行政院國家科學委員會專題研究計畫　成果報告

數位典藏資訊之版權保護與驗證技術之研究(二)

計畫類別：個別型計畫
計畫編號：NSC92-2422-H-009-010-
執行期間：92 年 03 月 01 日至 93 年 02 月 28 日
執行單位：國立交通大學資訊科學學系

計畫主持人：蔡文祥
共同主持人：吳大鈞

報告類型：完整報告

處理方式：本計畫可公開查詢

中　華　民　國 93 年 5 月 18 日

行政院國家科學委員會補助專題研究計畫
成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　數位典藏資訊之版權保護與驗證技術之研究（二）　※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　A Study on Copyright Protection & Authentication　※
※　Techniques for Digital Information Archiving（2）　※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：個別型計畫
計畫編號：NSC92－2422－H－009－010
執行期間：92 年 3 月 1 日至 93 年 2 月 29 日

計畫主持人：交通大學蔡文祥教授兼副校長
共同主持人：高雄第一科技大學吳大鈞副教授

執行單位：國立交通大學資訊科學系

中　華　民　國　93　年　05　月　25　日

目錄

第一部份：成果報告成果略述……………………………………………………………………01

第二部分：計畫技術內容概述……………………………………………………………………03

一、視訊資料之資訊隱藏技術之開發………………………………………………………03

二、浮水印植入軟體之多家授權機制及技術之開發………………………………………04

三、影像及視訊多家授權機制及技術之開發………………………………………………04

四、影像認證中心之運作……………………………………………………………………05

第三部份：已開發技術詳述………………………………………………………………………06

Chapter 1—Data Hiding in MPEG Videos for Covert Communication…………………………07

Chapter 2—Content Verification of MPEG Videos by Random Signal Hiding………………33

Chapter 3—Multiple Authorizations for Images, Videos or Software Packages ……….....…56

Chapter 4—Video Authentication for Copyright Claim…………………………………………63

附件一：科學博物館所提建議之回覆……………………………………….……….....……66

附件二：交大與台中自然科學博物館會議記錄…………………………………………………67

附件三：參加「第二屆數位典藏技術研討會」發表之論文—

“Hiding Authenticable General Digital Information behind Binary Images with Reduced Distortion” ………………………………………………………………………………69

附件四：交通大學影像認證中心註冊證明書……………………………………………………80

第一部份 期末執行成果略述

本計畫「數位典藏資訊之版權保護與驗證技術之研究」執行時間為 92 年 3 月 1 日至 93 年 2 月 28 日，需完成之工作項目包括：

1．視訊資料之資訊隱藏技術之開發；
2．浮水印植入軟體之多家授權機制及技術之開發；
3．影像及視訊多家授權機制及技術之開發;
4．影像認證中心之運作。

針對上述四項技術，本計畫團隊經過一年積極的研究，已經完成視訊資料之資訊隱藏技術之開發、浮水印植入軟體之多家授權機制及技術之開發、影像多家授權機制技術之開發、視訊多家授權機制，以及影像認證中心之運作等成果。

此外，在去年 7 月 18 日，本計畫團隊與推廣辦公室合辦「數位典藏專業培訓課程系列四—數位典藏版權資訊研討會」，邀集數位典藏單位以及相關學術單位共二十多個單位參加。除了講解版權保護與驗證技術之發展歷程及現況之外，同時也發給參加人員「資訊保護家 2.0 版」應用軟體，讓各單位能在現場實際操作及發現問題。本計畫團隊也針對此次蒐集的問題詳加研究並予以改進。

同時，本計畫也與台中自然科學博物館進行合作，除了去年與其數位典藏人員進行交流，並聽取其意見，計畫團隊也針對其建議改善資訊保護家軟體，其意見與改善情形詳見附件一；此外，去年計畫團隊在發展出初步研究成果之際，於 92 年 11 月 14 日前往自然科學博物館展示軟體，並再次與其計畫人員交換意見、進行研討，本計畫綜合整理此次科博館對於目前軟體所提供的意見（詳見附件二），同時針對科博館的意見，也提出解決的方法並持續修改軟體，以期更加符合使用者的需求。

而在去年 7 月 22 日、23 日由數位典藏技術分項所舉辦的第二屆數位典藏技術研討會中，本計畫發表了一篇論文—「Hiding Authenticable General Digital Information Behind Binary Images with Reduced Distortion」，其詳細內容如附件三。

為了讓典藏單位能更了解浮水印的技術內涵，本計畫在今年 11 月 1 日出刊的《國家數位典藏通訊》中，介紹本計畫所研發之軟體及相關技術。

此外依所研究的影像認證中心之運作，本計畫在一年前成立了交通大學影像認證中心，而在本年度內，本中心設計了註冊證明書（詳見附件四），作為典藏單位到認證中心來註冊的憑據。

由於目前本計畫正積極的發展「資訊保護家 3.0 版」軟體，希望能將這一年所發展的技術整合至軟體中，並在今年再次舉辦軟體訓練課程，讓本計畫團隊所發展的資訊保護技術能

對數位典藏單位有實質上的幫助。

值得一提的是本計畫團隊參加 2004 年數位典藏國家型科技計畫浮水印技術評比競賽獲得「一般彩色影像組」與「大型彩色影像組」第一名。

第二部分：計畫技術內容概述

在技術部分，本計畫所需完成之工作項目有視訊資料之資訊隱藏技術之開發、浮水印植入軟體之多家授權機制及技術之開發、影像及視訊多家授權機制及技術之開發、以及影像認證中心之運作等四項，在這個部分將針對這些技術作概略的說明。

一、視訊資料之資訊隱藏技術之開發

視訊資料之資訊隱藏技術研發，主要進行下列兩項工作。

A. 以資料隱藏技巧植入註解資料於 MPEG 檔案中

由於視訊的傳輸必須要經過壓縮後才能達到快速傳輸的效果，因此，視訊的註解資訊隱藏的技術首先要考慮到視訊的壓縮。目前有關於視訊隱藏的研究中，所能隱藏的機密資訊限制在某幾類可以容忍失真的資訊形式上，如聲音、靜態與動態影像，因此本計畫的目的之一便是研究視訊隱藏任意格式秘密資訊的可能性。一般來說，若要將一秘密資訊隱藏在視訊中，可選擇在壓縮前、壓縮時以及壓縮後三種不同時機進行藏入動作。而資料取出的方法，相對也有三種不同進行時機與之對應。本計畫團隊所採取的方式是在壓縮後將秘密資訊藏入視訊檔案之中。

B. 以資料隱藏技巧植入認證資料於 MPEG 檔案中

由於認證浮水印較數位簽章的認證方式有更高的可攜性，因此，在本計畫中，我們採用易碎浮水印的機制，研究的方法是根據人類視覺模式，將易碎浮水印以肉眼不可見的方式植入視訊資料中。植入時並考慮幾何座標的位置，即在視訊某部分被修改的情況下，也能夠影響所植入的浮水印，並根據浮水印的破壞情況，可以判別被竄改的部分。我們浮水印認證方式的設計符合以下的需求：

(1) 就一個畫面而言，我們將所隱藏之認證資訊必須平均分布於整個畫面中。因為畫面中的每一個位置均有可能會被改變，一旦畫面內容被改變，所隱藏的認證資訊也跟著被破壞。

(2) 當認證資訊隱藏到畫面中，會造成畫面的品質少部份的破壞或失真，所以我們所使用的隱藏技術必須對畫質的影響不大。

(3) 認證資訊添加的過程所需要的時間愈少愈好。

如上所述，視訊資料量龐大，通常會以壓縮的方式以便儲存與傳輸。因此，在隱藏認證浮水印時，也需要考慮到隱藏的時機。為了快速的添加浮水印，我們採取在壓縮後才進行藏入的方式，藉著分析壓縮標準的編碼方式，在藏入浮水印時，將原來的編碼以另一組編碼取代，而不破壞視訊的可看性。在認證時，可以直接由編碼所代表的數值得到所隱藏的浮水印，並進行認證工作。此方式的好處是在添加和認證的過程中，不需要解壓縮的動作，不僅可以達到認證的效果，也可以提高效率。

二、浮水印植入軟體之多家授權機制及技術之開發

本計畫團隊今年度發展出「資訊保護家 2.0 版」應用軟體，提供「數位典藏國家型科科技計畫」之主題計畫將其 BMP、JPG 以及 GIF、TIFF 無壓縮等格式的靜態影像，以不同的方法藏入多種版權資訊，以及進行自我驗證的動作。藉由「資訊保護家 2.0 版」，使用者可針對上述四種數位影像格式，嵌入多種資訊以達到不同的功效。這些資訊分別為「註解資訊」、「可視版權保護浮水印」、「不可視版權保護浮水印」，以及「驗證資訊」四種。

先前使用的「資訊保護家 1.0 版」應用軟體採用不同的序號來管理，擁有此套軟體者，能對其擁有影像執行藏入多種資訊及進行自我驗證的動作。而沒有此軟體者就無法一窺影像中之奧秘。使用此軟體亦可對先前已藏入資訊之影像再藏入新資料，此時系統會覆蓋原先藏入之舊資料。但倘若某個影像 A 已由「典藏機構甲」加入浮水印資訊而得影像 B，而「典藏機構乙」若沒有先行驗證影像 B 中之浮水印資訊，仍可用其擁有的軟體對影像 B 執行藏入新的浮水印的動作而得影像 C，新浮水印資訊會覆蓋掉原「典藏機構甲」所加入之浮水印，也就是說影像 C 中無法得到「典藏機構甲」的浮水印。這時「典藏機構甲」與「典藏機構乙」可能就會產生糾紛。當然，如果「典藏機構甲」曾經將影像在本認證中心註冊則自然可以判定版權。

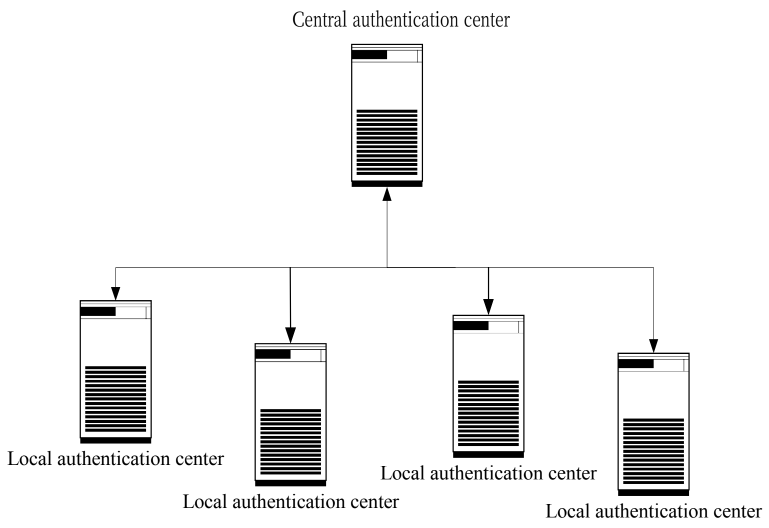
本計畫為避免此種因不小心而產生的典藏困擾與版權糾紛，發展出多家授權機制及技術，並整合至「資訊保護家 2.0 版」。本計畫團隊所採行的做法是給每一典藏機構所擁有「資訊保護家」應用軟體一個不同的設定值。當影像由某典藏機構之「資訊保護家」植入浮水印的同時，也會自動植入該設定值。當任何影像要植入浮水印訊號時，系統會先自動查驗此影像中是否已經存在不是該典藏機構的設定值，如果有的話則無法對此影像進行植入新資料之動作。加入此機制後，因為只有使用原機構的「資訊保護家」才能對此己加入設定值的影像植入新浮水印資訊，所以就不會發生前述糾紛。影像自我驗證及抽取資訊等動作則不檢查該設定值，也就是說各典藏機構都可以利用其所擁有的「資訊保護家」查看其他典藏機構的影像中之浮水印資訊。

三、影像及視訊多家授權機制及技術之開發

同一張影像(或視訊)若授權給多家機構運用，有可能發生其中某一授權機構侵權，卻無法得知是哪一機構違法。舉例言之，某一機構可能將所獲授權影像(或視訊)再出售圖利，但影像(或視訊)被發現時，卻無法得知是那一機構出售的，因為有多家機構皆擁有該授權影像(或視訊)。因此我們藉由浮水印技術，在影像(或視訊)中藏入註解資訊來判定典藏單位的授權機制，舉例來說，就是在註解資訊中藏入授權單位名稱、授權時間、授權內容合約等，以此來達到影像(或視訊)多家授權的機制。

四、 影像認證中心之運作

在此我們希望建立一套視訊認證機制，用以保護數位資料的完整性及其版權。。我們所提出的視訊的認證程序和認證中心是一個兩層的架構，最上層的稱為 Central AC，底層的稱為 Local AC，下圖則是認證中心的架構圖。Central AC 是整個認證中心架構的最頂端，當底下的 Local AC 發生爭議時就必須請求 Central AC 作公正的驗證工作。而 Local AC 則是一般的片商、多媒體工作室或是電視台…等需要作視訊認證的地方。


Central authentication center
Local authentication center
Local authentication center
Local authentication center
Local authentication center

第三部分 已開發技術詳述

Chapter 1
Data Hiding in MPEG Videos for Covert Communication

1.1 Introduction

Due to the high availability of the Internet and the advance of multimedia techniques, there are many applications of digital data on the network. For example, images may be used for covert communication by data hiding techniques. However, the data hiding capacity of one image is usually not large. When the amount of secret data to be transmitted is huge, we have to use a lot of images for covert communication. A video usually consists of a long series of images, so we propose in this study a video data hiding method for the covert communication of large amounts of data. The proposed method is introduced in this chapter. In Section 1.1.1, some related problem definitions are given, and in Section 1.1.2 the basic ideas of the proposed method are presented. In Section 1.2, the proposed data hiding method is described, and the corresponding data extraction method is stated in Section 1.3. In Section 1.4, several experimental results are shown to prove the feasibility of the proposed method. Finally, some discussions and a summary of the proposed method are made in the last section of this chapter.

