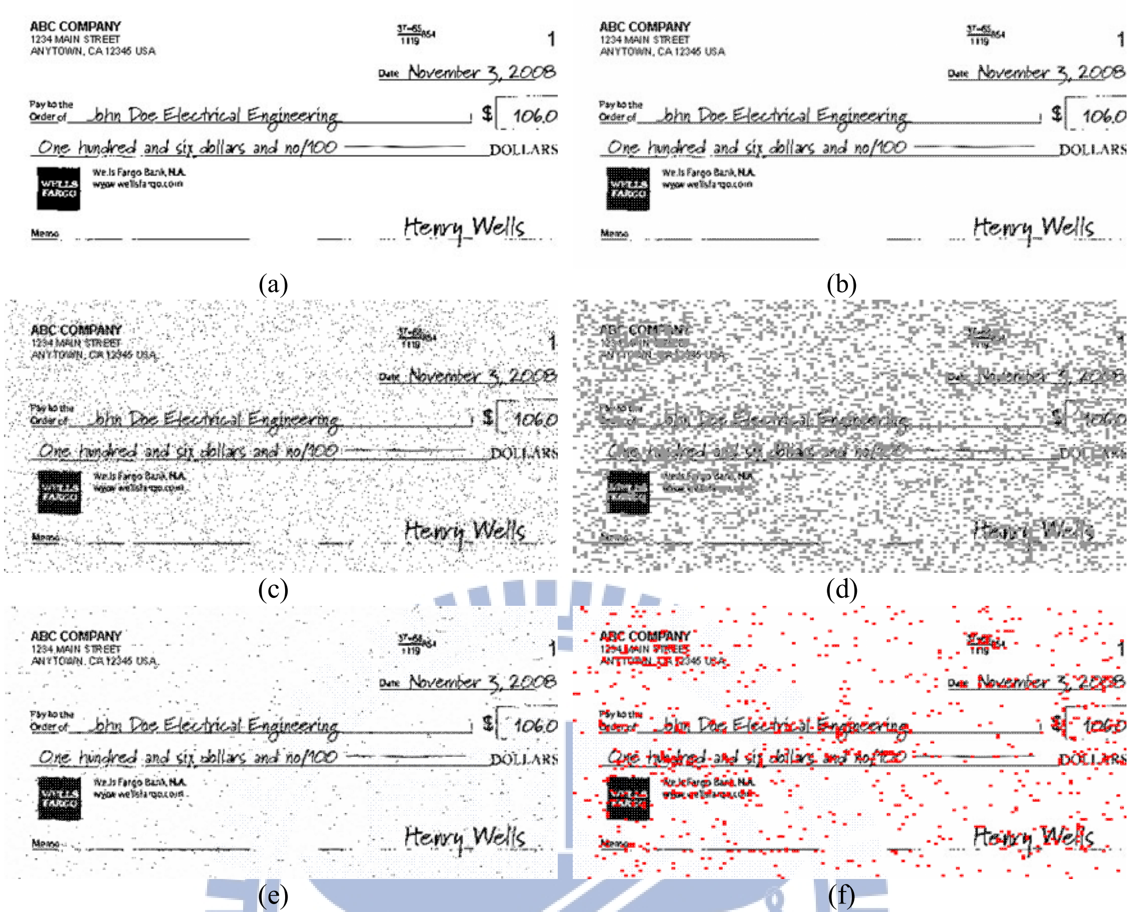


Fig. 3.16 Authentication result of a PNG document image of a check attacked by added noises. (a) Original cover image. (b) A stego-image with embedded data. (c) Tampered image with added noises. (d) Result with tampered blocks detected and marked as gray. (e) Data repair result. (f) Data repair result with red dots indicating unrepaired tampered blocks.

Table 3.3 Statistics of experimental results of using an image of check.

Experimental result (image size = 504×226)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of repaired blocks (repair ratio)	false acceptance ratio	false rejection ratio
Exp. 7 shown in Fig. 3.15	18984	1434 (7.55%)	1434 (100%)	1432 (99.86%)	0%	0%
Exp. 8 shown in Fig. 3.16	18984	6695 (35.27%)	5549 (82%)	5549 (100%)	17.12%	0%

C. Comparison of Performances with Other Methods

A comparison of the capabilities of the proposed method with those of four existing methods is shown in Table 3.4. All but the proposed method will create distortion in the stego-image during the authentication process. More importantly, only the proposed method has the capability of repairing the tampered parts of an authenticated image.

Furthermore, among the methods with tampering localization capabilities at the block level like Yang and Kot [3], Tzeng and Tsai [5], and the proposed method, the proposed method provides a finer authentication precision with the block size of 2×3. Specifically, the method in [3] needs larger *macro-blocks* to yield pixel flippabilities for embedding authentication data. In the case of using smaller blocks, Tzeng and Tsai's method [5] has a high possibility to generate noise pixels as mentioned in [2], and so they conducted experimental results with the larger block size of 64×64.

As to the distribution of *authenticated image parts*, because there exists no flippable pixel for use by the methods of [1-3] to embed data in *all-white* regions (such as marginal regions) of a document image, the distribution of authenticated image parts tends to be restricted to be on lines or strokes in the document, while the proposed method does not have this limitation. Nevertheless, in [1-3] the authenticity of an image part including such all-white regions can be still ensured by the use of cryptographic signatures embedded in other regions of the image. At last, the methods of [1-3] manipulate pixel flippability and the method of [5] enforces pixel replacement for the aim of data embedding. The proposed method is the only one which makes use of the alpha channel plane instead of the bit plane.

Table 3.4 Comparison of document image authentication methods.

	Distortion in stego-imag e	Tampering localization capability	Repair capability	Reported authentication precision	Distribution of authenticated image parts	Manipulation of data embedding
Wu & Liu [1]	Yes	No	No	Variable size	Non-blank part	Pixel flippability
Yang & Kot [3]	Yes	Yes	No	33×33 block	Non-blank part	Pixel flippability
Yang & Kot [2]	Yes	No	No	Variable size	Non-blank part	Pixel flippability
Tzeng & Tsai [5]	Yes	Yes	No	64×64 block	Entire image	Pixel replacement
Proposed method	No	Yes	Yes	2×3 block	Entire image	Alpha channel pixel replacement

3.8 Summary

A new blind image authentication method with a data repair capability for binary-like grayscale document images based on secret sharing has been proposed. Both the generated authentication signal and the content of a block are transformed into partial shares by the Shamir method, which are then distributed in a well-designed manner into an alpha channel plane to create a stego-image in the PNG format. The undesired opaque effect visible in the stego-image coming from embedding the partial shares is eliminated by mapping the share values into a small range of alpha channel values near their maximum transparency value of 255.

In the process of image block authentication, a block in the stego-image is regarded as having been tampered with if the computed authentication signal does not match that extracted from corresponding partial shares in the alpha channel plane. For self-repairing of the content of a tampered block, the reverse Shamir scheme is used to compute the original content of the block from any two untampered shares. Measures for enhancing the security of the data embedded in the alpha channel plane were also proposed. Experimental results have been shown to prove the effectiveness of the proposed method. Future studies may be directed to choices of other block sizes and related parameters (prime number, coefficients for secret sharing, number of authentication signal bits, etc.) to improve data repair effects. Applications of the proposed method to authentication and repairing of attacked color images may also be tried.

Chapter 4

A New Approach to Binary Image Authentication via Uses of PNG Images and Secret Sharing Technique with an Image Recovery Capability

4.1 Introduction

This study explores the issue of authenticating *binary* images which are popular for preserving text-type or line-art documents. It is hoped that if a critical image part is authenticated to have been altered illicitly, its original content can be recovered. Such binary image content verification and self-recovery capabilities are useful for authentication of many kinds of document data. However, the simple bi-level nature of binary images leads to shortage of data-embeddable space as well as creation of easily-perceptible changes after authentication signals are embedded into the image pixels. Therefore, a good solution to binary image authentication should take into account not only the possibility of creating sufficient space for data embedding but also the necessity of keeping the visual quality of the resulting image.

4.2 Merits of Proposed Method

Some other merits of the proposed method are described in the following.

(1) *Causing no destruction to the cover binary image* — Existing binary image authentication methods usually embed authentication data into the cover image at the price of causing image destruction. Different from such methods, the proposed method utilizes the pixels' values of the alpha channel for the purpose of image authentication and data recovery, leaving the intensity channel of the PNG image *untouched* and so causing *no destruction* to it.

(2) *Having the reversibility to the original binary image* — The proposed method simply removes the alpha channel and rescales the sample depth of the stego-image to obtain the original binary image, thus achieving the reversibility to

the input image content. This is beneficial for certain applications like art image archiving.

(3) *Creating good concealments by secret sharing* — The proposed method embeds data in the form of shares into the alpha channel of the binary PNG image. The effect may be regarded as *double-fold* concealments, one fold contributed by the shares as a disguise of the original image data and the authentication signals, and the other fold contributed by the use of the alpha channel which is created by the proposed method to be nearly transparent with imperceptible noise.

(4) *Fully using channels in images for data embedding* — Different from the commonly-seen binary image with only one channel – the intensity channel, the PNG image has an additional channel – the alpha channel, which is fully utilized by the proposed method to embed the share data.

Being blind in the process of image authentication — Some authentication methods for binary images need auxiliary data such as a lookup table to decode embedded signals. In contrast, the proposed method requires no auxiliary information other than the stego-image in the process of authentication.

4.3 Proposed Method for generation of a stego-image

In the proposed method, a binary cover image transformed into the PNG format is first divided into nonoverlapping blocks of size 2x3, and an authentication signal is generated from each of them. Next, the generated data for use in later processes of authentication and self-recovery of block contents are gathered to form a secret message. The message then is transformed into n shares by the Shamir's (k, n)-threshold secret sharing scheme described in Chapter 3.2. Finally, the shares are embedded into the alpha-channel of the cover image to form a stego-image. Later, if some, but no more than k, of the shares for an image block are attacked and destroyed, then k of the remaining untouched shares are retrieved to recover the original image block content. In this way, as mentioned previously, the chance of data survival against attacks is raised. These main concepts of the proposed method are described in detail in this section. The values of (k, n) are taken to be (2, 6), respectively.

A. Algorithm for generation of a stego-image

A detailed algorithm for describing the generation of a stego-image in the PNG format of the proposed method is presented in the following, with the used symbols listed in Table 4.1.

***Algorithm 1: generation and embedding of authentication signals and shares*.**

Input: a binary image I with pixel values 0 and 1 in the intensity channel, and a key K.

Output: a stego-image I' composed of the intensity channel and an additional alpha channel in the PNG format with embedded data for authentication and self-recovery.

Steps.

1. (*Transforming the cover image into the PNG format and scaling the image sample depth*) Transform I into a PNG image with the original intensity channel, denoted as M, and an additional alpha channel, denoted as L, by the following steps.

   (1) Scale the sample depth (sd) of the intensity values of M from $sd_{\text{in}} = 1$ up to $sd_{\text{out}} = 8$ by the following linear equation to create a new value v' for each pixel's intensity value v in M:

$$v' = \lfloor v \times (ms_{\text{out}}/ms_{\text{in}}) + 0.5 \rfloor \tag{3}$$

   where

   (a) ms_{in} = the maximum sample (pixel) intensity value in the input image I $= 2^{sd\text{in}} - 1 = 2^1 - 1 = 1$;

   (b) ms_{out} = the maximum sample (pixel) intensity value in the output image I' $= 2^{sd\text{out}} - 1 = 2^8 - 1 = 255$;

   (c) $\lfloor \cdot \rfloor$ is the integer floor function.

(2) Create an 8-bit alpha channel L for I.

2. (*Beginning of looping*) In the raster-scan order, take a 2×3 block B with pixel values p_1 through p_6 from the intensity channel M.

Table 4.1 List of symbols used in Algorithms 3 and 4.

Symbol	Meaning
I	an input binary image
I'	a generated stego-image
K	a key
M	the intensity channel in I
L	the alpha channel in I
L'	the alpha channel in I' with authentication data
sd	the sample depth of an image
sd_{in}	the sample depth of an input image
sd_{out}	the sample depth of an output image
B	a 2×3 block of I
B'	a 2×3 block of I'
v	a pixel value in I
v'	a pixel value in I after sample depth scaling
$p_1, p_2, \ldots, p_6$	six pixel values in B
$p_1', p_2', \ldots, p_6'$	Thresholding result of six pixel values in B (if $p_i = 0$, set $p_i = 0$; otherwise, set $p_i = 1$, where $i = 1, 2, \ldots, 6$.)
m_1	authentication data in binary form
m_2	authentication data in binary form
m_1'	authentication data in decimal form
m_2'	authentication data in decimal form
q_i	the ith partial share
q_i'	the ith partial share after value mapping

3. (*Creation of authentication data*) If $p_i = 0$, set $p_i' = 0$; otherwise, set $p_i' = 1$, where $i = 1, 2, \ldots, 6$.
4. (*Creation of secret and coefficient values for secret sharing using Algorithm 1*) Create two binary values $m_1 = p_1'p_2'p_3'$; $m_2 = p_4'p_5'p_6'$ and transform them into decimal numbers m_1' and m_2', respectively.
5. (*Partial share generation*) Set p, c_i, and x_i in Eq. (1) of Algorithm 1 described in Chapter 3.2 to be the following values:

   (a) $p = 11$ (the smallest prime number larger than 7);

(b) $d = m_1'$, $c_1 = m_2'$;

(c) $x_1 = 1$, $x_2 = 2$, …, $x_6 = 6$;

and perform Algorithm 1 of Chapter 3.2 as a (2, 6)-threshold secret sharing scheme to generate six partial shares q_1 through q_6 using the following equations:

$$q_i = F(x_i) = (d + c_1 x_i)_{\text{mod } p}, \tag{4}$$

where $i = 1, 2, \ldots, 6$.

6. (*Mapping of partial share values*) Add 244 to each of q_1 through q_6, resulting in the new values of q_1' through q_6', respectively, which fall in the nearly total transparency range of 244 through 254.
7. (*Embedding of two partial shares at the current block*) Take the block B' in the alpha channel L which in position corresponds to B in the intensity channel M, select in the raster-scan order the values p_1' and p_2' of the first two pixels in B', and change them to be q_1' and q_2', respectively.
8. (*Embedding the remaining partial shares at random positions*) Perform the following steps to embed the remaining partial shares.
   (1) Use the input key K to select randomly four pixels in L which are *outside* B', *unselected yet* in this step, and *not* the first two pixels of any block, and let their values be p_3' through p_6'.
   (2) Set the values p_3' through p_6' respectively to be the four mapped partial shares q_3' through q_6' of B generated in Step 6 above.
9. (*End of looping*) If there exists any unprocessed block in the intensity channel M, then go to Step 2; otherwise, denote the resulting alpha channel L to be L' and the resulting intensity channel M to be M'; and take M' and L' *together* as the desired stego-image I'.

After Step 1, the black and white pixels in the binary cover image are represented by values 0 and 255, respectively, in the resulting image. The sample depth scaling is achieved by using the linear equation (3) defined by the PNG standard [57] in which a binary image is regarded as an extreme case of a grayscale image with sample depth 1.

The possible values q_1 through q_6 yielded by the formulas in (4) above and inserted in the alpha channel L of image I are between 0 and 10 because the value of p used in (4) is 11. And after performing Step 6 of the above algorithm, the values q_1

through q_6 become q_1' through q_6', respectively, which fall in a small range of integers from 244 to 254. The embedding of these values within such a small range means that *very similar* values will appear everywhere in the alpha channels after Steps 7 and 8 are completed, resulting in a *nearly uniformly transparent binary* PNG image with an *imperceptible* effect, as mentioned previously. In addition, we use Fig. 4.1 to illustrate Steps 7 and 8 of Algorithm 1 where a core idea of the proposed method is presented: two shares of the generated six are embedded within the current block and the other four embedded at four randomly-selected pixels outside the block, with each selected pixel not being either of the first two in any block.

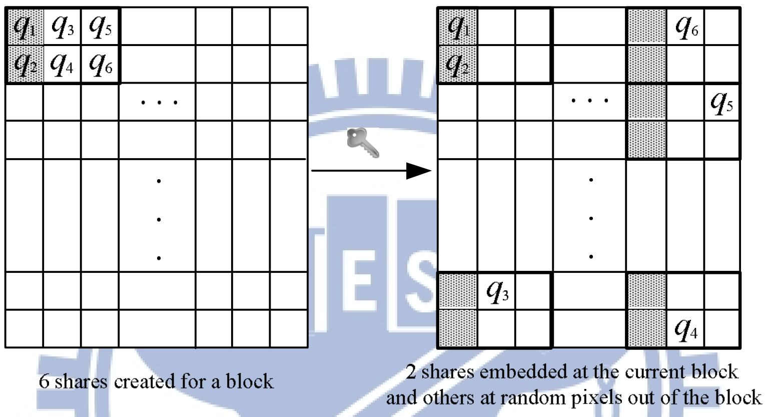

q_1 q_3 q_5
q_2 q_4 q_6
6 shares created for a block
q_1
q_2
q_6
q_5
q_3
q_4
2 shares embedded at the current block and others at random pixels out of the block

Fig. 4.1 Illustration of embedding six shares created for a block — two shares embedded within the current block and the other four in four randomly-selected pixels outside the block, with each selected pixel not being either of the first two in any block.

4.4 Proposed Image Authentication and Recovery Process

A detailed algorithm describing the proposed stego-image authentication process, including both verification and self-recovery of the original image content, is presented in the following.

Algorithm 2*: *image authentication and data recovery.

Input: a stego-image I' generated by Algorithm 1 with an intensity channel M' and an alpha channel L'; and the key K used there.

Output: a binary image I_r with tampered blocks marked and their data recovered if possible.

1. (*Beginning of looping*) Take in the raster-scan order a block B from the intensity channel M' of I' with pixel values p_1 through p_6, and let the alpha channel values of block B' in L', which in position corresponds to B, be denoted as q_1' through q_6',

respectively.

2. (*Extraction of the hidden authentication data*) Perform the following steps to extract the embedded authentication data in the alpha channel L'.

   (1) Take the first two alpha channel values q_1' and q_2' in B'.

   (2) Subtract 244 from q_1' and q_2' to obtain q_1 and q_2, respectively.

   (3) Take (1, q_1) and (2, q_2) as *two shares* of B'; and with them as input, perform Algorithm 2 described in Chapter 3.2 to extract the two values d and c_1 (the secret and the first coefficient value) from q_1 and q_2.

3. (*Computation of authentication data*) For each block B in the intensity channel M' with pixel values p_1 through p_6, compute the authentication data a_1' and a_2' as follows.

   (1) If $p_i = 0$, set $p_i' = 0$; otherwise, set $p_i' = 1$, for $i = 1, 2, \ldots, 6$.

   (2) Create two binary values $a_1 = p_1'p_2'p_3'$ and $a_2 = p_4'p_5'p_6'$, and transform them into decimal numbers d' and c_1', respectively.

4. (*Matching the hidden and computed authentication data and marking tampered blocks*) Match respectively the extracted authentication data d and c_1 with the computed data d' and c_1', and if any mismatch occurs, mark B and all the mapped partial shares embedded in B' as *tampered*.

5. (*End of looping*) If there exists any unprocessed block in M', then go to Step 1; otherwise, continue.

6. (*Extraction of the remaining partial shares*) For each block B' in the alpha channel L', perform the following steps to extract the remaining four partial shares of B' from the alpha channel values of the other blocks in L'.

   (1) Use the key K to collect the four pixels which were randomly selected for B' in Step 8 of Algorithm 1, and take out the data q_3' through q_6' embedded in them.

   (2) Subtract 244 from each of q_3' through q_6', respectively, to obtain *four partial shares* q_3 through q_6, respectively.

7. (*Recovering the lost data for tampered regions*) For each block B' marked as tampered, perform the following steps to recover its original content.

   (1) From the six previously computed or collected shares q_1 through q_6 of B', choose, if possible, two shares, denoted as r_1' and r_2', which are *not* marked

as tampered.

(2) Subtract 244 from r_1' and r_2' to obtain r_1 and r_2, respectively.

(3) Take r_1 and r_2 as *two partial shares of* B'; and with them as input, perform Algorithm 2 of Chapter 3.2 to extract the values of d and c_1 (the secret and the first coefficient value) from r_1 and r_2.

(4) Transform d and c_1 into 3-bit numbers and concatenate them to form a 6-bit string S.

(5) Replace the pixel values of block B corresponding to B' with S as the result of data recovery.

8. (*Image sample depth rescaling*) Rescale the sample depth of the final M' from 8 down to 1 in a reverse way of Step 1 of Algorithm 1 as the desired self-recovered binary image I_r.

In Step 8, by rescaling the sample depth of a recovered image I' generated by Step 7, the pixel values 0 and 255 representing black and white go back to their original versions 0 and 1. Also, it is easy to see that the original binary image, if untampered, can be obtained losslessly at the end of the algorithm above.

4.5 Security Consideration

For ease to understand the proposed method, the values of x_1 through x_6 used in Step 5(c) of Algorithm 1 and Step 2(3) of Algorithm 2 are taken to be constants. But for the purpose of preventing the embedded data from being forged, we also allow x_1 through x_6 for each block to be selected *randomly* within the allowed integer range of $0 \leq x_i < p$ (= 11) [56] by using a secret key. Accordingly, the probability of correctly guessing the selected values of x_1 through x_6 for a block is $1/[(11\times10\times9\times8\times7\times6)] \approx 3.006\times10^{-6}$. Therefore, in an $m\times n$ cover image with a total of $[m\times n/(2\times3)] = m\times n/6$ blocks, the aforementioned probability is $1/[(11\times10\times9\times8\times7\times6)]^{m\times n/6} \approx 1/(3.006\times10^{-6})^{m\times n/6}$, which is very small for common image sizes of $m\times n$. The detail of using a key K' to choose random values for x_1 through x_6 is described as follows.

Step 1. Use K' as the seed for a random number generator to get a number sequence $R = r_1, r_2, \ldots$.

Step 2. For each image block, perform the following steps.

(1) Take an element r_j from R in order.

(2) For each x_i with $i = 1$ through 6, compute x_i by $x_i = (r_j)_{\text{mod } 11}$, and if the computed x_i is equal to any of x_1, x_2, …, and x_{i-1}, discard x_i and go to Step 2(1) to re-compute another value.

In the process of later authentication, the same values of x_1 through x_6 for each block used in Algorithm 1 can be obtained in the same way.

Furthermore, resistance to possible attacks to the proposed authentication method is also considered. For this, the cut-and-paste attack proposed by Hollimon and Memon [60] is analyzed. The attack defeats certain block-based oblivious watermarking schemes by counterfeiting an existing watermark in an unwatermarked image without knowledge of the watermark insertion key. Specifically, given a watermarked image with n blocks, $X = \{X_1, X_2, …, X_n\}$, containing a watermark W, the attack proposed by Hollimon and Memon [60] is to allow an adversary to construct a counterfeit image also with n blocks, $Y' = \{Y_1', Y_2', …, Y_n'\}$, such that the same watermark data can be extracted both from all X_i and Y_i' by using the same key where $1 \leqq i \leqq n$. Here the counterfeit image Y' is an approximation of a target image Y. It is mentioned that two image blocks Z_1 and Z_2 are said in [60] to be K-equivalent if $D_K(Z_1) = D_K(Z_2) = W$ where D_K denotes a watermark extraction function with a key K. Accordingly, X_i and Y_i' are K-equivalent here. A conclusion made in [60] says that embedding information into a host image in a block-wise independent fashion is vulnerable to the counterfeit attack.

However, the method proposed in this study is immune to such a type of counterfeit attack because the *block dependence* is established by distributing four partial shares q_3' through q_4' of each block to other randomly-chosen blocks in the alpha channel (See Step 8 of Algorithm 1). More specifically, as illustrated in Fig. 4.2 below, given a watermarked block B, even though an adversary may replace the content of B with that of another block C from which the same authentication data as those embedded in B can be extracted by the first two shares q_1' and q_2' (See Step 2 of Algorithm 2), yet after collecting four randomly-located partial shares q_3' through q_6' from other counterfeit blocks, block C will be checked to be counterfeit because the authentication signal extracted from any two partial shares out of q_3' through q_6' and that extracted from q_1' and q_2' will not match.

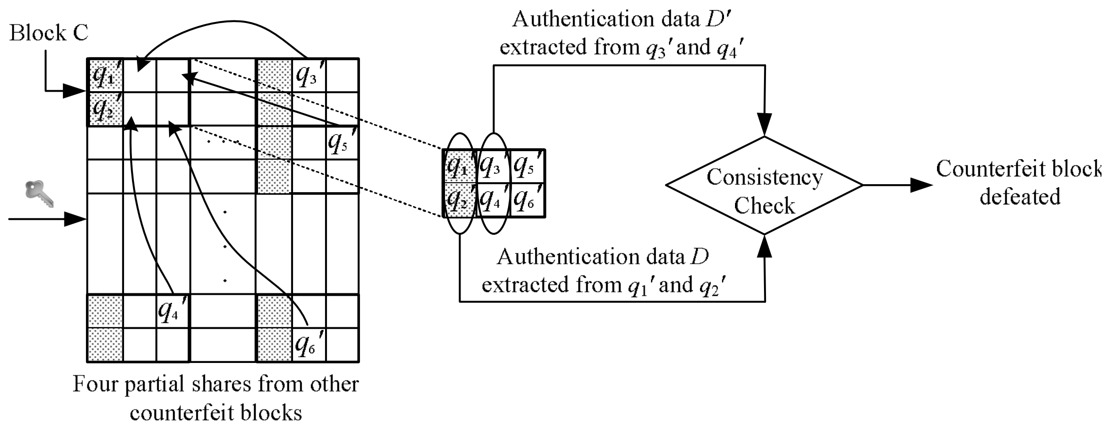

Block C
Authentication data D' extracted from q_3' and q_4'
q_1'
q_2'
q_3'
q_4'
q_5'
q_6'
Consistency Check
Counterfeit block defeated
Authentication data D extracted from q_1' and q_2'
Four partial shares from other counterfeit blocks

Fig. 4.2 Illustration of immunity of proposed method against cut-and-paste counterfeit attack.

Accordingly, those randomly-distributed partial shares, i.e., q_3' through q_6' of each block, are essential to the security consideration in the proposed method. The probability of correctly guessing the positions of all distributed partial shares is then analyzed as follows. As described in Algorithm 1, the first two pixels in the raster-scan order in each block in the alpha channel are reserved for keeping the first two of the six partial shares of this block, leaving the other four pixels of the block to keep four partial shares of other blocks located at random positions. That is, in an $m\times n$ cover image with a total of $[(m\times n)/(2\times3)] = m\times n/6$ blocks, each block provides four pixel positions to keep partial shares from other blocks, yielding a total of $k = (m\times n/6)\times4$ candidate pixel positions to be randomly chosen by a secret key. As a result, for an adversary without the key to guess correctly the positions of all the randomly-distributed partial shares in a stego-image, the probability is $1/[k\times(k-1)\times\ldots\times1] = 1/k!$ This very small probability for the common image size of $m\times n$ means that it is infeasible to collect blindly the randomly-distributed partial shares to pass the check of authentication-signal consistency mentioned above. This also means that an adversary cannot simply select certain individual blocks, which are K-equivalent to corresponding blocks in the original image, to construct an effective counterfeit image [60].

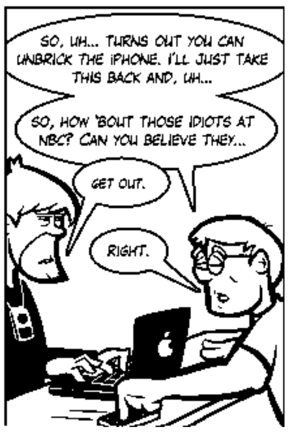
Finally, it is mentioned that the authentication data may possibly be erased by discarding or modifying the content of the alpha channel without producing any alteration to the content of the intensity channel. How does the proposed method work for this case? First, the major concern of image authentication is to avoid a counterfeit image to pass the authentication process. This concern is different from that of maintaining robustness in the application of copyright protection — keeping a watermark such as a logo in a stego-image to survive attacks. Therefore, at an authentication side, if a stego-image is checked to include no alpha channel, it means the integrity of the stego-image is seriously lost. In other words, the alpha channel is regarded as a necessary part of the stego-image and discarding or modifying it without touching the intensity channel to deceive the authentication processes will fail.

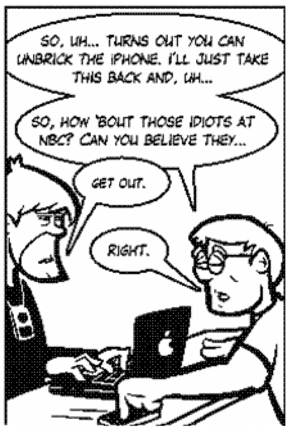
Besides, in designing an image authentication method, it is expected that the method is not only capable of detecting noticeable alterations such as the aforementioned attack (removing the authentication data by eliminating the entire alpha channel) but also able to deal with sophisticated attacks which seem flawless, such as imperceptible content modifications made by painting, superimposing, cut-and-paste attacks, etc. The proposed method has taken theses cases into considerations.

4.6 Experimental Results

A. Experimental Results Using a Binary Comic Image

In Fig. 4.3(a), a PNG binary comic image with size 3.71 KB is taken to be the input image in the experiment. The corresponding stego-image with size 6.52 KB generated by the proposed method is shown in Fig. 4.3(b), which is visually identical to the cover image although authentication data have been embedded into the alpha channel of the stego-image. It is noted that the increase of the stego-image size is caused by the appended alpha channel. However, this state of a stego-image represented in such a PNG image may be regarded as a temporary transition, because the original 1-bit binary image can be obtained by removing the alpha channel and rescaling the sample depth of the stego-image after the authentication process is finished. In addition, Fig. 4.3(c) is given to show the opaque effect resulting from skipping Step 6 of Algorithm 1 which maps partial share values into the alpha channel value range of 244 through 254.


SO, UH... TURNS OUT YOU CAN UNBRICK THE IPHONE. I'LL JUST TAKE THIS BACK AND, UH...
SO, HOW 'BOUT THOSE IDIOTS AT NBC? CAN YOU BELIEVE THEY...
GET OUT.
RIGHT.
(a)


SO, UH... TURNS OUT YOU CAN UNBRICK THE IPHONE. I'LL JUST TAKE THIS BACK AND, UH...
SO, HOW 'BOUT THOSE IDIOTS AT NBC? CAN YOU BELIEVE THEY...
GET OUT.
RIGHT.
(b)

(c)

Fig. 4.3 Result of a binary comic image processed by proposed method. (a) PNG cover binary image of 216×324. (b) A stego-image with embedded data. (c) Another stego-image created without conducting partial share value mapping.

Two common image editing operations *superimposing* and *painting* were used to simulate tampering with the stego-image. Some tampered images yielded by the superimposing operation are presented in Fig. 4.4. It was found that the superimposing operation, like that provided by the image editing software Adobe Photoshop or Corel PhotoImpact destroys the content of the alpha channel values by replacing all the original alpha channel values at the attacked part with the new values of 255. Since the largest alpha channel values created by the proposed method is 254 (see Step 6 of Algorithm 1), all pixels with the unique values of 255 in the alpha channel plane may be easily detected as tampered by a modified version of Step 4 of Algorithm 2, which we describe as follows:

Step 4′ (*Check of superimposing attacks and extraction of the hidden authentication data*) Check if both q_1' and q_2' are 255; if so, *then* (1) regard the corresponding block B in I' as tampered by superimposing, (2) mark B, B', and all the partial shares embedded in L as tampered, and (3) go to Step 6; *otherwise*, perform the original operations of Step 4 of Algorithm 2.

Figs. 4.4(b) shows the result of superimposing a fake face on the person at the right side of the stego-image Fig. 4.4(a). Fig. 4.4(c) shows the authentication result yielded by Algorithm 2, with the gray blocks indicating the detected tampered image parts. As can be seen, the superimposing part has been detected successfully. For each of the detected tampered blocks, if at least two untampered shares of it can be collected, its original content can be recovered, yielding the result shown in Fig. 4.4(d); otherwise, the tampered block is left *unrecovered*, as shown by the red dots in Fig. 4.4(e). As a comparison, we show the result of data recovery accomplished with a wrong key in Fig. 4.4(f). An erroneous image recovery result was obtained as expected.