1.1.1 Problem Definition

When applying video data hiding techniques for covert communication, the data hiding capacity and the imperceptibility of the hidden data are two of the major concerns. In most existing methods, DCT coefficients in the frequency domain are used to hide data. However, not all types of frames of MPEG videos are suitable for data embedding in the DCT domain. At least, it is not easy to embed data in inter-coded frames, including the P and B frames that are coded by motion compensated prediction. In the MPEG standard, motion compensation prediction is adopted to reduce temporal redundancy, so that most DCT coefficients are zero and so are not encoded in P and B frames. And this is the reason why it is hard to hide data in P or B frames.

A solution for this problem is just to use the I frames of MPEG videos to hide data. However, most of the frames in an MPEG video sequence are inter-coded frames, instead of just I frames. To increase the data hiding capacity, embedding data in inter-coded frames is necessary. Aiming at this goal, an adaptive and efficient data hiding method utilizing both DCT coefficients and motion vectors of I, P, and B frames is developed in this study. The imperceptibility of the hidden data is also maintained by a method which selects proper frequency coefficients and motion vectors for data hiding.

1.1.2 Proposed Ideas

For I frames, it is proposed in this study to hide data into the middle frequency band of the DCT coefficients, which is one of the three frequency bands in an 8╳8 DCT block defined in this study, as illustrated in Figure 1.1. On the other hand, the locations of the DCT coefficients used to hide data are selected dynamically in order to promote the security of the hidden data.

The idea of hiding data in P and B frames proposed in this study is to make a slight modification of the motion vectors in the frames. But not all motions vectors can be used to hide data. The reason will be explained in Section 1.2.2.

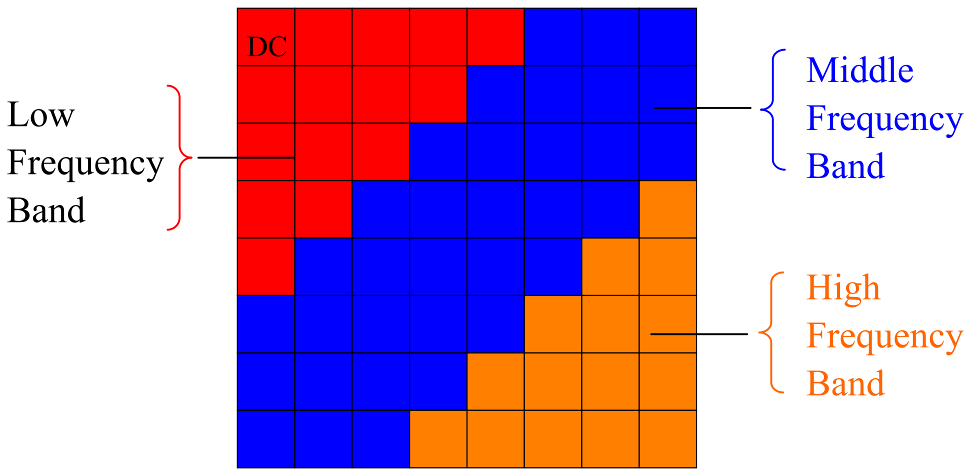


Figure 1.1 Locations of the three frequency bands in a DCT block defined in this study.

1.2 Hiding Secret data in MPEG Videos

In this section, the proposed processes of hiding data into different types of frames of MPEG videos will be described. An illustration of the hiding method is shown in Figure 1.2. In Section 1.2.1, the process for hiding data in I frames will be described. The process for hiding data in P frames will be described in Section 1.2.2. Finally, the process for hiding data in B frames will be described in Section 1.2.3.

1.2.1 Process for Hiding Data in I Frames

Because I frames are coded without referencing to other frames, all macroblocks in an I frame are intra-coded. This coding fashion is similar to the compression technique of the JPEG standard. Therefore, a DCT-based method which is usually used for JPEG images is developed in this study to hide data into I frames.

Each 8×8 luminance block in an I frame can be used to hide four bits of data by making a slight modification of the DCT coefficients, and the imperceptivity of the hidden data can still be maintained because only coefficients at proper locations in the DCT domain are selected to hide data in the proposed method.

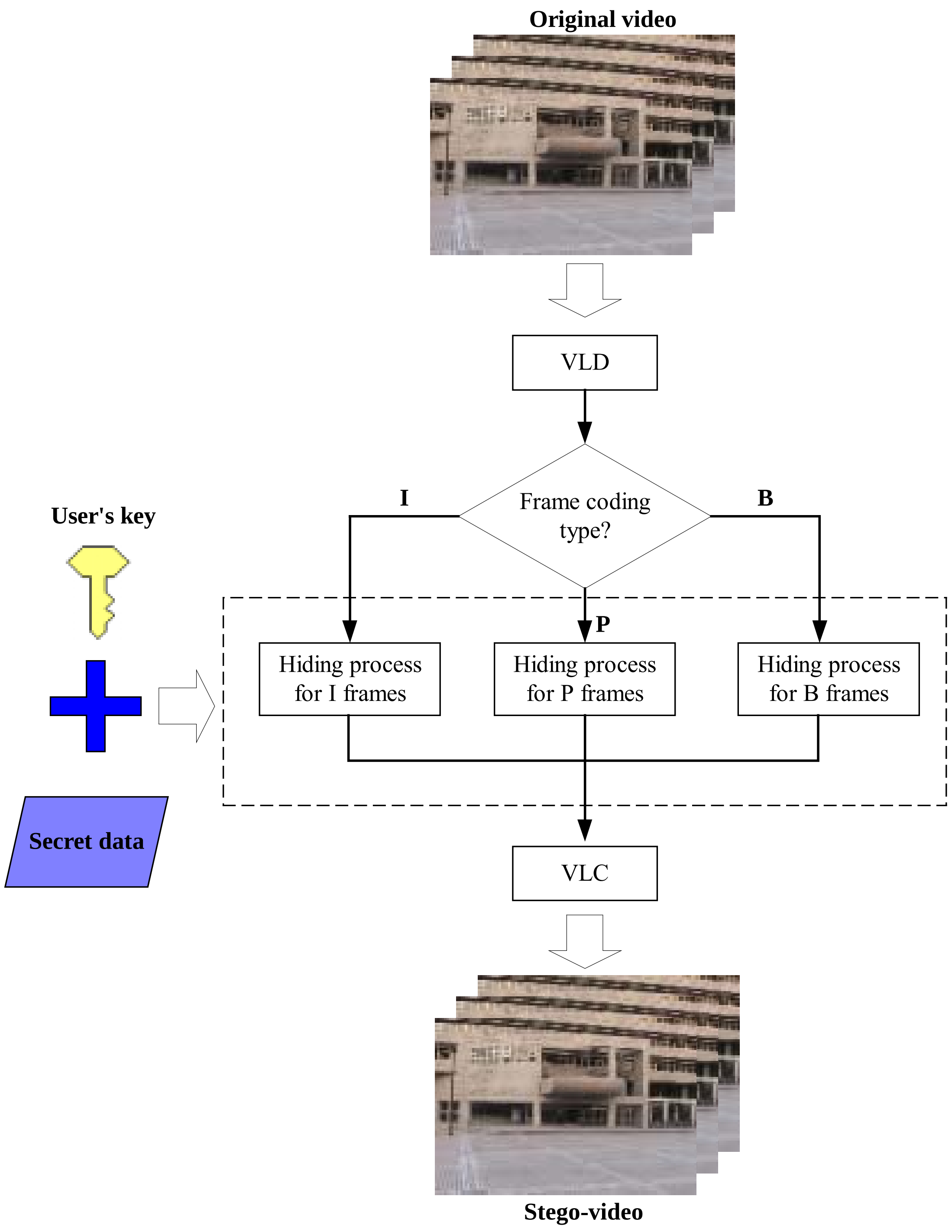

Original video
VLD
Frame coding type?
I
B
P
User's key
Hiding process for I frames
Hiding process for P frames
Hiding process for B frames
Secret data
VLC
Stego-video

Figure 1.2 Illustration of the proposed hiding method.

The selection of the coefficient locations in the DCT domain for data hiding will influence the degree of the imperceptibility of the hidden data, and so is an important task. In this study, the DCT coefficients with larger magnitudes are selected to hide data, because changing a larger DCT coefficient by decrementing or incrementing it by a small value will be less perceivable, compared with making the same decrement or increment in a smaller DCT coefficient. In addition, the user's key will be hidden in the first I frame in order to ensure that the hidden data can be extracted only by a user who has the correct key.

The quantized DCT coefficients of each 8╳8 block of a given I frame are taken as input to the data hiding process for I frames after performing variable length decoding on the input video. A detailed algorithm of the process is described in the following.

***Algorithm* 1**: Hiding process for I frames.

Input: an I frame F in the quantized DCT domain, a user's key R, and a secret data file D.

Output: a stego-frame F'.

Steps:

1. With the aim of hiding 4-bit data in each 8╳8 luminance block of F, compute data hiding capacity L as follows:

$$L = \frac{(mb_{\mathrm{R}} \times mb_{\mathrm{C}}) \times 4 \times 4}{8}, \tag{1.1}$$

where mb_{R} is the number of macroblocks in one row of F, and mb_{C} is the number of macroblocks in one column of F. And get L bytes from the secret data D.

2. Define four sections in the middle band in the DCT domain in a zigzag scanning order, as shown in Figure 3.3, and then find the DCT coefficient C_i whose magnitude is the maximum in each section.

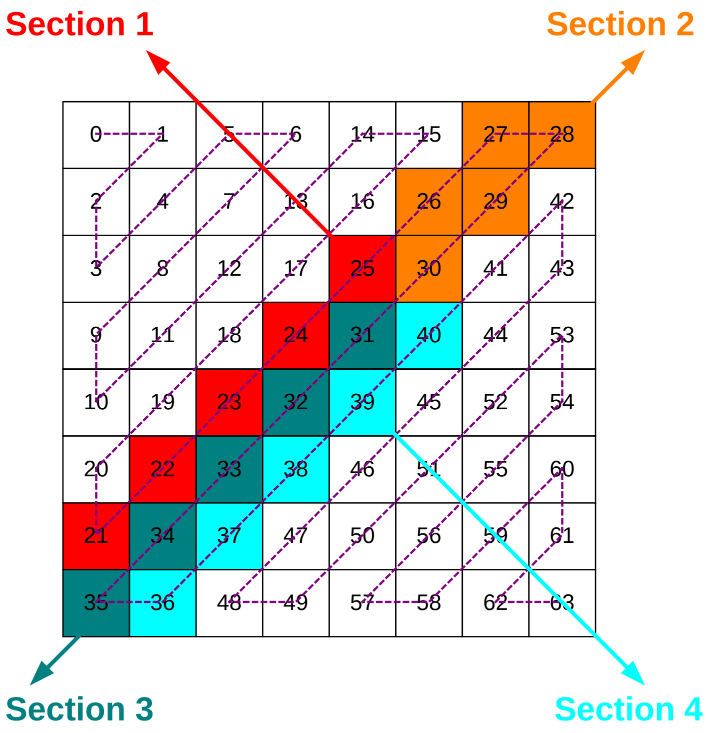


Figure 1.3 Four defined sections in an 8╳8 luminance block.

3. Hide a bit *d* of *D* into each C_i according to the following two types of rules.

   (1) *When* $C_i \geq 0$:

$$\begin{cases} \textit{if } D(i)=1 \textit{ and } C_i \textit{ is even, then set } C_i = C_i + 1; \\ \textit{if } D(i)=0 \textit{ and } C_i \textit{ is odd, then set } C_i = C_i + 1; \\ \textit{otherwise, leave } C_i \textit{ unchanged.} \end{cases} \tag{1.2}$$

   (2) *When* $C_i < 0$:

$$\begin{cases} \textit{if } D(i)=1 \textit{ and } C_i \textit{ is even, then set } C_i = C_i - 1; \\ \textit{if } D(i)=0 \textit{ and } C_i \textit{ is odd, then set } C_i = C_i - 1; \\ \textit{otherwise, leave } C_i \textit{ unchanged.} \end{cases} \tag{1.3}$$

A flowchart of the hiding process for I frames is shown in Figure 3.4.