We have also conducted experiments of enlarging the tampered image parts during the attacks to test the performance of the proposed method. The corresponding statistics of the performance is shown in Table 4.2 in which five parameters: *tampering ratio*, *detection ratio*, *recovery ratio*, *false acceptance ratio*, and *false rejection ratio*, are as defined in the following:

(1) *tampering ratio* = (the number of tampered blocks)/(the total number of blocks);

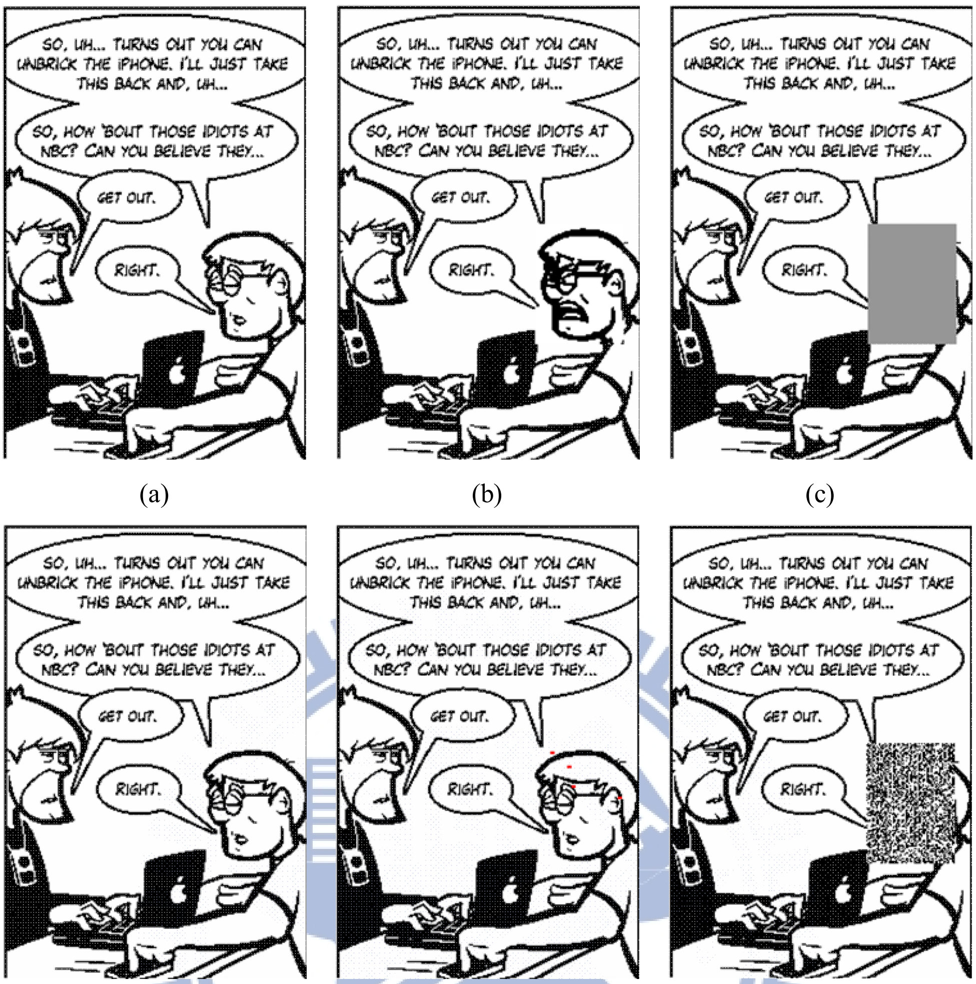
(2) *detection ratio* = (the number of detected blocks)/(the number of tampered blocks);

(3) *recovery ratio* = (the number of recovered blocks)/(the number of detected blocks);

(4) *false acceptance ratio* = (the number of tampered blocks marked as untampered)/(the total number of tampered blocks);

(5) *false rejection ratio* = (the number of untampered blocks marked as tampered)/(the total number of untampered blocks).

It can be observed that the recovery ratio becomes worse when the tampering ratio grows. This is reasonable because when the tampered area becomes larger, fewer partial shares for image recovery will survive. We illustrate the relationship between the tampering ratio and the recovery ratio in Fig. 4.5. Note that the detection ratios presented in Table 4.2 are all 100% due to the ease in the detection of the alpha channel values of 255 (using Step 4′ described above) at the image parts attacked by superimposing, as mentioned previously. Likewise, the alpha channel value corresponding to an intact block will not be 255 and can be easily checked to be so, yielding a false rejection rate of 0%. On the contrary, the alpha channel value corresponding to a tampered block is 255 which is easy to check as well, yielding a false acceptance rate of 0%.


SO, UH... TURNS OUT YOU CAN UNBRICK THE iPHONE. I'LL JUST TAKE THIS BACK AND, UH...
SO, HOW 'BOUT THOSE IDIOTS AT NBC? CAN YOU BELIEVE THEY...
GET OUT.
RIGHT.

(a) (b) (c)

(d) (e) (f)

Fig. 4.4 Another authentication result of a stego-image of comic attacked by the superimposing operation. (a) A stego-image. (b) Tampered image yielded by superimposing a fake face on the person at the right side in Fig. 4.4(a). (c) Result with tampered blocks detected and marked as gray. (d) Result of image recovery. (e) Result of image recovery with red dots indicating unrecovered tampered blocks. (f) Erroneous image recovery result obtained with a wrong key.

Two other examples of tampered images yielded by the common operation of painting provided by well-known image editing software are presented in Figs. 4.6 and 4.7. Again, painting using Adobe Photoshop will replace the alpha channel values by 255, just like the superimposing operation mentioned previously. However, it was found that the painting operation provided by Corel PhotoImpact does *not* change the alpha channel values. In Fig. 4.6(a), the word iPHONE in the conversation and the brand label of the laptop appearing in Fig. 4.3(b) are modified by the painting operation. Fig. 4.6(b) shows the authentication result in which gray blocks were used again to indicate image parts where mismatching authentication data were detected. And the result of image recovery is shown in Fig. 4.6(c) from which we can see that the original content of the altered region reappears without loss.

Table 4.2 Statistics of experimental results of attacks using superimposing operations.

Experimental result (image size = 216×324)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of recovered blocks (recovery ratio)	false acceptance ratio	false rejection ratio
Exp. 1 shown in Fig. 4.4	11664	903 (7.7%)	903 (100%)	899 (99.56%)	0%	0%
Exp. 2	11664	2301 (20%)	2301 (100%)	2241 (97.39%)	0%	0%
Exp. 3	11664	4674 (40%)	4674 (100%)	3824 (81.81%)	0%	0%
Exp. 4	11664	6968 (60%)	6968 (100%)	3645 (52.31%)	0%	0%
Exp. 5	11664	9310 (80%)	9310 (100%)	1702 (18.28%)	0%	0%

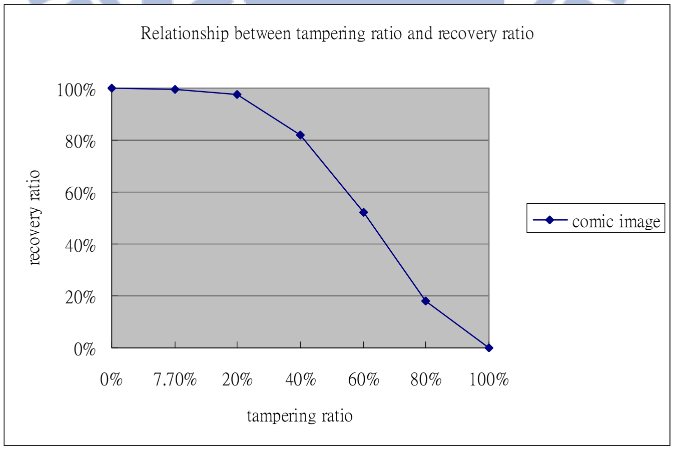
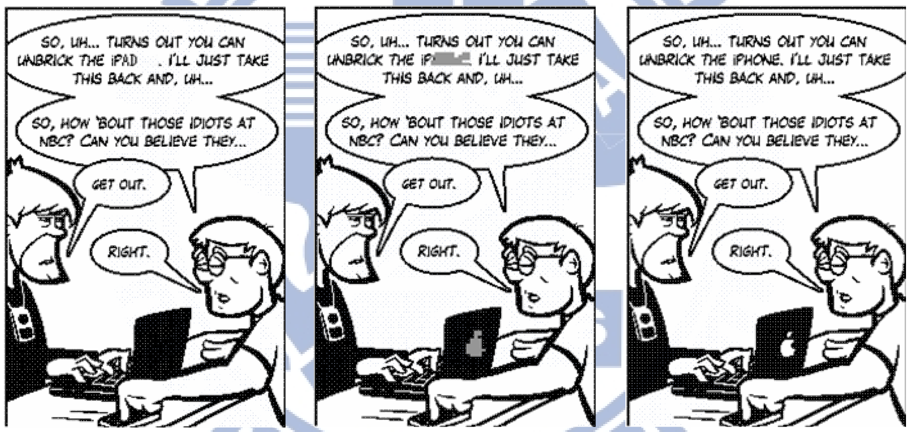


Fig. 4.5 Relationship between tampering ratios and recovery ratios of Table 4.2 for the comic image.

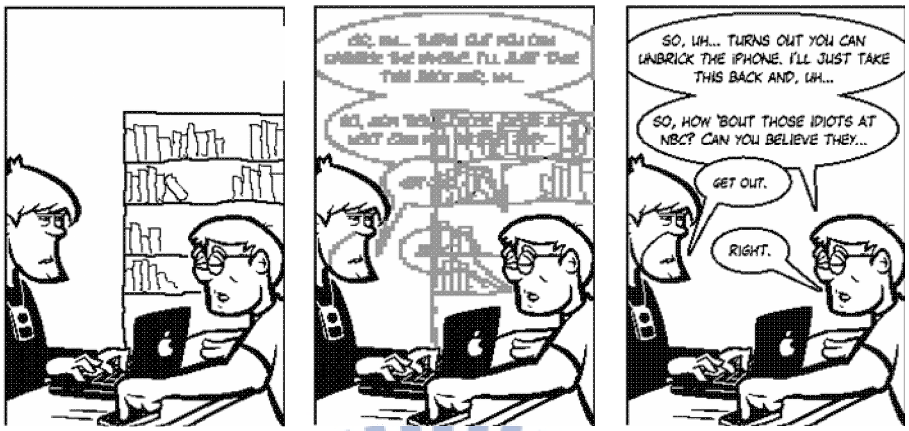
Fig. 4.7 shows the results of removing the image content of conversations and painting a counterfeit cabinet. The untouched content of the alpha channel values still yields image recovery results with the original contents reappearing very clearly. Table 4.3 shows the statistics corresponding to these experimental results of modification by painting. Since the stego-image was tampered with by painting which kept the content of the alpha channel plane intact as mentioned previously, the hidden data for authentication and recovery are not destructed. Therefore, the computed authentication data from the alpha channel values are always true, leading to the false rejection rate of 0%. Since a few tampered blocks are coincidentally identical to that of its untampered version, such a type of tampered block will reasonably be regarded as untampered and left unmarked, as indicated by several black blocks appearing in the detected region in gray shown in Fig. 4.7(b).


SO, UH... TURNS OUT YOU CAN UNBRICK THE IPAD . I'LL JUST TAKE THIS BACK AND, UH...
SO, HOW 'BOUT THOSE IDIOTS AT NBC? CAN YOU BELIEVE THEY...
GET OUT.
RIGHT.
SO, UH... TURNS OUT YOU CAN UNBRICK THE IPHONE. I'LL JUST TAKE THIS BACK AND, UH...

(a) (b) (c)

Fig. 4.6 Authentication result of a stego-image of comic attacked by painting. (a) A tampered image. (b) Result of authentication with tampered blocks detected and marked as gray. (c) Result of image recovery.

On the other hand, since the authentication data of each block are composed of d and c_1 both with values coming from the set of {0, 1, 2, …, 7} (see Steps 3 and 4 in Algorithm 2), there exists a probability of 1/8 for a coincidental match between the extracted d' and the computed d to occur. Likewise, the probability for a coincidental match between the extracted c_1' and the computed c_1 to occur is also 1/8. Consequently, there is a total probability of $1/8 \times 1/8 = 1/64$ for an erroneous block authentication to occur, yielding a false acceptance ratio of (1/64)% = 1.56% in theory. The corresponding statistics of the false acceptances are given in Table 4.3.


SO, UH... TURNS OUT YOU CAN UNBRICK THE iPHONE. I'LL JUST TAKE THIS BACK AND, UH...
SO, HOW 'BOUT THOSE IDIOTS AT NBC? CAN YOU BELIEVE THEY...
GET OUT.
RIGHT.

(a) (b) (c)

Fig. 4.7 Authentication results of a stego-image of comic attacked by painting. (a) A tampered image. (b) Result of authentication with tampered blocks detected and marked as gray. (c) Result of image recovery.

Table 4.3 Statistics of experimental results of attacks using painting operations.

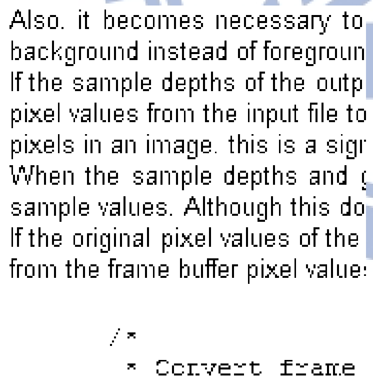
Experimental result (image size = 216×324)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of recovered blocks (recovery ratio)	false acceptance ratio	false rejection ratio
Exp. 6 shown in Fig. 4.6	11664	74 (0.63%)	73 (100%)	73 (98.65%)	1.35%	0%
Exp. 7 shown in Fig. 4.7	11664	2391 (20.50%)	2380 (100%)	2380 (97.39%)	0.46%	0%

It should be mentioned that almost all image processing operations which can be applied to a stego-image have been tried in this study, and it is found that, in addition to "superimposing," the operation of "erasing" will also change the alpha values, but to be 0 instead of 255. To deal with this case, we do not have to regard the alpha value 0 as a distinguishing one for image authentication, but can just use Algorithm 2 proposed in this study to perform authentication normally. The reason is that the authentication signals extracted from the alpha channel with such alpha values will be incorrect, and mismatches of authentication signals will occur after these incorrect signals are compared with those computed from the current block contents in the intensity channel.

Actually, even though there might be other image processing operations which will change the alpha values to be other than 255, the general version of the proposed method described in Algorithms 1 and 2 still works because the proposed method detects attacks by matching authentication signals computed from the intensity channel and those extracted from the alpha channel (See Step 4 of Algorithm 2).

B. Experimental Results Using a Document Image

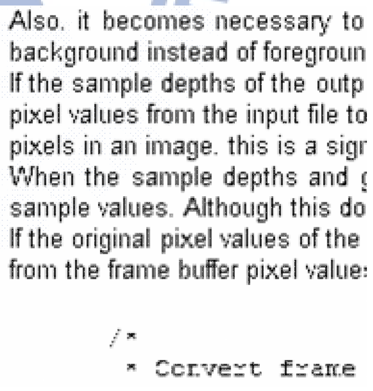
Experimental results yielded by the use of a document image are shown in Figs. 4.8 through 4.11 where Figs. 4.8(a) and 4.8(b) are respectively the cover document image in the PNG format with size 1.57 KB and the stego-image generated by the proposed method with size 2.63 KB. Again, the increase of the stego-image size results from the appended alpha channel.



Also, it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image, this is a sigr
When the sample depths and g
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

```
  /*
   * Convert frame
```

(a)



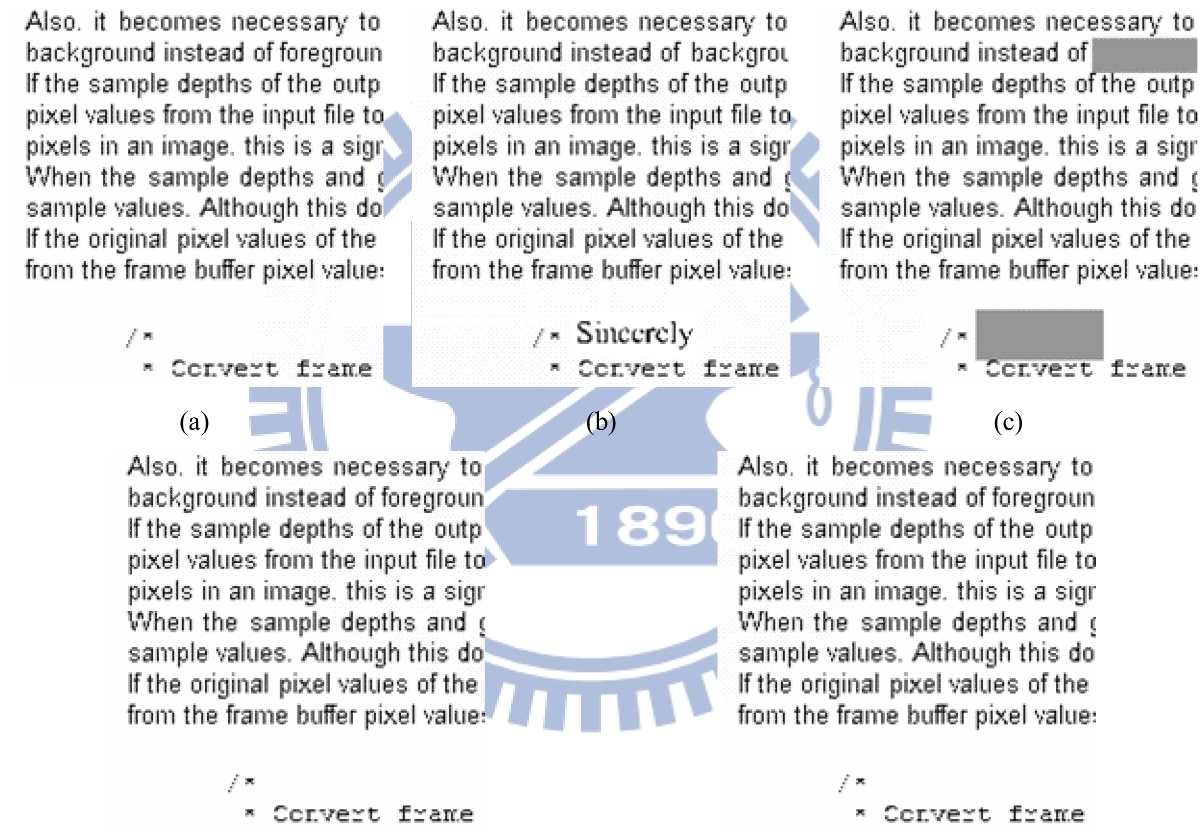
Also, it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image, this is a sigr
When the sample depths and g
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

```
  /*
   * Convert frame
```

(b)

Fig. 4.8 Result of a binary document image processed by proposed method. (a) PNG cover image of 195×196. (b) A stego-image with embedded data.

Fig. 4.9(b) shows the result of text content tampering by superimposing some counterfeit text parts on Fig. 4.9(a). The result of tampering detection by the proposed authentication algorithm is shown in Fig. 4.9(c) in which the modified part of "backgrou" and the added part "Sincerely" were successfully marked. Since the tampering ratio is not large, the destruction occurring in the alpha channel content is not much, yielding a good image recovery result without any unrecovered blocks as shown by Figs. 4.9(d) and 4.9(e). In addition, we also enlarged the tampered regions in the document stego-image to test the performance of the proposed method. The corresponding statistics are given in Table 4.4. Fig. 4.10 is an illustration of the relationship between the tampering ratio and the recovery ratio for the stego-image of the document.



Also. it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigr
When the sample depths and
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/*
 * Convert frame

(a)

Also. it becomes necessary to
background instead of backgrou
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigr
When the sample depths and
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/* Sincerely
 * Convert frame

(b)

Also. it becomes necessary to
background instead of
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigr
When the sample depths and
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/*
 * Convert frame

(c)

Also. it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigr
When the sample depths and
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/*
 * Convert frame

(d)

Also. it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigr
When the sample depths and
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/*
 * Convert frame

(e)

Fig. 4.9 Authentication result of a stego-image of document attacked by superimposing operations. (a) A stego-image. (b) Tampered image yielded by superimposing counterfeit text parts on Fig. 4.9(a). (c) Result with tampered blocks detected and marked as gray. (d) Result of image recovery. (e) Unrecovered blocks shown in red (none for this example).

Table 4.4 Statistics of experimental results of attacks using superimposing operations.

Experimental result		No. of	No. of	No. of	false	false
(image size =	No. of	tampered	detected	recovered	acceptance	rejection

195×196)	blocks	blocks (tampering ratio)	blocks (detection ratio)	blocks (recovery ratio)	ratio	ratio
Exp. 8 shown in Fig. 4.9	6370	432 (6.78%)	432	432 (100%)	0%	0%
Exp. 9	6370	1269 (20%)	1269 (100%)	1237 (97.48%)	0%	0%
Exp. 10	6370	2540 (40%)	2540 (100%)	2097 (82.60%)	0%	0%
Exp. 11	6370	3825 (60%)	3825 (100%)	2011 (52.58%)	0%	0%
Exp. 12	6370	5073 (80%)	5073 (100%)	961 (18.94%)	0%	0%

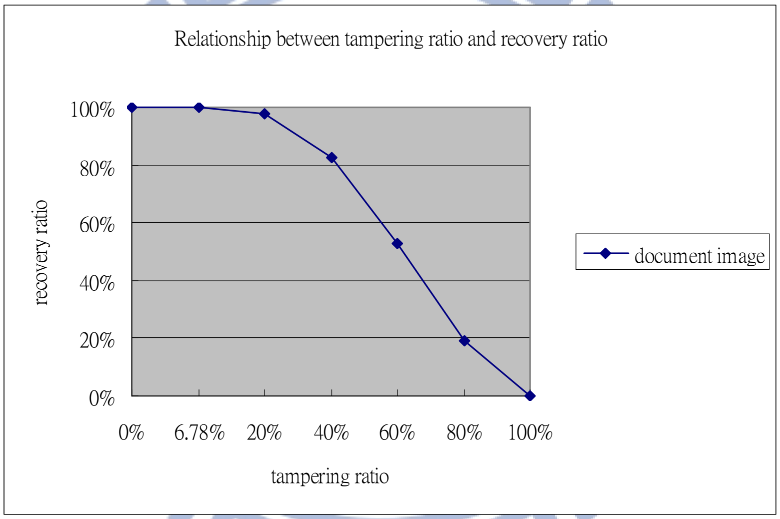
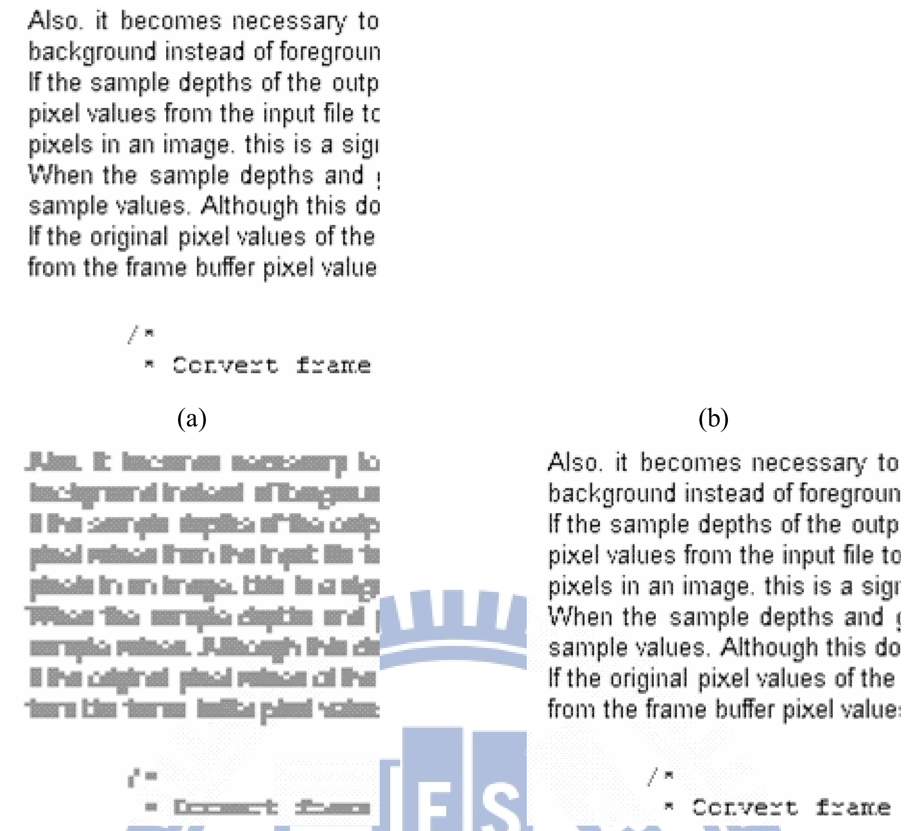


Fig. 4.10 Relationship between tampering ratio and recovery ratio for the document image.

Fig. 4.11 shows the experimental results of removing the entire image content by the painting operation provided by Corel PhotoImpact. As mentioned previously, since the content of the alpha channel values are untouched, the tampered blocks were all detected and the recovery result for those blocks meets the performance expectation as shown in Figs 4.11(c) and 4.11(d), respectively. The statistics of this experiment is shown in Table 4.5.


Also. it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file tc
pixels in an image. this is a sigı
When the sample depths and ı
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value

/*
 * Convert frame

Also. it becomes necessary to
background instead of foregroun
If the sample depths of the outp
pixel values from the input file to
pixels in an image. this is a sigı
When the sample depths and ı
sample values. Although this do
If the original pixel values of the
from the frame buffer pixel value:

/*
 * Convert frame

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

Fig. 4.11 Authentication result of a stego-image of document attacked by painting operations. (a) A stego-image. (b) Tampered image yielded by painting white color on entire content of Fig. 4.11(a). (c) Result with tampered blocks detected and marked as gray. (d) Result of image recovery.

Table 4.5 Statistics of experimental results of attacks using painting operations.

Experimental result (image size = 216×324)	No. of blocks	No. of tampered blocks (tampering ratio)	No. of detected blocks (detection ratio)	No. of recovered blocks (recovering ratio)	false acceptance ratio	False rejection ratio
Exp. 13 shown in Fig. 4.11	6370	1967 (30.88%)	1967 (100%)	1967 (100%)	0%	0%

At last, a result of size comparisons between the cover images shown in Figs. 4.3(a) and 4.8(a), and the corresponding resulting stego-images shown in Figs. 4.3(b) and 4.8(b), is given in Table 4.6. The proposed method makes use of the additional alpha channel to accommodate authentication data while the other existing methods utilize only the intensity channel; therefore, the proposed method yields stego-images with larger sizes comparatively, which are disadvantageous to image transmission and keeping, requiring longer time and larger storage space. The larger image size generated by the proposed method may be regarded as a price for gaining the image recovery capability as well as the finer authentication precision.

Table 4.6 Size comparison between cover images and resulting stego-images.

	Fig. 4.3	Fig. 4.8
Size of cover image in PNG format (without the alpha channel)	3.71 KB	1.57 KB
Size of stego- image in PNG format (with data embedded in the alpha channel)	6.52 KB	2.63 KB

C. Comparison of Performances with Existing Methods

For the purpose of presenting the contributions made by the proposed method, a comparison in terms of important capabilities between the proposed method and five existing binary image authentication methods is given in Table 4.7.

Contrastive with the methods in [1-3, 7] which flip qualified pixels to embed data and the method in [5] which changes pixel values for embedding authentication codes, the proposed method is the only one which causes no destruction to the content of an input binary image. It is also noted that the destruction caused by data embedding in the methods of [1-3, 5, 7] is irreversible, while the proposed method has the reversibility that enables an untampered image to transform back reappear with no loss to the original cover image.

Methods [1-2, 7] check the image content integrity by inspecting the patterns of the extracted signatures or watermarks; however, these methods lack the capability to precisely localize the tampering region. In view of this, methods [3, 5] provide the tampering localization capabilities; however, the method in [3] needs larger *macro-blocks*, composed of many finer blocks, to seek enough amounts of flippable pixels for embedding authentication data, leading to a coarser authentication precision. As to the method of [5], it tends to yield a stego-image which includes noise pixels in the blank area due to authentication code embedding. And so, the use of a large block size of 64×64 was recommended. Compared with methods in [3, 5] which have the tampering localization capability, the proposed method localizes tampered regions with a finer authentication precision.

Finally, it is worth to mention that only the proposed method has the capability of recovering the tampered parts of an authenticated binary image. Through the integration of the secret sharing technique and the larger data embeddable space provided by the alpha channel supported by the PNG format, the self-recovery capability is fulfilled so that a complete image authentication system including tampering detection, tampering localization, and tampering recovery is established by the proposed method.

Table 4.7 Comparison of binary image authentication methods.

Method	No destruction to cover image	Reversibility	Tampering localization capability	Recovery capability	Reported authentication precision
Wu & Liu [1]	No	No	No	No	Macro-block
Yang & Kot [2]	No	No	No	No	Macro-block
Yang & Kot [3]	No	No	Yes	No	33×33 block
Tzeng & Tsai [5]	No	No	Yes	No	64×64 block
Lee et al. [7]	No	No	No	No	Macro-block
Proposed method	Yes	Yes	Yes	Yes	2×3 block

4.7 Summary

A new secret-sharing-based authentication method for binary images with a data recovery capability has been proposed. A cover binary image is put under protection first by being transformed into the PNG format at the scaled sample depth of 8 with an alpha channel. Next, data for authentication and image content recovery are transformed into shares by the Shamir method, which then are distributed in a designed manner into the alpha channel. The opaque effect with undesired noise visible in the resulting stego-image is eliminated by mapping the computed share values into a small range of alpha-channel values near their maximum value of 255. The result is a transparent binary PNG stego-image with uniformly distributed and nearly imperceptible noise. In the authentication process, a block is regarded as tampered if the current block content does not match the authentication data computed from the shares embedded in the alpha channel plane. For self-recovery of the content of a tampered block, a reverse Shamir scheme is used to compute the original content of the block from two shares collected from untampered blocks. By rescaling the sample depth and removing the alpha channel of the authenticated image, the original version of the 1-bit binary cover image can be regained. Experimental results have been shown to prove the effectiveness of the proposed method.

Chapter 5

A Data Hiding Method Based on Information Sharing via PNG Images for Applications of Color Image Authentication and Metadata Embedding

5.1 Introduction

Data hiding is a process of embedding information into a certain digital file that acts as a host. Through data hiding techniques, information such as authentication signals can be embedded into a digital file for the purpose of verifying the integrity or fidelity of the file [61-62]. In the application of copyright protection, an owner of a digital file can use data hiding techniques to embed a visible or invisible digital watermark into the file content to claim the ownership of the content [63-64]. Another application of data hiding is covert communication [65-66] in which people hide a secret message into a *cover file*, resulting in a *stego-file*; and a receiver of the latter can extract the hidden message from the stego-file to complete the communication.

In addition, *information sharing* was proposed to protect the security of concerned data by transforming a secret message into several *shares* which are then distributed to a number of participants to keep. Such a *secret sharing* scheme is useful for reducing the risk of incidental data loss and advantageous for keeping a balance among the participants: only when all the shares or a sufficient number of them are collected from the participants can the secret message be recovered correctly. This concept of secret sharing was proposed first by Shamir [56]. Conventionally, data hiding and information sharing are two irrelevant issues in the domain of information security. In this study, a new data hiding method based on the concept of information sharing is proposed for hiding data into PNG (portable network graphics) images.

In the *information-sharing-based* data hiding method proposed in this study, a PNG image is used as the cover image in which the alpha-channel value of each pixel is set to be 255 initially. That is, the cover image is a totally transparent color one at the beginning of the proposed data hiding process. A data string to be hidden is transformed into shares by the Shamir's secret sharing method, which is then embedded into the alpha-channel plane of the cover PNG image. Coefficient parameters involved in the Shamir method are used as *carriers* of the data to be hidden in the proposed method. A prime number used in the method, which is found to dominate the resulting visual quality and data hiding capacity of the stego-image, is properly selected. Also, a mapping function is designed for adjusting the alpha-channel values to create uniform transparency in the alpha-channel plane, resulting in an imperceptible effect in the stego-image. The original *R*, *G*, and *B* channels are untouched so that the original image appearance revealed by the color information of these three channels is kept. Fig. 5.1 illustrates these core ideas of the proposed method.