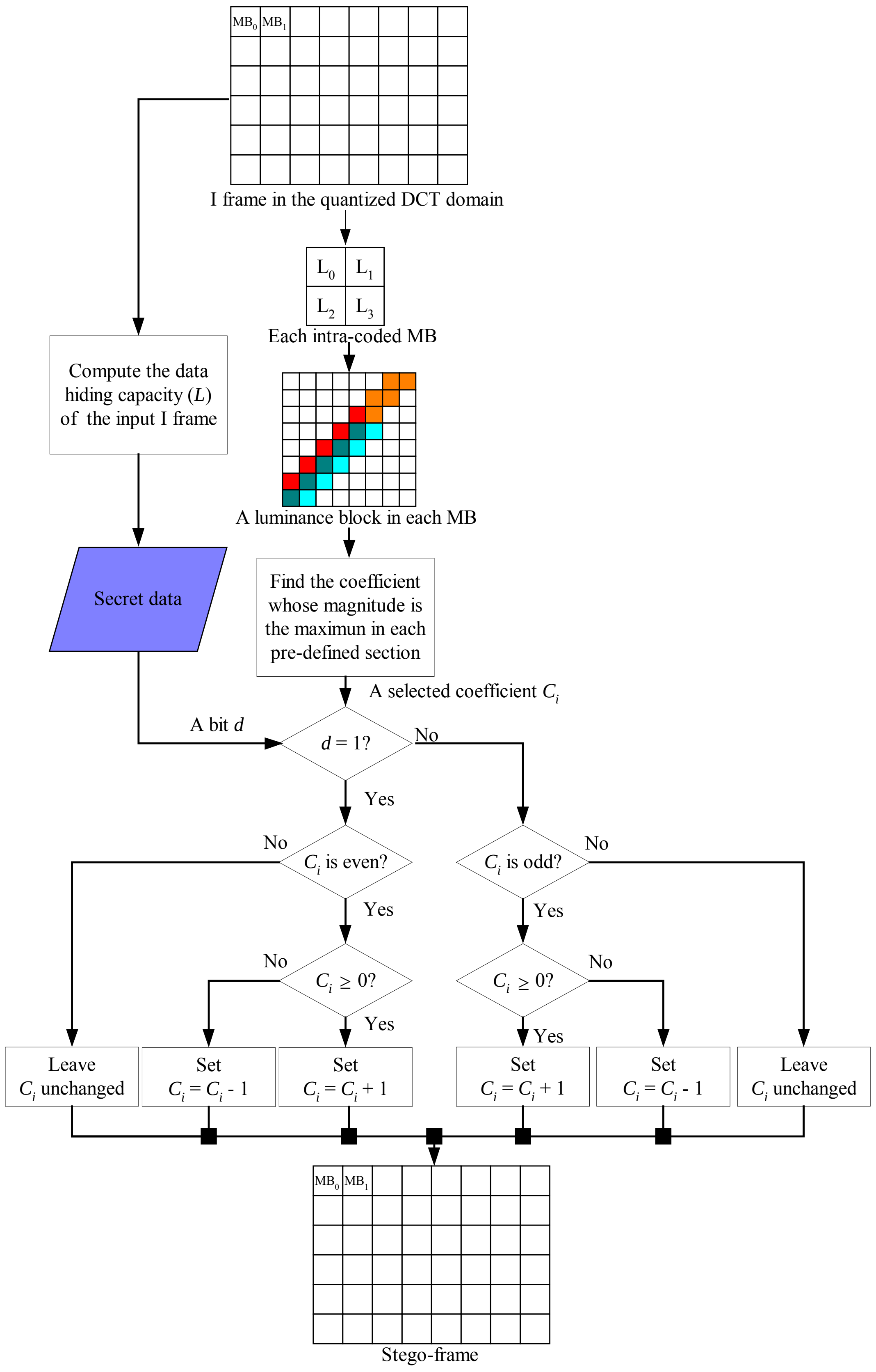


Figure 1.4 Flowchart of the hiding process for I frames.

1.2.2 Process for Hiding Data in P Frames

Inter-coded frames are encoded by motion compensation prediction to reduce temporal redundancy. Therefore, hiding data in the motion vectors can utilize efficiently the information in the video bitstream to increase the data hiding capacity of the inter-coded frames. Moreover, there are still some intra-coded macroblocks which can be used to hide data in inter-coded frames.

In P frames, there are forward-coded and intra-coded macroblocks; therefore, two different hiding processes are proposed.

A. Hiding process for FMBs

The motion vector of each forward-coded macroblock consists of a horizontal component and a vertical component. Each component will be used to hide data in the proposed method. However, not all of the forward-coded macroblocks are suitable for hiding data. First, only those macroblocks whose horizontal or vertical component magnitudes are large are selected. In other words, only those macroblocks which represent faster physical motions are utilized. This way, making a modification of the motion vector to hide data, will cause less perceivable degradation of the stego-video quality. Next, the magnitudes of the horizontal and vertical components of each selected forward-coded macroblock are compared with each other. The component whose magnitude is larger is selected to hide data, because changing a component whose magnitude is larger will be less perceivable, compared with changing another whose magnitude is smaller. A flowchart of the hiding process for forward-coded macroblocks is shown in Figure 1.5 and the corresponding detailed algorithm is described in the following.

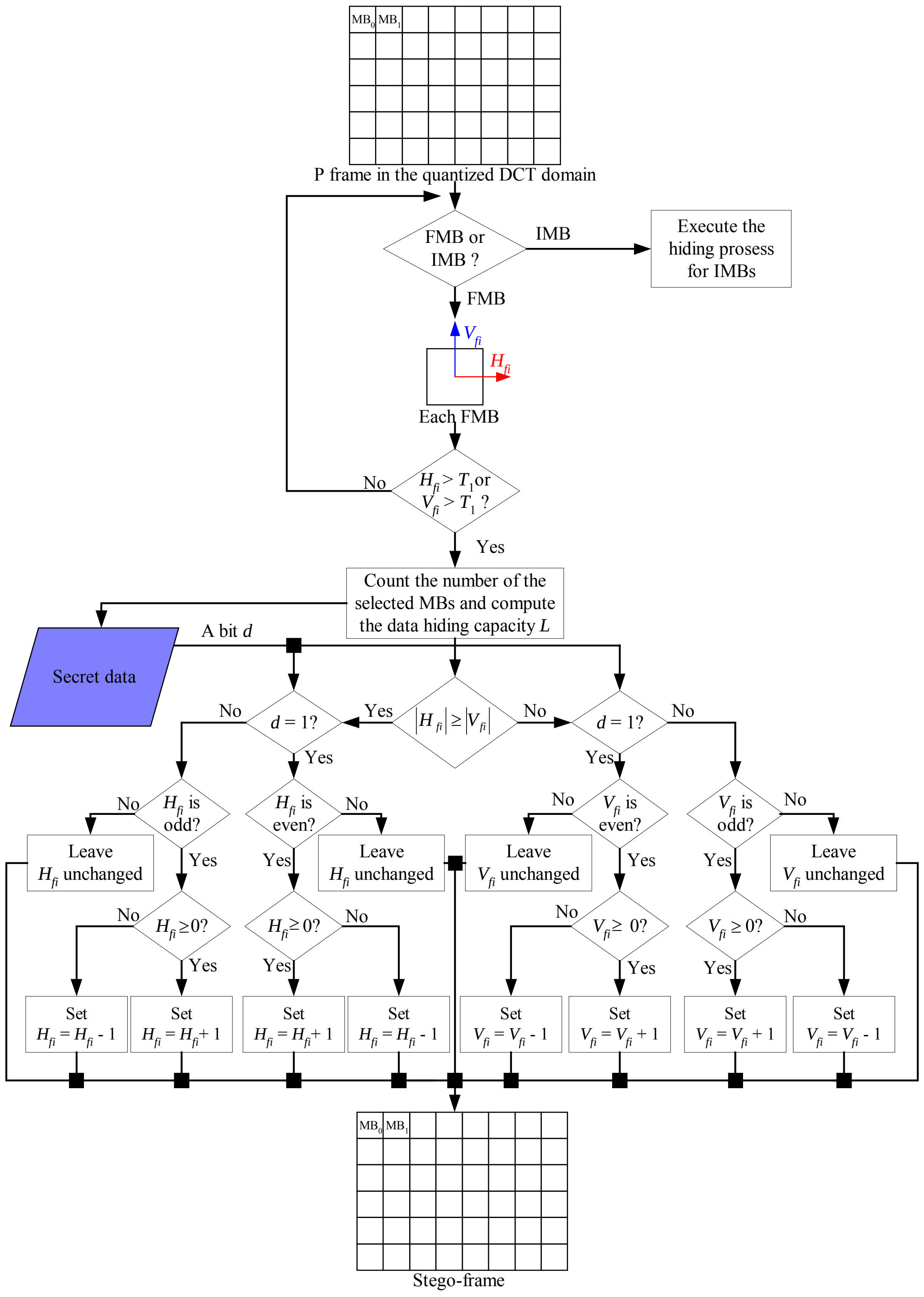


Figure 1.5 Flowchart of the hiding process for FMBs.

***Algorithm* 2**: Data hiding process for the FMBs of P frames.

Input: a P frame F in the quantized DCT domain and a secret data file D.

Output: a stego-frame F'.

Steps:

1. For each forward-coded macroblock of the input P frame F, use the following rule to decide if the macroblock is proper to hide data:

$$|H_{fi}| > T_1 \text{ or } |V_{fi}| > T_1, \tag{1.4}$$

where H_{fi} and V_{fi} are the horizontal component and the vertical one of the i-th forward-coded macroblock, respectively; and T_1 is a pre-defined threshold value. And then count the number N of selected proper macroblocks to compute a data hiding capacity $L = N/8$.

2. Hide a bit d of D into a corresponding selected motion vector according to the following four types of rules.

- *When* $|H_{fi}| \geq |V_{fi}|$ *and* $|H_{fi}| \geq 0$:

$$\begin{cases} \text{if } d = 1 \text{ and } H_{fi} \text{ is even, then set } H_{fi} = H_{fi} + 1; \\ \text{if } d = 0 \text{ and } H_{fi} \text{ is odd, then set } H_{fi} = H_{fi} + 1; \\ \text{otherwise, leave } H_{fi} \text{ unchanged.} \end{cases} \tag{1.5}$$

- *When* $|H_{fi}| \geq |V_{fi}|$ *and* $|H_{fi}| < 0$:

$$\begin{cases} \text{if } d = 1 \text{ and } H_{fi} \text{ is even, then set } H_{fi} = H_{fi} - 1; \\ \text{if } d = 0 \text{ and } H_{fi} \text{ is odd, then set } H_{fi} = H_{fi} - 1; \\ \text{otherwise, leave } H_{fi} \text{ unchanged.} \end{cases} \tag{1.6}$$

- *When* $|V_{fi}| > |H_{fi}|$ *and* $|V_{fi}| \geq 0$:

$$\begin{cases} \text{if } d = 1 \text{ and } V_{fi} \text{ is even, then set } V_{fi} = V_{fi} + 1; \\ \text{if } d = 0 \text{ and } V_{fi} \text{ is odd, then set } V_{fi} = V_{fi} + 1; \\ \text{otherwise, leave } V_{fi} \text{ unchanged.} \end{cases} \tag{1.7}$$

- *When* $|V_{fi}| > |H_{fi}|$ *and* $|V_{fi}| < 0$:

$$\begin{cases} \text{if } d = 1 \text{ and } V_{fi} \text{ is even, then set } V_{fi} = V_{fi} - 1; \\ \text{if } d = 0 \text{ and } V_{fi} \text{ is odd, then set } V_{fi} = V_{fi} - 1; \\ \text{otherwise, leave } V_{fi} \text{ unchanged.} \end{cases} \tag{1.8}$$

Notice that the selection of the value of T_1 is a tradeoff between the data hiding capacity and the resulting video quality. In other words, the smaller T_1 is, the more data can be hidden into a video, however, at the expense of the resulting video quality.

B. Hiding process for IMBs

The hiding process for the intra-coded macroblocks of P frames is similar to the method used for I frames, except that now the number of the intra-coded macroblocks in an input P frame should be counted to compute the data hiding capacity before the data hiding work is started. A flowchart of the data hiding process for intra-coded macroblocks is shown in Figure 1.6 and a corresponding detailed algorithm is described in the following.

***Algorithm* 3**: Data hiding process for the IMBs of P frames.

Input: a P frame F in the quantized DCT domain and a secret data file D.

Output: a stego-frame F'.

Steps:

1. Count the number N of the intra-coded macroblocks of the input P frame F, and compute the data hiding capacity $L = (N\times4\times4)/8$.
2. Define four sections in the middle band in the DCT domain in a zigzag scanning order, as shown in Figure 1.3, and find the DCT coefficient C_i whose magnitude is the maximum in each section.

3. Hide a bit *d* of *D* into the coefficient C_i according to the Equations (1.2) and (1.3).

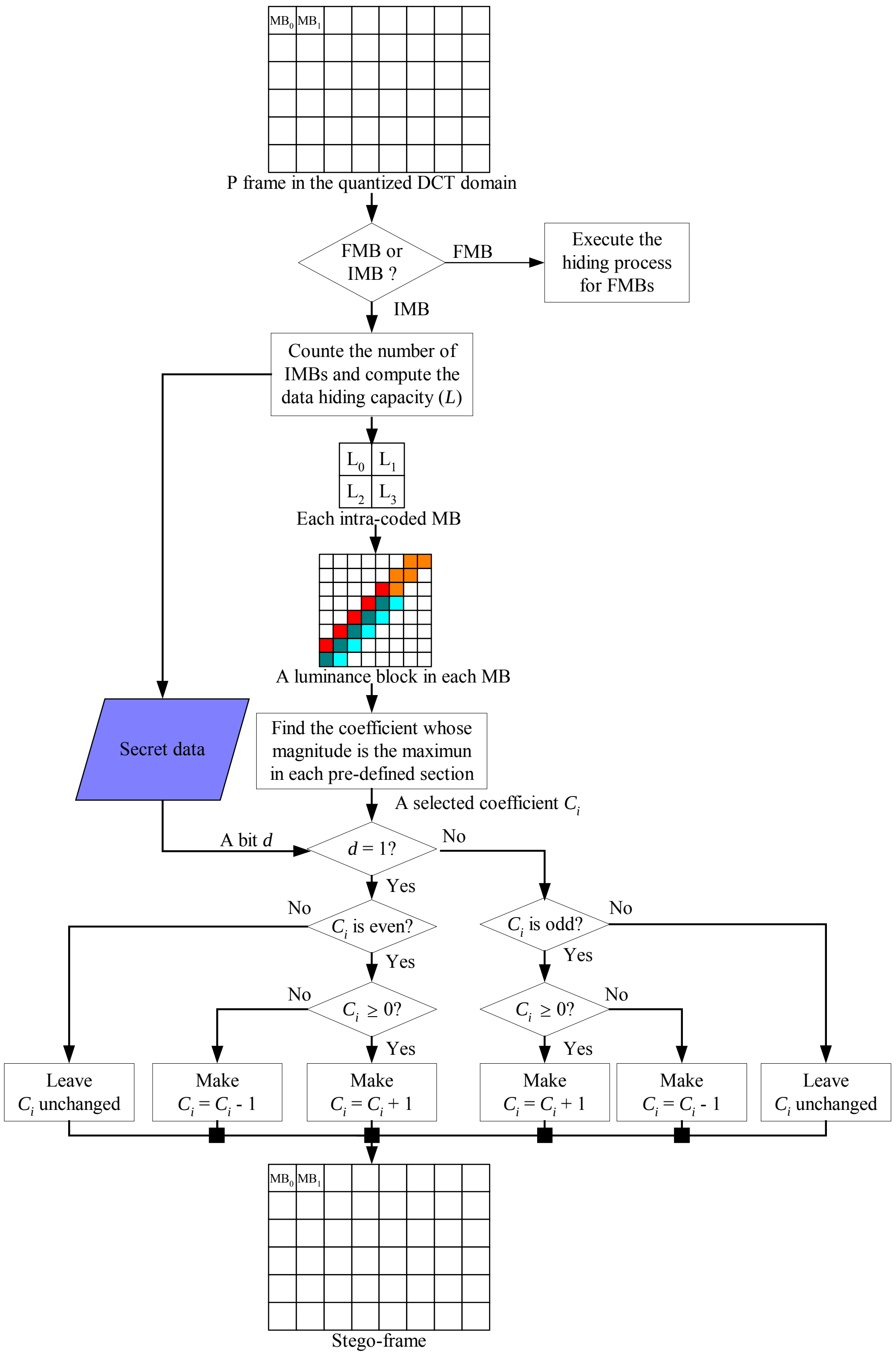


Figure 1.6 Flowchart of the hiding process for IMBs.

1.2.3 Process for Hiding Data in B Frames

The hiding process for the FMBs and IMBs of B frames are similar to that for P frames;

moreover, there is the new type of backward-coded macroblock (BMB) which can be used to hide data in B frames. A flowchart of the hiding process for backward-coded macroblocks is shown in Figure 1.7 and a corresponding detailed algorithm is described in the following.

***Algorithm* 4**: Hiding process for the BMBs of B frames.

Input: a B frame F in the quantized DCT domain and a secret data file D.

Output: a stego-frame F'.

Steps:

1. For each forward-coded macroblock of the input B frame F, use the following rule to decide if the macroblock is proper to hide data:

$$|H_{bi}| > T_2 \quad or \quad |V_{bi}| > T_{2,} \tag{1.9}$$

   where H_{bi} and V_{bi} are the horizontal and vertical components of the i-th forward-coded macroblock, respectively, and T_2 is a pre-defined threshold value. And then count the number N of the selected macroblocks to compute the data hiding capacity $L = N/8$.

2. Hide a bit d of D into the corresponding selected motion vector according to the following four types of rules.
   - *When* $|H_{bi}| \geq |V_{bi}|$ *and* $|H_{bi}| \geq 0$:

$$\begin{cases} if\ d = 1\, and\ H_{bi}\ is\ even, then\ set\ H_{bi} = H_{bi} + 1; \\ if\ d = 0\, and\ H_{bi}\ is\ odd, then\ set\ H_{bi} = H_{bi} + 1; \\ otherwise,\ leave\ H_{bi}\ unchanged. \end{cases} \tag{1.10}$$

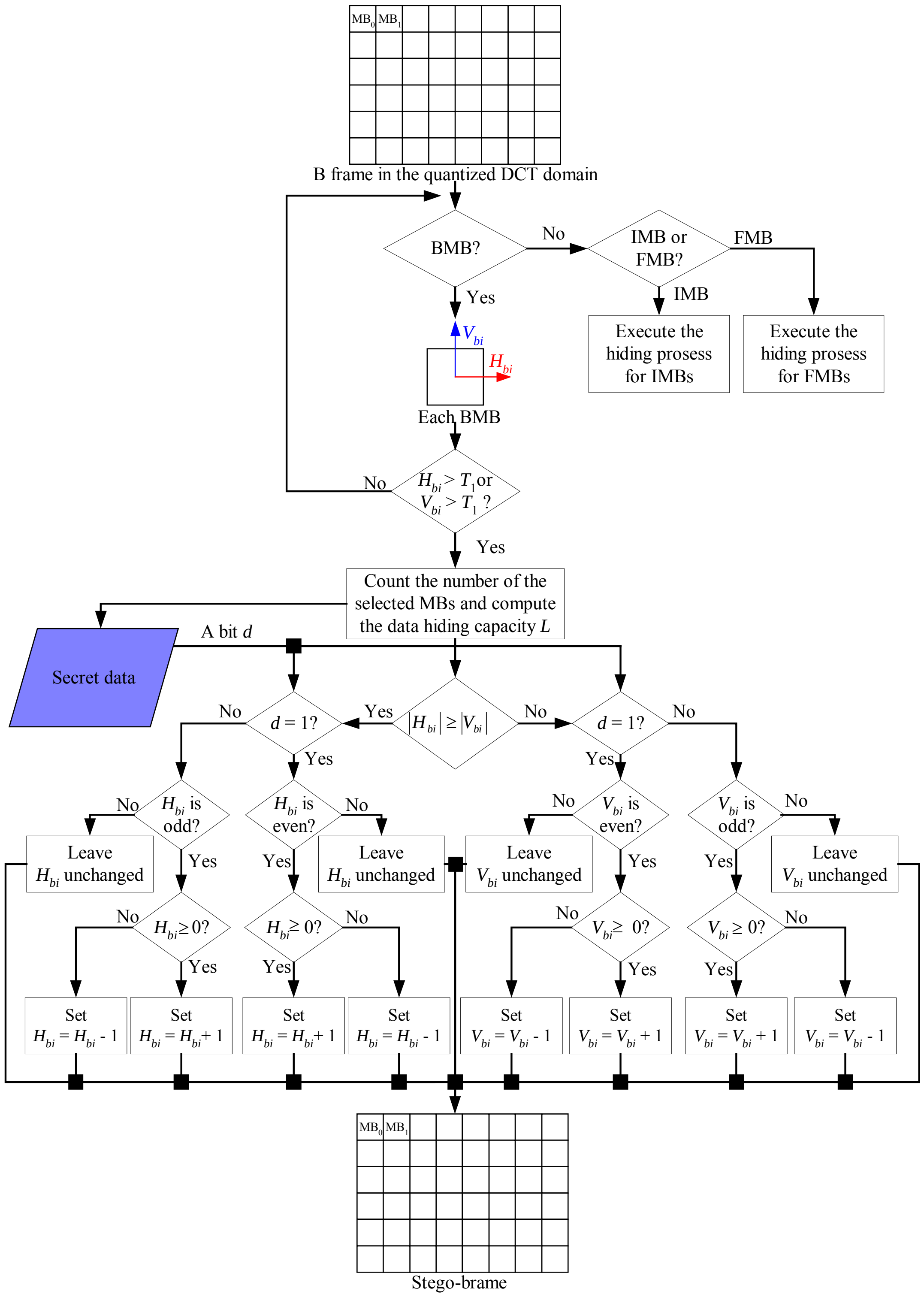

MB0
MB1
B frame in the quantized DCT domain
BMB?
No
IMB or FMB?
FMB
IMB
Execute the hiding prosess for IMBs
Execute the hiding prosess for FMBs
Yes
V_{bi}
H_{bi}
Each BMB
No
$H_{bi} > T_1$ or $V_{bi} > T_1$?
Yes
Count the number of the selected MBs and compute the data hiding capacity L
Secret data
A bit d
$|H_{bi}| \geq |V_{bi}|$
Yes
No
d = 1?
No
Yes
d = 1?
No
Yes
H_{bi} is odd?
H_{bi} is even?
V_{bi} is even?
V_{bi} is odd?
Leave H_{bi} unchanged
Leave H_{bi} unchanged
Leave V_{bi} unchanged
Leave V_{bi} unchanged
$H_{bi} \geq 0$?
$H_{bi} \geq 0$?
$V_{bi} \geq 0$?
$V_{bi} \geq 0$?
Set $H_{bi} = H_{bi} - 1$
Set $H_{bi} = H_{bi} + 1$
Set $H_{bi} = H_{bi} + 1$
Set $H_{bi} = H_{bi} - 1$
Set $V_{bi} = V_{bi} - 1$
Set $V_{bi} = V_{bi} + 1$
Set $V_{bi} = V_{bi} + 1$
Set $V_{bi} = V_{bi} - 1$
Stego-brame

Figure 1.7 Flowchart of the hiding process for BMBs.

- *When* $|H_{bi}| \geq |V_{bi}|$ *and* $|H_{bi}| < 0$:

$$\begin{cases} if\ d = 1\ and\ H_{bi}\ is\ even, then\ set\ H_{bi} = H_{bi} - 1; \\ if\ d = 0\ and\ H_{bi}\ is\ odd, then\ set\ H_{bi} = H_{bi} - 1; \\ otherwise,\ leave\ H_{bi}\ unchanged. \end{cases} \tag{1.11}$$

- *When* $|V_{bi}| > |H_{bi}|$ *and* $|V_{bi}| \geq 0$:

$$\begin{cases} if\ d = 1\ and\ V_{fi}\ is\ even, then\ set\ V_{bi} = V_{bi} + 1; \\ if\ d = 0\ and\ V_{bi}\ is\ odd, then\ set\ V_{bi} = V_{bi} + 1; \\ otherwise,\ leave\ V_{bi}\ unchanged. \end{cases} \tag{1.12}$$

- *When* $|V_{bi}| > |H_{bi}|$ *and* $|V_{bi}| < 0$:

$$\begin{cases} if\ d = 1\ and\ V_{bi}\ is\ even, then\ set\ V_{bi} = V_{bi} - 1; \\ if\ d = 0\ and\ V_{bi}\ is\ odd, then\ set\ V_{bi} = V_{bi} - 1; \\ otherwise,\ leave\ V_{bi}\ unchanged. \end{cases} \tag{1.13}$$

1.3 Extracting Secret Data from MPEG Videos

In this section, the processes of extracting the hidden data from an input MPEG video will be described. An illustration of the proposed data extraction method is illustrated in Figure 1.8. In Section 1.3.1, the process for extracting data from an I frame will be described. Next, the process for extracting data from a P frame will be described in Section 1.3.2. Finally, the process for extracting data from a B frame will be described in Section 1.3.3.

1.3.1 Process for Extracting Data from I Frames

The proposed data extraction process starts with the verification of the user's key hidden in the first I frame. After performing variable length decoding on the input video, the quantized DCT coefficients of each 8╳8 block of the I frame are retrieved and taken as input to the extraction process for I frames. A flowchart of the extraction process is shown in Figure 1.9 and a corresponding detailed algorithm is described in the following.