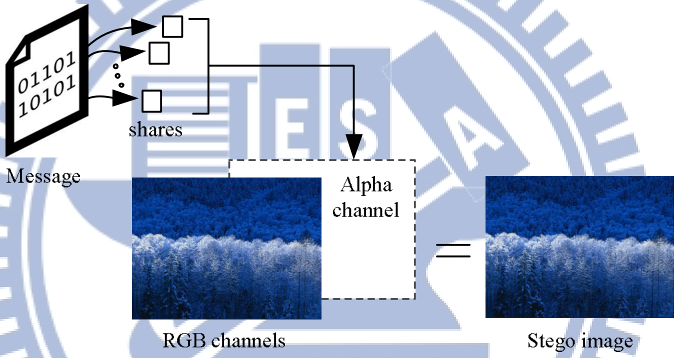


Fig. 5.1 Illustration of proposed data hiding method via PNG images.

The proposed data hiding method is suitable for applications of *image authentication* and *metadata hiding*. In particular, the application of the proposed method to image authentication is investigated in detail in this study and relevant algorithms are proposed in this chapter. In more detail, the proposed method for data hiding is based on the so-called (k, n)-threshold secret sharing method proposed by Shamir [56], where n is the number of participants in the secret sharing activity and k is a threshold specifying the minimum number of shares which should be collected to recover the secret. The detail of the method has been reviewed in aforementioned Chapter 3.2.

5.2 Merits of Proposed Method

Some other merits of the proposed method are described in the following.

(1) *Causing no destruction to the original cover image* — The proposed method manipulates the alpha-channel plane but leaves the RGB color channels untouched in the entire data hiding and extraction process, resulting in a stego-image with a destruction-free color content, which is beneficial for certain applications like metadata hiding for digital archiving.

(2) *Having the discardability of the alpha channel* — Since data are embedded into the alpha channel in the proposed method, the alpha channel can be discarded for the purpose of obtaining the original image content. In the application of image authentication, an *intact* protected image can be regained after the process of authentication. The proposed method does not leave permanent distortion to an input image as that done by conventional image authentication methods.

(3) *Fully using channels in images for data hiding* — Different from commonly-seen color images with three color channels, namely, R, G, and B, the PNG image format has a fourth channel, namely, *alpha*. By the proposed method, the alpha channel of a PNG image can be used to offer an additional data hiding channel for various applications.

5.3 Proposed Data Embedding Method

Based on the Shamir method [56] described previously, the basic idea of the proposed method for hiding a given data string M in a cover PNG image I to yield a stego-image I' is described as four major steps as follows.

1. Transform M into a sequence M' of integers.
2. Take sequentially a number of integers from M' as the values of d and c_i in Eqs. 3.(1) to compute partial shares $F(x_i)$.
3. Embed $F(x_i)$ into I by replacing some alpha-channel values of I with $F(x_i)$.

4. Repeat (2) and (3) until no integer is left in M', resulting in a stego-image I'.

A block diagram describing the processes in the proposed method is shown in Fig. 5.2, and the details of the ways we embed and extract the shares are presented as algorithms in the following.

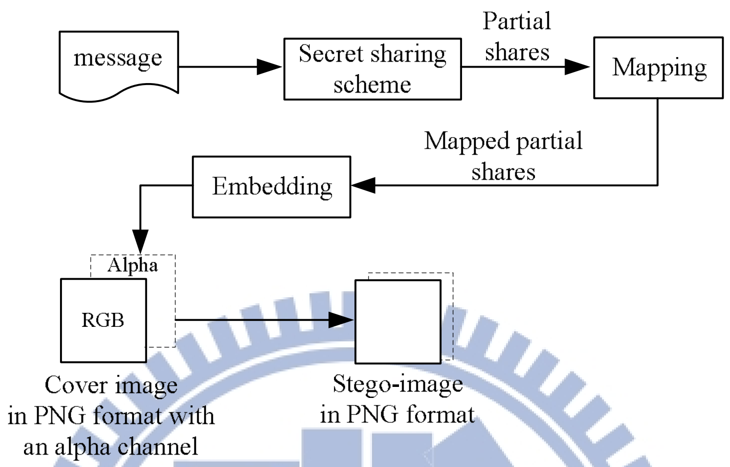


Fig. 5.2 Block diagram of proposed data hiding processes.

Algorithm 1: data embedding by secret sharing using a PNG image.

Input: a cover PNG image I and a secret message M in the form of a binary data string.

Output: a stego-image I' in the PNG format.

Steps.

Step 1. (*Initialization*) Divide M into t-bit segments with t = 3, and transform each segment into an integer, resulting in an integer sequence $M' = d_1d_2d_3...$ where $0 \le d_i \le 7$.

Step 2. (*Beginning of looping*) Take the first four elements from M' as m_1, m_2, m_3, and m_4, starting from the beginning of M'.

Step 3. (*Partial share creation*) Set p, c_i, and x_i in Eqs. (1) of Algorithm 1 described in Chapter 3.2 to be the following values:

(a) $p = 11$ (the smallest prime number larger than 7);

(b) $d = m_1$, $c_1 = m_2$, $c_2 = m_3$, and $c_3 = m_4$;

(c) $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, and $x_4 = 4$,

resulting in the following equations:

$$q_1 = F(x_1) = (m_1 + m_2x_1 + m_3x_1^2 + m_4x_1^3)_{\text{mod } p},$$

$$q_2 = F(x_2) = (m_1 + m_2x_2 + m_3x_2^2 + m_4x_2^3)_{\mathrm{mod}\,p},$$
$$q_3 = F(x_3) = (m_1 + m_2x_3 + m_3x_3^2 + m_4x_3^3)_{\mathrm{mod}\,p},$$
$$q_4 = F(x_4) = (m_1 + m_2x_4 + m_3x_4^2 + m_4x_4^3)_{\mathrm{mod}\,p}; \qquad (4)$$

and compute the partial shares of q_1 through q_4 accordingly.

Step 4. (*Mapping of partial share values*) Add 245 to each of q_1 through q_4 to form q_1', q_2', q_3', and q_4', respectively.

Step 5. (*Data embedding*) Embed q_1' through q_4' into the alpha-channel plane of I in the following way.

5.1 Take in a raster-scan order four unprocessed pixels of I and set their alpha-channel values to be q_1' through q_4', respectively.

5.2 Remove m_i through m_{i+3} from M'.

Step 6. (*End of looping*) If M' is not empty, then go to Step 2 to process the next four integers in M'; otherwise, take the final I as the desired stego-image I'.

The above algorithm can be regarded as a (4, 4)-threshold secret sharing method. The possible values of q_1 through q_4 yielded by Eqs. (4) in Step 3 of the above algorithm and inserted in the alpha channels of I are between 0 and 10 because the value of p used in Eqs. (4) is 11. After performing Step 4 of the algorithm, the values of q_1' through q_4' form a small range of integer values from 245 to 255 which are then embedded into the alpha channels of the cover image I. The distribution of the alpha-channel values within such a small range of values means that very similar values appear everywhere in the alpha channels, resulting in a nearly *uniformly transparent* PNG image, as desired.

Also, it can be seen from the algorithm that every four 3-bit segments of the secret data string are embedded into the alpha-channel values of four pixels of the cover image I to yield the stego-image I'. This means that if the size of the cover image is S, then the data hiding capacity is $R = (4\times3)\times(S/4) = 3S$ bits. This is for the case of $t = 3$ where t is as mentioned in Step 1. More generally, if every four t-bit segments are transformed and embedded similarly, then it is easy to figure out that $R = (4\times t)\times(S/4) = tS$ bits, which means that the data hiding capacity is proportional to the chosen value of t. Since S is the dimension of the cover image, this capacity of tS is large in general.

However, it should be noted that the larger the value of t is chosen to be, the lower the visual quality of the stego-image will become. The reason is that a larger value of t, according to Step 2 of Algorithm 1 described in Chapter 3.2, implies that a larger value of p is chosen, and so the possible values of q_1 through q_k, according to Step 3 of Algorithm 1 of Chapter 3.2, will be spread in a larger range of values from 0 through $p - 1$ due to the use of the mod-p operation. This will cause a wider range of alpha-channel values even after the value mapping of Step 4 of Algorithm 1 described above is conducted. This wider alpha-channel value range in turn leads to a more obvious non-uniform transparency effect appearing on the stego-image. This also explains the reason why we segment, in Step 1 of Algorithm 1, the message M into segments of $t = 3$ bits for use in Eqs. (4), which is a compromise between the resulting data hiding capacity and stego-image quality according to our experimental experience. Of course, if a higher image quality of the stego-image is required, we may use a smaller t like $t = 2$ at the sacrifice of the data hiding capacity.

In addition, it is noted that Algorithm 1 takes every *four* integer numbers of the string M' each time and embeds them into the alpha channels of *four* pixels of the cover image. It is not difficult to figure out that the algorithm can be *generalized* to take n integers each time and embeds them into n pixels. For this, just modify part of Step 3 to be as follows:

…

(b) $d = m_1$, $c_1 = m_2$, $c_2 = m_3$, …, $c_n = m_n$;

(c) $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, …, $x_n = n$,

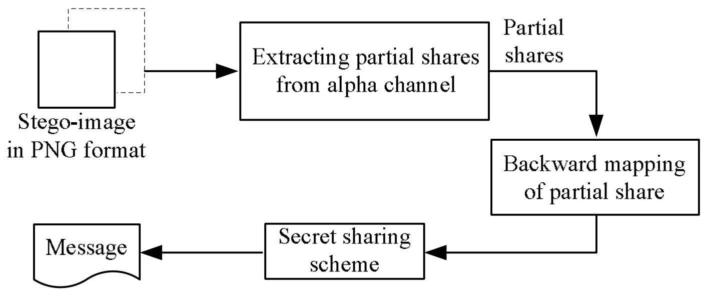
resulting in the following equations:

$$q_1 = F(x_1) = (m_1 + m_2x_1 + m_3x_1^2 + ... + m_nx_1^n)_{\text{mod } p},$$
$$q_2 = F(x_2) = (m_1 + m_2x_2 + m_3x_2^2 + ... + m_nx_2^n)_{\text{mod } p},$$
$$q_3 = F(x_3) = (m_1 + m_2x_3 + m_3x_3^2 + ... + m_nx_3^n)_{\text{mod } p},$$
$$...$$
$$q_n = F(x_n) = (m_1 + m_2x_n + m_3x_n^2 + ... + m_nx_n^n)_{\text{mod } p}. \tag{4'}$$

The resulting generalized algorithm yields, as can be figured out again, a data hiding capacity of $R = (n\times t)\times(S/n) = tS$ bits which is identical to the original algorithm.

5.4 Proposed Data Extraction Method

Now, the process of hidden data extraction is described in the following. A block diagram describing the proposed data extraction processes is shown in Fig. 5.3.



Stego-image in PNG format
Extracting partial shares from alpha channel
Partial shares
Backward mapping of partial share
Secret sharing scheme
Message

Fig. 5.3 Block diagram of proposed data extraction processes.

Algorithm 2: data extraction from a stego-image.

Input: a stego-image I' created by Algorithm 1 in the PNG format.

Output: the binary data string M hidden in I'.

Steps.

Step 1. (*Initialization*) Create an empty string M.

Step 2. (*Beginning of looping*) Take in a raster-scan order four alpha-channel values q_1', q_2', q_3', and q_4' from I'.

Step 3. (*Backward mapping of partial share values*) Subtract 245 from each of q_1' through q_4' to obtain q_1 through q_4, respectively.

Step 4. (*Data extraction*) Perform the secret recovery process described by Algorithm 2 of Chapter 3.2 to extract four integers m_1 through m_4 hidden in q_1 through q_4, transform m_1 through m_4 into binary numbers, and append them in a sequential order to the end of M.

Step 5. (*End of looping*) If all shares embedded in I' are processed, then take the final M as output; otherwise, go to Step 2.

Similarly to generalization of Algorithm 1 as mentioned previously, Algorithm 2 above may also be generalized to take care of data extraction from images yielded by the generalized version of Algorithm 1. The details are simple and so omitted.

5.5 Application of Proposed Method to Image Authentication

A possible application of data hiding is *image authentication*. To implement image authentication by the proposed data hiding method, an input image with RGB channels (like a BMP image) to be protected is transformed first into a PNG image with the alpha-channel values all set initially to be 0. The PNG image is processed next to generate color-dependent authentication signals, which are then embedded into the alpha channel by using the previously-proposed data hiding method to yield the resulting stego-image. Then, in the later authentication signal verification process, the embedded authentication signals are extracted from the alpha channel plane of a stego-image and verified against those computed from the color values of the pixels of the image. A pixel with mismatched signals is regarded as being tampered with. Fig. 5.4 illustrates the processes of authentication signals generation and embedding in the proposed color image authentication method, and Fig. 5.5 illustrates the image verification processes of the proposed method. More details are described as two algorithms (Algorithms 3 and 4) in the following.

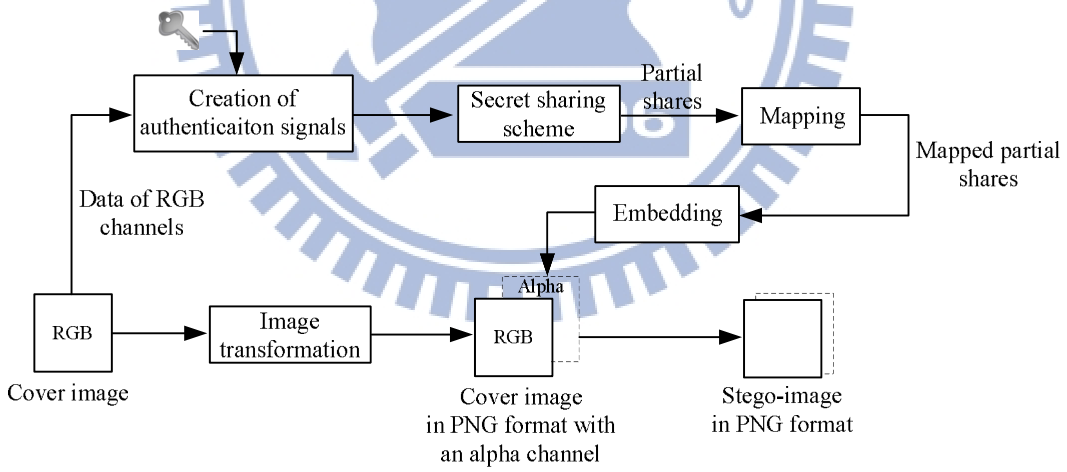


Fig. 5.4 Block diagram of generating a stego-image with authentication signals.

Algorithm 3: authentication signal generation and embedding.

Input: a PNG image I and a key K.

Output: a protected image I' with authentication signals embedded.

Steps.

Step 1. (*Initialization and beginning of looping*) Take in a raster-scan order three pixels p_1, p_2, and p_3 from I, and use K as the seed for a random number generator to get a number sequence $S = s_1, s_2, \ldots$.

Step 2. (*Creation of authentication signals*) For each of p_1, p_2 and p_3, denoted as p_i, perform the following steps.

2.1 Take the R, G, and B values of p_i, denoted as V_R, V_G, and V_B, respectively.

2.2 Take in order three elements from S, denoted as s_j, s_{j+1}, and s_{j+2}, to perform the following operations to obtain three bits r, g, and b, respectively:

$$(V_R + s_j)_{\text{mod } 2} = r;$$
$$(V_G + s_{j+1})_{\text{mod } 2} = g;$$
$$(V_B + s_{j+2})_{\text{mod } 2} = b.$$

2.3 Concatenate r, g, and b as a 3-bit string rgb, and transform it into a decimal integer as the authentication signal D_i for p_i.

Step 3. (*Authentication signal embedding into three pixels*) Perform the previously-described generalized version of Algorithm 1 to embed the three authentication signals D_1, D_2, and D_3 into three pixels of I, except that in the step of *mapping of partial share values* (Step 4) in Algorithm 1, 244 instead of 245 is added to each partial share value.

Step 4. (*End of looping*) If there exists any unprocessed pixel in I, then go to Step 2; otherwise, take the final I as the desired stego-image I'.

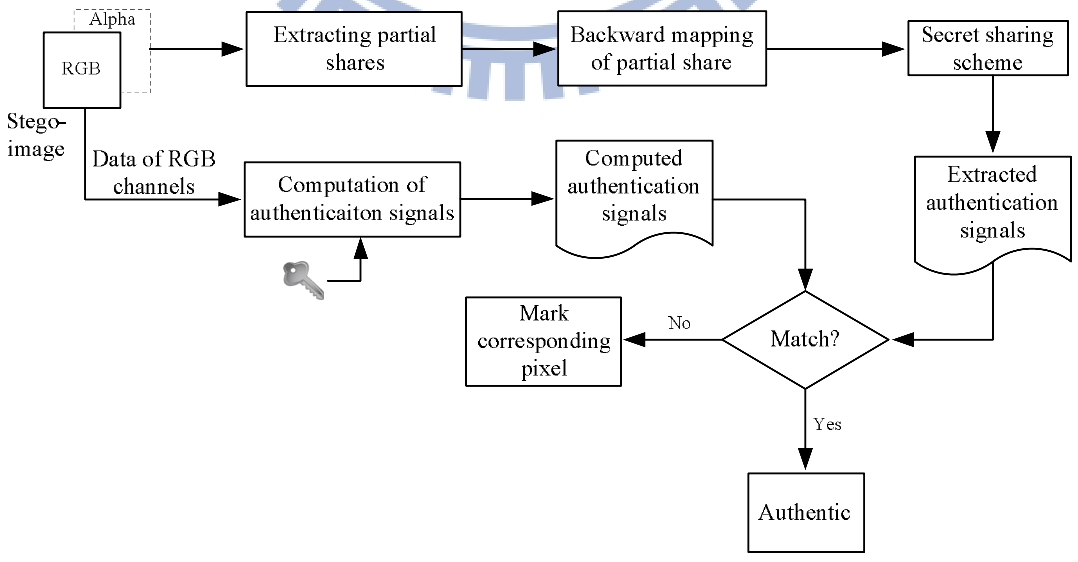

Alpha
RGB
Stego-image
Extracting partial shares
Backward mapping of partial share
Secret sharing scheme
Data of RGB channels
Computation of authenticaiton signals
Computed authentication signals
Extracted authentication signals
Mark corresponding pixel
No
Match?
Yes
Authentic

Fig. 5.5 Block diagram of verifying a stego-image.

Algorithm 4: authentication signal verification.

Input: a protected stego-image I' created by Algorithm 3, and a key K used there.

Output: image I' with altered pixels being marked if found.

Steps.

Step 1. (*Initialization and beginning of looping*) Take in a raster-scan order three pixels p_1, p_2 and p_3 from I'; let their alpha-channel values be denoted as q_1', q_2', and q_3', respectively; and use K as the seed for a random number generator to generate a random number sequence $S = s_1, s_2, \ldots$.

Step 2. (*Extraction of authentication signals embedded in alpha channels*) Perform the previously-described generalized version of Algorithm 2 with q_1', q_2', and q_3' as input to extract the authentication signals D_1', D_2', and D_3'cept that in the step of *backward mapping of partial share values* (Step 3) in Algorithm 2, 244 instead of 245 is subtracted from each of q_1' through q_3'.

Step 3. (*Computation of authentication signals from pixels' color values*) For each of p_1, p_2 and p_3, denoted as p_i, perform the following steps.

3.1 Take the R, G, and B values of p_i, denoted as V_R, V_G, and V_B, respectively.

3.2 Take in order three elements from S, denoted as s_j, s_{j+1}, and s_{j+2}, to perform the following operations to obtain three bits r, g, and b, respectively.

$$(V_R + s_j)_{\text{mod } 2} = r;$$
$$(V_G + s_{j+1})_{\text{mod } 2} = g;$$
$$(V_B + s_{j+2})_{\text{mod } 2} = b.$$

3.3 Concatenate r, g, and b as a 3-bit string rgb, and transform it into a decimal integer as the authentication signal D_i for p_i.

Step 4. (*Matching of extracted and computed authentication signals*) Match D_1, D_2, and D_3 with D_1', D_2', and D_3', respectively, and if there exists any mismatched pair, then mark the corresponding pixel as being tampered with in the input image I'.

Step 5. (*End of looping*) If there exists any unprocessed pixel in I', then go to Step 2; otherwise, take the final I', possibly marked, as output.

5.6 Security Consideration

The secret key K used in Step 2.2 of Algorithm 3 provides a measure to protect the authentication signals to be counterfeited. More specifically, since three bits consist of an authentication signal for each pixel, the probability of correctly guessing an authentication signal for each pixel is 1/8. To enhance further the security of the proposed method, an additional measure is adopted but not included in the above algorithms for clarity of algorithm descriptions. It is *randomization* of the constant values of x_1 through x_n used in Step 3 of the generalized version of Algorithm 1 within the allowed integer range of $0 \le x_i < p$ [56] with the help of another secret key. Then, the probability of correctly guessing values of x_1 through x_3 used in a set of three pixels can be figured out to be

$$1/[p \times (p - 1)\times\ldots\times(p - n + 1)] = 1/[11\times(11 - 1)\times(11 - 3 + 1))] = 1/990 \approx 0.1\%.$$

As a result, the possibility of correctly guessing all these values used in a stego-image of size S is $1/[p\times(p - 1)\times\ldots\times(p - n + 1)]^{S/n}$ which is small enough in general cases. Through the use of above security measures, it is almost impossible for an attacked color stego-image with counterfeit authentication signals to pass the proposed image authentication processes.

5.7 Experimental Results

A. Experimental results of data hiding using three test color images

A lot of experiments have been conducted to test the proposed algorithms on a set of color images. Some results using three test images, named Lena, Jet, and Tiffany, as given in Figs. 5.6(a), 5.6(c), and 5.6(e), respectively, are shown here. The results of applying the proposed method using Algorithm 1 to embed a long sequence of binary message data into the three images are shown in Figs. 5.6(b), 5.6(d), and 5.6(f), respectively. As can be seen from the figures, the stego-images are visually almost identical to the cover images, respectively, although the alpha-channel contents of the stego-images include embedded message data. Note that in Algorithm 1, the value of t, which is the number of bits taken as a segment and transformed into an integer for use in the secret sharing process performed by Algorithm 1, was taken to be 3.

Next, we show the effect of choosing different values of t in Step 1 of Algorithm 1. Recall that the data hiding capacity has been computed to be $R = tS$ where S is the size of the cover image. Although accordingly *t* may be chosen to be the maximum of 7 to increase the data hiding capacity, the yielded stego-image quality degrades very much, as illustrated by the results shown in Fig. 5.7, where Fig. 5.7(a) is the stego-image of Lena yielded by Algorithm 1 with *t* taken to be 7 which includes, as can be seen, a lot of white noise. For comparison, a better result of using $t = 3$ shown in Fig. 5.6(b) is repeated in Fig. 5.7(b) in which the white noise created by the algorithm is almost imperceptible, as mentioned previously.

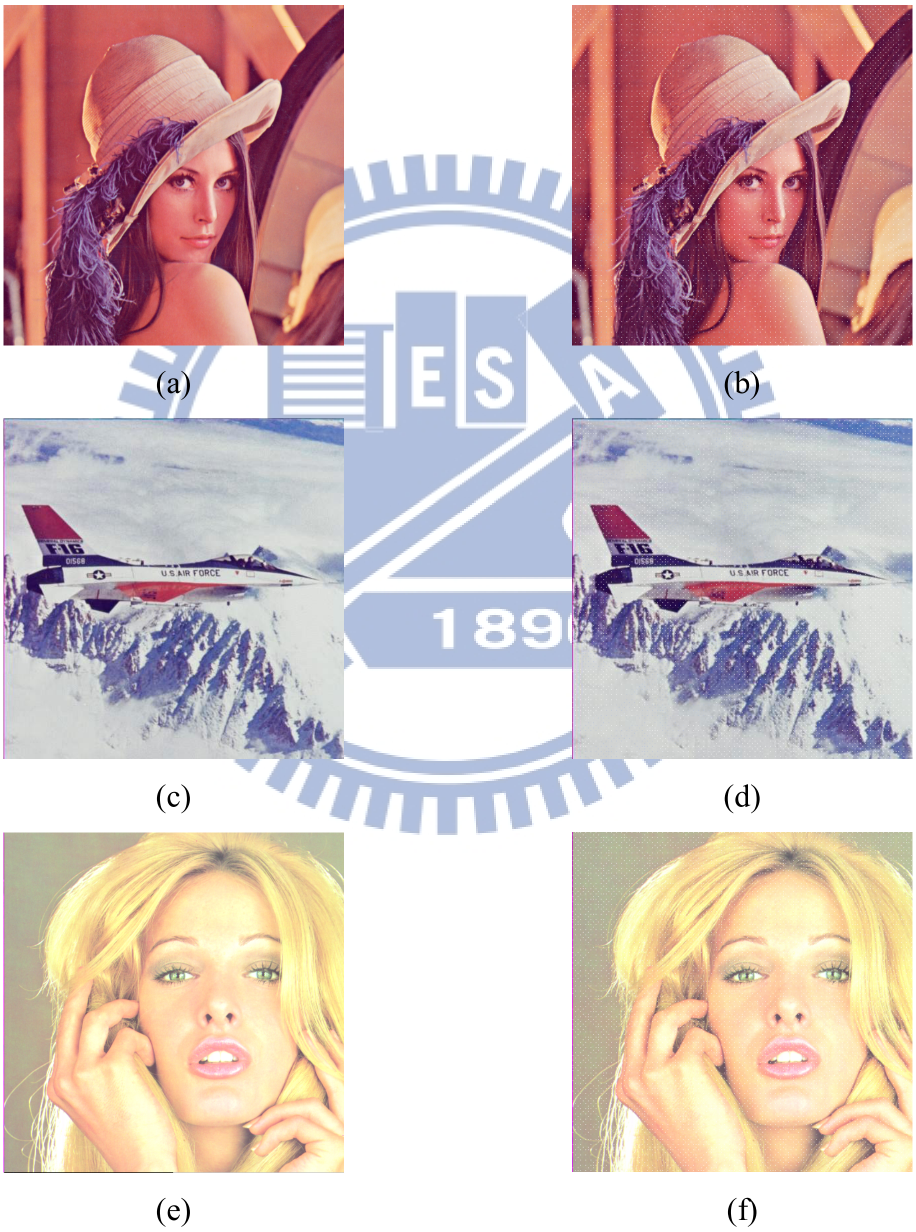


Fig. 5.6 Results of applying Algorithm 1 to embed a long sequence of binary message data into three images. (a) Cover image Lena. (b) Stego-image of Lena. (c) Cover image Jet. (d) Stego-image of Jet. (e) Cover image Tiffany. (f) Stego-image of Tiffany.

Furthermore, we show in more detail the data hiding capacities and the corresponding stego-image qualities for all possible values of t = 1, 2, …, 7 for the three images Lena, Jet, and Tiffany used as cover images. Specifically, in Table 5.1 we show, in addition to the data hiding capacities, the qualities of the alpha-channel planes of the three corresponding stego-images in terms of the PSNR measure; and in Figure 5.8, we show the stego-images corresponding to t = 1, 2, …, 7. The PSNR values of the alpha-channel planes of the stego-images were computed with the original alpha-channel values in the cover images being taken to be all 255. As can be seen, when t = 3, although the PSNR is at the level of 32.69 which is not high enough, yet the corresponding stego-image quality shown by Fig. 5.8(c) is still visually good. In addition, the corresponding data hiding capacity is $R = tS = 3\times512\times512$ = 75K = 786432 bits, which is good enough for applications.

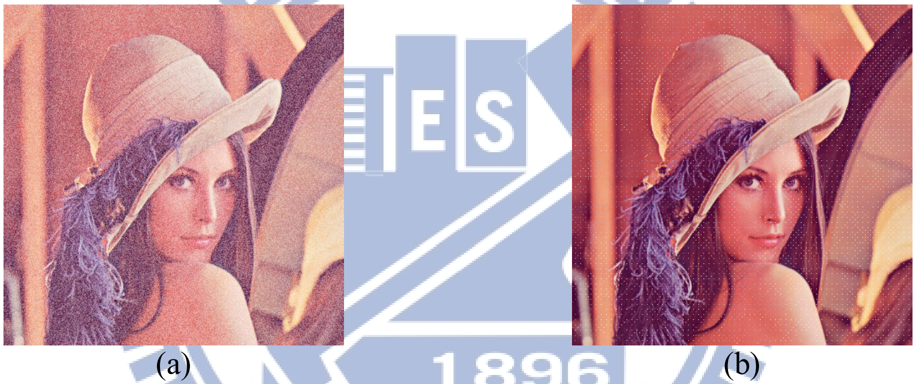


Fig. 5.7 An example of experimental results showing compromise between data hiding capacity and stego-image quality. (a) Stego-image of Lena yielded by choosing t = 7 with more data hidden but showing more white noise coming from the alpha channel. (b) Stego-image of Lena yielded by choosing t = 3 with less data hidden but with almost no noise due to nearly total transparency in the alpha channel.

Table 5.1 Data hiding capacities and stego-image qualities for t = 1 through 7 (DHC = data hiding capacity; PSNR = peak of signal to noise ratio).

	Lena			Jet			Tiffany		
t value	DHC (bits)	PSNR of alpha channel (dB)	PSNR of RGB channels (dB)	DHC (bits)	PSNR of alpha channel (dB)	PSNR of RGB channels (dB)	DHC (bits)	PSNR of alpha channel (dB)	PSNR of RGB channels (dB)
t = 1	262,144	45.44	∞	262,144	45.44	∞	262,144	45.44	∞
t = 2	524,288	40.34	∞	524,288	40.34	∞	524,288	40.34	∞
t = 3	786,432	32.69	∞	786,432	32.69	∞	786,432	32.69	∞
t = 4	1,048,576	28.68	∞	1,048,576	28.68	∞	1,048,576	28.68	∞
t = 5	1,310,720	21.72	∞	1,310,720	21.72	∞	1,310,720	21.72	∞
t = 6	1,572,864	16.74	∞	1,572,864	16.74	∞	1,572,864	16.74	∞
t = 7	1,835,008	10.61	∞	1,835,008	10.61	∞	1,835,008	10.61	∞

In fact, it can be seen from Fig. 5.8(d) that even when *t* is taken to be 4, the stego-image quality is still acceptable with the data hiding capacity increased to 100K bits. On the contrary, if image quality is the most serious concern, then *t* may be reduced to 2 or even 1 at the sacrifice of the data hiding capacity. Note that the data hiding capacity is related to the *t* value only; it is independent of the cover image content. It is also noted that, different from other methods, the channels of *R*, *G*, and *B* of the cover image are not processed by the proposed method, yielding a lossless result in the color channels.