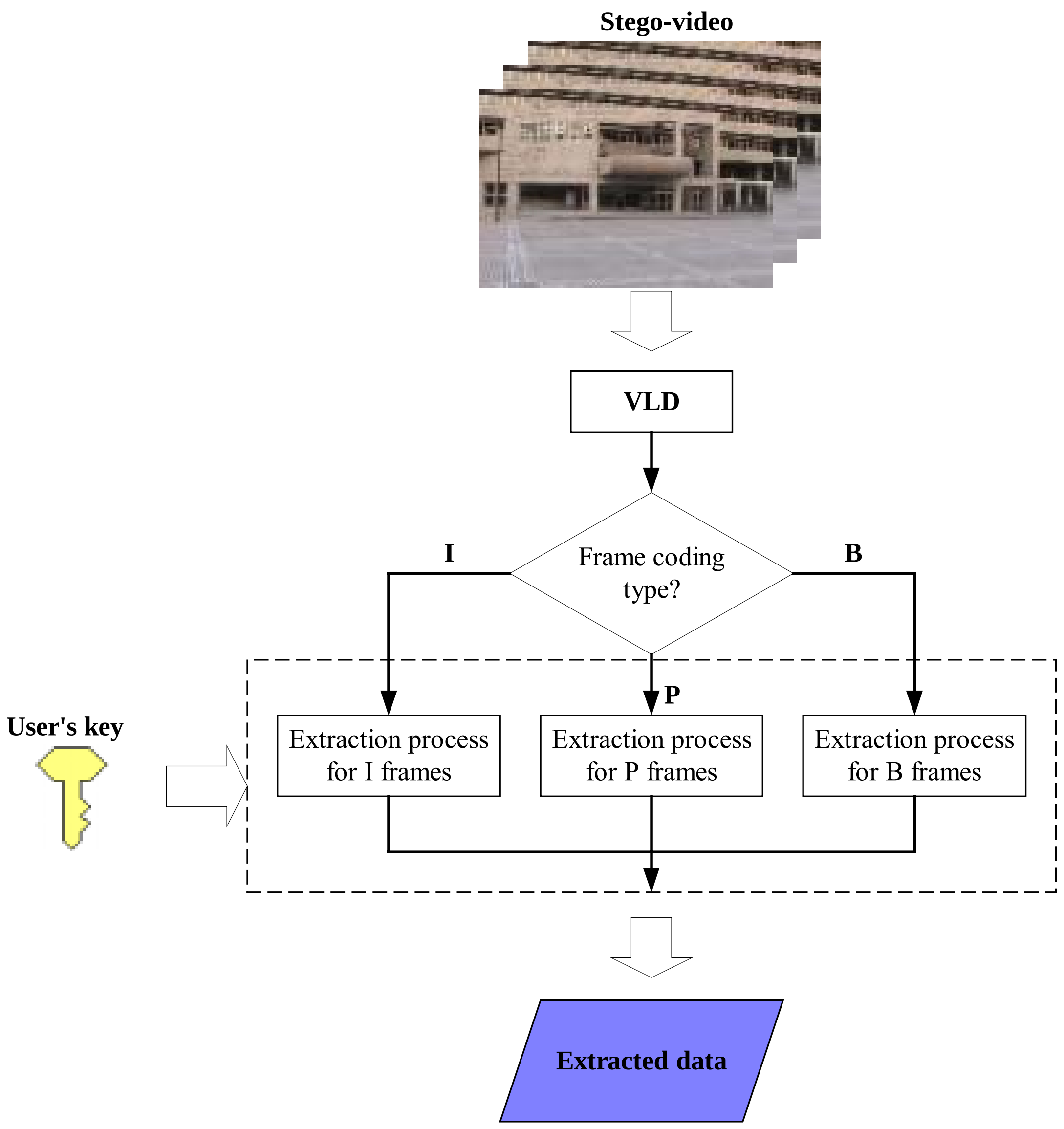
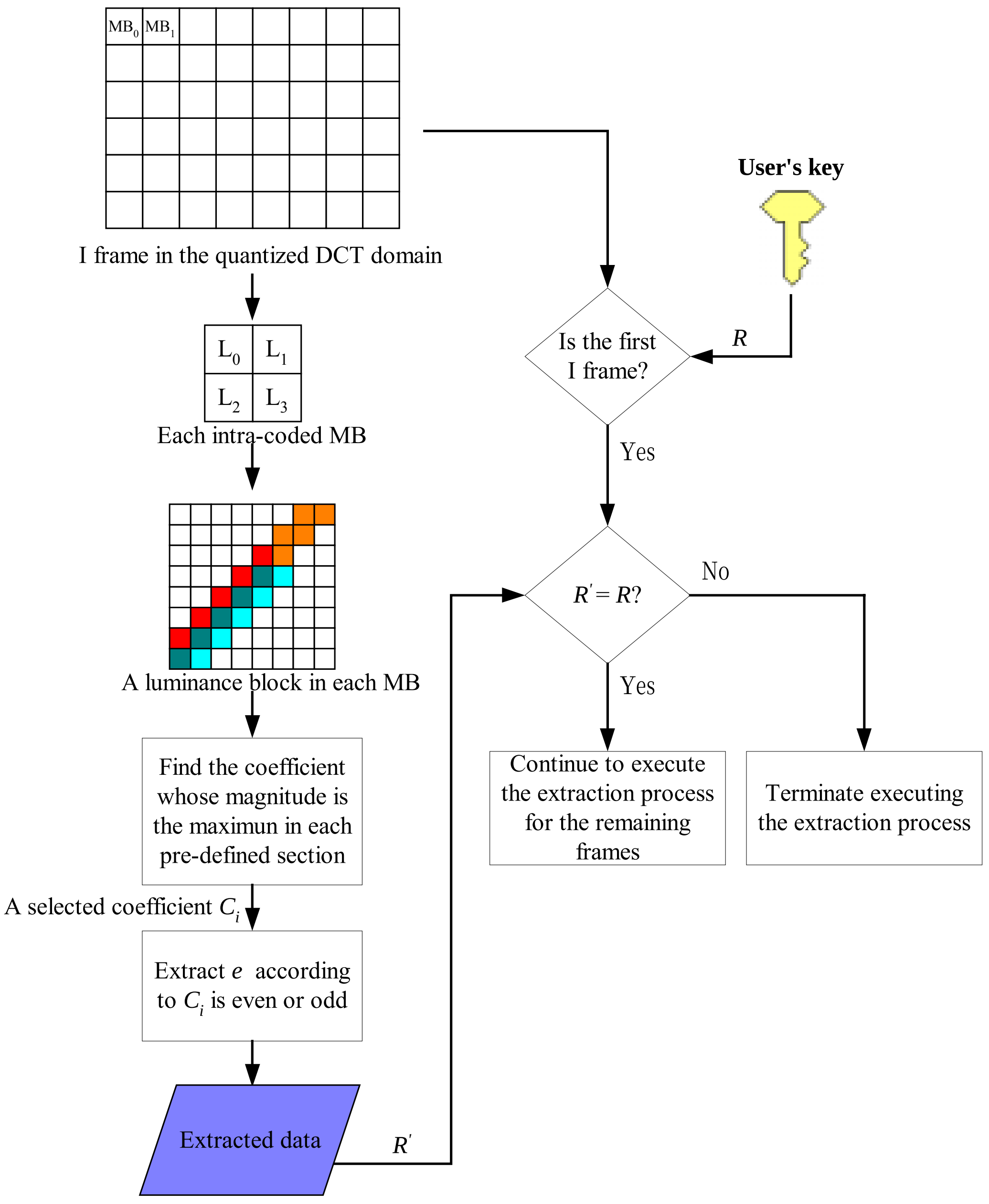

Stego-video
VLD
Frame coding type?
I
B
P
User's key
Extraction process for I frames
Extraction process for P frames
Extraction process for B frames
Extracted data

Figure 1.8 Illustration of proposed data extraction method.



MB_0
MB_1
I frame in the quantized DCT domain
L_0
L_1
L_2
L_3
Each intra-coded MB
A luminance block in each MB
Find the coefficient whose magnitude is the maximun in each pre-defined section
A selected coefficient C_i
Extract e according to C_i is even or odd
Extracted data
R'
User's key
R
Is the first I frame?
Yes
$R' = R$?
No
Yes
Continue to execute the extraction process for the remaining frames
Terminate executing the extraction process

Figure 1.9 Flowchart of the extraction process for I frames.

***Algorithm* 5:** Extraction process for I frames.

Input: an I frame F in the quantized DCT domain and a user's key R.

Output: an extracted data file E.

Steps:

1. For each 8×8 luminance block of the input frame F, find the DCT coefficient C_i whose magnitude is the maximum in each pre-defined section to decide the locations used to hide data.
2. Extract a bit e as part of E from the coefficient C_i according to the following rule:

$$e = \begin{cases} 1, & \text{if } C_i \text{ is odd;} \\ 0, & \text{otherwise.} \end{cases} \tag{1.14}$$

When F is the first frame of the input video, get the correct key R' from the extraction result to verify the correctness of the input key R. If R is not identical to R', the extraction process is terminated; otherwise, continue to execute the extraction process for the remaining frames.

1.3.2 Process for Extracting Data from P Frames

A. Extraction process for FMBs

A flowchart of the extraction process for forward-coded macroblocks is shown in Figure 3.10 and a corresponding detailed algorithm is described in the following.

***Algorithm* 6**: Extraction process for the FMBs of P frames.

Input: a P frame F in the quantized DCT domain.

Outputs: an extracted data file E.

Steps:

1. For each forward-coded macroblock of the input P frame F, use the following rule to check whether there are data hidden in it:

   $$|H_{fi}| > T_1 \ or \ |V_{fi}| > T_{1,}$$

   where H_{fi} and V_{fi} are the horizontal and vertical components of the ***i***-th forward-coded macroblock, respectively, and T_1 is a pre-defined threshold value.

2. Extract a bit e as part of E from each corresponding motion vector according to the following two types of rules.

   - *When* $|H_{fi}| \geq |V_{fi}|$:

   $$e = \begin{cases} 1, & \text{if } H_{fi} \text{ is odd;} \\ 0, & \text{otherwise.} \end{cases} \tag{1.15}$$

   - *When* $|H_{fi}| < |V_{fi}|$:

   $$e = \begin{cases} 1, & \text{if } V_{fi} \text{ is odd;} \\ 0, & \text{otherwise.} \end{cases} \tag{1.16}$$

B. Extraction process for IMBs

A flowchart of the extraction process for the intra-coded macroblocks is shown in Figure 1.11 and a corresponding detailed algorithm is described in the following.

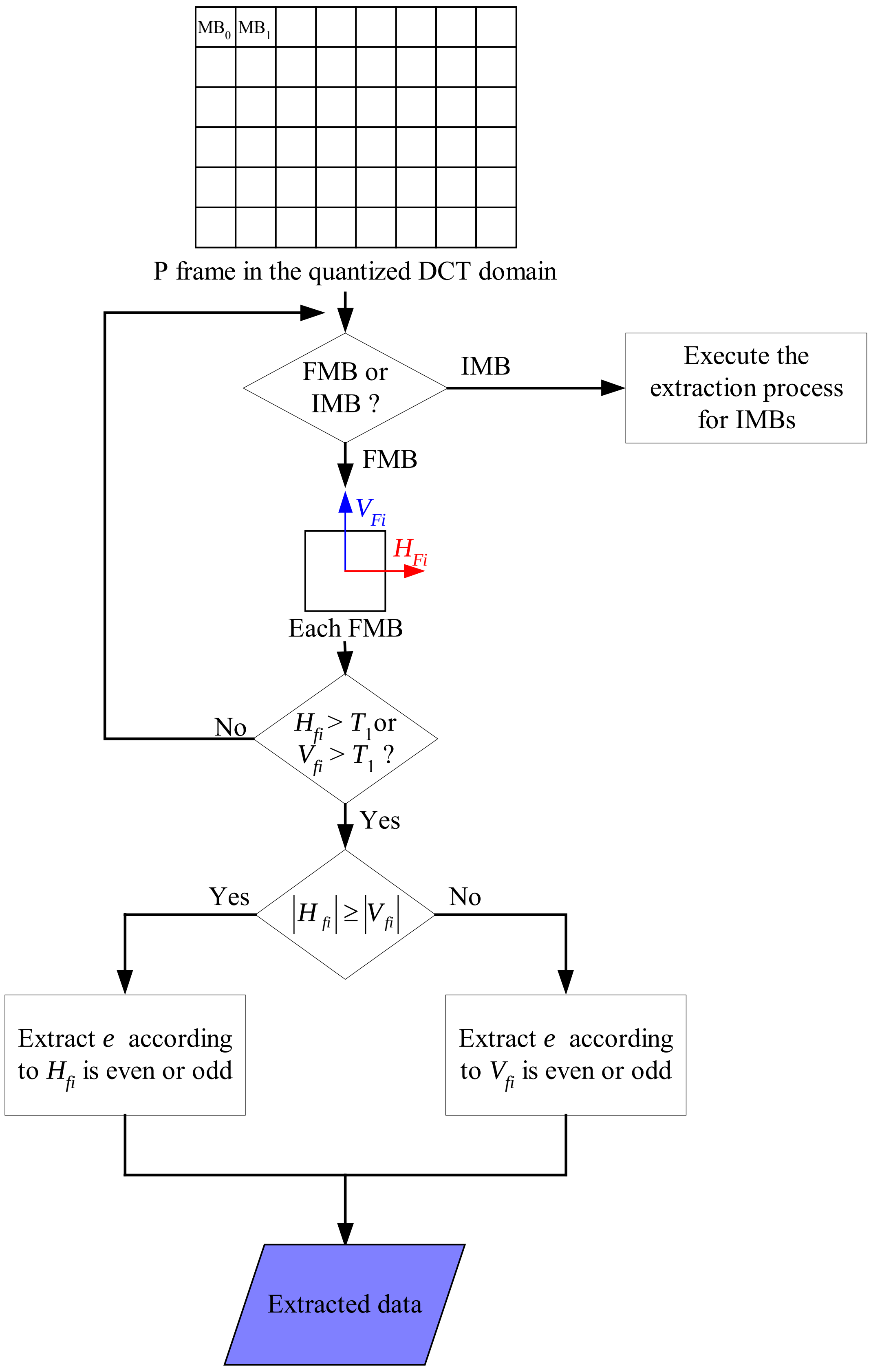
***Algorithm* 7**: Extraction process for the IMBs of P frames.

Input: a P frame F in the quantized DCT domain.

Output: an extracted data file E.

Steps:

1. For each 8×8 luminance block of each intra-coded macroblock of the input P frame F, find the DCT coefficient C_i whose magnitude is the maximum in each pre-defined section to decide the locations used to hide data.
2. Extract a bit e as part of E from the coefficient C_i according to Equation (1.14).



MB_0
MB_1
P frame in the quantized DCT domain
FMB or IMB ?
IMB
Execute the extraction process for IMBs
FMB
V_{Fi}
H_{Fi}
Each FMB
$H_{fi} > T_1$ or $V_{fi} > T_1$?
No
Yes
$|H_{fi}| \geq |V_{fi}|$
Yes
No
Extract e according to H_{fi} is even or odd
Extract e according to V_{fi} is even or odd
Extracted data

Figure 1.10 Flowchart of the extraction process for FMBs.

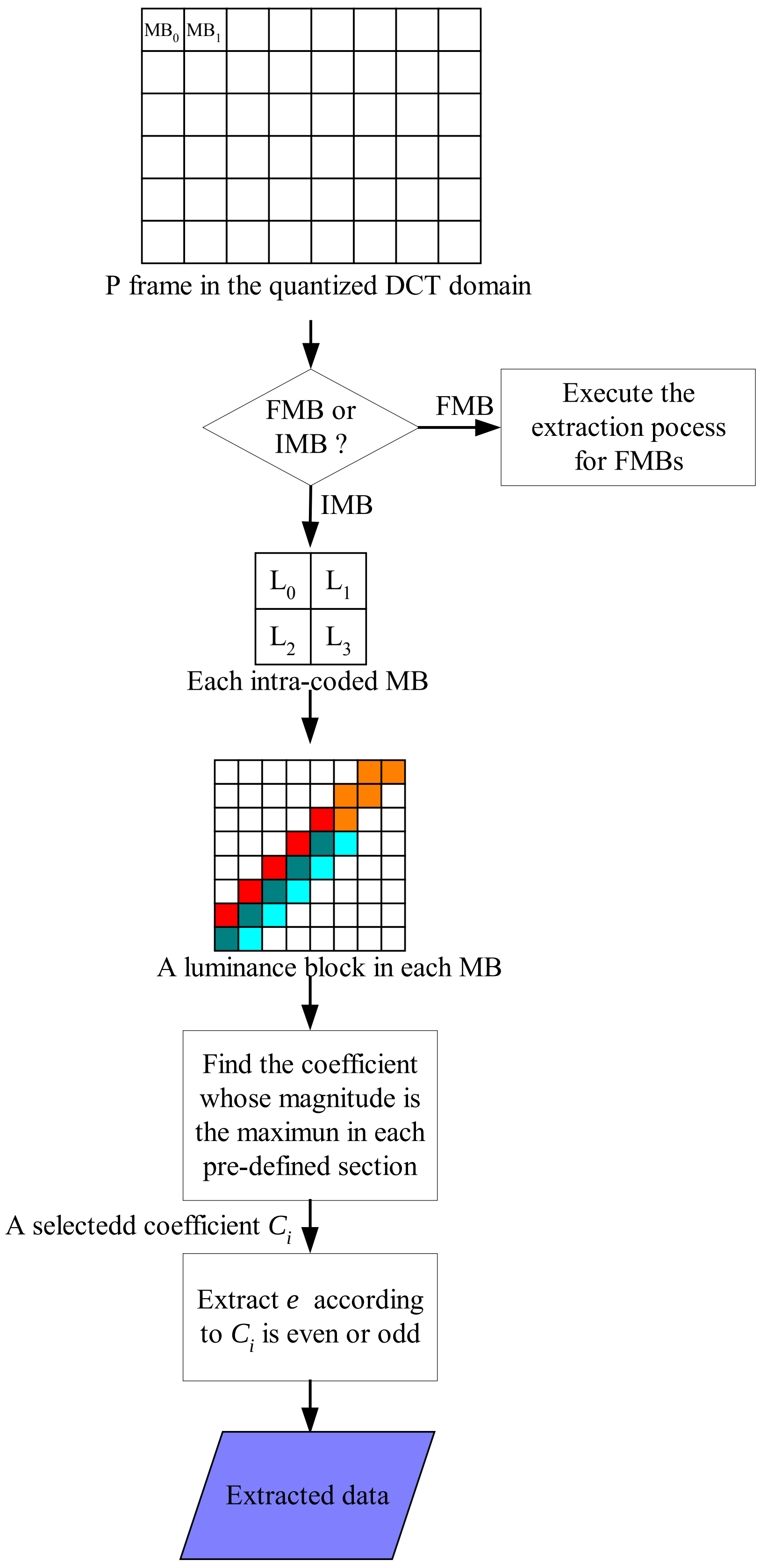

MB$_0$
MB$_1$
P frame in the quantized DCT domain
FMB or IMB ?
FMB
Execute the extraction pocess for FMBs
IMB
L$_0$
L$_1$
L$_2$
L$_3$
Each intra-coded MB
A luminance block in each MB
Find the coefficient whose magnitude is the maximun in each pre-defined section
A selectedd coefficient C_i
Extract e according to C_i is even or odd
Extracted data

Figure 1.11 Flowchart of the extraction process for IMBs.

1.3.3 Process for Extracting Data from B Frames

A flowchart of the data extraction process for backward-coded macroblocks is shown in Figure 1.12 and a corresponding detailed algorithm is described in the following.

***Algorithm* 8**: Extraction process for the BMBs of B frames.

Input: a B frame F in the quantized DCT domain.

Output: the extracted data file E.

Steps:

1. For the motion vector of each backward-coded macroblock, use the following rule to check whether the data are hidden in it.

$$|H_{bi}| > T_2 \ or \ |V_{bi}| > T_{2,}$$

where H_{bi} and V_{bi} are the horizontal and vertical components of the i-th forward-coded macroblocks, respectively, and T_1 is a pre-defined threshold value.

2. Extract a bit e as part of E from each corresponding motion vector according to the following two types of rules.

- *When* $|H_{bi}| \geq |V_{bi}|$:

$$e = \begin{cases} 1, & \text{if } H_{bi} \text{ is odd;} \\ 0, & \text{otherwise.} \end{cases} \qquad (1.17)$$

- *When* $|H_{bi}| < |V_{bi}|$:

$$e = \begin{cases} 1, & \text{if } V_{bi} \text{ is odd;} \\ 0, & \text{otherwise.} \end{cases} \qquad (1.18)$$

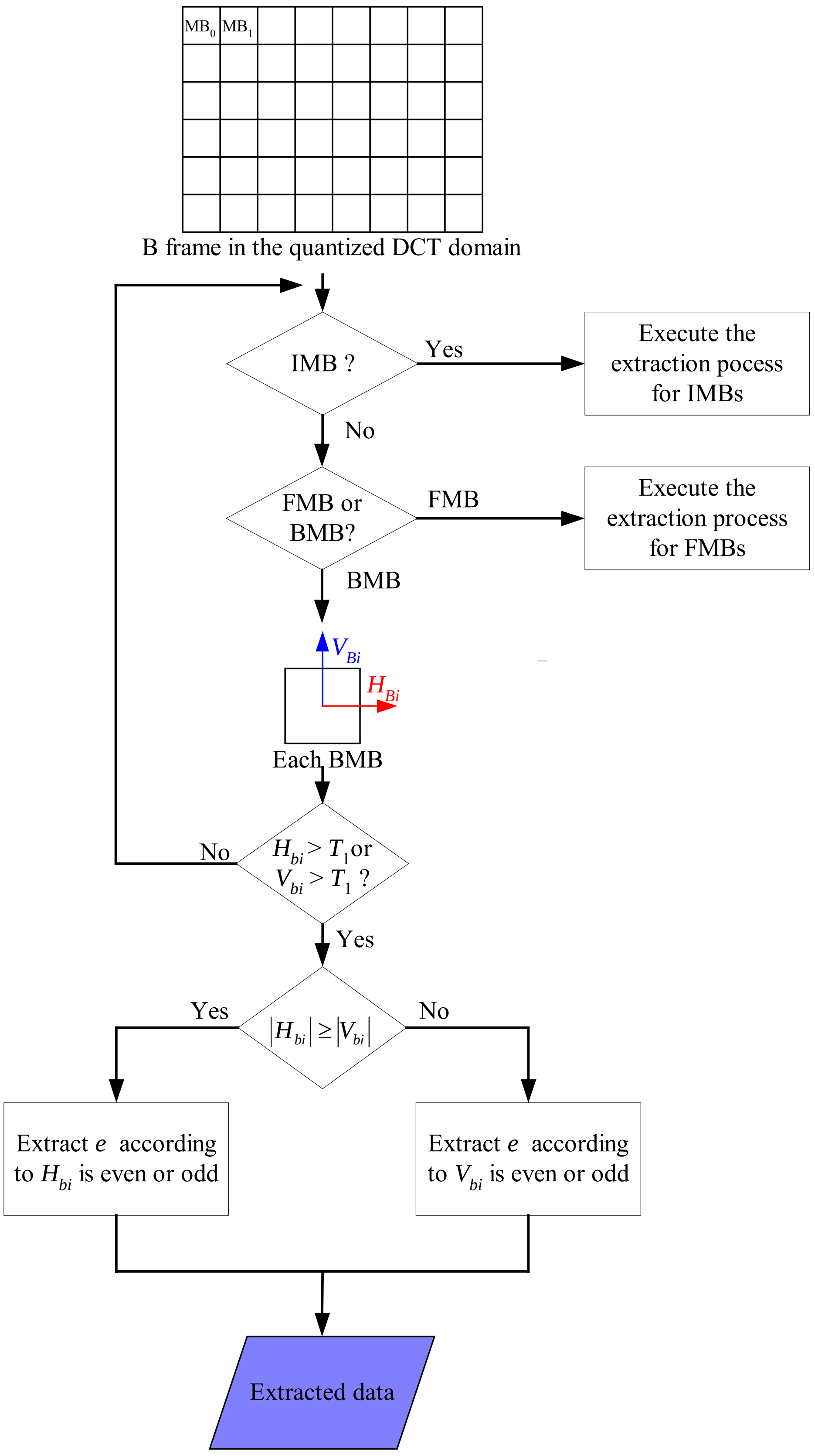
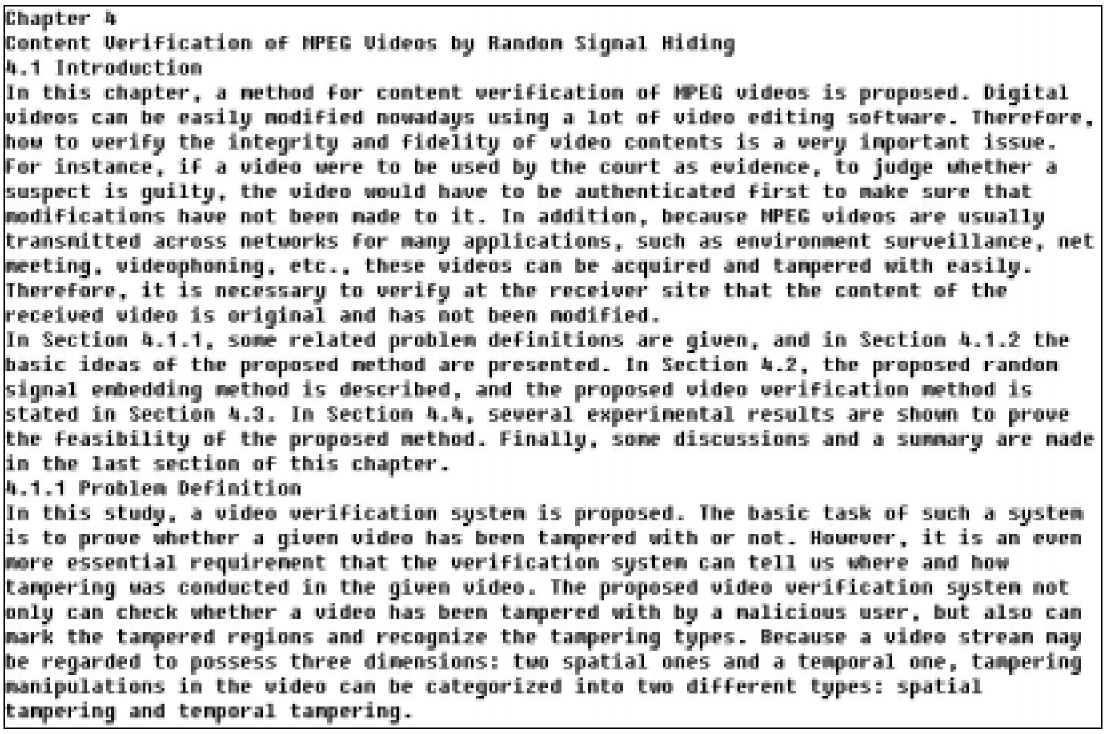

MB_0
MB_1
B frame in the quantized DCT domain
IMB ?
Yes
Execute the extraction pocess for IMBs
No
FMB or BMB?
FMB
Execute the extraction process for FMBs
BMB
V_{Bi}
H_{Bi}
Each BMB
No
$H_{bi} > T_1$ or $V_{bi} > T_1$?
Yes
Yes
$|H_{bi}| \geq |V_{bi}|$
No
Extract e according to H_{bi} is even or odd
Extract e according to V_{bi} is even or odd
Extracted data

Figure 1.12 Flowchart of the extraction process for BMBs.