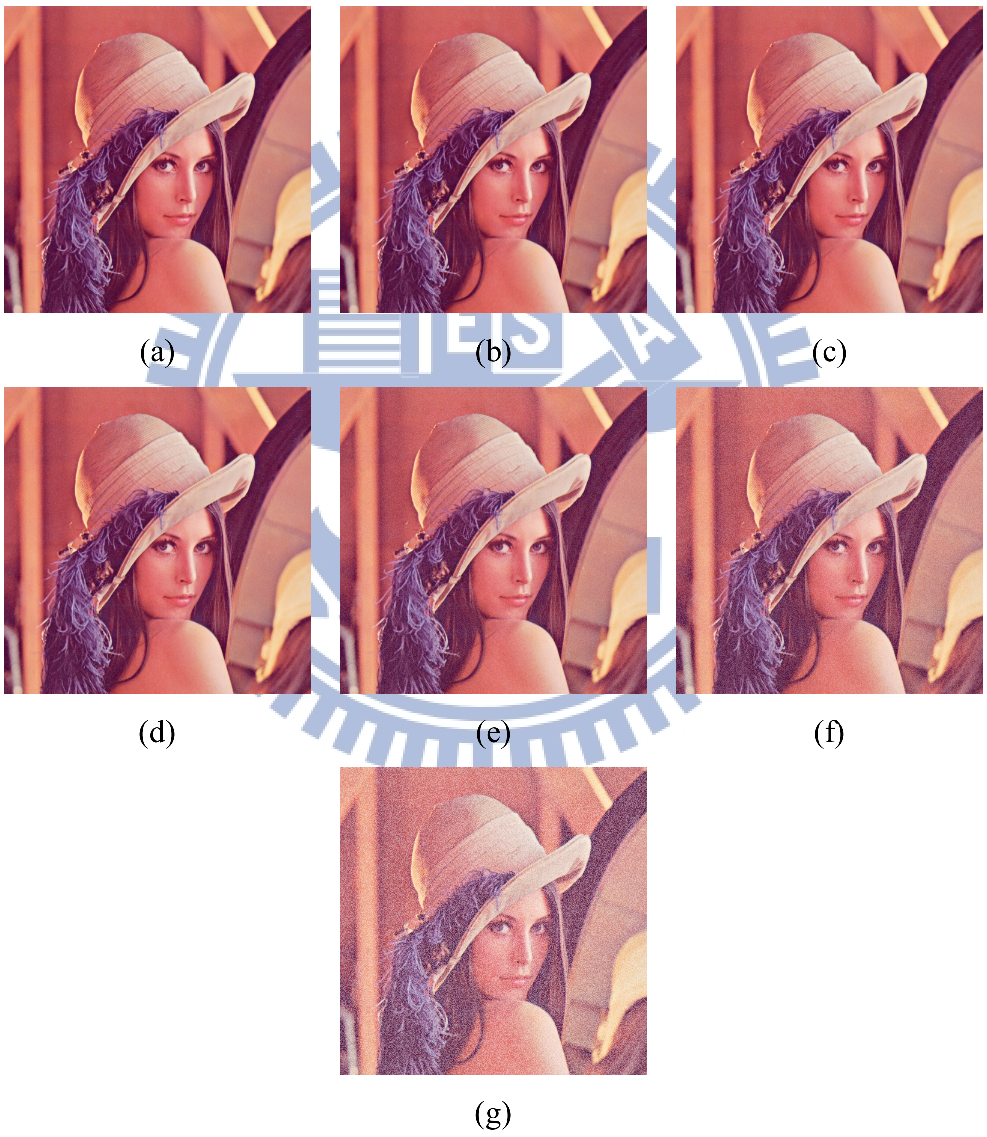


Fig. 5.8 Stego-images yielded by Algorithm 1 for t = 1 through 7. (a) through (g) correspond to t = 1, 2, …, 7, respectively.

B. Experimental results of color image authentication

In this section, we show some experimental results of applying the proposed image authentication algorithms (Algorithms 3 and 4) to authenticate stego-images attacked by two common image editing operations, i.e., superimposing and painting. It was found that if the superimposing operation is used in the attack, the alpha channel values will be replaced with the new value 255 at the attacked part. Since the largest alpha channel value generated by the proposed method is 254 (see Step 3 in Algorithm 3), attacked pixels can be easily detected and marked by checking the existence of the specific value 255 in the alpha channel. As an example, Fig. 5.9(a) shows a cover image "Tiffany" and Fig. 5.9(b) is the stego-image of Tiffany. In Fig. 5.9(c), the stego-image was attacked by superimposing a fake mouth on the face. Fig. 5.9(d) shows the authentication result in which all altered pixels were detected and marked in black. The statistics corresponding to the experiment is shown in Table 5.2. As can be observed, both the false acceptance ratio and false rejection ratio are 0% for the reason that, as mentioned previously, alpha channel values 255 only occur at attacked pixels.

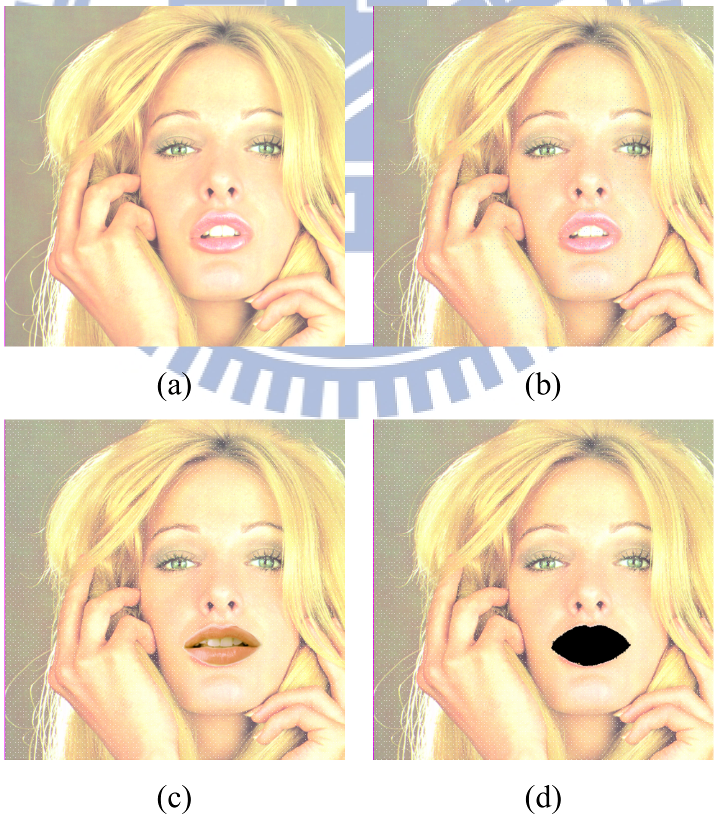


Fig. 5.9 Authentication result of an attacked stego-image of Tiffany with a pasted fake mouth. (a) Cover image. (b) Stego-image with authentication signals. (c) Modified stego-image. (d) Result with altered pixels detected and marked in black.

Another example of a tampered image attacked by superimposing a rose to the hair part of Tiffany is shown in Fig. 5.10(b). The authentication result is shown in Fig. 5.10(c) in which all pixels consisting of the added rose were successfully detected and marked in black. The corresponding statistics is also given in Table 5.2.

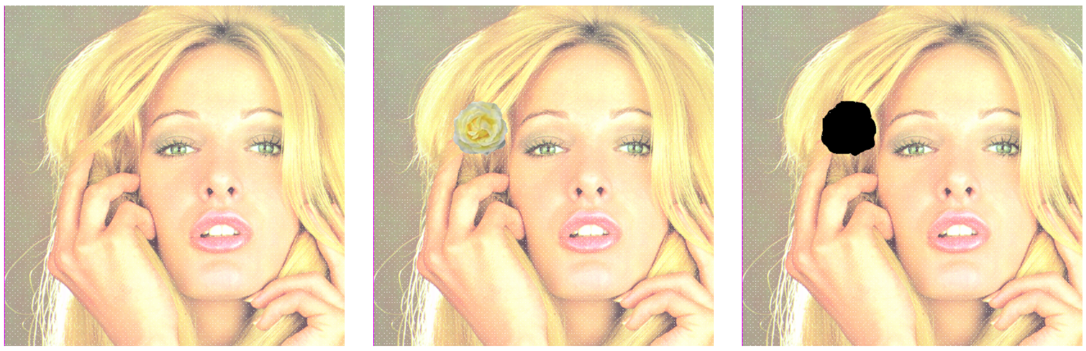
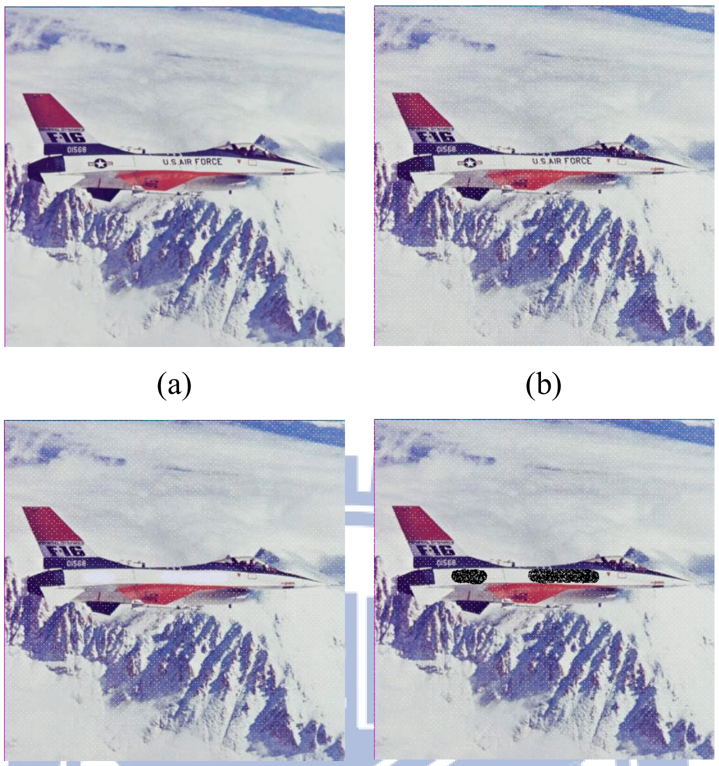


Fig. 5.10 Authentication result of an attacked stego-image of Tiffany with an added rose. (a) Stego-image with authentication signals. (b) Modified stego-image. (c) Result with altered pixels detected and marked in black.

Table 5.2 Statistics of experimental results.

Experimental result (image size = 512×512)	No. of tampered pixels	No. of detected blocks (detection ratio)	false acceptance ratio	false rejection ratio
Exp. 1 shown in Fig. 5.9	5886	5886 (100%)	0%	0%
Exp. 2 shown in Fig. 5.10	5207	5207 (100%)	0%	0%

Some experimental results of using painting operations to attack stego-images are shown in Fig. 5.11. Specifically, Figs. 5.11(a) and 5.11(b) are respectively an input cover image "jet" and a generated stego-image. In Fig. 5.11(c), a logo and words printed on the body of the plane were smeared by painting color similar to the plane body. The authentication result generated from Algorithm 4 is shown in Fig. 5.11(d) in which tampered regions were successfully detected and marked in black. However, as can be seen, some tampered pixels were not detected and appeared as noise in the marked region. This phenomenon results from the case that the authentication signals extracted from the alpha channel incidentally match the authentication signals computed from the tampered pixels. This is also the reason why the false acceptance ratio exists. It is noted that the alpha channel content keeps intact after the painting operation and so the extracted authentication signals are always true, yielding the false rejection ratio is 0%. Related statistics of the experiment is given in Table 5.3.


(a) (b)

(c) (d)

Fig. 5.11 Authentication result of an attacked stego-image of jet with the smeared logo and words. (a) Stego-image with authentication signals. (b) Modified stego-image. (c) Result with altered pixels detected and marked in black.

Since the authentication signal of each pixel is composed of three bits, there is a probability of 1/8 for an erroneous authentication, leading to a false acceptance ratio of around 12.5%. As an example, Fig. 5.12(b) shows that the stego-image of jet was tampered with by smearing the entire plane out of the image content. The authentication result is shown in Fig. 5.12(c) in which 85.02% of tampered pixels were detected and marked in black. In other words, 14.98% of tampered pixels incidentally passed the authentication process, which meets the probabilistic expectation of around 12.5% authentication misses. The corresponding statistics is also given in Table 5.3.

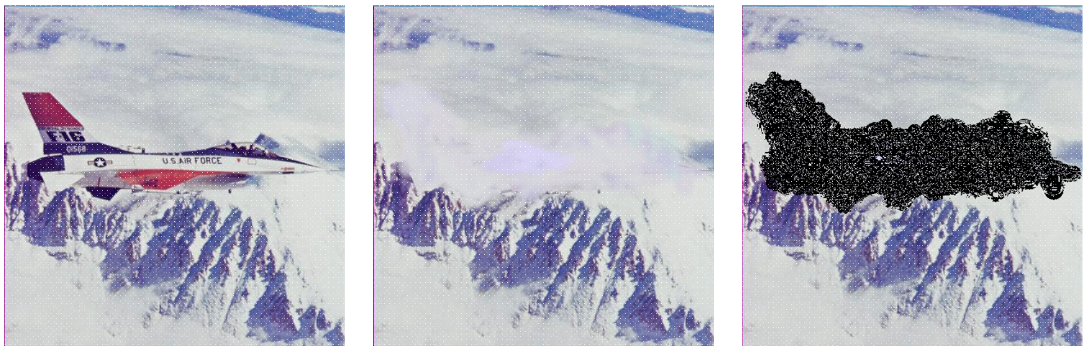


Fig. 5.12 Authentication result of an attacked stego-image in which the jet has been smeared. (a) Stego-image with authentication signals. (b) Modified stego-image. (c) Result with altered pixels detected and marked in black.

Table 5.3 Statistics of experimental results.

Experimental result (image size = 512×512)	No. of tampered pixels	No. of detected blocks (detection ratio)	false acceptance ratio	false rejection ratio
Exp. 3 shown in Fig. 5.11	5886	5379 (91.39%)	8.61%	0%
Exp. 4 shown in Fig. 5.12	57037	48491 (85.02%)	14.98%	0%

C. Comparison with existing methods

Several existing authentication methods related to color images are taken to compare with the proposed method for the purpose of presenting contributions made in this study. A comparison table in terms of important capabilities is given in Table 5.4. As can be observed, since the proposed method makes use of the alpha channel instead of conventional illumination [65] or RGB [67-68] channels for embedding authentication signals, the proposed method is the only one which has the characteristic of losslessness of the image content.

Both the method in [68] and the proposed method can localize tampered parts at the pixel-level; however, the localization capability only works when the alteration occurs in the blue channel in [68]. This fact results from that the method of [68] uses the blue channel for embedding authentication signals generated from channels of red and green. As a result, when the image content in the red and green channels altered, the method of [68] is able to detect the existence of the attack but unable to localize alterations precisely. Moreover, it is worth to note that the method of [65] embeds watermarks in the illumination channel and leaves chrominance channel untouched. Therefore, modifications involved in illumination can be detected efficiently but those made in the chrominance channel will be neglected. In contrast, the proposed method, which generates authentication signals from data of the RGB channels, provides high sensitivity to alterations occurring in each channel.

At last, the method in [65] allows reasonable image processing such as JPEG compression without raising false positive alarms and is practical for some applications. In this aspect, the proposed method yields a stego-image in the PNG format which in normal cases will not be compressed further, reducing the possibility of erroneous authentication caused by incidental image compression.

Table 5.4 Comparison of existing color image authentication methods.

	Losslessness of the image content	Tampering localization capability	Level of localization precision	detection of chrominance modification	Channel of containing embedded data	Tolerance of incidental image compression
Lu & Liao [65]	No	Yes	Block-level	No	Illumination channel	Yes
Peng & Liu [67]	No	Yes	Block-level	Yes	RGB channels	No
Byun et al. [68]	No	Only when alteration made in the blue channel	Pixel-level	Yes	Blue channel	No
Proposed method	Yes	Yes	Pixel-level	Yes	Alpha channel	Free from the misgiving

5.8 Summary

A new type of data hiding via PNG images based on information sharing has been proposed. The Shamir's secret sharing method is used first in a novel way to generate partial shares from a given data string. The alpha-channel plane of a cover PNG image is utilized next to embed the partial shares, yielding a stego-image with undesirable white noise. The white noise is then eliminated skillfully by choosing a small prime number, dividing the input data string into 3-bit segments, and mapping computed share values into a range of large alpha-channel values which create high transparency. Generalization of the method to allow compromise between the resulting data hiding capacity and stego-image quality has also been proposed.

Moreover, detailed algorithms for applying the proposed method to color image authentication have been proposed. Shown by the results is the applicability of the proposed authentication algorithms to detect attacks implemented by common image-content alternation operations. Good experimental results prove the effectiveness of the proposed methods in the aspects of tampering detection ratio, characteristic of losslessness, and false acceptance and rejection ratios. Future studies may be directed to applications of the proposed method to copyright protection, digital rights management, recovery of altered image contents, etc.

Chapter 6
An Optimal Pixel-level Self-repairing Authentication Method for Grayscale Images under a Minimax Criterion of Distortion Reduction

6.1 Introduction

With the era of cloud computing coming, data stored originally in personal computers mostly will eventually be moved to and processed in powerful servers at far ends. However, how can one be sure that personal data accessed from cloud servers are intact? Undoubtedly, this problem of data security has become a significant issue in the age of cloud computing.

This study explores the security issue of keeping digital image data. The use of an *image authentication* technique provides a solution to this issue. A new grayscale image authentication method is proposed in this paper. By embedding *fragile* authentication signals into a *cover image* to be protected to create a *stego-image*, illicit modifications made to the stego-image may be localized to the pixel-level precision by the proposed method such that the integrity and fidelity of the original image content can be checked.

6.2 Merits of Proposed Method

The proposed method has at least four merits. (1) First, different from other methods [71-72] which generate the authentication signal and the repairing data as two separate items, the proposed method uses the above-mentioned *single* bin code to function as the two items *simultaneously*, leading to use of less storage for embedding these data in the image. (2) The use of less storage leads further to the possibility of conducting more precise pixel-level authentication because it becomes now possible to allow every pixel to include the pixel authentication signal (saved as the three LSBs) in addition to the original pixel content (kept in the five MSBs). Note that most related methods with data repairing capabilities authenticate images at the block level

[73-75], yielding coarser tampering localization and data repairing results. (3) Furthermore, a secret key is used in the proposed method for randomly choosing pixels for embedding the generated authentication signals, thus increasing the security of the stego-image yielded by the proposed method. (4) Finally, because of the first merit of using less storage for authentication and repairing data mentioned previously, the proposed method is *blind* [76] in nature — no information other than the image itself is needed for conducting the data repairing process. Note that the methods of [10, 69, 70, 71, 77] need to know the prior information of the hidden digital signatures or watermarks [76] used in the authentication process. Besides, extra information like codebooks or other overhead data is required in some existing methods with data repairing capabilities [69, 71, 75].

6.3 Proposed Method for generation of a stego-image

A. Authentication Signal Generation and Embedding

In the proposed method for grayscale image authentication and self-repairing, the 8-bit gray value g of each pixel in an input image is divided into two parts — the five MSBs of g and the remaining three LSBs. The former is used to generate an authentication signal for the pixel itself, with the signal also working as an index for generating the data for repairing the pixel's gray value when the pixel is authenticated to have been tampered with. The five MSBs *ideally* are expected to be embedded *directly* in a randomly-selected pixel elsewhere and can be retrieved later for use in the two previously-mentioned purposes of authentication signal and repairing data generations. However, due to the limited data hiding capacity in the image, it is difficult to embed the large-volume data consisting of such MSBs of all the pixels into the input image; and even if they could be embedded, noticeable distortion would be created. Consequently, we propose in this study to use a *bin-mapping* scheme for the purpose of compressing these MSB data before embedding them. Specifically, we map the gray-value range specified by the five MSBs into eight equal-length intervals called *bins*, with each bin being indexed by an integer called a *bin number*, or equivalently, by a 3-bit binary number, called a *bin code*. The eight bins and their corresponding bin numbers and bin codes are shown in Table 6.1. The bin code of each pixel is then taken as the authentication signal of the pixel and embedded into the three remaining LSBs (the previously-mentioned second part) of *another* pixel randomly chosen by a pre-selected secret key. An illustration of these ideas of authentication signal (bin code) generation and embedding is given in Fig. 6.1, and the detail is described as an algorithm in the following.

Table 6.1 Bins, bin numbers, bin codes, and representative values of bins used in this study.

Bin (an interval)	Bin number (an integer)	Bin code (a binary number)	Representative value of bin
[0, 3]	0	000	2
[4, 7]	1	001	6
[8, 11]	2	010	10
[12, 15]	3	011	14
[16, 19]	4	100	18
[20, 23]	5	101	22
[24, 27]	6	110	26
[28, 31]	7	111	30

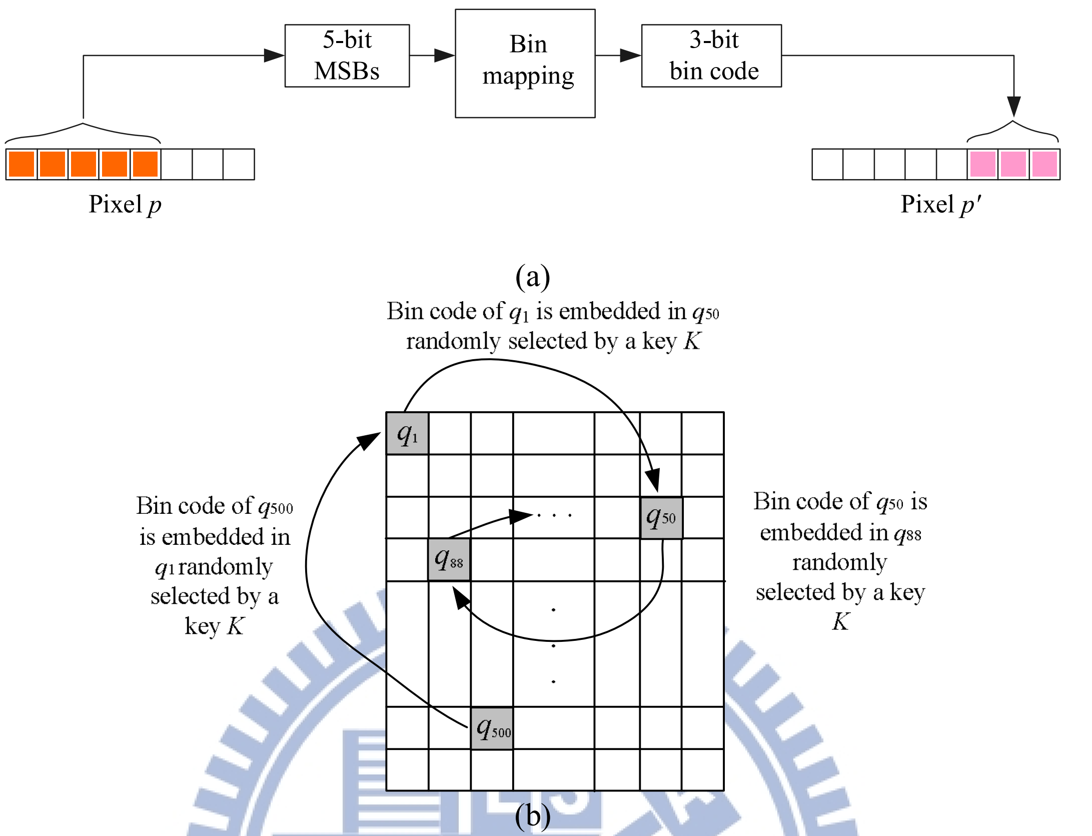

5-bit MSBs
Bin mapping
3-bit bin code
Pixel p
Pixel p'
(a)
Bin code of q_1 is embedded in q_{50} randomly selected by a key K
Bin code of q_{500} is embedded in q_1 randomly selected by a key K
Bin code of q_{50} is embedded in q_{88} randomly selected by a key K
q_1
q_{50}
q_{88}
q_{500}
(b)

Fig. 6.1 Illustration of bin code (authentication signal) generation and embedding. (a) Mapping 5-bit MSBs to a 3-bit bin code. (b) Bin codes embedded into pixels randomly selected by a secret key K.

Algorithm 1: authentication signal generation and embedding.

Input: a grayscale cover image I, a random number generator f, and a secret key K.

Output: a stego-image I_s with authentication signals embedded.

Steps.

Step 1. (*Beginning of looping*) In a raster-scan order, select a pixel p from the image I.

Step 2. (*Authentication signal generation and embedding*) Perform the following steps to generate an authentication signal for p and embed it into another randomly-selected pixel.

1.1 Transform the gray value of p into eight bits, b_7, b_6, …, b_0.

1.2 Transform the five MSBs, b_7, b_6, …, b_3, of p into an integer d.

1.3 Map the integer d into a bin indexed by a bin number B computed by the function $B = \lfloor d/4 \rfloor$ where $\lfloor \cdot \rfloor$ specifies the integer floor function.

1.4 Transform B into a 3-bit bin code $s = c_2c_1c_0$ for use as the authentication signal for p.

1.5 Select randomly a pixel p' in I other than p using the input random number generator f with the input key K as the seed, and regard pixel p' as *corresponding to p*.

1.6 Embed the 3-bit authentication signal $s = c_2c_1c_0$ of p into p' by replacing the three LSBs of p' with s.

Step 3. (*End of looping*) If there remain unprocessed pixels in I, then go to Step 1; otherwise, take the final I as the desired stego-image I_s.

In the above algorithm, Steps 2.2 through 2.4 are used to show how the concept of bin mapping of our method is applied. In practice, these steps may be reduced to be simply as follows for use in real applications.

2.2 Take the 3 MSBs $b_7b_6b_5$ of p to yield a bin code denoted as $s = c_2c_1c_0$.

6.4 Proposed Image Authentication and Recovery Process

During the image authentication process, an authentication signal is computed from the five MSBs of every pixel p. Also, the authentication signal embedded in the three LSBs of the pixel p' corresponding to p, which was randomly selected previously in Algorithm 1, is retrieved. The two authentication signals then are compared with each other. If mismatching occurs, pixel p is regarded as having been tampered with. In this case, we use again the three LSBs of pixel p', which is also the bin code of p, as an index to generate a data item for use in repairing the tampered gray values of p. The generated data item is taken to be the middle value of the bin indexed by the bin code, which is called the *representative value* of the bin and denoted by M. Specifically, M is computed as $M = \lceil (a + b)/2 \rceil$ for a bin with range $[a, b]$ where $\lceil \cdot \rceil$ specifies the integer ceiling operation. The representative value M for each bin used in this study is shown in the rightmost column of Table 6.1, though it may be computed analytically directly (for the detail, see Section III later). Finally, after padding three trailing 0's to M, the result is used to repair the tampered pixel. A diagram illustrating the above idea of authentication signal matching and tampered pixel detection is shown in Fig. 6.2. And another diagram illustrating the idea of tampered pixel repairing is shown in Fig. 6.3. Detailed algorithms implementing these ideas are described subsequently.

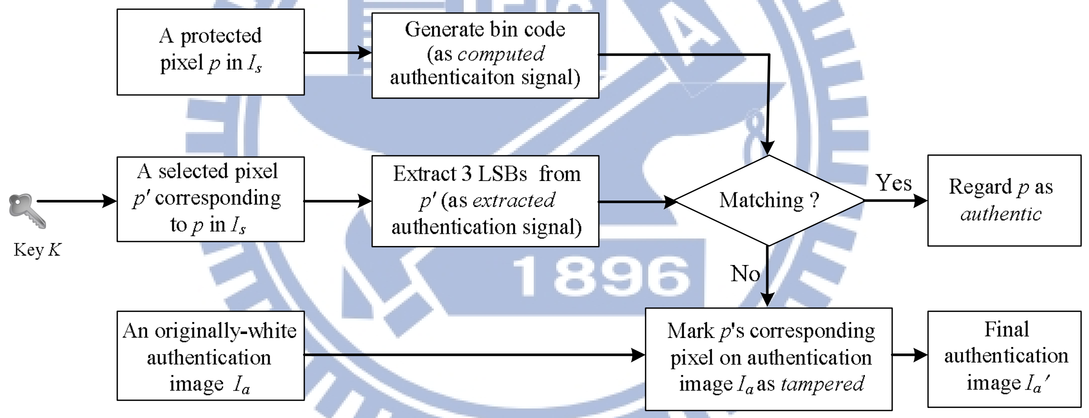

A protected pixel p in I_s
Generate bin code (as *computed* authenticaiton signal)
Key K
A selected pixel p' corresponding to p in I_s
Extract 3 LSBs from p' (as *extracted* authentication signal)
Matching ?
Yes
Regard p as *authentic*
No
An originally-white authentication image I_a
Mark p's corresponding pixel on authentication image I_a as *tampered*
Final authentication image I_a'

Fig. 6.2 Diagram of authentication signal matching and tampered pixel marking (detail to be described in Algorithm 2).

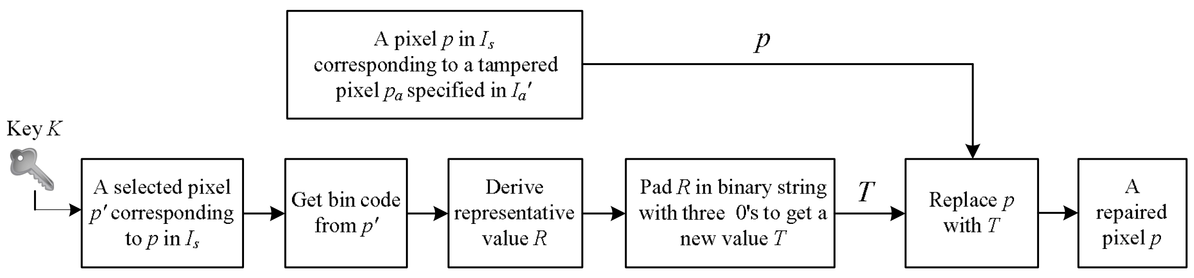

A pixel p in I_s corresponding to a tampered pixel p_a specified in I_a'
p
Key K
A selected pixel p' corresponding to p in I_s
Get bin code from p'
Derive representative value R
Pad R in binary string with three 0's to get a new value T
T
Replace p with T
A repaired pixel p

Fig. 6.3 Diagram of tampered pixel repairing (detail to be described in Algorithm 2).

Algorithm 2: image authentication, tampering detection, and data repairing.

Input: a stego-image I_s generated by Algorithm 1 presumably, an originally-white *authentication image* I_a, and the random number generator f and the secret key

K used in Algorithm 1.

Output: an image I_r with tampered pixels, if any, being repaired.

Step 1. (*Beginning of looping for pixel authentication*) Take in a raster-scan order a pixel p from I_s, and perform the following steps.

Stage 1 --- computation of authentication signals.

1.1 Transform the gray value of p into eight bits $b_7, b_6, \ldots, b_0$.

1.2 Transform the five MSBs $b_7, b_6, \ldots, b_3$ of p into an integer d.

1.3 Map the integer d into a bin indexed by a bin number B computed by $B = \lfloor d/4 \rfloor$.

1.4 Transform B into a 3-bit bin code $s = c_2c_1c_0$ which is also regarded as an authentication signal, called the *computed authentication signal*.

Stage 2 --- extraction of the hidden authentication signal.

1.5 Use the random number generator f and the input key K as the seed to select from I_s randomly a pixel p' corresponding to p, where a previously-embedded authentication signal for p is located presumably.

1.6 Transform the gray value of p' into eight bits $b_7', b_6', \ldots, b_0'$, extract the three LSBs to form a string $s' = b_2'b_1'b_0'$, called the *extracted authentication signal*.

Stage 3 --- authentication signal matching and tampered pixel marking.

1.7 Match the computed authentication signal $s = c_2c_1c_0$ and the extracted one $s' = b_2'b_1'b_0'$ bit by bit; and if mismatching occurs, regard p as having been tampered with and mark its corresponding pixel on the authentication image I_a as a black point.

1.8 (*End of looping*) If there remain unprocessed pixels in I_s, then go to Step 1; otherwise, take the final I_a as a new authentication image I_a' for use in the next stage of the algorithm for image repairing.

Stage 4 --- tampered pixel repairing.

Step 2. (*Beginning of looping for tampered pixel repairing*) For each black point p_a in I_a' selected in the raster-scan order, perform the following steps.

2.1 For the pixel p' in I_s corresponding to p_a, use the input random number generator f with the input key K as the seed to select randomly a pixel p'' where a previously-embedded authentication signal for p' is located presumably.

2.2 Transform the gray value of p'' into eight bits b_7'', b_6'', …, b_0'', extract the three LSBs $b_2''b_1''b_0''$, and transform $b_2''b_1''b_0''$ into an integer B'' which specifies the index of the bin into which the gray value of p' falls.

2.3 Repair the tampered pixel p' by the following steps.

2.3.1 Derive the representative value M of the bin indexed by B''.

2.3.2 Transform M into a 5-bit binary string $r_7r_6r_5r_4r_3$.

2.3.3 Pad three trailing 0's to $r_7r_6r_5r_4r_3$ to get an 8-bit string $T = r_7r_6r_5r_4r_3000$.

2.3.4 Transform T into an integer d' and replace the gray value of p' with d' as the repairing result.

Step 3. (*End of looping*) If there remain unprocessed black pixels in I_a, then go to Step 2; otherwise, take the final I_s as the desired output image I_r.

6.5 Proof of Optimality of Proposed Method for Image Distortion Reduction

In the proposed method presented above, the eight bits of each pixel's gray value is separated into two parts, five MSBs and three LSBs, with the former used for keeping the pixel content and the latter used for embedding the authentication signal. It seems that we may generalize this specific choice of *pixel-bit division*, $(m, l) = (5, 3)$, where m denotes the number of MSBs and l the number of LSBs with $m + l = 8$.