1.4 Experimental Results

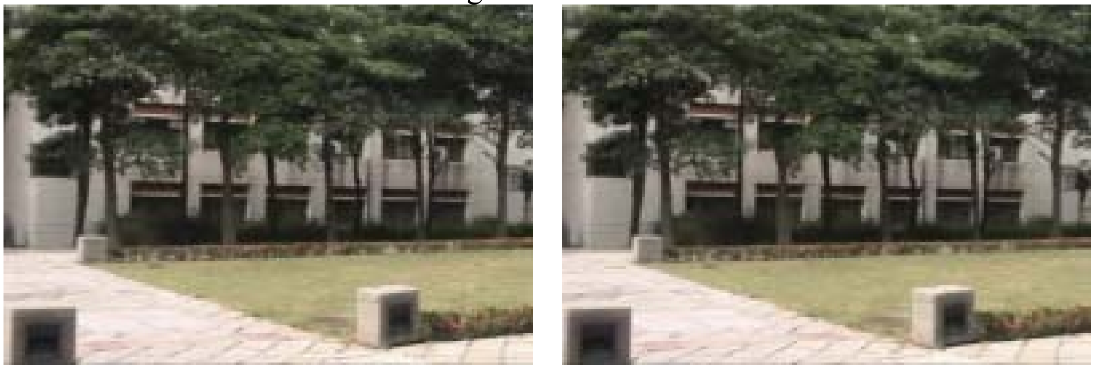
In our experiments, a video with frame size 352╳240 was used as the input to hide a secret data file with the size of 2137 bytes. The data capacity of this video is 2554 bytes, so the secret data can be hidden into it completely. The secret data are shown in Figure 1.13, and six frames of the input video are shown in Figure 1.14. 655, 15, 31, 10, 31, and 30 bytes of data were hidden into the six frames, respectively. Six frames of the resulting stego-video are shown in Figure 1.15 and the PSNR values are shown in Table 1.1. The extracted data are shown in Figure 1.16. From Table 1.1, the PSNR values of the six frames are all acceptable. It shows that by applying the proposed method, the secret data can be hidden into MPEG videos imperceptibly.

In addition, the average data hiding capacity in a second is from 2KBs to 3KBs by our experimental experience. It has been found that the data hiding capacity is proportional to the amount of the motions coded in a video.



Chapter 4
Content Verification of MPEG Videos by Random Signal Hiding
4.1 Introduction
In this chapter, a method for content verification of MPEG videos is proposed. Digital
videos can be easily modified nowadays using a lot of video editing software. Therefore,
how to verify the integrity and fidelity of video contents is a very important issue.
For instance, if a video were to be used by the court as evidence, to judge whether a
suspect is guilty, the video would have to be authenticated first to make sure that
modifications have not been made to it. In addition, because MPEG videos are usually
transmitted across networks for many applications, such as environment surveillance, net
meeting, videophoning, etc., these videos can be acquired and tampered with easily.
Therefore, it is necessary to verify at the receiver site that the content of the
received video is original and has not been modified.
In Section 4.1.1, some related problem definitions are given, and in Section 4.1.2 the
basic ideas of the proposed method are presented. In Section 4.2, the proposed random
signal embedding method is described, and the proposed video verification method is
stated in Section 4.3. In Section 4.4, several experimental results are shown to prove
the feasibility of the proposed method. Finally, some discussions and a summary are made
in the last section of this chapter.
4.1.1 Problem Definition
In this study, a video verification system is proposed. The basic task of such a system
is to prove whether a given video has been tampered with or not. However, it is an even
more essential requirement that the verification system can tell us where and how
tampering was conducted in the given video. The proposed video verification system not
only can check whether a video has been tampered with by a malicious user, but also can
mark the tampered regions and recognize the tampering types. Because a video stream may
be regarded to possess three dimensions: two spatial ones and a temporal one, tampering
manipulations in the video can be categorized into two different types: spatial
tampering and temporal tampering.

Figure 1.13 A secret data file.



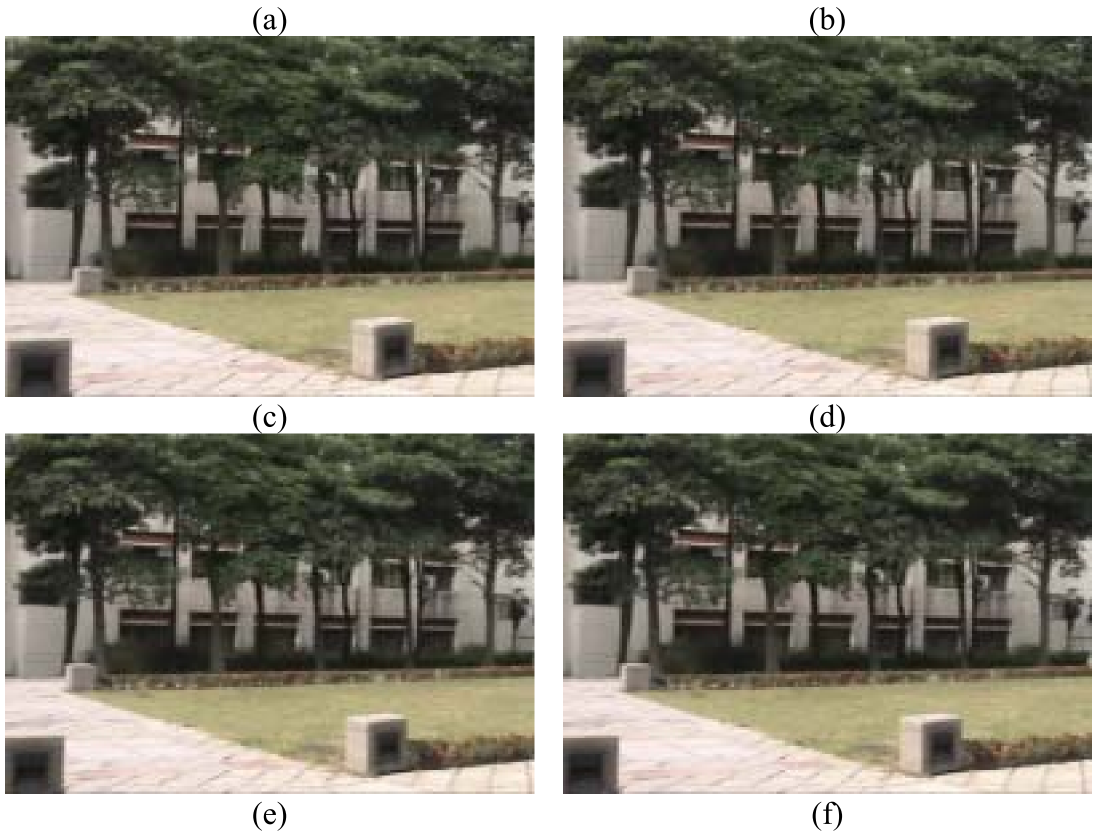
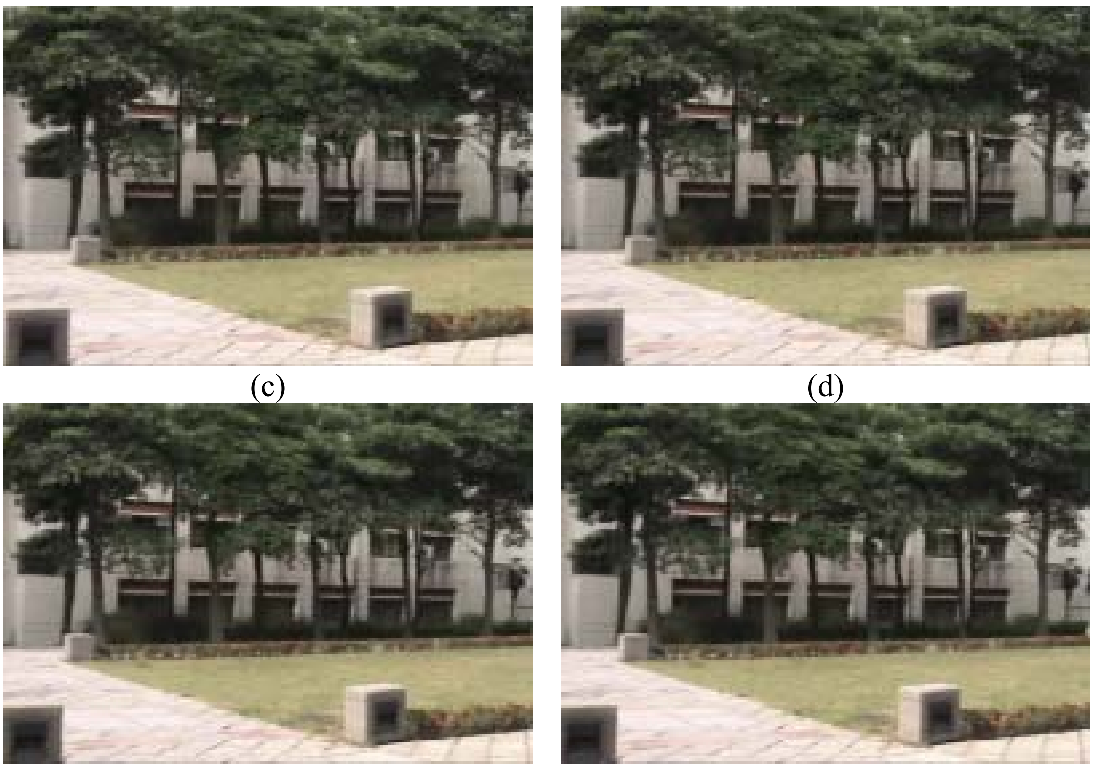


Figure 1.14 Six frames of the input video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).

Table 1.1 The PSNR values of the stego-video.

	f_0 (I)	f_1 (B)	f_2 (B)	f_3 (P)	f_4 (B)	f_5 (B)
PSNR	37.5	38.0	38.1	41.0	38.0	39.6


(c) (d)

(e) (f)

Figure 1.15 Six frames of the stego-video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).

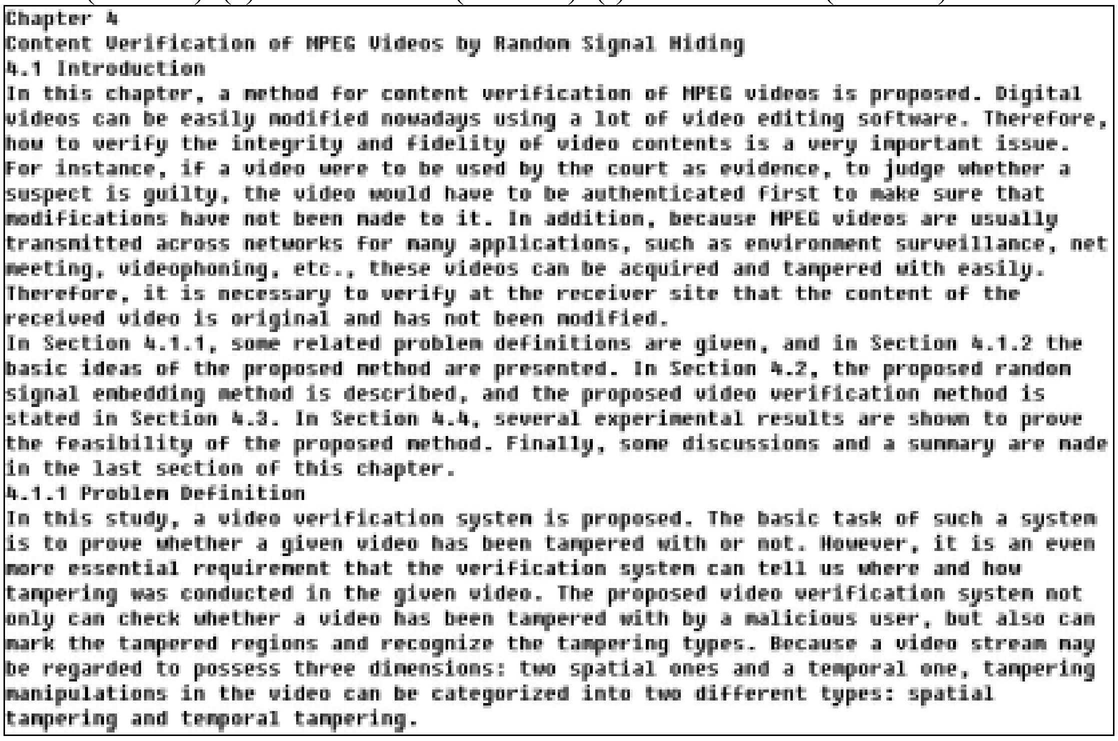

```
Chapter 4
Content Verification of MPEG Videos by Random Signal Hiding
4.1 Introduction
In this chapter, a method for content verification of MPEG videos is proposed. Digital
videos can be easily modified nowadays using a lot of video editing software. Therefore,
how to verify the integrity and fidelity of video contents is a very important issue.
For instance, if a video were to be used by the court as evidence, to judge whether a
suspect is guilty, the video would have to be authenticated first to make sure that
modifications have not been made to it. In addition, because MPEG videos are usually
transmitted across networks for many applications, such as environment surveillance, net
meeting, videophoning, etc., these videos can be acquired and tampered with easily.
Therefore, it is necessary to verify at the receiver site that the content of the
received video is original and has not been modified.
In Section 4.1.1, some related problem definitions are given, and in Section 4.1.2 the
basic ideas of the proposed method are presented. In Section 4.2, the proposed random
signal embedding method is described, and the proposed video verification method is
stated in Section 4.3. In Section 4.4, several experimental results are shown to prove
the feasibility of the proposed method. Finally, some discussions and a summary are made
in the last section of this chapter.
4.1.1 Problem Definition
In this study, a video verification system is proposed. The basic task of such a system
is to prove whether a given video has been tampered with or not. However, it is an even
more essential requirement that the verification system can tell us where and how
tampering was conducted in the given video. The proposed video verification system not
only can check whether a video has been tampered with by a malicious user, but also can
mark the tampered regions and recognize the tampering types. Because a video stream may
be regarded to possess three dimensions: two spatial ones and a temporal one, tampering
manipulations in the video can be categorized into two different types: spatial
tampering and temporal tampering.
```

Figure 1.16 The extracted data file.