For example, we may choose alternatively to use two LSBs in a pixel for embedding the authentication signal and the remaining six bits for keeping the pixel content, so that $(m, l) = (6, 2)$. Or, by a reverse consideration, we may choose to adopt $(m, l) = (4, 4)$ as well. Is there a criterion to decide which choice is better? The answer proposed in this study is to consider the resulting image distortion.

It will be proved in this section that the choice of $m = 5$ and $l = 3$ as done in this study is *optimal* in the sense of minimizing the resulting total image distortion incurred *both* by authentication signal embedding *and* by tampered pixel repairing. The proof is conducted in a step-by-step reasoning manner as described in the following.

Proof of the optimal choice of the number of bits for use as the authentication signal.

Stage 1 — optimization criterion consideration in terms of resulting image distortion.

(1) First, we consider simultaneously at the *pixel-level* the maximum distortion D_1 resulting from the process of embedding authentication signals as well as the maximum distortion D_2 resulting from the process of repairing tampered pixels, and take their sum $D = D_1 + D_2$ as the criterion function for optimization in choosing the values of (m, l), i.e., for dividing the eight bits of a pixel's gray value into two parts for the purposes described previously. The goal is to obtain a choice of (m, l) which minimizes the value of D, or equivalently, the maximum distortion coming from authentication signal embedding and tampered pixel repairing for each pixel.

(2) Since $m + l = 8$, we just have to choose an optimal value for l under the above-mentioned *minimax criterion*, and take the value of m to be $m = 8 - l$.

Stage 2 — derivation of distortion incurred by authentication signal embedding.

(3) As mentioned, l LSBs of a pixel p are used to compose a bin code which is then taken to be the authentication signal s of p and embedded in another pixel p' (see Step 2 of Algorithm 1). And this will incur a maximum gray-value changes of $2^l - 1$ coming from either of the two cases of bit changes from l 0's to l 1's and from l 1's to l 0's.

(4) Therefore, the maximum gray-value distortion occurring at each pixel resulting from authentication signal embedding is $D_1 = 2^l - 1$.

Stage 3 — derivation of distortion resulting from tampered pixel repairing.

(5) The width of the total range of gray values specified by the m MSBs of a pixel is 2^m which is divided into 2^l bins (see Step 2 of Algorithm 1), so the width W_{bin} of each bin is

$$W_{\text{bin}} = 2^m/2^l = (2^{8-l})/2^l = 2^{8-2l}$$

because $m + l = 8$ as explained before.

(6) Accordingly, if the range of the xth bin B_x is denoted by $[L, R]$, then it is easy to figure out that $L = (x - 1)\times 2^{8-2l}$ and $R = x\times 2^{8-2l} - 1$ and, where $x = 1, 2, \ldots, 2^l$ (see Table 6.1 for numerical examples of $[L, R]$).

(7) Then, the representative value M of B_x (computed in Step 2 of Algorithm 2), which is the middle value between L and R, is just

$$\begin{aligned} M = (L + R)/2 &= [((x - 1)\times 2^{8-2l}) + (x\times 2^{8-2l} - 1)]/2 \\ &= x\times 2^{8-2l} - 2^{7-2l} - 2^{-1}. \end{aligned}$$

(8) With M as the representative value for all the gray values in bin B_x used in repairing a tampered pixel p, the maximum gray-value difference D' between the *repaired* m MSBs of pixel p and the original m ones is $M - L$ (or $R - M$) which may be computed to be

$$D' = M - L = (x\times 2^{8-2l} - 2^{7-2l} - 2^{-1}) - (x - 1)\times 2^{8-2l}$$
$$= 2^{7-2l} - 2^{-1}.$$

(9) Since we pad l trailing zeros to the m MSBs of the representative value M (see Step 2 in Algorithm 2) to compose an 8-bit number to repair the tampered pixel p, the maximum gray-value distortion after repairing p is

$$D_2 = D'\times 2^l + (2^l - 1)$$

where the term $2^l - 1$ specifies the partial distortion coming from the extreme case that the original last l bits of p are all 1′s.

(10) By using the result of D' derived previously in (8), D_2 may be derived in more detail to be

$$D_2 = (2^{7-2l} - 2^{-1})\times 2^l + (2^l - 1).$$
$$= 2^{7-l} + 2^{l-1} - 1.$$

Stage 4 — minimization of the overall distortion.

(11) The maximum gray-value distortion D considered for a pixel as mentioned previously in (1) now can be computed from the results of (4) and (10) above to be

$$D = D_1 + D_2 = (2^l - 1) + (2^{7-l} + 2^{l-1} - 1)$$
$$= 2^{7-l} + 3\times 2^{l-1} - 2.$$

(12) Taking the derivative of D with respect to l, we get

$$dD/dl = 2^{7-l}\times ln2\times[d(7-l)/dl] + 3\times 2^{l-1}\times ln2\times[d(l-1)/dl]$$

$$= 2^{7-l}\times ln2\times(-1) + 3\times 2^{l-1}\times ln2\times(+1)$$

$$= ln2\times(3\times 2^{l-1} - 2^{7-l})$$

where $ln2$ is the natural logarithm value of 2.

(13) Setting $dD/dl = 0$, we can get the following equation

$$ln2\times(3\times2^{l-1} - 2^{7-l}) = 0$$

which may be solved to get

$$2^{7-l} = 3\times2^{l-1},$$

or equivalently,

$$(2^{7-l})/(2^{l-1}) = 2^{8-2l} = 3.$$

(14) Taking the base-2 logarithm values of the two sides of the above equality and simplifying the result, we get finally the solution of l as:

$$l = 4 - [\log_2 3]/2$$

which may be evaluated explicitly to be approximately equal to 3.2075.

(15) Accordingly, since l is the number of LSBs which should be an integer, it is taken to be the integers 3 and 4 for which the corresponding values of the gray-value distortion D are $D(3) = 2^{7-3} + 3\times2^{3-1} - 2 = 26$ and $D(4) = 2^{7-4} + 3\times2^{4-1} - 2 = 30$, respectively. Therefore, the optimal l is finally decided to be 3 which is exactly the number of bits we use to compose an authentication signal as described previously. This completes the proof.

6.6 Experimental Results

Many experiments have been conducted to test the proposed method and one result is shown in Fig. 6.4, where Fig. 6.4(a) is an input surveillance image with the size of 480×360. The result of applying Algorithm 1 to generate and embed authentication signals into Fig. 6.4(a) is shown in Fig. 6.4(b) with a PSNR value of 37.51. Actually, a general lower bound may be computed for this PSNR value, as done by the following reasoning.

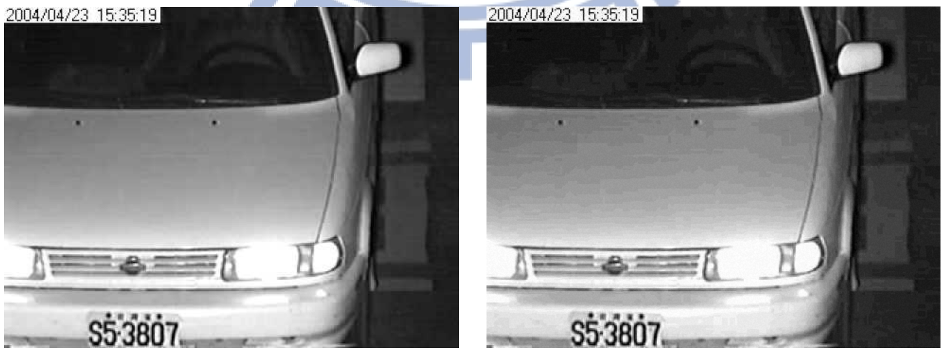
(1) With l being the number of bits in a pixel used for embedding the authentication signal, the *largest* mean square error value MSE of a stego-image with respect to the cover image is $(2^l - 1)^2$ because at each pixel, the largest gray-value difference is $2^l - 1$ after an l-bit authentication signal is embedded there, as described previously.

(2) Accordingly, the peak-signal-to-noise-ratio value $PSNR$ by definition is just

$$\begin{aligned} PSNR &= 10\times\log_{10}(255^2/MSE) \\ &= 10\times\log_{10}[255^2/(2^l - 1)^2] \\ &= 20\times\log_{10}[255/(2^l - 1)] \\ &= 20\times\log_{10}(255/7) \\ &\approx 31.23 \end{aligned}$$

where 255 is the maximum gray value of an 8-bit pixel and l is 3 for our case here.

(3) That is, the lowest bound for the PSNR value is approximately 31.23, which means that the quality of the stego-image is good enough for general applications.



(a) (b)

Fig. 6.4 Generation of stego-image from an input surveillance image. (a) Input image taken by a monitor. (b) Stego-image with PSNR value 37.51.

Back to the presentation of the first case in our experimental results, Fig. 6.5(a) shows a tampering result with a tampering ratio of 0.74% in which two numbers "3" and “7” on the car plate shown in Fig. 6.4(b) were replaced with fake numbers “7” and “5”, respectively. Fig. 6.5(b) shows the obtained authentication image after applying Stages 1 through 3 of Algorithm 2 to Fig. 6.5(a). As can be seen, the tampered pixels covered by the fake numbers have been detected correctly. However, some noise points can be seen to appear in Fig. 6.5(b). These noise points indicate that the pixels in the original image corresponding to these noise points are also erroneously authenticated as having been tampered with. The reason for this noise phenomenon is explained in the following.

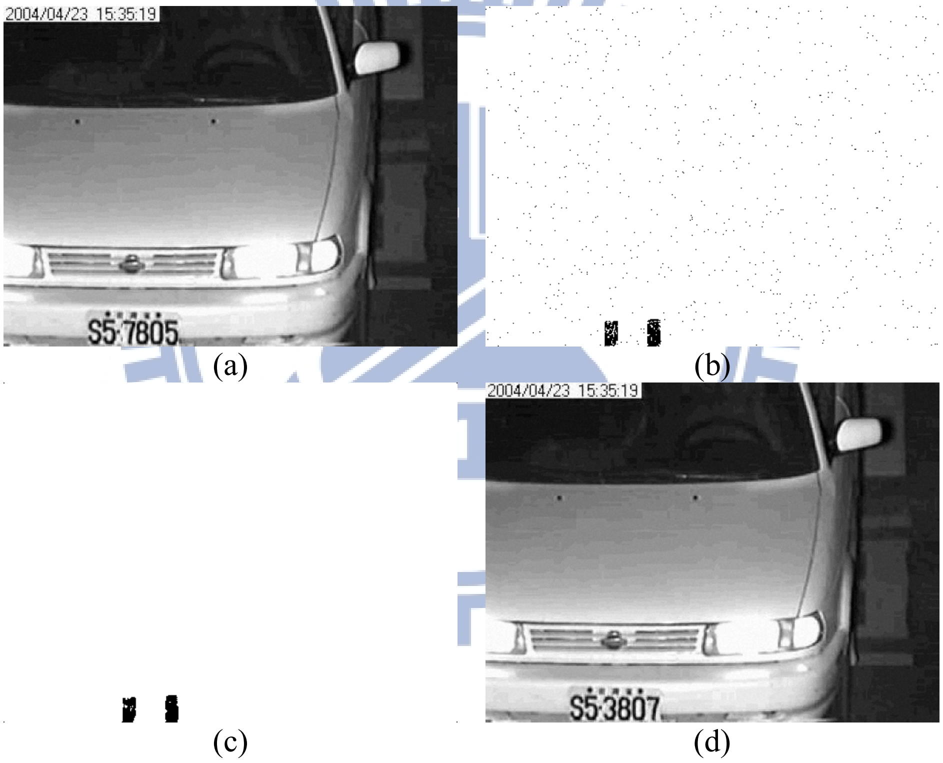


Fig. 6.5 Authentication result of a surveillance image taken by a monitor with tampered area. (a) Image with modification of two car plate numbers. (b) Authentication image with noise. (c) Final authentication image. (d) Final repairing result with PSNR 45.60 with respect to stego-image.

If a pixel A is authenticated as having been tampered with, it means that the authentication signal of a pixel B, which is embedded at pixel A, is also damaged. This in turn means that B will also be authenticated as having been tampered with, even when B is in fact not so. This effect of mutual affection leads to erroneous marking of single points in the authentication image as tampered pixels, creating a pepper-and-salt noise phenomenon like that seen in Fig. 6.5(b). To remove this effect, we applied the median filtering operation to eliminate such noise points before performing the pixel repairing operations described in Stage 4 of Algorithm 2. The final authentication image resulting from doing so to Fig. 6.5(b) is shown in Fig. 6.5(c), in which, as can be seen, most pepper-and-salt points have been eliminated, but 90 false acceptance pixels and 1 false rejection pixels are left. To deal further with this authentication image, image repairing was conducted and the result is shown in Fig. 6.5(d), in which we see that the original numbers “3” and “7” have been repaired successfully at their original positions. Also, with the tampered pixel repaired, the image has a PSNR value of 45.6 with respect to the stego-image shown in Fig. 6.4(b).

Another experimental result of replacing the entire car plate with a fake one is shown in Fig. 6.6. Compared with the previous experimental result with the tampering ratio being 0.74%, the tampering ratio in this case was raised to be 2.25%. It can be seen in Fig. 6.6(b) that the phenomenon of noise points caused by the effect of mutual affection becomes more conspicuous than that in the previous case because of the higher tampering ratio. After noise elimination was performed on Fig. 6.6(b), the final authentication image of Fig. 6.6(c) was obtained, which includes 551 false acceptance pixels (due to the reason that the five MSBs of each of them coincide with those of the original image) and 16 false rejection pixels (due to the reason that their authentication signals embedded in the tampered area were destroyed). Finally, the repaired image in which the original car plate reappeared clearly with a PSNR value of 36.38 is shown in Fig. 6.6(d). Some relevant statistics of the two cases mentioned above are given in Table 6.2.

Table 6.2 Statistics of experiments using a surveillance image of Fig. 6.4(a).

Surveillance image (480×360)	Total # of tampered pixels (tampering ratio)	PSNR of recovered image with respect to stego-image	Total # of false acceptance pixels	Total # of false rejection pixels
Case 1 shown in Fig. 5	784 (0.45%)	45.60	90	1
Case 2 shown in Fig. 6	3895 (2.25%)	36.38	551	16
Case 3 (not shown)	8640 (5%)	29.33	2026	59
Case 4 (not shown)	17280 (10%)	23.94	4201	265
Case 5 (not shown)	34560 (20%)	17.87	8009	2411
Case 6 (not shown)	51840 (30%)	13.59	15079	9775
Case 7 (not shown)	69120 (40%)	10.42	30211	21997
Case 8 (not shown)	86400 (50%)	8.16	53353	34174

To show the relation of the performance of tampering localization and repairing to the degree of tampering as well as the use of median filtering, the statistics of the *false judgments* (including false acceptance pixels and false rejection pixels) and the PSNR values of a series of repaired images listed in the order of increasing tampering ratios are given in Table 6.2. In addition, an illustration of the statistics is shown in Fig. 6.7. Note that the total numbers of false acceptance pixels plus false rejection pixels comprises the ordinate of the number of *falsely judged pixels* in Fig. 6.7.

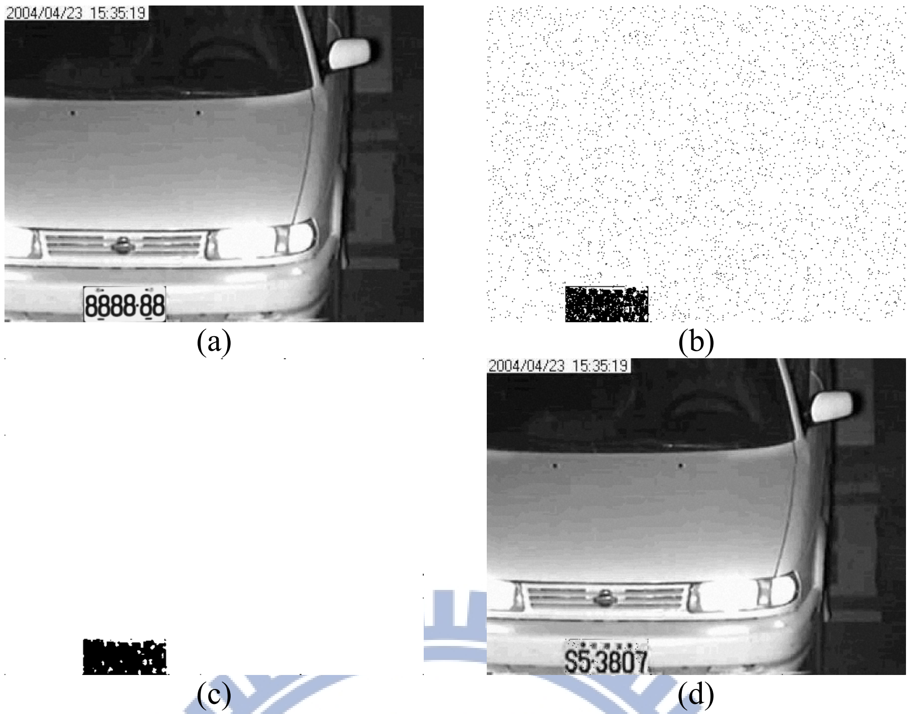


Fig. 6.6 Authentication result of a surveillance image taken by a monitor with tampered area. (a) Image with modification of entire car plate. (b) Authentication image with noise. (c) Final authentication image. (d) Final repairing result with PSNR 36.38 with respect to stego-image.

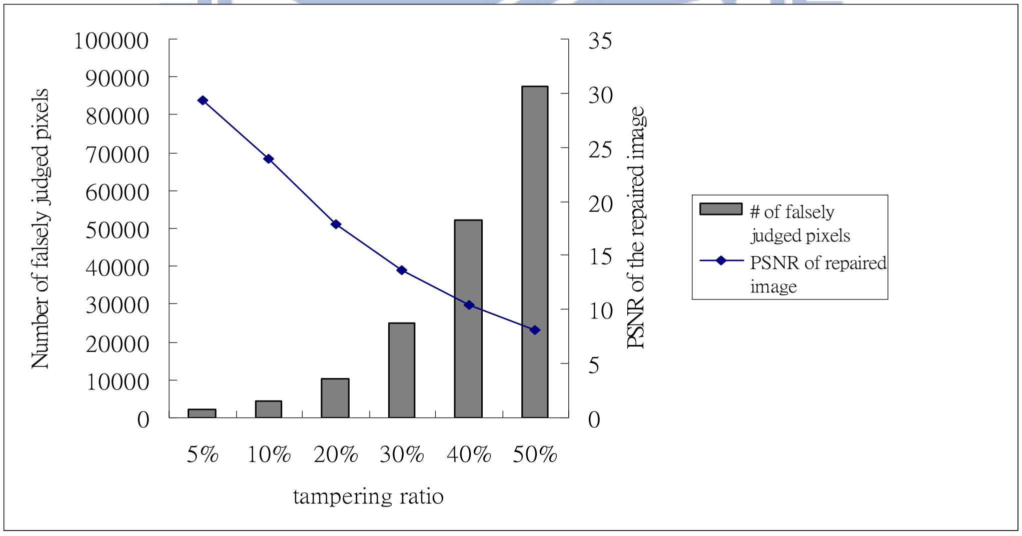


Fig. 6.7 Relations of performances among tampering ratios, false tampering detection, and tampering repairing using surveillance image of Fig. 6.4(a).

In a subsequent experiment, we used another test image, Lena, of size 512×512 as shown in Fig. 6.8(a), and the stego-image yielded by the proposed method is shown in Fig. 6.8(b) whose PSNR value is 39.34.

Fig. 6.8 Generation of stego-image from another image. (a) Input image Lena. (b) Stego-image with PSNR 39.34.

In this experiment, we selected the area of Lena's hair and modified it by adding a rose flower shape of 2084 pixels on it. The modification result is shown in Fig. 6.9(a). Fig. 6.9(b) shows the authentication result without noise elimination, and the final authentication image is shown in Fig. 6.9(c) in which 2041 tampered pixels of the flower were detected and most isolated points were removed after median filtering. Finally, we repaired each of those detected pixels by referencing the bin code as the authentication signal embedded in a certain pixel whose position in Fig. 6.9(a) was located by a key. The repairing result in this case is shown in Fig. 6.9(d), and the PSNR value with respect to the stego-image is 47.00. Some other statistics about this case is given in Table 6.3.

As done in the previous experiments using a surveillance image, we also gradually extended the tampered area in the Lena image to test the effectiveness of the proposed method. Table 6.3 lists the statistics of our experiments conducted in this way. Furthermore, an illustration corresponding to the statistics of Table 6.3 is shown in Fig. 6.10.

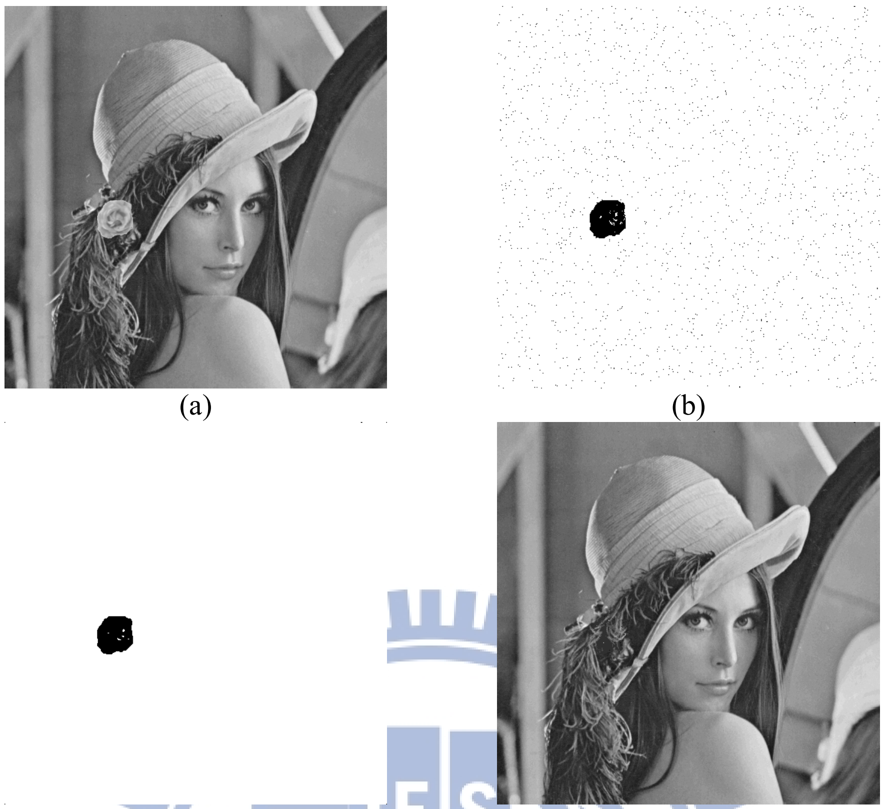


Fig. 6.9 Authentication result of a grayscale image with an added flower shape composed of 2084 pixels. (a) Image with modification of a hair portion. (b) Authentication image with noise. (c) Final authentication image. (d) Final repairing result with PSNR 47.00 with respect to stego-image of Fig. 6.9(b).

According to the results and statistics of all the conducted experiments, the proposed method is seen to be effective enough till the tampering ratio reaches about 10%. This overall result is better than that of the method described in [78] which works effectively when the tampering ratio is smaller than 1.1%.

Table 6.4 lists a comparison of the proposed method with other pixel-level image authentication methods [70][78] in terms of capabilities of self-recovery and tampered-pixel detection. We conducted an experiment that was also conducted in [78] with 2084 tampered pixels. The experimental result is exactly that of Fig. 6.9 given above. From Table 6.4, it can be seen that the proposed method provides better performance in the aspects of tampered pixels detection and tampering ratio limitation, and has the additional self-recovery capability. In addition, due to the characteristic of

Table 6.3 Statistics of experiments using image Lena of Fig. 6.8(a).

Lena (512×512)	Total # of tampered pixels (tampering ratio)	PSNR of recovered image with respect to stego-image	Total # of false acceptance pixels	Total # of false rejection pixels
Case 1 shown in Fig. 9	2084 (0.79%)	47.00	43	12
Case 2 (not shown)	13100 (5%)	32.58	4	92
Case 3 (not shown)	26200 (10%)	25.93	87	430
Case 4 (not shown)	52400 (20%)	19.19	1169	4617
Case 5 (not shown)	78720 (30%)	14.12	8114	18866
Case 6 (not shown)	104640 (40%)	10.85	27678	42277
Case 7 (not shown)	131072 (50%)	8.52	65399	65477

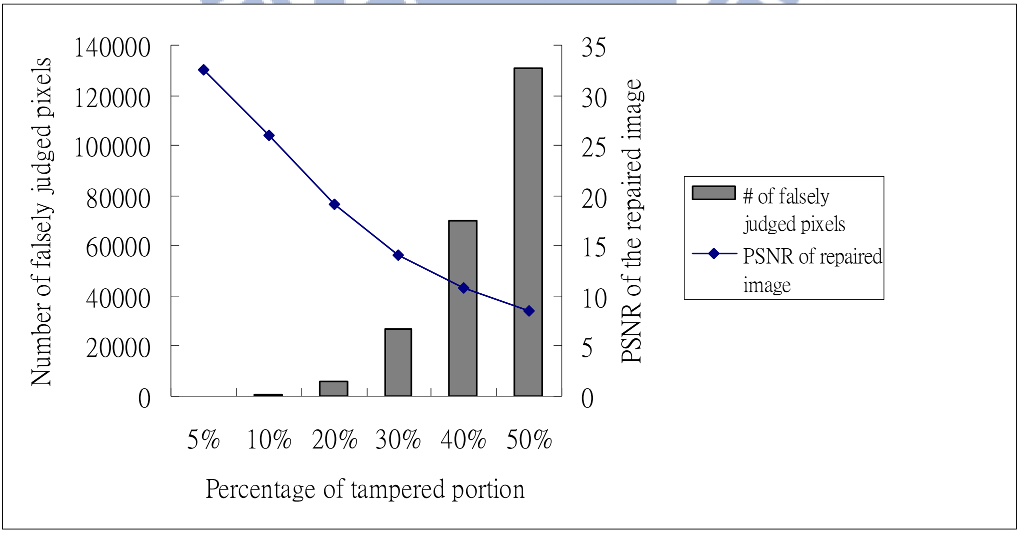


Fig. 6.10 Relations of performances among tampering ratios, tampering detection, and tampering repairing using image Lena of Fig. 6.8(a).

pixel-level authentication, we can recover the tampered area by the unit of pixel and so can recognize the detailed part existing in the original image after the recovery work.

Table 6.4 Comparison of performance of proposed method with those of [8] and [9].

Authentication methods	Pixel-level	Recoverable	# of correctly detected pixels out of 2084 tampered pixels	Limitation of tampering ratio
Method in [8]	Yes	No	Around 1042	Unrestricted
Method in [9]	Yes	No	1996	≦1.1%
Proposed method	Yes	Yes	2041	≦10%

To reveal further the characteristics of the proposed method, an image authentication method [71] based on the similar concept of using compressed codes was compared with. As can be observed in Table 6.5, the proposed method can recover tampered areas at the pixel level, instead of at the block level as done by [71]. And it is also noted that in [71] an auxiliary data item, a code book, is needed for image repairing. This leads to inconvenience and *non-blindness* in the image recovery process because extra storage space is required for the auxiliary data and the image repairing work cannot be done without referring to the auxiliary data. On the contrary, the proposed method is characterized as *blindness*.

Table 6.5 Comparison of performance of proposed method with those of [71].

Authentication methods	Protected area	Pixel level	Free from the need of auxiliary information for recovery	Schemes used for image recovery
Method of [71]	Partial (region of importance)	No	No	Fractal code and image painting
Proposed method	Unrestricted	Yes	Yes	Bin code

Some issues deserve further investigation in the future, for example, noise attacks. Though this kind of attacks can be detected with the aid of human vision in the proposed method, a feasible criterion which can be used to distinguish noise points in the authentication image caused by mutual affection from those resulting from noise attack is desired.

6.7 Summary

A grayscale image authentication method with a capability of localizing tampered image regions and repairing them at the pixel level has been proposed. Based on a bin-mapping scheme of dividing the 5-bit grayscale into eight bins, a 3-bit bin code is generated for use as an authentication signal for each input image pixel. The authentication signals are embedded into other pixels selected randomly by a secret key. The signals are utilized not only for detecting and localizing tampered pixels but also for generating representative values for repairing the tampered pixels. This double-function merit of the authentication signal leads to the possibility of pixel-level tampering detection and the blindness characteristic of the proposed method. Also shown is a proof of the optimality of the proposed method in choosing three bits out of the eight ones of a pixel as an authentication signal under a minimax criterion of minimizing the maximum total gray-value distortion incurred by authentication signal embedding and tampered pixel repairing. Experimental results have shown the effectiveness of the proposed method for authenticating and repairing tampered real images. Future works may be directed to extending the method to deal with color images.

Chapter 7
A Steganographic Method Based on Information Sharing for Hiding Secret Data in Spreadsheets with a Self-Authentication Capability

7.1 Introduction

Covert communication is a technique of concealing secret information into a *cover medium* in an imperceptible way or with a camouflage effect such that only a sender and an intended receiver know the existence of the hidden data in the resulting *stego-medium*. In the literature, emphases were put on the use of multimedia like images, videos, and audios [79-82] because these media in general provide larger embeddable spaces and cause less suspicion due to their wide distributions. And weaknesses existing in human beings' visual capabilities are often exploited to design effective covert communication methods.

In this study, we propose a new data hiding method for covert communication which applies Shamir's (k, n)-threshold secret sharing scheme with $n = k + 1$ to a given secret item to yield k+1 shares, and the generated $k + 1$ shares are embedded into the number items in a spreadsheet as if they are part of the spreadsheet content. The purpose of transforming the secret data into secret shares by the $(k, k+1)$-threshold secret sharing scheme is *not* to enforce robustness, but to yield a blind self-authentication capability for the embedded secret. Conventionally, the concept of (k, n)-threshold secret sharing is applied to provide destruction-tolerant capabilities. That is, any k shares collected from n ones may be processed to reveal the shared secret even though up to $(n - k)$ shares are destroyed. But in the proposed method, the scheme of (k, $k+1$)-threshold secret sharing is developed for the first time to provide instead a self-authentication capability by checking the *value-consistency* of $k+1$ results coming from all $k+1$ combinations to determine whether the extracted secret is intact or not. That is, only when the results computed from any k shares collected from $k+1$ shares are *all identical* in value can the extracted secret be decided to be intact. Fig. 7.1 illustrates these core ideas of the proposed method.

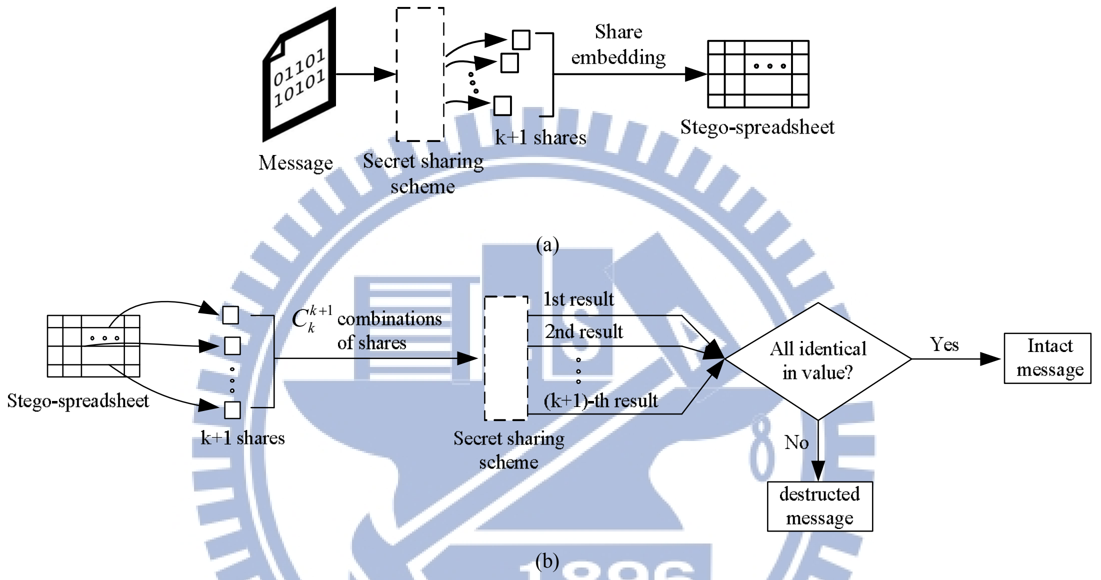
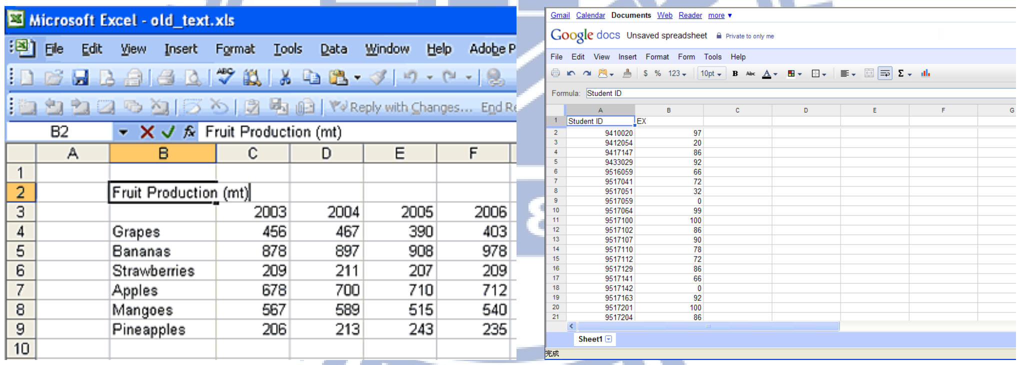


Fig. 7.1 Illustration of proposed covert communication method via spreadsheets by secret sharing. (a) Generation of a stego-spreadsheet. (b) Self-authentication of the extracted message.

Moreover, to conceal the presence of hidden data, secret shares are spread throughout the cover spreadsheet in a *sparsely* fashion. And a spreadsheet containing numeral items with a *high scatter level* is more suitable to be used as a cover spreadsheet for better concealment.

7.2 Merits of Proposed Method

Merits of the proposed method include the following. (1) A receiver can confirm the correctness of the extracted secret message. (2) Compared with some methods using hash codes or parity bits as redundant data to ensure the authenticity of retrieved data, only a minor redundancy, i.e., the $(k + 1)$-th share, is needed in the proposed method. (3) By adaptively choosing involved parameters, i.e. the value of p, used in the polynomial of Shamir's method for the selected spreadsheet, the numerical items' values generated by the method will fall into a reasonable range of values, arousing little suspicion during covert communication. (4) Using spreadsheets as cover media, the proposed method is free from *unintentional* destruction of hidden data like data compression during the secret transmission or data keeping process, in contrast with cover media like images or videos which are often compressed *ignorantly* in such a process. Two examples of such documents, Microsoft Excel and Google Docs, are shown in Fig. 7.2.


Microsoft Excel - old_text.xls
File Edit View Insert Format Tools Data Window Help Adobe P
Reply with Changes... End R
B2
Fruit Production (mt)
Gmail Calendar Documents Web Reader more
Google docs Unsaved spreadsheet
Private to only me
File Edit View Insert Format Form Tools Help
Formula: Student ID
Sheet1
完成

	A	B	C	D	E	F
1						
2		Fruit Production (mt)				
3			2003	2004	2005	2006
4		Grapes	456	467	390	403
5		Bananas	878	897	908	978
6		Strawberries	209	211	207	209
7		Apples	678	700	710	712
8		Mangoes	567	589	515	540
9		Pineapples	206	213	243	235
10						

	A	B
1	Student ID	EX
2	9410020	97
3	9412054	20
4	9417147	86
5	9433029	92
6	9516059	66
7	9517041	72
8	9517051	32
9	9517059	0
10	9517064	99
11	9517100	100
12	9517102	86
13	9517107	90
14	9517110	78
15	9517112	72
16	9517129	86
17	9517141	66
18	9517142	0
19	9517163	92
20	9517201	100
21	9517204	86

(a) (b)

Fig. 7.2 Examples of spreadsheets. (a) Microsoft Excel. (b) Google Docs.

7.3 Review of Shamir's Method for Secret Sharing

In the (k, n)-threshold secret sharing scheme proposed by Shamir [56] with $k \leq n$, a secret d in the form of an integer is transformed into shares which then are distributed to n participants to keep; and as long as up to k of the n shares can be collected, the original secret can be recovered. The detail of the scheme may be described as two algorithms in the following.

Algorithm 1: (k, n)-threshold secret sharing.

Input: a secret d in the form of an integer, the number n of participants, and a threshold k not larger than n.

Output: n shares in the form of integers for n participants to keep.

Steps.

Step 6. Choose randomly a prime number p which is larger than the secret d.

Step 7. Select $k - 1$ integer values $c_1, c_2, \ldots, c_{k-1}$ within the range of 0 through $p - 1$.

Step 8. Select n distinct real values for the variables $x_1, x_2, \ldots, x_n$.

Step 9. Use the following $(k - 1)$-degree polynomial to compute n function values $F(x_i)$, called *partial shares*:

$$F(x_i) = (d + c_1x_i + c_2x_i^2 + \ldots + c_{k-1}x_i^{k-1})_{\text{mod } p}, \tag{1}$$

for $i = 1, 2, \ldots, n$.

Step 10. Deliver the 2-tuple $(x_i, F(x_i))$ as a *share* to the ith participant, where $i = 1, 2, \ldots, n$.

Since there are k coefficients, including d and c_1 through c_{k-1}, in (1) above, it is necessary to collect at least k shares from the n participants to form k equations of the form of (1) to solve these k coefficients in order to recover the secret d. This explains the term, *threshold*, for k and the name, (k, n)-*threshold*, for the Shamir method [56]. Below is a description of the equation-solving process for secret recovery.

Algorithm 2: secret recovery.

Input: k shares collected from the n participants where k is the threshold mentioned in Algorithm 1; and the prime number p which was chosen in Step 1 of Algorithm 1.

Output: the secret d hidden in the shares and the coefficients c_i used in the equations described by (1) in Algorithm 1, where $i = 1, 2, \ldots, k - 1$.

Steps.

1. Use the k shares $(x_1, F(x_1)), (x_2, F(x_2)), \ldots, (x_k, F(x_k))$ to set up the following equations:

$$F(x_j) = (d + c_1x_j + c_2x_j^2 + \ldots + c_{k-1}x_j^{k-1})_{\text{mod } p}, \tag{2}$$

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

2. Solve the k equations above by Lagrange's interpolation to obtain the desired secret value d [58] as follows:

$$d = (-1)^{k-1}[F(x_1)\frac{x_2x_3...x_k}{(x_1-x_2)(x_1-x_3)...(x_1-x_k)} + F(x_2)\frac{x_1x_2...x_k}{(x_2-x_1)(x_2-x_3)...(x_2-x_k)}$$
$$+...+F(x_k)\frac{x_1x_2...x_{k-1}}{(x_k-x_1)(x_k-x_2)...(x_k-x_{k-1})}]_{\bmod p}.$$

3. Compute the values c_1 through c_{k-1} by expanding the following equality and comparing the result with (2) in Step 1 while regarding the variable x in the equality below to be x_j in (2):

$$F(x) = [F(x_1)\frac{(x-x_2)(x-x_3)...(x-x_k)}{(x_1-x_2)(x_1-x_3)...(x_1-x_k)} + F(x_2)\frac{(x-x_1)(x-x_3)...(x-x_k)}{(x_2-x_1)(x_2-x_3)...(x_2-x_k)}$$
$$+...+F(x_k)\frac{(x-x_1)(x-x_2)...(x-x_{k-1})}{(x_k-x_1)(x_k-x_2)...(x_k-x_{k-1})}]_{\bmod p}.$$

Step 3 in the above algorithm is included for the purpose of computing the values of the parameters c_i in the proposed method. In other applications, if only the secret value d need be recovered, this step may be eliminated.

7.4 Proposed Method for generation of a stego-image

In the proposed method, an appropriate cover spreadsheet S which contains numeric data for disguising generated secret shares is prepared first. Next, a secret message M to be hidden is divided into several segments, and taken as input to Shamir's (k, n)-threshold secret sharing scheme [56] with carefully chosen parameters to generate secret shares. Then, numeric items in S which are selected by a secret key are replaced with the shares to generate a stego-spreadsheet S'. In this process, the parameters involved in Eq. (1) of Algorithm 1 are adjusted to satisfy the characteristics of the input secret message and the prepared cover spreadsheet. These parameters include: (a) the number m of bits in each message segment, which is also taken to be the identical numbers of bits in all of the coefficients d, c_1 through c_{k-1}; (b) the number k of message segments processed by the Shamir scheme each time, which is also the minimum number k of secret shares needed to be collected to recover the secret; (c) the total number n of generated shares, which is set to be $k + 1$ specifically; (d) and the prime number p, which is the smallest integer larger than all the values of the coefficients d, c_1 through c_{k-1}, and the variables x_1 through x_n used in Eq. (1) [56].

A detailed algorithm describing the process is presented in the following.

Algorithm 3*: *generation of a stego-spreadsheet.

Input: a binary secret message M divided into m-bit segments, a spreadsheet S, a secret key K, and three pre-selected integers k, n (= k + 1), and m.

Output: a stego-spreadsheet S'.

Steps.

Stage 1 — share generation.

Step 1. Choose the smallest prime number p which is larger than $2^m - 1$.

Step 2. Take sequentially k unprocessed m-bit segments from M to form a group G, called *segment group*, and perform the following steps to transform the segment group into partial shares.

2.1 Transform the k m-bit message segments in G into integers and take the results to be d, c_1, c_2, …, c_{k-1}, respectively.

2.2 Take x_1 through x_n to be the integers 1 through n, respectively, where $n = k + 1$.

2.3 Use the following (k − 1)-degree polynomial to compute n partial shares $F(x_i)$:

$$F(x_i) = (d + c_1x_i + c_2x_i^2 + \ldots + c_{k-1}x_i^{k-1})_{\bmod p}, \tag{3}$$

where i = 1, 2, …, n.

2.4 Save all $F(x_i)$ in order into a *partial-share set* F_{ps}.

Step 3. If the message segments in M are not exhausted, then go to Step 2 to process another segment group; otherwise, continue.

Stage 2 — partial share embedding.

Step 4. Take an unprocessed partial share $F(x_i)$ from F_{ps}, and perform the following steps.

4.1 Use the secret key K to randomly select a numeric item I in S.

4.2 Replace I with $F(x_i)$.

If there exist unprocessed partial shares in F_{ps}, go to Step 4; otherwise, take the final S as the output S'.

7.5 Proposed Image Authentication and Recovery Process

The proposed blind self-authentication capability for verifying a recovered secret message is fulfilled by the $(k, k+1)$-threshold secret sharing scheme. In the past, the concept of (k, n)-threshold secret sharing is often applied to develop methods for secret image sharing [58][83-85] or image repairing [85] with destruction-tolerant capabilities — any k shares collected from the n ones may be processed to reveal the shared secret even though up to $(n - k)$ shares are destroyed. But in the proposed method here, the scheme of $(k, k+1)$-threshold secret sharing is developed to provide a self-authentication capability for verifying the correctness of a recovered segment group in the secret message — *any* k shares collected from the $k+1$ ones should, after the secret recovery process of Algorithm 2 is conducted, reveal the same secret value in normal cases, meaning that no damage ever occurs to the $k+1$ shares; otherwise, it can be decided that some shares must have been destroyed. By making use of this characteristic, blind self-authentication of each segment group in the recovered secret message is carried out, and verification of the integrity and fidelity of the secret message thus achieved. A detailed algorithm of secret message recovery and self-authentication is described in the following.

Algorithm 4: secret data recovery and self-authentication.

Input: a stego-spreadsheet S'; the prime number p, the three integers k, n $(= k + 1)$, and m, and the secret key K used in Algorithm 3.

Output: a secret message M hidden in S' presumably, and a report about the authenticity of the segments within M.

Steps.

Stage 1 — message segment computation.

Step 1. Use the secret key K to select randomly numeric items in S'; take out their values which presumably are the partial shares $F(x_i)$ embedded by Algorithm 3; and put the items sequentially into a set F_{ps} as a partial-share set.

Step 2. Take out in order n partial shares from F_{ps}, set their corresponding x values as 1 through n, respectively, and perform the following steps to recover a binary segment M_i of the secret message M, if possible.

2.1 For every k partial shares $F_1, F_2, \ldots, F_k$ in the n ones and their corresponding x values $x_1, x_2, \ldots, x_k$, perform the following steps.

2.1.1 Use the k shares (x_1, F_1), (x_2, F_2), …, (x_k, F_k) to set up the following equations:

$$F_j = F(x_j) = (d + c_1x_j + c_2x_j^2 + \ldots + c_{k-1}x_j^{k-1})_{\bmod p}, \qquad (4)$$

where $j = 1, 2, ..., k$.

2.1.2 Compute the values d and c_1 through c_{k-1} by expanding the following equality and comparing the result with (4) in Step 2.1.1 above while regarding the variable x in the equality below to be x_j in (4):

$$F(x) = [F(x_1)\frac{(x-x_2)(x-x_3)...(x-x_k)}{(x_1-x_2)(x_1-x_3)...(x_1-x_k)} + F(x_2)\frac{(x-x_1)(x-x_3)...(x-x_k)}{(x_2-x_1)(x_2-x_3)...(x_2-x_k)}$$
$$+ ... + F(x_k)\frac{(x-x_1)(x-x_2)...(x-x_{k-1})}{(x_k-x_1)(x_k-x_2)...(x_k-x_{k-1})}]_{\bmod p}.$$

2.1.3 Put the computed values of d and c_1 through c_{k-1} as a set into a buffer B.

(There will be $n = k + 1$ sets of values of d and c_1 through c_{k-1} at the end of Step 2).

Stage 2 — self-authentication of the computed message segment.

Step 3. Take out the n sets of the coefficient values of d and c_1 through c_{k-1} in B and perform the following operations.

3.1 Transform the coefficients d and c_1 through c_{k-1} into k binary segments, and concatenate them as a message segment M_i.

3.2 If all the n sets of the coefficient values are identical to one another, then mark M_i as *authentic* and append it to the end of the desired secret message M; else, mark M_i as *having been tampered with* and continue.

If all shares embedded in S' are processed, then take the final M as the output; otherwise, go to Step 2.

7.6 Security Consideration

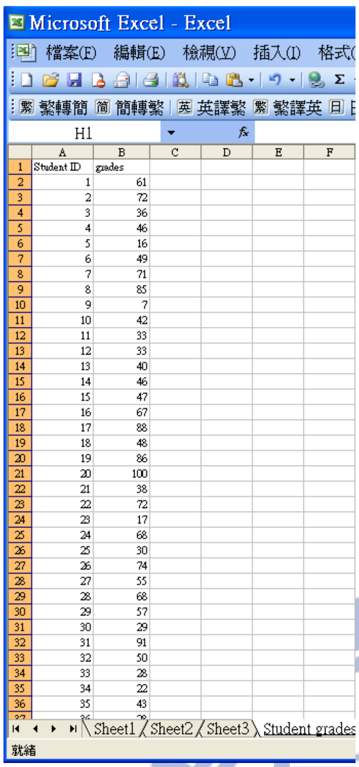
A statistical anomaly caused by information embedding is a reliable clue to detect the presence of the steganographic content. For the purpose of resisting such statistical analysis, two strategies are used in the proposed method. One is to spread secret shares throughout the cover spreadsheet in a *sparsely* and randomly distributed fashion so that less affection is incurred to the statistical properties of the cover spreadsheet after information embedding. This way of achieving undetectability for a hidden message used in the proposed method follows the concept of the *frequency-hopping spread spectrum* technique in which radio signals are transmitted by many frequency channels selected according to a pseudorandom sequence known to the sender and the receiver. The other strategy is to choose *comparatively insignificant parts* of numeric data in the spreadsheet for embedding secret shares in order to keep a low level of embedding strength for maintaining the statistical properties in a stego-spreadsheet. For example, we may choose the decimal fractions of the numbers in a cover spreadsheet and replace their values with those of the secret shares, resulting in insignificant alterations to the statistical property in the stego-spreadsheet.

The proposed method not only can passively prevent the stego-spreadsheet from detection but also can actively ensure the fidelity and integrity of the transmitted secret. In the active attack model [86], if an adversary subtly made modifications to passing-by stego-spreadsheets for the purpose of misleading a receiver, the blind self-authentication capability provided by the proposed method can be used to check the authenticity of the retrieved secret message. When the authenticity check fails, it reveals that the communication between the two sides has been threatened and appropriate measures should be adopted.

7.7 Experimental Results

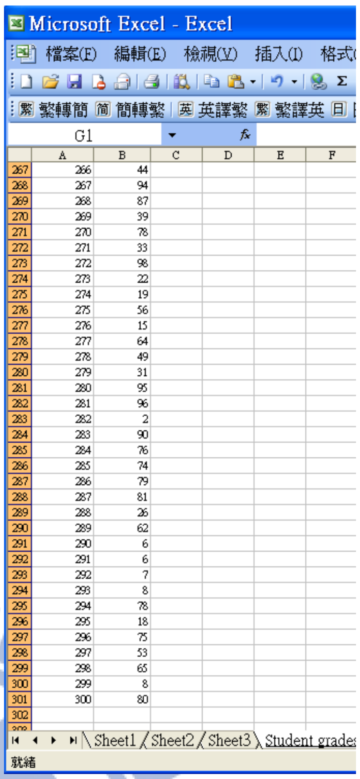
A. Experimental Results Using Spreadsheets Recording Students' Scores

A result of the experiments we conducted using the proposed method was based on the use of a cover spreadsheet recording 300 students' scores saved as an Excel file as shown in Fig. 7.3. Note that this is just an example; the type of cover spreadsheet and the content of it need not be restricted to be so.


Microsoft Excel - Excel
檔案(F) 編輯(E) 檢視(V) 插入(I) 格式
繁轉簡 簡轉繁 英譯繁 繁譯英
H1
Sheet1 Sheet2 Sheet3 Student grades
就緒

	A	B
1	Student ID	grades
2	1	61
3	2	72
4	3	36
5	4	46
6	5	16
7	6	49
8	7	71
9	8	85
10	9	7
11	10	42
12	11	33
13	12	33
14	13	40
15	14	46
16	15	47
17	16	67
18	17	88
19	18	48
20	19	86
21	20	100
22	21	38
23	22	72
24	23	17
25	24	68
26	25	30
27	26	74
28	27	55
29	28	68
30	29	57
31	30	29
32	31	91
33	32	50
34	33	28
35	34	22
36	35	43

(a)


Microsoft Excel - Excel
檔案(F) 編輯(E) 檢視(V) 插入(I) 格式
繁轉簡 簡轉繁 英譯繁 繁譯英
G1
Sheet1 Sheet2 Sheet3 Student grades
就緒

	A	B
267	266	44
268	267	94
269	268	87
270	269	39
271	270	78
272	271	33
273	272	98
274	273	22
275	274	19
276	275	56
277	276	15
278	277	64
279	278	49
280	279	31
281	280	95
282	281	96
283	282	2
284	283	90
285	284	76
286	285	74
287	286	79
288	287	81
289	288	26
290	289	62
291	290	6
292	291	6
293	292	7
294	293	8
295	294	78
296	295	18
297	296	75
298	297	53
299	298	65
300	299	8
301	300	80
302		

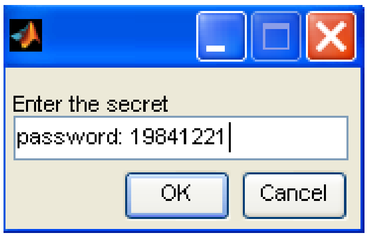
(b)

Fig. 7.3 A cover spreadsheet with 300 numeric items of students' test scores. (a) List of the first 36 items in the spreadsheet. (b) List of the last 34 items in the spreadsheet.

The values of the involved parameters p, m and k in Eq. (3) of the Shamir method were set to be 101, 6, and 7, respectively. The value of the prime number p was taken to be 101 because it is the smallest integer larger than the full marks of 100 of the students' test scores. The value of $m = 6$ means that the length of each segment of the input secret message M was taken to be 6 bits, which satisfies the requirement of $2^m - 1 = 63 < p$ mentioned in Step 1 of Algorithm 3. And each message segment in M was transformed into an integer for use as one of the coefficients d, c_1, c_2…, c_{k-1} in Eq. (3). As for $k = 7$, it means that the value n is $n = k + 1 = 8$ in the applied (k, n)-threshold secret sharing scheme, and that every 7 message segments in M are used as the coefficients d, c_1, c_2, …, c_6 of the polynomial in Eq. (3). Then, a total of 8 (= 7 + 1) secret shares were generated by Algorithm 3, yielding a self-authentication capacity of checking every 7 message segments in M.

Furthermore, as shown in Fig. 7.4, the input secret message M was taken to be the note: "password: 19841221". In this case, the 18 characters of the message were transformed into a binary string with $18 \times 7 = 126$ bits (7 bits per ASCII-coded character). The 126 bits then were divided into 3 segment groups with each group composed of 7 segments and each segment consisting of m = 6 bits. The three segment groups correspond to the following three message sections:

Group 1: "Passwo"; Group 2: "rd: 19"; Group 3: "841221."

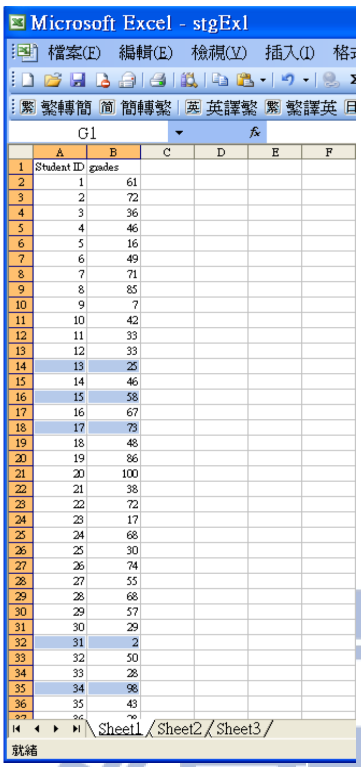

Enter the secret

password: 19841221

Fig. 7.4 A dialogue for entering input secret message.

Totally, the 3 segment groups generated 3×8 = 24 secret shares which at last, by the use of a secret key, were randomly embedded into the cover spreadsheet to yield a stego-spreadsheet. We list the first 36 items in the stego-spreadsheet in Fig. 7.5(a), where items having been replaced with the secret shares are marked in blue. A list of the first 36 items in the cover spreadsheet is given in Fig. 7.5(b) for comparison.

If the stego-spreadsheet is intentionally modified illegally, Algorithm 4 will detect such tampering by the self-authentication operation (see Step 3). Besides, if some embedded secret shares survive the modification, Algorithm 4 can reconstruct the partially correct secret message from them by the recovery steps (Steps 2 through 4). Some experimental results of these functions are described now. Fig. 7.6 shows a modified stego-spreadsheet where items 16 through 26 were tampered with by replacing them with other numbers. Within the 11 modified items, items 15 and 17 include two embedded secret shares. The secret message extracted from such a modified spreadsheet using Algorithm 4 is shown in Fig. 7.7. As can be seen, segment groups 2 and 3 of the secret message were reconstructed correctly, while segment group 1 is authenticated to have been modified and marked by the algorithm with asterisk symbols "*."


Microsoft Excel - stgExl

檔案(F) 編輯(E) 檢視(V) 插入(I)

繁轉簡 簡轉繁 英譯繁 繁譯英

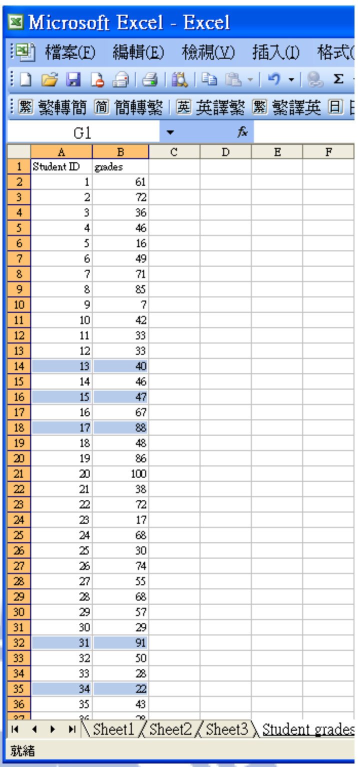
G1

	A	B	C	D	E	F
1	Student ID	grades				
2	1	61				
3	2	72				
4	3	36				
5	4	46				
6	5	16				
7	6	49				
8	7	71				
9	8	85				
10	9	7				
11	10	42				
12	11	33				
13	12	33				
14	13	25				
15	14	46				
16	15	58				
17	16	67				
18	17	73				
19	18	48				
20	19	86				
21	20	100				
22	21	38				
23	22	72				
24	23	17				
25	24	68				
26	25	30				
27	26	74				
28	27	55				
29	28	68				
30	29	57				
31	30	29				
32	31	2				
33	32	50				
34	33	28				
35	34	98				
36	35	43				

Sheet1 / Sheet2 / Sheet3

就緒

(a)


Microsoft Excel - Excel

檔案(F) 編輯(E) 檢視(V) 插入(I) 格式(

繁轉簡 簡轉繁 英譯繁 繁譯英

G1

	A	B	C	D	E	F
1	Student ID	grades				
2	1	61				
3	2	72				
4	3	36				
5	4	46				
6	5	16				
7	6	49				
8	7	71				
9	8	85				
10	9	7				
11	10	42				
12	11	33				
13	12	33				
14	13	40				
15	14	46				
16	15	47				
17	16	67				
18	17	88				
19	18	48				
20	19	86				
21	20	100				
22	21	38				
23	22	72				
24	23	17				
25	24	68				
26	25	30				
27	26	74				
28	27	55				
29	28	68				
30	29	57				
31	30	29				
32	31	91				
33	32	50				
34	33	28				
35	34	22				
36	35	43				

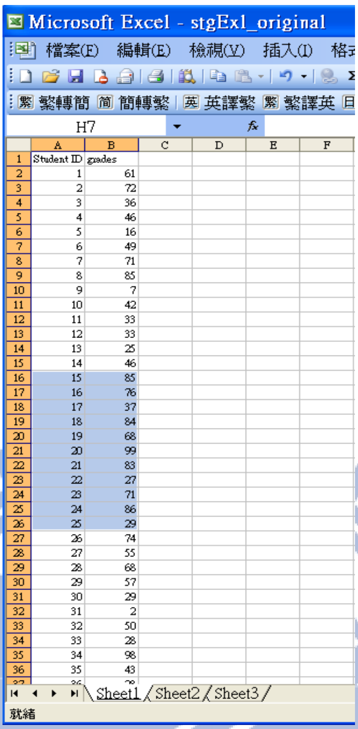
Sheet1 / Sheet2 / Sheet3 \ Student grades

就緒

(b)

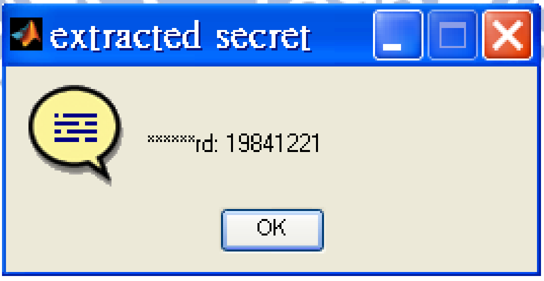
Fig. 7.5 Comparison of a cover spreadsheet and the stego-spreadsheet generated from it. (a) The stego-spreadsheet. (b) The cover spreadsheet.

In this case, the strategy of yielding a low embedding rate mentioned previously is used to achieve the goal of creating undetectability of the stego-spreadsheet. In order to ensure that this strategy works, the two-sample Kolmogorov-Smirnov test (KS test), which is a non-parametric statistical test and is useful to check whether two data samples come from the same probability distribution, is used to quantitatively compare the probability distribution of numeric data in a stego-spreadsheet with that in a cover spreadsheet. The null hypothesis is that two data samples come from the same underlying distribution at the 5% significance level, and the alternative hypothesis is that they are from different distributions. The result of applying the test to the contents of the cover spreadsheet and the stego-spreadsheet shown in Fig. 7.5 is shown in Table 7.1 given below, in which the resulting hypothesis 0 means that the test cannot reject the null hypothesis, that is, a third party cannot think that the probability distribution of the stego-spreadsheet is different from that of the cover spreadsheet. The limit of the embedding rate at which the two-sample KS-test will reject the null hypothesis, according to our experiments, is 50.67% in this case. This means that the embedding rate should be smaller than 50.67% in order to keep the undetectability property of the stego-spreadsheet when a steganalyst has the information of the probability distribution related to the stego-spreadsheet.


Microsoft Excel - stgExl_original
檔案(F) 編輯(E) 檢視(V) 插入(I) 格式
繁 繁轉簡 簡 簡轉繁 英 英譯繁 繁 繁譯英
H7
Sheet1 Sheet2 Sheet3
就緒

	A	B
1	Student ID	grades
2	1	61
3	2	72
4	3	36
5	4	46
6	5	16
7	6	49
8	7	71
9	8	85
10	9	7
11	10	42
12	11	33
13	12	33
14	13	25
15	14	46
16	15	85
17	16	76
18	17	37
19	18	84
20	19	68
21	20	99
22	21	83
23	22	27
24	23	71
25	24	86
26	25	29
27	26	74
28	27	55
29	28	68
30	29	57
31	30	29
32	31	2
33	32	50
34	33	28
35	34	98
36	35	43

Fig. 7.6 A tampered spreadsheet with fake items 16 through 26.



******rd: 19841221

Fig. 7.7 An extracted secret message with a message segment retrieved from tampered items in the stego-spreadsheet marked by symbols "*."

How to choose an embedding rate which is secure against such a statistical test depends on the *scatter level* of the chosen numeric data of the cover spreadsheet. Here, the scatter level is computed as the variance of numeric data values. In terms of this parameter, three spreadsheets Scores1, Scores2, and Scores3 with the scatter level from high to low were tested further in our experiments using the same setting of parameters. Scores1 is just the one used in the first experiment mentioned above and the corresponding statistics is shown in Table 7.1. The results of using Scores2 and

Scores3 are shown in Tables 7.2 and 7.3, respectively. From Table 7.2, the limit of the embedding rate using Scores2 is seen to be 26% which is lower than that using Scores1. As for Scores3, the corresponding limit of the embedding rate is down to be 6.04% as seen in Table 7.3. These experimental statistics indicate that the numeric data of a cover spreadsheet with a higher scatter level can yield a higher embedding rate without causing statistical anomalies. This fact can also be seen from the message embedding bit rate *per numeric item*, also shown in the tables. Specifically, the upper bound of the embedding bit rate per numeric item in Scores 1 is 2.66 b, which is higher than those in Scores 2 (1.36 b) and Scores 3 (0.32 b).

Table 7.1 Experimental results of using strategy 1 with a cover spreadsheet with high scatter level of numeric data.

Scores 1 (300 numeric items with variance 917.76 and size 25K)	# of replaced numeric items I	Resulting hypothesis (5%)	p value	Capacity= $\lfloor I/n \rfloor \times m \times k$ (bits)	Embedding bit rate per numeric item	Embedding bit rate
Embedding rate 5%	16	0 (cannot reject)	1	2×6×7=84	0.28 b	1/298
Embedding rate limit 50.67%	152	1 (reject)	0.0309	19×6×7=798	2.66 b	1/31

Table 7.2 Experimental results of using strategy 1 with a cover spreadsheet with medium scatter level of numeric data.

Scores 2 (1296 numeric items with variance 465.62 and size 105K)	# of replaced numeric items	Resulting hypothesis (5%)	p value	Capacity= $\lfloor I/n \rfloor \times m \times k$ (bits)	Embedding bit rate per numeric item	Embedding bit rate
Embedding rate 5%	64	0 (cannot reject)	0.9999	8×6×7=336	0.26 b	1/313
Embedding rate limit 26%	336	1 (reject)	0.049	42×6×7=1764	1.36 b	1/60

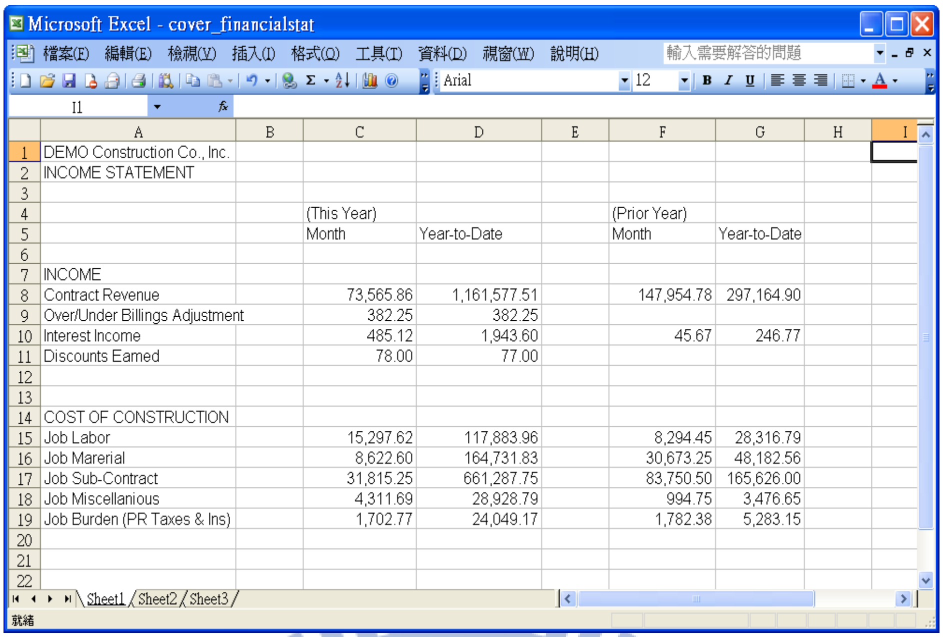
Table 7.3 Experimental results of using strategy 1 with a cover spreadsheet with low scatter level of numeric data.

Scores 3 (2250 numeric items with variance 283.11 and size 31K)	# of replaced numeric items	Resulting hypothesis (5%)	p value	Capacity= $\lfloor I/n \rfloor \times m \times k$ (bits)	Embedding bit rate per numeric item	Embedding bit rate
Embedding rate 5%	112	0 (cannot reject)	0.3557	14×6×7=588	0.26 b	1/53
Embedding rate limit 6.04 %	136	1 (reject)	0.0383	17×6×7=714	0.32 b	1/43

B. Experimental Results Using a Spreadsheet of a Financial Statement

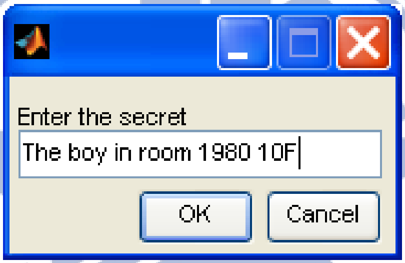
Another experimental result using the Microsoft Excel file of a financial statement of a company as the cover spreadsheet is shown in Figs. 7.8 through 7.11. Fig. 7.8 shows the cover spreadsheet with 32 candidate numeric items for data embedding. In this case, the strategy of choosing insignificant parts of numeric data in the cover spreadsheet for embedding secret shares is used to keep a low level of embedding strength for consideration of the undetectability of the generated stego-spreadsheet. Fig. 7.9 shows the input secret message which was transformed into 32 shares by Algorithm 3. Correspondingly, the decimal fractions of all of the 32 numeric items in the cover spreadsheet of Fig. 7.8 were used to embed the shares. Each share was transformed into two digits and embedded to the right of the decimal point of a numeric item. The resulting stego-spreadsheet is shown in Fig. 7.10 which looks like a common spreadsheet. As done in the previous experiment, the two-sample Kolmogorov-Smirnov test was used, and the result is shown in Table 7.4 which supports the use of the strategy, accomplishing the goal of yielding statistical undetectability in the stego-spreadsheet.

Again assume that an aggressive adversary made subtle modifications to the stego-object. Fig. 7.11 shows the stego-spreadsheet with 3 numeric items (highlighted) being modified. The secret message extracted from the modified stego-spreadsheet is shown in Fig. 7.12(b) in which the destructed part of the secret message is marked by asterisk symbols. As a comparison, the secret message extracted from the intact stego-spreadsheet shown in Fig. 7.10 is shown in Fig. 7.12(a).


Microsoft Excel - cover_financialstat

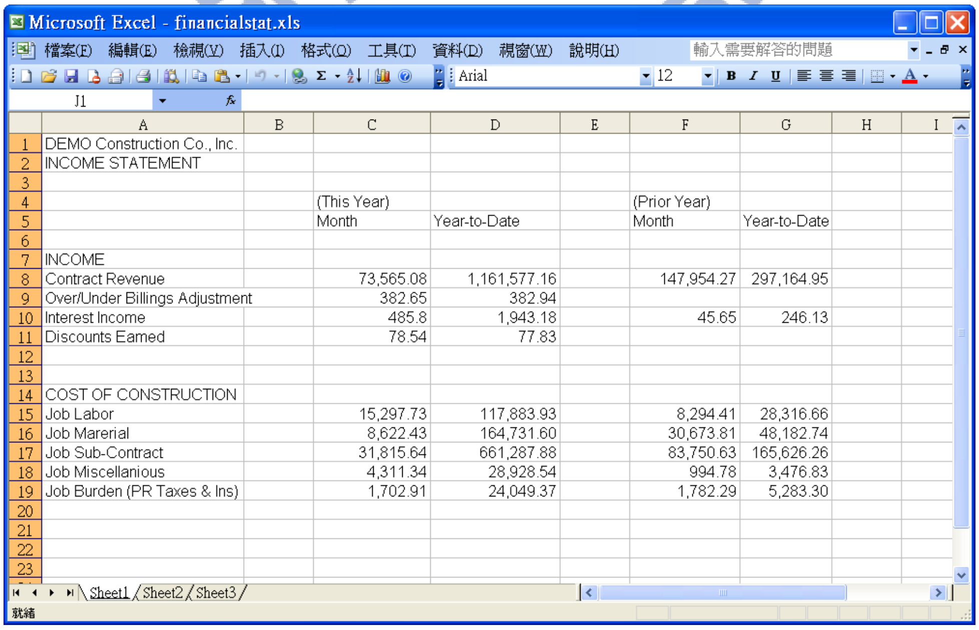
	A	B	C	D	E	F	G
1	DEMO Construction Co., Inc.						
2	INCOME STATEMENT						
3							
4			(This Year)			(Prior Year)	
5			Month	Year-to-Date		Month	Year-to-Date
6							
7	INCOME						
8	Contract Revenue		73,565.86	1,161,577.51		147,954.78	297,164.90
9	Over/Under Billings Adjustment		382.25	382.25			
10	Interest Income		485.12	1,943.60		45.67	246.77
11	Discounts Eamed		78.00	77.00			
12							
13							
14	COST OF CONSTRUCTION						
15	Job Labor		15,297.62	117,883.96		8,294.45	28,316.79
16	Job Marerial		8,622.60	164,731.83		30,673.25	48,182.56
17	Job Sub-Contract		31,815.25	661,287.75		83,750.50	165,626.00
18	Job Miscellanious		4,311.69	28,928.79		994.75	3,476.65
19	Job Burden (PR Taxes & Ins)		1,702.77	24,049.17		1,782.38	5,283.15

Fig. 7.8 A cover spreadsheet of financial statement with 32 numeric items.


Enter the secret

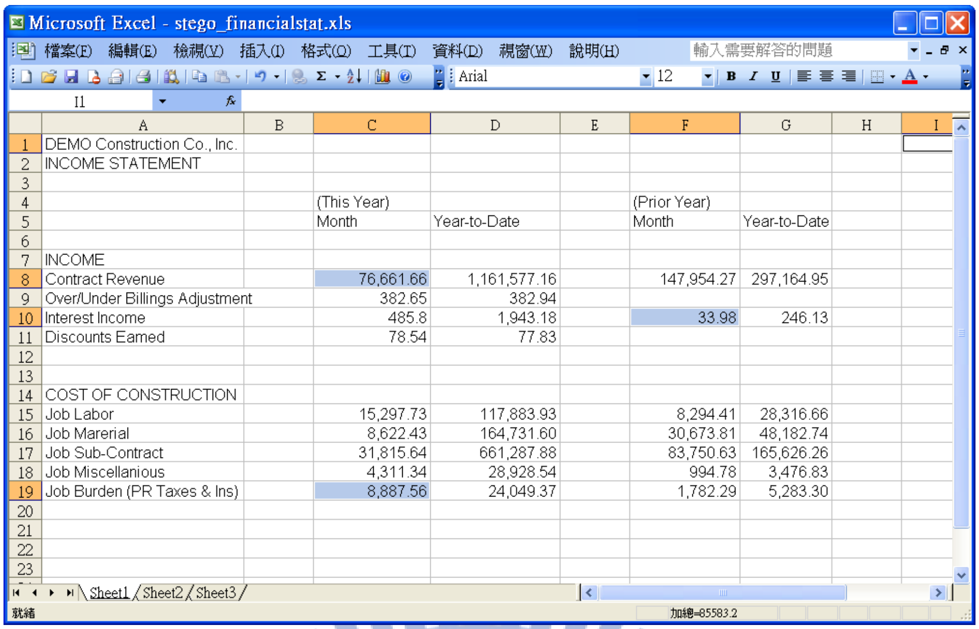
The boy in room 1980 10F

Fig. 7.9 A dialogue with the input secret message.


Microsoft Excel - financialstat.xls

	A	B	C	D	E	F	G
1	DEMO Construction Co., Inc.						
2	INCOME STATEMENT						
3							
4			(This Year)			(Prior Year)	
5			Month	Year-to-Date		Month	Year-to-Date
6							
7	INCOME						
8	Contract Revenue		73,565.08	1,161,577.16		147,954.27	297,164.95
9	Over/Under Billings Adjustment		382.65	382.94			
10	Interest Income		485.8	1,943.18		45.65	246.13
11	Discounts Eamed		78.54	77.83			
12							
13							
14	COST OF CONSTRUCTION						
15	Job Labor		15,297.73	117,883.93		8,294.41	28,316.66
16	Job Marerial		8,622.43	164,731.60		30,673.81	48,182.74
17	Job Sub-Contract		31,815.64	661,287.88		83,750.63	165,626.26
18	Job Miscellanious		4,311.34	28,928.54		994.78	3,476.83
19	Job Burden (PR Taxes & Ins)		1,702.91	24,049.37		1,782.29	5,283.30

Fig. 7.10 A stego-spreadsheet in which the decimal fractions of the numeric items have been modified by embedded shares.

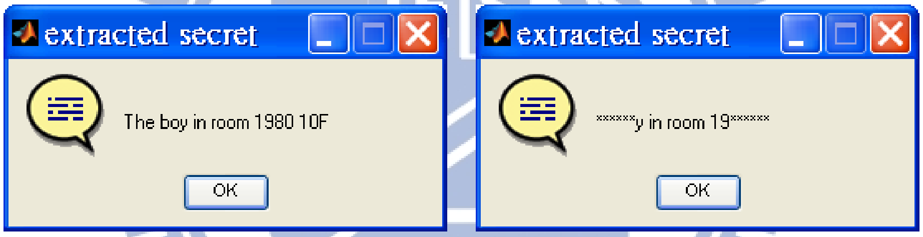


Microsoft Excel - stego_financialstat.xls

	A	B	C	D	E	F	G	H	I
1	DEMO Construction Co., Inc.								
2	INCOME STATEMENT								
3									
4			(This Year)			(Prior Year)			
5			Month	Year-to-Date		Month	Year-to-Date		
6									
7	INCOME								
8	Contract Revenue		76,661.66	1,161,577.16		147,954.27	297,164.95		
9	Over/Under Billings Adjustment		382.65	382.94					
10	Interest Income		485.8	1,943.18		33.98	246.13		
11	Discounts Eamed		78.54	77.83					
12									
13									
14	COST OF CONSTRUCTION								
15	Job Labor		15,297.73	117,883.93		8,294.41	28,316.66		
16	Job Marerial		8,622.43	164,731.60		30,673.81	48,182.74		
17	Job Sub-Contract		31,815.64	661,287.88		83,750.63	165,626.26		
18	Job Miscellanious		4,311.34	28,928.54		994.78	3,476.83		
19	Job Burden (PR Taxes & Ins)		8,887.56	24,049.37		1,782.29	5,283.30		
20									
21									
22									
23									

就緒 加總=85583.2

Fig. 7.11 A stego-spreadsheet with 3 numeric items (highlighted) being modified.



extracted secret

The boy in room 1980 10F

extracted secret

******y in room 19******

(a) (b)

Fig. 7.12 The recovered secret message. (a) Message extracted from the intact stego-spreadsheet shown in Fig. 7.10. (b) Message extracted from the modified stego-spreadsheet shown in Fig. 7.11.

Table 7.4 Experimental results of using strategy 2 for a cover spreadsheet of a financial statement.

Financial statement (32 numeric items and size 15K)	# of replaced numeric items	Result of Hypothesis (5%)	p value	Capacity= $\lfloor I/n \rfloor \times m \times k$ (bits)	Embedding bit rate per numeric item	Embedding bit rate
Embedding rate 100%	32	0 (cannot reject)	1	4×6×7=168	5.25 b	1/89

C. Comparison with existing Methods

For the purpose of presenting the contributions made in this study, a comparison of the capabilities of the proposed method with those of some existing covert communication methods is given in Table 7.5.

Most existing data hiding methods for covert communication [86-90, 91-92] were developed based on the premise that an adversary always works in the passive mode. However, the active attack model [86] mentioned previously that an adversary is allowed to introduce subtle modifications to passing-by stego-objects between the two parties is possible in practical covert communication. Contrastive with the existing methods, the proposed method is the only one which has the capability against active attacks and simultaneously takes the passive steganalytic attack into the consideration. Furthermore, the destructed part of a secret message caused by active attacks can be localized precisely by the proposed method, that is, the proposed method has the capability of modification localization which is useful for verifying the integrity of the secret message in the proposed method.

Furthermore, auxiliary information for message decoding is required in some methods like [86]. Extra storage space is thus required to save the information for both parties in the communication, adding a burden to the system in practical use. Contrarily, like the methods of [87-90, 91-92] the proposed method does not need any auxiliary information. In addition, the methods in [86, 91] increase the size of the generated stego-file due to the procedure of adding encoding codes or changing tracking records for data embedding. In contrast, the manipulation of substitution/replacement for data embedding used in methods of [87-90] as well as the proposed method keep the size of a cover file unchanged after it is transformed into a stego-version.

The embedding bit rate of the proposed method is comparatively smaller than that yielded by the methods of [87-89] using images as cover media. However, it is noted that these methods are vulnerable to the well-known RS steganalysis. This study aims at providing a new way of covert communication, and the issue of improving the embedding capacity deserves further investigation in the future.

Table 7.5 Comparison of existing steganographic methods and proposed method.

	Manipulation of data embedding	Against active attack	Modification localization capability	free from need of auxiliary information for message extraction	Keeping the size of a cover file after transformed into stego-version
[87, 88, 89]	LSB-based (image)	No	No	Yes	Yes
[90]	parities of palette colors (image)	No	No	Yes	Yes
[91]	Certain ASCII codes (PDF)	No	No	Yes	No
[92]	Character space varying (PDF)	No	No	Yes	Yes
[86]	Change tracking technique (MS word document)	No	No	No	No
Proposed method	Partial replacement of numeric items (spreadsheet)	Yes	Yes	Yes	Yes

7.8 Summary

A new covert communication method with a self-authentication capability for hiding data in spreadsheets using Shamir's $(k + 1, k)$ secret sharing scheme has been proposed in this study. The segment groups of a secret message are transformed into secret shares and then embedded as if they are part of the content in a cover spreadsheet, yielding a camouflage effect and generating a self-authentication capability. Each segment group of the secret message extracted from a stego-spreadsheet can be blindly authenticated by checking the results computed from all the $k + 1$ possible combinations of k shares out of $k + 1$ ones — if the resulting $k + 1$ copies of the recovered secret are all identical to one another, then the stego-spreadsheet is decided to be intact. In case the stego-spreadsheet is authenticated to have been modified, the altered part of the hidden secret message may be identified, and the untampered part recovered correctly. Experimental results have been shown to prove the feasibility and effectiveness of the proposed method. Derivations of the data embedding capacity and authentication precision have also been conducted, and discussions on the steganalysis issue included. Future studies may be directed to applications of the proposed method to multimedia protection in the field of fragile watermarking.

Chapter 8
A Blind Reversible Data Hiding Method Based on a New Iterative Histogram-Shifting Technique

8.1 Introduction

Reversible data hiding methods embed message data into host images and are capable of restoring resulting stego-images to their original states after the hidden data are extracted. Histogram shifting is an efficient technique used in many reversible data hiding methods. In this study, an iterative reversible data hiding method which is composed of two phases and yields high data embedding rates is proposed. The method utilizes the spatial similarity of neighboring pixels to create a difference histogram. In the first phase of histogram shifting, the peak point of the difference histogram is used to accommodate some message data, and in the second phase the value of the peak point, combined with the remaining message data, is embedded into the difference histogram again using another histogram-shifting scheme.

8.2 Merits of Proposed Method

The proposed method has the following merits: (1) yielding high data embedding capacities and low image distortions — with the two-phase histogram-shifting process, the hiding capacity of the proposed method is significantly improved, compared with that of [93], with the quality of the resulting stego-image kept at the same level as that of [93]; (2) utilizing the iteration technique — the data hiding process can be performed iteratively with the difference histogram resulting from the last iteration being taken as input to the current iteration; (3) having the advantageous property of blindness — before extracting the hidden data, the number of iterations, the value of the used peak point in each iteration, and the information of possible overflow and underflow pixels need not be known in advance; instead, one can just perform the proposed algorithms and all information needed to retrieve the hidden data can be gained during the data extraction process.

8.3 Proposed Data Embedding Method

The data embedding process of the proposed method is composed of two stages. In the first stage, a difference histogram is generated and in the second, given message data are hidden by two-phase histogram shiftings. Specifically, in the first stage, a given host image is divided into $n \times n$ blocks and the value of the central pixel of each block is subtracted from the values of the other pixels in the block to generate a set of difference values. The histogram of these difference values, called the *difference histogram*, is generated, which has a good property for embedding more data — most resulting difference values are centered on the value of zero and are good for use as data carriers via value shifting.

The second stage of the data embedding process includes two phases. In the first phase, the peak point x_0, supposed to be zero, of the difference histogram is used to embed the first part of the message data, and a new difference histogram is yielded subsequently. In the second phase, a new peak point x_0' is found from the new difference histogram and the neighboring points $x_0' \pm 1$ are then chosen to embed data, keeping the peak point x_0' intact. It is noted that in this phase the value of the peak point x_0 used in the first phase is regarded as message data and embedded before the remaining message data.

Accordingly, in the process of data extraction of the proposed method, firstly the new peak point x_0' of the resulting difference histogram is located and the neighboring points $x_0' \pm 1$ are checked to extract some hidden data which are composed of the peak value x_0 used in the first phase as well as part of the message data. Then, a similar reverse version of those operations conducted in the first phase of the data embedding process is performed to extract the remaining message data. The details of the ways we embed and extract the data are presented as two algorithms in the following.

Algorithm 1. Data embedding by iterative two-phase histogram shifting.

Input: a cover image I and a message string M in binary form.

Output: a stego-image I_s with M embedded.

Steps.

Stage 1 — generation of a difference histogram from the cover image.

Step 1. (*Block Division*) Divide cover image I into blocks of size $n \times n$.

Step 2. (*Generation of a difference histogram*) Perform the following steps to generate a difference histogram.

(1) For each block B in I, select the central pixel P_C of B and subtract the image value of P_C from those of the other pixels in B to generate a set of difference values, with each non-central pixel in B being *associated with a difference value*.

(2) Collect the associated difference values of all the pixels in I to produce a set N of ordered numbers, and generate accordingly a difference histogram h of N.

Stage 2 — data embedding to yield a stego-image.

Phase 1 --- embedding the message data using the original peak point.

Step 3. Find the peak in h and get its location x_0.

Step 4. (*Collecting the "central-valued" pixels*) Collect in a raster-scan order all pixels in I, whose associated difference values are equal to x_0, as a set S_0.

Step 5. (*Collecting the "left-side-valued" pixels*) Collect in a raster-scan order all pixels in I, whose associated difference values are smaller than x_0, as a set S_L.

Step 6. (*Shifting S_L*) Decrement the associated difference value of each pixel in S_L by one.

Step 7. (*Beginning of looping and message bit embedding*) Take an unprocessed bit m sequentially from input message string M, fetch in a raster scan order from I an unprocessed pixel P in S_0, and decrement the associated difference value v of P, which is x_0, by one if $m = 1$ or keep v unchanged if $m = 0$.

Step 8. (*End of looping*) If there exist unprocessed pixels in S_0, then go to Step 7; otherwise, denote the resulting difference histogram by h' and continue.

Phase 2 --- embedding the remaining message data by using side points near a new peak.

Step 9. (*Creation of an intermediate message string*) Construct a new message string M' by concatenating the data x_0 represented as an 8-bit string, and the remaining part of M.

Step 10. Find the peak in h' and get its location x_0'.

Step 11. (*Collecting "left-side-valued" and "right-side-valued" pixels*) Collect in a raster-scan order all the pixels in I with their associated difference values smaller than x_0' as a set S_L', and those with their associated difference values larger than x_0' as a set S_R'.

Step 12. (*Shifting* S_L' and S_R') Conduct the following two operations:

(1) Decrement the associated difference value of each pixel in S_L' by one.

(2) Increment the associated difference value of each pixel in S_R' by one.

Step 13. (*Beginning of looping and message bit embedding*) Take an unprocessed bit m' sequentially from the new message string M', fetch in a raster-scan order from I an unprocessed pixel P, denote its associated difference value by v, and perform the following operations:

(1) if $v = x_0' - 2$, then increment v by 1 when $m' = 1$ or keep v unchanged when $m' = 0$;

(2) if $v = x_0' + 2$, then decrement v by 1 when $m' = 1$ or keep v unchanged when $m' = 0$.

Step 14. (*End of looping*) If there exist unprocessed pixels in I, then go to Step 13; otherwise, conduct the following operations:

(1) if M' is not exhausted, then denote the resulting difference histogram by h and go to Step 3 of Phase 1;

(2) if M' is exhausted, then conduct the following operations to yield an output stego-image I_s and exit:

for each block B in I_s with its central pixel's value being v_C and for each non-central pixel P in B with image value v_i, if the associated difference value of P is v, then replace v_i with $v_i' = v_C + v$.

8.4 Proposed Data Extraction Method

In the processes of data extraction, a reverse version of the above data hiding process is performed. That is, in Stage 2, Phase 2 is executed first and then Phase 1. Since the peak point x_0' mentioned previously remains intact after data embedding in Phase 2, the value x_0' is used as the mark to indicate the location into which the data are embedded in the histogram.

Algorithm 2. Data extraction.

Input: a stego-image I_s yielded by Algorithm 1.

Output: the message string M in binary form embedded in I_s and the original cover image I from which I_s is produced.

Steps.

Step 1. (*Initialization*) Create two binary strings M_a and M_b with initial contents being empty.

Phase 1 (corresponding to Phase 2 in Algorithm 1) --- extracting the hidden message data using side points near a new peak.

Step 2. **(***Block Division***)** Divide stego-image I_s into blocks of size n×n.

Step 3. **(***Generation of a difference histogram***)** Perform the following steps to generate a difference histogram.

(3) For each block B in I_s, select the central pixel P_C of B and subtract the image value of P_C from those of the other pixels in B to generate a set of difference values, with each non-central pixel in B being *associated with a difference value.*

(4) Collect the associated difference values of all the pixels in I_s to produce a set N of ordered numbers, and generate accordingly a difference histogram

h of *N*.

Step 4. Find the peak in *h* and get its location x_0.

Step 5. (*Beginning of looping and message bit extraction*) Scan pixels in I_s in a raster-scan order, take an unprocessed pixel whose associated difference value v is equal to $x_0 - 1$, $x_0 + 1$, $x_0 - 2$, or $x_0 + 2$, and perform the following steps:

(1) if $v = x_0 - 2$ or $x_0 + 2$, then extract a bit "0" and append it to the end of M_b;

(2) if $v = x_0 - 1$, then extract a bit "1," append it to the end of M_b, and decrement v by one;

(3) if $v = x_0 + 1$, then extract a bit "1," append it to the end of M_b, and increment v by one.

Step 6. (*End of looping*) If there exist unprocessed pixels in I_s, then go to Step 5; otherwise, continue.

Step 7. Collect all pixels in I_s whose associated difference values are smaller than x_0 as a set S_L, and those whose associated difference values are larger than x_0 as a set S_R.

Step 8. (*Reverse histogram shifting*) Perform the following steps to generate a shifted difference histogram h_{tmp}:

(1) increment the associated difference value of each pixel in S_L by one;

(2) decrement the associated difference value of each pixel in S_R by one.

Phase 2 (corresponding to Phase 1 in Algorithm 1) --- extracting message data using the peak point.

Step 9. (*Peak value extraction*) Take the first eight bits $b_1b_2b_3b_4b_5b_6b_7b_8$ from M_b generated from Phase 1 above, and transform it into a decimal number, called p.

Step 10. (*Data extraction*) For h_{tmp}, perform the following operations.

3.1 (*Bit extraction*) Take sequentially an unprocessed pixel in I_s whose associated difference value v is equal to p or $p - 1$, and perform the following operations:

(1) if $v = p$, then extract a bit "0" and append it to the end of M_a;

(2) if $v = p - 1$, then extract a bit "1," append it to the end of M_a, and increment v by one.

3.2 Collect in a raster-scan order all pixels in I_s whose associated difference values are smaller than p as a set S_L'.

Step 11. (*Reverse histogram shifting*) Increment the difference value of each pixel in S_L' by one to generate a shifted difference histogram h_{tmp}'.

Step 12. Conduct the following operations to yield the original image I and exit:

for each block B in I_s with its central pixel's value being v_C and for each non-central pixel P in B with image value v_i, if the associated difference value of P is v, then replace v_i with $v_i' = v_C + v$.

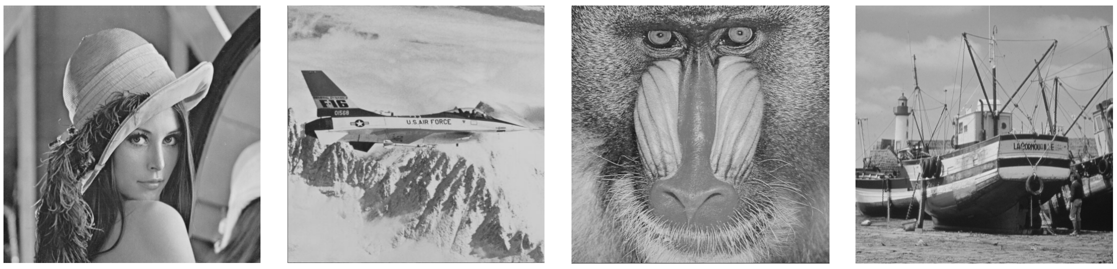
By Algorithm 2, the hidden message string can be extracted successfully, and the original host image recovered losslessly.

It is noted that the procedure described above may be perform iteratively. Specifically, the difference histogram resulting from this iteration can be taken as the input to the next iteration. When iterative hiding procedures are executed, it is suggested that a user may use a 32-bit string to record the total number of hidden bits. The 32-bit string, which follows the secret message string, is then hidden as well in the last iteration. In this way, in the later procedure of data extraction, the 32-bit string can be obtained after the initial phase of data extraction and so the total number of bits that need to be extracted subsequently will be known. This implies that a user can extracted all hidden data bits correctly without the knowledge of the number of iterations that have been done in the data hiding stage. In addition, the maximum number of bits that can be recorded by the 32-bit string is 2^{32} bits, which is large enough in general uses.

8.5 Experimental Results

In this study, the proposed method was compared mainly with the method proposed by Kim et al. [93] for demonstrating the effectiveness because their method outperforms other existing histogram-shifting-based reversible data hiding methods such as [94-95, 96-97] in terms of embedding capacity and image distortion. Furthermore, according to [93], the best performance of their method is produced when the sampling factors $\triangle u$ and $\triangle v$ in their method are set to be (3, 3), which mean the desired sub-sampling intervals in the row and column directions, respectively. Therefore, in the experiments conducted in this study, Kim et al.'s method with sampling factors (3, 3) is selected to be the objective for performance comparison.

In Fig. 8.1, four test images used in the experiments are shown and each of them is a grayscale one of size 512×512 with the gray values ranging from 0 through 255. To measure the performance of the proposed method, nine iterations of the proposed 2-phase data hiding process described in Algorithm 1 were carried out for each test image. And the block size of 3×3, which has been manifested in [97] as the optimal block size for the consideration of the spatial similarity of neighboring pixels, is chosen for use in the experiments. The corresponding statistics of the experimental results of the proposed method are given in Table 8.1. Similarly, nine levels of the data hiding process based on Kim et al.'s method were also implemented and the corresponding statistics of the experimental results are shown in Table 8.2. As can be observed, under the same circumstance of executing one level and one iteration by Kim et al.'s method and by the proposed method, respectively, the proposed method obtains improvements of 46.11%, 46.14%, 41.95%, and 76.55% on the data embedding capacity using the images of Lena, Airplane, Baboon, and Boat, respectively, and meanwhile maintains the image quality to be as good as that yielded by Kim et al.'s method. To observe further the performance comparison between the two methods, Figs. 8.2(a) through 8.2(d) are shown to illustrate the tendency in terms of the parameters of embedding capacity and image quality as the data hiding levels and iterations conducted by both methods increase.



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

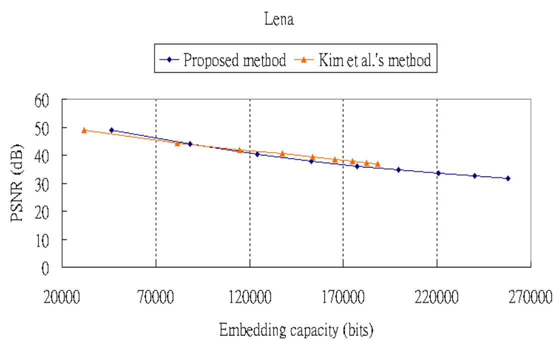
Fig. 8.1 Test images used in experiments. (a) Lena. (b) Airplane. (c) Baboon. (d) Boat.

Table 8.1 Statistics of experimental results of proposed method using block size 3×3.

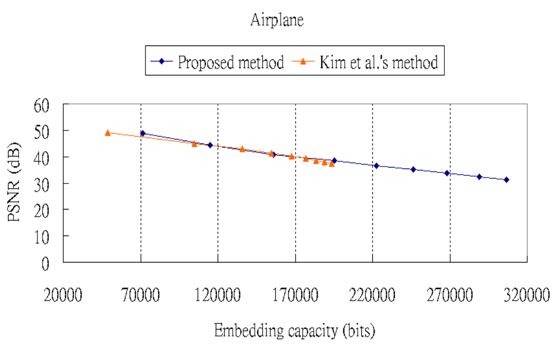
# of Levels	1	2	3	4	5	6	7	8	9
Lena (PSNR of marked image)	46561 (48.77)	88131 (43.95)	124217 (40.36)	152913 (37.92)	177199 (36.12)	199566 (34.63)	220520 (33.45)	240206 (32.70)	257669 (31.74)
Airplane (PSNR of marked image)	71279 (48.84)	115042 (44.41)	155967 (40.83)	195213 (38.43)	222685 (36.65)	246289 (35.20)	268296 (33.83)	288919 (32.51)	306730 (31.35)
Baboon (PSNR of marked image)	10290 (48.67)	24264 (42.95)	34447 (39.45)	44266 (37.03)	56889 (35.29)	66375 (33.79)	77697 (32.64)	86186 (31.58)	94460 (30.63)
Boat (PSNR of marked image)	37856 (49.47)	67420 (43.42)	96174 (40.16)	121255 (38.26)	140937 (36.57)	162148 (35.05)	179059 (33.53)	198443 (32.33)	213915 (31.26)

Table 8.2 Statistics of experimental results of Kim et al's method using the sampling factor ($\triangle u$, $\triangle v$)= (3, 3).

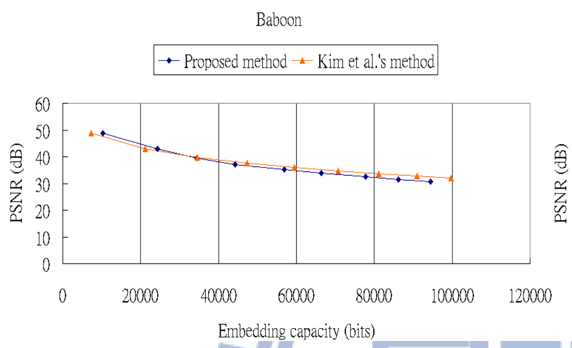
# of iterations	1	2	3	4	5	6	7	8	9
Lena (PSNR of marked-image)	31865 (48.98)	81511 (44.40)	114536 (42.00)	137326 (40.47)	153558 (39.37)	165641 (38.53)	174964 (37.86)	182264 (37.30)	188210 (36.85)
Airplane (PSNR of marked-image)	48774 (49.17)	104898 (45.06)	135903 (42.84)	154882 (41.34)	167828 (40.22)	176870 (39.32)	183720 (38.57)	189107 (37.94)	193484 (37.40)
Baboon (PSNR of marked-image)	7249 (48.72)	21141 (43.02)	34592 (39.80)	47424 (37.59)	59459 (35.95)	70783 (34.66)	81216 (33.62)	91008 (32.75)	99716 (32.01)
Boat (PSNR of marked-image)	21442 (48.87)	60903 (43.89)	93559 (41.32)	117827 (39.67)	135222 (38.46)	147692 (37.49)	156937 (36.66)	164112 (35.94)	170145 (35.33)



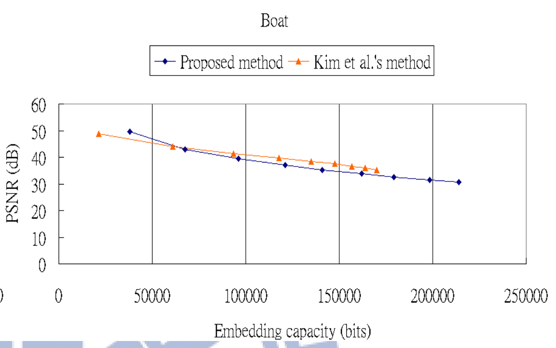
(a)



(b)



(c)



(d)

Fig. 8.2 Illustration of comparison of embedding capacity versus image distortion between the proposed method and Kim et al.'s method using (a) image Lena, (b) image Airplane, (d) image Baboon, and (d) image Boat.

At last, to deal with a general problem of pixel-value overflows and underflows during histogram shifting, the proposed method adopts a pre-processing strategy proposed in Tsai et al. [97] in which pixel values of "0" are incremented by one and those of "255" decremented by one. This pre-processing work is conducted at the beginning of every data embedding iteration so that the overflow/underflow problem will not occur.

8.6 Summary

A new histogram-shifting-based blind reversible data hiding method composed of two-phase iterations for more effective data hiding has been proposed. With the cooperation of two data hiding phases, the proposed method skillfully stores the value of the peak point yielded in Phase 1 via the use of Phase 2, solving a common problem of how to share the information of the used peak points between the sender and the recipient. By the development of such a two-phase data hiding process as well as the consideration of the spatial similarity of neighboring pixels, the proposed method significantly improves the data hiding capacity while keeping the low degradation of image quality when compared with other methods. Experimental results have been shown to prove the effectiveness of the proposed method.

Chapter 9
Conclusions and Suggestions for Future Works

In this thesis, various methods for the issue of multimedia security including image authentication, covert communication, and reversible data hiding have been proposed. Via the use of the PNG images, repairable image authentication techniques have been developed for grayscale document images and binary images. We also have developed a color image authentication method which is based on the proposed data hiding method. For grayscale images, a self-repairing image authentication using the bin code has been proposed. In addition, a covert communication method, which takes spreadsheets as the cover medium and uses (k, k+1)-threshold method to yield a blind self-authentication capability for the embedded secret, has been proposed. At last, we have proposed a blind reversible data hiding method which is based on a new iterative histogram-shifting technique.

Among the proposed methods, four are developed for image authentication, one for covert communication, and one based on reversible data hiding. In the following, conclusions of each method and suggestions for future researches are given as follows.

(1) A new blind image authentication method with a data repair capability for binary-like grayscale document images based on secret sharing has been proposed. Both the generated authentication signal and the content of a block are transformed into partial shares by the Shamir method, which are then distributed in a well-designed manner into an alpha channel plane to create a stego-image in the PNG format. In the process of image block authentication, a block in the stego-image is regarded as having been tampered with if the computed authentication signal does not match that extracted from corresponding partial shares in the alpha channel plane. For self-repairing of the content of a tampered block, the reverse Shamir scheme is used to compute the original content of the block from any two untampered shares. Future studies may be directed to choices of other block sizes and related parameters (prime number, coefficients for secret sharing, number of authentication signal bits, etc.) to improve data repair effects.

Applications of the proposed method to authentication and repairing of attacked color images may also be tried.

(2) A new secret-sharing-based authentication method for binary images with a data recovery capability has been proposed. A cover binary image is put under protection first by being transformed into the PNG format at the scaled sample depth of 8 with an alpha channel. Next, data for authentication and image content recovery are transformed into shares by the Shamir method, which then are distributed in a designed manner into the alpha channel. In the authentication process, a block is regarded as tampered if the current block content does not match the authentication data computed from the shares embedded in the alpha channel plane. For self-recovery of the content of a tampered block, a reverse Shamir scheme is used to compute the original content of the block from two shares collected from untampered blocks. By rescaling the sample depth and removing the alpha channel of the authenticated image, the original version of the 1-bit binary cover image can be regained. Future works may be directed to developing semi-fragile binary image authentication methods.

(3) A new type of data hiding via PNG images based on information sharing has been proposed. The Shamir's secret sharing method is used first in a novel way to generate partial shares from a given data string. The alpha-channel plane of a cover PNG image is utilized next to embed the partial shares, yielding a stego-image with undesirable white noise. The white noise is then eliminated skillfully by choosing a small prime number, dividing the input data string into 3-bit segments, and mapping computed share values into a range of large alpha-channel values which create high transparency. Moreover, detailed algorithms for applying the proposed method to color image authentication have been proposed. Shown by the results is the applicability of the proposed authentication algorithms to detect attacks implemented by common image-content alternation operations. Future studies may be directed to applications of the proposed method to copyright protection, digital rights management, recovery of altered image contents, etc.

(4) A grayscale image authentication method with a capability of localizing tampered image regions and repairing them at the pixel level has been proposed. Based on a bin-mapping scheme of dividing the 5-bit grayscale into eight bins, a 3-bit bin code is generated for use as an authentication signal for each input image pixel.

The authentication signals are embedded into other pixels selected randomly by a secret key. The signals are utilized not only for detecting and localizing tampered pixels but also for generating representative values for repairing the tampered pixels. This double-function merit of the authentication signal leads to the possibility of pixel-level tampering detection and the blindness characteristic of the proposed method. Also shown is a proof of the optimality of the proposed method in choosing three bits out of the eight ones of a pixel as an authentication signal under a minimax criterion of minimizing the maximum total gray-value distortion incurred by authentication signal embedding and tampered pixel repairing. Future works may be directed to extending the method to deal with color images.

(5) A new covert communication method with a self-authentication capability for hiding data in spreadsheets using Shamir's (k + 1, k) secret sharing scheme has been proposed in this study. The segment groups of a secret message are transformed into secret shares and then embedded as if they are part of the content in a cover spreadsheet, yielding a camouflage effect and generating a self-authentication capability. Each segment group of the secret message extracted from a stego-spreadsheet can be blindly authenticated by checking the results computed from all the k + 1 possible combinations of k shares out of k + 1 ones — if the resulting k + 1 copies of the recovered secret are all identical to one another, then the stego-spreadsheet is decided to be intact. In case the stego-spreadsheet is authenticated to have been modified, the altered part of the hidden secret message may be identified, and the untampered part recovered correctly. Future studies may be directed to applications of the proposed method to multimedia protection in the field of fragile watermarking.

(6) A new histogram-shifting-based blind reversible data hiding method composed of two-phase iterations for more effective data hiding has been proposed. With the cooperation of two data hiding phases based on histogram shifting, the proposed method skillfully stores the value of the peak point yielded in Phase 1 via the use of Phase 2, solving a common problem of how to share the information of the used peak points between the sender and the recipient. By the development of such a two-phase data hiding process as well as the consideration of the spatial similarity of neighboring pixels, the proposed method significantly improves the

data hiding capacity without degrading the image quality when compared with other methods. Future works may be directed to applications of the proposed method to losslessly image authentication.

References

[1] M. Wu and B. Liu, “Data hiding in binary images for authentication and annotation,” *IEEE Trans. Multimedia*, vol. 6, no. 4, pp. 528–538, Aug. 2004.

[2] H. Yang and A. C. Kot, “Pattern-based data hiding for binary images authentication by connectivity-preserving,” *IEEE Trans. Multimedia*, vol. 9, no. 3, pp. 475–486, April 2007.

[3] H. Yang and A. C. Kot, “Binary image authentication with tampering localization by embedding cryptographic signature and block identifier,” *IEEE Signal Processing Letters*, vol. 13, no. 12, pp. 741–744, Dec. 2006.

[4] H. Y. Kim and A. Afif, “Secure authentication watermarking for halftone and binary images,” *Int. J. of Imaging Systems & Technology*, vol. 14, no. 4, pp. 147–152, 2004.

[5] C. H. Tzeng and W. H. Tsai. “A new approach to authentication of binary images for multimedia communication with distortion reduction and security enhancement,” *IEEE Communications Letters*, vol. 7, no. 9, pp. 443–445.

[6] Y. Lee, J. Hur, H. Kim, Y. Park and H. Yoon, “A new binary image authentication scheme with small distortion and low false negative rates,” *IEICE Trans. on Communications*, vol. E90-B, no. 11, Nov. 2007.

[7] Y. Lee, H. Kim, and Y. Park, “A new data hiding scheme for binary image authentication with small image distortion,” *Information Science*, vol. 179, no. 22, pp. 3866–3884, Nov. 2009.

[8] M. U. Celik, G. Sharma, E. Saber and A. M. Tekalp, “Hierarchical watermarking for secure image authentication with localization,” *IEEE Trans. Image Process.*, vol. 11, no. 6, pp. 585-595, 2002.

[9] P. W. Wong and N. Memon, “Secret and public key image watermarking schemes for image authentication and ownership verification,” *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1593-1601, 2001.

[10] C. S. Lu and H. Y. M. Liao, “Structural digital signature for image authentication: an incidental distortion resistant scheme,” *IEEE Trans. Image Process.*, vol. 5, no. 2, pp. 161-173, 2003.

[11] C. S. Lu and H. Y. M. Liao, “Multipurpose watermarking for image authentication and protection,” *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1579-1592, 2001.

[12] C. H. Tzeng and W. H. Tsai, "A new technique for authentication of image/video for multimedia applications," *Proc. ACM Multimedia Workshops — Multimedia & Security: New Challenges*, Ottawa, Ontario, Canada, pp. 23-26, 2001.

[13] C. W. Yang and J. J. Shen, “Recover the tampered image based on VQ indexing,” *Signal Processing*, vol. 90, no. 1, pp. 331-343, Jan. 2010.

[14] T. Y. Lee and S. D. Lin, “Dual watermark for image tamper detection and recovery,” *Patt. Recog.*, vol. 41, pp. 3497-3506, 2008.

[15] S. H. Liu, H. X. Yao, W. Gao, and Y. L. Liu, “An image fragile watermark scheme based on chaotic image pattern and pixel-pairs,” *Appl. Math. Comput.*, vol. 185, no. 2, pp. 869-882, 2007.

[16] X. Zhang and S. Wang, “Statistical fragile watermarking capable of locating individual tampered pixels,” *IEEE Signal Process. Letters*, vol. 14, no. 10, pp. 727-730, Oct. 2007.

[17] X. Zhang and S. Wang, “Fragile watermarking scheme using a hierarchical mechanism,” *Signal Processing*, vol. 89, no. 4, pp. 675-679, 2009.

[18] C. W. Lee and W. H. Tsai, “A grayscale image authentication method with a pixel-level self-recovering capability against image tampering,” *Proc. 2011 IAPR Int’l Conf. on Machine Vision Applications*, Nara, Japan, pp. 328-331, June, 2011.

[19] M. Wu, H. Yu, and A. Gelman, “Multi-level data hiding for digital image and video,” *Proc. of SPIE Photonics East*, Boston, MA, USA, 1999.

[20] Gopalan, K., et al, “Covert speech communication via cover speech by tone insertion,” *Proc. of the 2003 IEEE Aerospace Conference*, Big Sky, MT, USA, March 2003.

[21] J. J. Chae and B. S. Manjunath, “Data hiding in Video,” *Proc. of 1999 IEEE International Conference on Image Processing*, Kobe, Japan, vol. 1, pp. 243-246, 1999.

[22] S. K. Kapotas, E. E. Varsaki, and A. N. Skodras, “Data hiding in H. 264 encoded video sequences,” *Proc. of IEEE 9th Workshop on Multimedia Signal Processing (WMSP 2007)*, Chania, Crete, Greece, pp. 373-376, Oct. 2007.

[23] W. Bender, D. Gruhl, N. Morimoto and A. Lu, “Techniques for data hiding,” *IBM Syst. J.*, vol. 35, no. 3-4, pp. 313-336, 1996.

[24] D. C. Wu and W. H. Tsai, “A steganographic method for images by pixel-value differencing,” *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1613–1626, 2003.

[25] C. H. Yang, C. Y. Weng, S. J. Wang and H. M. Sun, “Adaptive data hiding in edge areas of images with spatial LSB domain systems,” *IEEE Transactions on Information forensics and security*, vol. 3, no. 3, pp. 488-497, Sept. 2008.

[26] J. Fridrich and R. Du, “Secure steganographic methods for palette images,” *Proc. of 3rd International Workshop Information Hiding*, Dresden, Germany, Sept. 1999; also in *Lecture Notes in Computer Science*, Springer-Verlag, Berlin,

vol. 1768, pp. 61-76, 2000.

[27] J. T. Brassil and N. F. Maxemchuk, “Copyright protection for the electronic distribution of text Documents,” *Proc. IEEE*, vol. 87, no. 7, pp. 1181-1196, July 1999.

[28] I. S. Lee and W. H. Tsai, “A new approach to covert communication via PDF Files,” *Signal Processing*, Vol. 90, no. 2, pp. 557-565, 2010.

[29] S. Zhong, X. Cheng and T. Chen, “Data hiding in a kind of PDF texts for secret communication,” *International Journal of Network Security* vol. 4, no. 1, pp. 17-26, Jan. 2007.

[30] 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, 2007.

[31] W. Bender, D. Gruhl, N. Morimoto and A. Lu, “Techniques for data hiding,” *IBM Systems Journal*, vol. 35, nos. 3-4, pp. 313-336 , 1996.

[32] R. Z. Wang, C. F. Lin and J. C. Lin, “Image hiding by optimal LSB substitution and genetic algorithm,” *Pattern Recog.*, vol. 34, no. 3, pp. 671-683, 2000.

[33] J. Mielikainen, “LSB matching revisited,” *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 285-287, 2006.

[34] D. C. Wu and W. H. Tsai, “A steganographic method for images by pixel-value differencing,” *Pattern Recog. Letters*, vol. 24, no. 9-10, pp. 1613-1626, 2003.

[35] C. H. Yang, C. Y. Weng, S. Wang and H. M. Sun, “Adaptive data hiding in edge areas of images with spatial LSB domain systems,” *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 488-497, 2008.

[36] Y. N. Wang and A. Pearmain, “Blind image data hiding based on self reference,” *Pattern Recognition Letters*, vol. 25, no. 15, pp. 1681-1689, 2004.

[37] C. F. Wu and W. S. Hsieh, “Digital watermarks using zerotree of DCT,” *IEEE Transactions on Consumer Electronics*, vol. 46, no. 1, pp. 87-94, 2000.

[38] Y. P. Wang, M. J. Chen and P. Y. Cheng, “Robust image watermark with wavelet transform and spread spectrum techniques,” *Proceedings of Thirty-Fourth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, pp. 1846-1850, 2000.

[39] A. Lumini and D. Maio, “A wavelet-based image watermarking scheme,” *Proceedings of Int. Conf. on Information Technology: Coding and Computing*, Las Vegas, Nevada, USA, pp. 122-127, 2000.

[40] S. H. Wang and Y. P. Lin, “Wavelet tree quantization for copyright protection watermarking,” *IEEE Trans. Image Processing*, vol. 13, no. 2, pp. 154-165, 2004.

[41] C. M. Pun, “A novel DFT-based digital watermarking system for images,” *Proceedings of 8th Int. Conf. on Signal Processing*, Guilin, Yunnan, China, pp.

1245-1248, 2006.

[42] S. Pereira and T. Pun, “Robust template matching for affine resistant image watermarks,” *IEEE Trans. on Image Processing*, vol. 9, no. 6, pp. 1123-1129, 2000.

[43] P. H. W. Wong, O. C. Au and J. W. C. Wong, “A Data Hiding Technique in JPEG Compressed Domain,” *Proceedings of 3rd SPIE Int. Conf. Security and Watermarking of Multimedia Contents*, San Jose, CA, USA, pp. 309-320, 2001.

[44] J. H. Lee, J. W. Chen and M. Y. Wu, “A Data Hiding Scheme to Reconstruct Missing Blocks for JPEG Image Transmission,” *Springer-Verlag Lecture Notes in Artificial Intelligence (LNAI)*, vol. 3682, pp. 554-559, 2005.

[45] L. S. T. Chen, S. J. Lin and J. C. Lin, “Reversible JPEG-based hiding method with high hiding-ratio,” *Int. Journal of Pattern Recog. and Artificial Intelligence*, vol. 24, pp. 1-23, 2010.

[46] J. Fridrich and R. Du, “Secure Steganographic Methods for Palette Images,” *Proc. 3rd Information Hiding Workshop, LNCS, Springer-Verlag*, NY, USA, pp. 47-60, 2000.

[47] C. H. Tzeng, Z. F. Yang and W. H. Tsai, “Adaptive data hiding in palette images by color ordering and mapping with security protection,” *IEEE Trans. on Communications*, vol. 52, no. 4, pp. 791-800, 2004.

[48] X. Zhang, S. Wang, and Z. Zhou, “Multibit assignment steganography in palette images,” *IEEE Signal Processing Letters*, vol. 15, pp. 553-556, 2008.

[49] J. H. Lee and M. Y. Wu, “Reversible Data-hiding method for palette-based images,” *Optical Engineering*, vol. 47, pp. 047008-1-047008-9, 2008.

[50] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, “Reversible data hiding,” *IEEE Trans. On Circuits Systems and Video Technology*, vol. 16, no.3, pp. 354-362, Mar. 2006.

[51] W. C. Kuo, D. J. Jiang, and Y. C. Huang, “A reversible data hiding scheme based on block division,” *Proceedings of 2008 International Congress on Image and Signal Processing (CISP 2008)*, Sanya, Hainan, China, vol. 1, pp. 365-369, 2008.

[52] C. W. Lee and W. H. Tsai, “A lossless large-volume data hiding method based on histogram shifting using an optimal hierarchical block division scheme,” *Journal of Information Science and Engineering*, vol. 27, no. 4, pp. 1265-1282, 2011.

[53] M. Fallahpour and M. H. Sedaaghi, “High capacity lossless data hiding based on histogram modification,” *IEICE Electron. Express*, vol. 4, no. 7, pp. 205-210, Apr. 2007.

[54] P. Y. Tsai, Y. C. Hu, and H. L. Yeh, “Reversible Image Hiding Scheme Using Predictive Coding and Histogram Shifting,” *Signal Processing*, vol. 89, no. 6, pp. 1129-1143, 2009.

[55] K. S. Kim, M. J. Lee, H. Y. Lee, and H. K. Lee, “Reversible data hiding

exploiting spatial correlation between sub-sampled images," *Pattern Recognition*, vol. 42, no. 11, pp. 3083-3096, 2009.

[56] A. Shamir, "How to share a secret," *Communication of ACM*, vol. 22, pp. 612–613, 1979.

[57] W3C PNG Development Group (2003, Oct.). *Portable Network Graphics (PNG) Specification (Second Edition), Information technology — Computer graphics and image processing — Portable Network Graphics (PNG): Functional specification. ISO/IEC 15948: 2003 (E).* Available: http://www.w3.org/TR/PNG/

[58] C. C. Lin and W. H. Tsai, "Secret image sharing with steganography and authentication," *Journal of Systems and Software*, vol. 73, pp. 405–414, 2004.

[59] W. H. Tsai, "Moment-preserving thresholding: a new approach," *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 3, pp. 377-393, 1985.

[60] M. Holliman and N. Memon, "Counterfeiting Attacks on Oblivious Block-wise Independent Invisible Watermarking Schemes," *IEEE Trans. Image Process.*, vol. 9, no. 3, pp. 432̃441, 2000.

[61] C. S. Lu and H. Y. M. Liao, "Multipurpose watermarking for image authentication and protection," *IEEE Trans. on Image Processing*, vol. 10, no. 10, pp. 1579-1592, 2001.

[62] G. J. Yu, C. S. Lu, and H. Y. M. Liao, "Mean quantization-based fragile watermarking for image authentication," *Optical Engineering*, vol. 40, no. 7, pp. 1396-1408, 2001.

[63] J. O. Ruanaidh and T. Pun, "Rotation, scale and translation invariant spread spectrum digital image watermarking," *Signal Processing*, vol. 66, no. 3, pp. 303-317, 1998.

[64] M. Barni, F. Bartolini and T. Furon, "A general framework for robust watermarking security," *Signal Processing*, vol. 83, no. 10, pp. 2069-2084, 2003.

[65] I. S. Lee and W. H. Tsai, "A new approach to covert communication via PDF Files," *Signal Processing*, vol. 90, no. 2, pp. 557-565, 2009.

[66] 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, 2007.

[67] Z. Peng and W. Liu, "Color image authentication based on spatiotemporal chaos and SVD," *Chaos, Solitons and fractals*, vol. 36, pp. 946-952, 2008.

[68] S. C. Byun, I. L. Lee, T. H. Shin and B. H. Ahn, "A public-key based watermarking for color image authentication," *IEEE Int. Conf. Multimedia and Expo*, vol. 1, pp. 593-596.

[69] C. W. Yang and J. J. Shen, "Recover the tampered image based on VQ indexing," *Signal Processing*, vol. 90, no. 1, pp. 331-343, Jan. 2010.

[70] S. H. Liu, H. X. Yao, W. Gao, and Y. L. Liu, "An image fragile watermark scheme based on chaotic image pattern and pixel-pairs," *Appl. Math. Comput.*, vol. 185, no. 2, pp. 869-882, 2007.

[71] S. S. Wang and S. L. Tsai, "Automatic image authentication and recovery using fractal code embedding and image inpainting," *Patt. Recog.*, no. 41, pp. 701-712, Feb., 2008.

[72] P. L. Lin, P. W. Huang and A. W. Peng, "A fragile watermarking scheme for image authentication with localization and recovery," *Proc. IEEE 6th Int'l Symp. on Multimedia Software Eng.,* Miami, Florida, USA, pp. 146-153, Dec., 2004.

[73] P. L. Lin, C. Hsieh and P. Huang, "A hierarchical digital watermarking method for image tamper detection and recovery," *Patt. Recog.,* no. 38, pp. 2519-2529, 2005.

[74] Y. Park, H. Kang, K, Yamaguchi and K. Kobayashi, "Watermarking for tamper detection and recovery," *IEICE Electronic Express*, vol. 5, no. 17, pp. 689-696, Sept. 2008.

[75] Y. J. Chang, Ran Zan Wang and J. C. Lin, "A sharing-based fragile watermarking method for authentication and self-recovery of image tampering," *EURASIP Journal on Advances in signal processing*, vol. 2008, pp. 1-17, Jan., 2008.

[76] B. Mahdian. and S. Saic, "Blind authentication using periodic properties of interpolation," *IEEE Trans. Inf. Forensics Security* vol. 3, no. 3, pp. 529-538, 2008.

[77] P. W. Wong and N. Memon, "Secret and public key image watermarking schemes for image authentication and ownership verification," *IEEE Trans. Image Process.*, vol. 10, no. 10, pp. 1593-1601, 2001.

[78] X. Zhang and S. Wang, "Statistical fragile watermarking capable of locating individual tampered pixels," *IEEE Signal Process. Letters*, vol. 14, no. 10, pp. 727-730, Oct. 2007.

[79] M. Wu, H. Yu, and A. Gelman, "Multi-level data hiding for digital image and video," *Proc. of SPIE Photonics East*, Boston, MA, USA, 1999.

[80] Gopalan, K., et al, "Covert speech communication via cover speech by tone insertion," *Proc. of the 2003 IEEE Aerospace Conference*, Big Sky, MT, USA, March 2003.

[81] J. J. Chae and B. S. Manjunath, "Data hiding in Video," *Proc. of 1999 IEEE International Conference on Image Processing*, Kobe, Japan, vol. 1, pp. 243-246, 1999.

[82] S. K. Kapotas, E. E. Varsaki, and A. N. Skodras, "Data hiding in H. 264 encoded video sequences," *Proc. of IEEE 9th Workshop on Multimedia Signal Processing (WMSP 2007)*, Chania, Crete, Greece, pp. 373-376, Oct. 2007.

[83] C. C. Thien and J. C. Lin, "Secret image sharing," *Computers and Graphics*, vol. 26, no. 1, pp. 765-770, 2002.

[84] L. S. T. Chen and J. C. Lin, "Multithreshold progressive image sharing with compact shadows," *Journal of Electronic Imaging*, vol. 19, no. 1, p. 013003, 2010.

[85] C. W. Lee and W. H. Tsai, "Authentication of binary document images in PNG format based on a secret sharing technique," *Proceedings of 2010 International Conference on System Science and Engineering (ICSSE 2010)*, p. 133, Taipei, Taiwan.

[86] 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, 2007.

[87] W. Bender, D. Gruhl, N. Morimoto and A. Lu, "Techniques for data hiding," *IBM Syst. J.*, vol. 35, no. 3-4, pp. 313-336, 1996.

[88] D. C. Wu and W. H. Tsai, "A steganographic method for images by pixel-value differencing," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1613–1626, 2003.

[89] C. H. Yang, C. Y. Weng, S. J. Wang and H. M. Sun, "Adaptive data hiding in edge areas of images with spatial LSB domain systems," *IEEE Transactions on Information forensics and security*, vol. 3, no. 3, pp. 488-497, Sept. 2008.

[90] J. Fridrich and R. Du, "Secure steganographic methods for palette images," *Proc. of 3rd International Workshop Information Hiding*, Dresden, Germany, Sept. 1999; also in *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, vol. 1768, pp. 61-76, 2000.

[91] I. S. Lee and W. H. Tsai, "A new approach to covert communication via PDF Files," *Signal Processing*, Vol. 90, no. 2, pp. 557-565, 2010.

[92] S. Zhong, X. Cheng and T. Chen, "Data hiding in a kind of PDF texts for secret communication," *International Journal of Network Security* vol. 4, no. 1, pp. 17-26, Jan. 2007.

[93] K. S. Kim, M. J. Lee, H. Y. Lee, and H. K. Lee, "Reversible data hiding exploiting spatial correlation between sub-sampled images," *Pattern Recognition*, vol. 42, no. 11, pp. 3083-3096, 2009.

[94] Z. Ni, Y. Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. On Circuits Systems and Video Technology*, vol. 16, no.3, pp. 354-362, Mar. 2006.

[95] W. C. Kuo, D. J. Jiang, and Y. C. Huang, "A reversible data hiding scheme based on block division," *Proceedings of 2008 International Congress on Image and Signal Processing (CISP 2008)*, Sanya, Hainan, China, vol. 1, pp. 365-369, 2008.

[96] M. Fallahpour and M. H. Sedaaghi, "High capacity lossless data hiding based on

histogram modification," *IEICE Electron. Express*, vol. 4, no. 7, pp. 205-210, Apr. 2007.

[97] P. Y. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible Image Hiding Scheme Using Predictive Coding and Histogram Shifting," *Signal Processing*, vol. 89, no. 6, pp. 1129-1143, 2009.

Vitae

Che-Wei Lee was born in Kaohsiung, Taiwan, R.O.C on October 21, 1980. He received the B. S. degree in civil engineering and the M. S. degree in electrical engineering from National Cheng Kung University in 2002 and 2005, respectively. He works toward his Ph. D degree in the Department of Computer Science at National Chiao Tung University since 2005. His research interests include image processing, multimedia information hiding, and video technologies.

List of Publications

Journal Papers

(1) **C. W. Lee** and W. H. Tsai, “A Secret-Sharing-Based Method for Authentication of Grayscale Document Images via the Use of the PNG Image with a Data Repairing Capability,” *IEEE Transactions on Image Processing*, vol. 21, no. 1. pp. 207-218, 2012.

(2) **C. W. Lee** and W. H. Tsai, “An Optimal Pixel-level Self-repairing Authentication Method for Grayscale Images under a Minimax Criterion of Distortion Reduction,” *Optical Engineering*, vol. 15, no. 5, pp. 057006-057006-10, 2012.

(3) **C. W. Lee** and W. H. Tsai, “A Lossless Large-volume Data Hiding Method Based on Histogram Shifting Using An Optimal Hierarchical Block Division Scheme,” *Journal of Information and Science and Engineering*, vol. 27, no. 4, pp. 1265-1282, 2011.

(4) **C. W. Lee** and W. H. Tsai, “A Steganographic Method Based on Information Sharing for Hiding Secret Data in Spreadsheets with an Authentication Capability,” *Journal of Systems and Software*, minor revision for publication.

(5) **C. W. Lee** and W. H. Tsai, “A New Approach to Binary Image Authentication via Uses of PNG Images and Secret Sharing Technique with an Image Recovery Capability,” *IEEE Transactions on Multimedia*, revised.

(6) **C. W. Lee** and W. H. Tsai, “A Data Hiding Method Based on Information Sharing via PNG Images for Applications of Color Image Authentication and Metadata Embedding,” *Signal Processing*, revised.

(7) **C. W. Lee** and W. H. Tsai, “A Blind Reversible Data Hiding Method Based on a New Iterative Histogram-Shifting Technique,” submitted to *IEEE Transactions on Circuits and Systems for Video Technology*.

Conference Papers

(1) **C. W. Lee** and W. H. Tsai, "A new steganographic method based on information sharing via PNG images," *Proceedings of 2nd International Conference on Computer and Automation Engineering* (*2nd ICCAE*), pp. 870-874, vol. 5, Singapore.

(2) **C. W. Lee** and W. H. Tsai, "Authentication of binary document images in PNG format based on a secret sharing technique," *Proceedings of 2010 International Conference on System Science and Engineering* (*ICSSE 2010*), p. 133, Taipei, Taiwan.

(3) **C. W. Lee** and W. H. Tsai, "A grayscale image authentication method with a pixel-level self-recovering capability against image tampering," *Proceedings of 12th IAPR Conference on Machine Vision Applications* (*MVA 2011*), Nara, Japan.

(4) **C. W. Lee** and W. H. Tsai, "A New Lossless Visible Watermarking Method via the Use of the PNG Image," in *Proc. 15th IASTED International Conference on Artificial Intelligence and Soft Computing*, pp. 195-199, Napoli, Italy.

Book Chapter

(1) **C. W. Lee** and W. H. Tsai, "A lossless data hiding method by histogram shifting based on an adaptive block division scheme," *in Pattern Recognition and Machine Vision – in Honor and Memory of the Late Professor King-Sun Fu,* Patrick S. P. Wang (Ed.), River Publishers, Aalborg, Denmark, pp. 1–14.

Patents

(1) **C. W. Lee** and W. H. Tsai, "A covert communication method based on information sharing via PNG images," *Republic of China Patent* (pending).

(2) **C. W. Lee** and W. H. Tsai, "A covert communication method based on information sharing via PNG images," *USA Patent* (pending).