Chapter 2

Content Verification of MPEG Videos by Random Signal Hiding

2.1 Introduction

In this chapter, a method for content verification of MPEG videos is proposed. Digital videos can be easily modified nowadays using a lot of video editing software. Therefore, how to verify the integrity and fidelity of video contents is a very important issue. For instance, if a video were to be used by the court as evidence, to judge whether a suspect is guilty, the video would have to be authenticated first to make sure that modifications have not been made to it. In addition, because MPEG videos are usually transmitted across networks for many applications, such as environment surveillance, net meeting, videophoning, etc., these videos can be acquired and tampered with ease. Therefore, it is necessary to verify at the receiver site that the content of the received video is original and has not been modified.

In Section 2.1.1, some related problem definitions are given, and in Section 2.1.2 the basic ideas of the proposed method are presented. In Section 2.2, the proposed random signal embedding method is described, and the proposed video verification method is stated in Section 2.3. In Section 2.4, several experimental results are shown to prove the feasibility of the proposed method.

2.1.1 Problem Definition

In this study, a video verification system is proposed. The basic task of such a system is to prove whether a given video has been tampered with or not. However, it is an even more essential requirement that the verification system can tell us where and how tampering was conducted in the given video. The proposed video verification system not only can check whether a video has been tampered with by a malicious user, but also can mark the tampered regions and recognize the tampering types.

Because a video stream may be regarded to possess three dimensions: two spatial ones and a temporal one, tampering manipulations in the video can be categorized into two different types: *spatial tampering* and *temporal tampering*. Spatial tampering means any modification on the image frame content, and temporal tampering means any manipulation performed on the image frame sequence.

In this study, temporal tampering of videos is categorized further into three types: *cropping*, *replacement*, and *insertion*. Cropping means deletion of some video frames by a malicious user. An illustration of frame cropping is shown in Figure 2.1. Insertion means addition of some fake video frames into the original video sequence. An illustration of frame insertion is shown in Figure 2.2. And Replacement means deletion of some video frames, followed by insertion of some other fake ones. An illustration of frame replacement is shown in Figure 2.3.

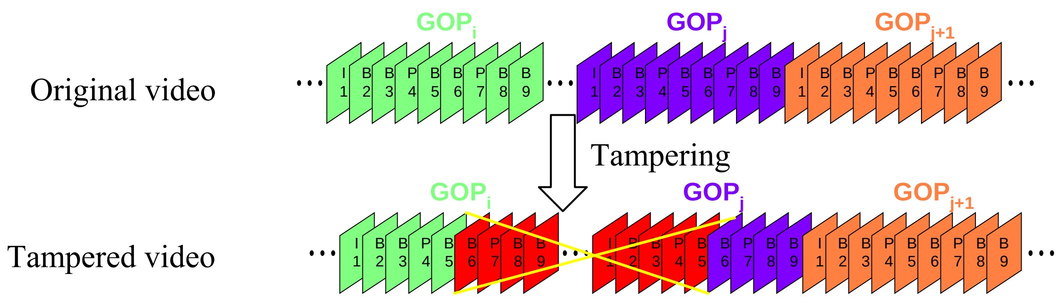


Figure 2.1 Illustration of cropping.

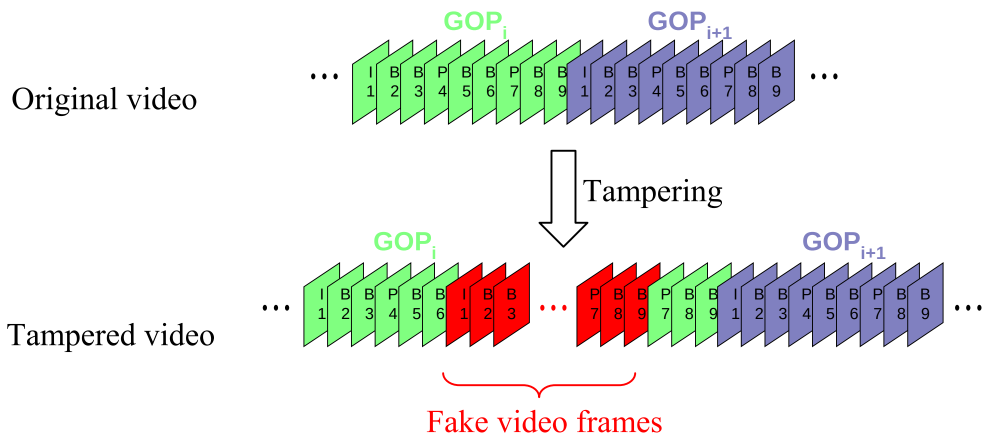


Figure 2.2 Illustration of insertion.

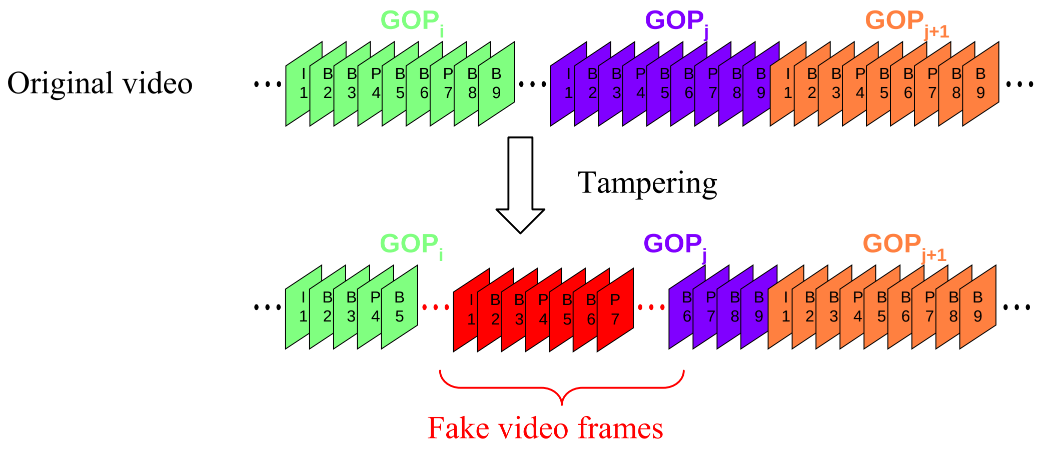


Figure 2.3 Illustration of replacement.

2.1.2 Proposed Idea

To detect spatial tampering, some random signals, called *authentication signals*, generated according to a user’s key are embedded in each frame of the video. For I frames, authentication signals are embedded into the coefficients of the DCT domain. For P and B frames, authentication signals are embedded into the motion vectors in the frames.

From our analysis of temporal tampering, two features are proposed in this study for use in detecting temporal tampering in the proposed method. One is the *index of the GOP of the video*. The other is the *number of the inter-coded frames in the GOP*. Both features will be embedded into the I frames of a video for the purpose of tampering detection.

2.2 Embedding Random Signals in MPEG Video

In this section, the proposed signal embedding method will be described. An illustration of the method is shown in Figure 2.4.

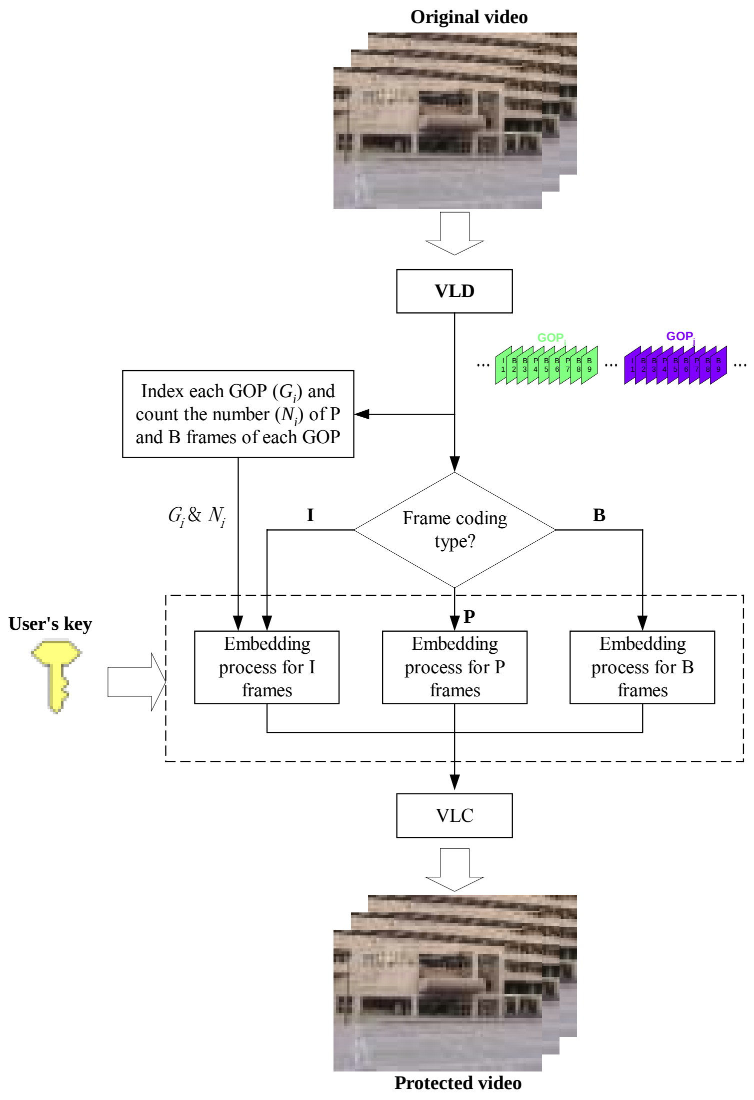

Original video
VLD
GOP
GOP
Index each GOP (G_i) and count the number (N_i) of P and B frames of each GOP
G_i & N_i
I
Frame coding type?
B
P
User's key
Embedding process for I frames
Embedding process for P frames
Embedding process for B frames
VLC
Protected video

Figure 4.4 Illustration of the proposed signal embedding method.

In Section 2.2.1, the process for embedding authentication signals in I frames will be described, followed by a description of the process for embedding authentication signals in P and B frames in Section 2.2.2.

2.2.1 Process for Embedding Random Signals in I Frames

In the proposed signal embedding process for I frames, two DCT coefficients, having the same quantization step size within the MPEG intra-quantization table of an 8╳8 luminance block, are selected as a pair to embed an authentication signal. Embedding is made possible by adjusting the relative values of the coefficient pair. Since the quantization step size of the two selected DCT coefficients are equal, the relative sizes between them will not be affected even when the coefficients are re-quantized. That is, the embedded authentication signals are robust to survive moderate image recompression.

In this study, the index of the *i*-th GOP of the input video is denoted as G_i, and the number of the inter-coded frames of the (*i*-1)-th GOP is denoted as N_i. G_i and N_i are embedded into some pre-defined macroblocks of the *i*-th I frame in the same fashion as embedding authentication signals, as mentioned previously. In order to extract these two types of features precisely in the verification process, the proposed system duplicates them many times before embedding them to reduce the probability of misrepresentation.

The MPEG intra-quantization table is shown in Table 2.1. Let (*x*, *y*) denote the location of a coefficient in an 8×8 block. In this study, the coefficients located at (5, 1) and (6, 0) are selected as a pair to embed an authentication signal. Moreover, the coefficients located at (4, 5) and (5, 4) are selected as a pair to embed G_i and N_i.

Table 2.1 Standard intra-quantization table in the MPEG compression standard (luminance component).

(x, y)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	8	16	19	22	26	27	29	34
1	16	16	22	24	27	29	34	37
2	19	22	26	27	29	34	34	38
3	22	22	26	27	29	34	37	40
4	22	26	27	29	32	35	40	48
5	26	27	29	32	35	40	48	58
6	26	27	29	34	38	46	56	69
7	27	29	35	38	46	56	69	83

A flowchart of the proposed signal embedding process for I frames is shown in Figure 2.5 and a corresponding detailed algorithm is described in the following.

***Algorithm* 1**: Signal embedding process for I frames.

Input: an I frame *F*, a user's key *R*, G_i, and N_i.

Output: a protected I frame F'.

Steps:

1. Denote the binary form of G_i as $G_i = g_1g_2\ldots g_{L_1}$, where L_1 is the length of G_i. Duplicate G_i

K times to form a new binary string G'_i.

2. Denote the binary form of N_i as $N_i = n_1n_2\ldots n_{L_2}$, where L_2 is the length of N_i. Duplicate N_i K times to form a new binary string N'_i.
3. For each 8×8 luminance block B, combine the input key R and the position P of B in F to form a seed for a random number generator to produce an authentication signal S.
4. Select the two DCT coefficient located at (5, 1) and (6, 0) in the intra-quantization table as a pair $P_1 = (C_1, C_2)$ to embed S. Before embedding S, compute $diff_1 = |C_1 - C_2|$. Embed S into P_1 according to the following two types of rules.
   - *When $diff_1 \le T_3$:*

$$\begin{cases} if\ S\ is\ odd, then\ set\ C_1 > C_2\ and\ |C_1 - C_2| = T_3; \\ if\ S\ is\ even, then\ set\ C_2 > C_1\ and\ |C_1 - C_2| = T_3. \end{cases} \tag{2.1}$$

   - *When $diff_1 > T_3$:*

$$\begin{cases} if\ S\ is\ odd\ and\ C_1 < C_2, then\ set\ C_1 = M_1 + (T_3/2)\ and\ C_2 = M_1 - (T_3/2); \\ if\ S\ is\ even\ and\ C_2 < C_1, then\ set\ C_1 = M_1 - (T_3/2)\ and\ C_2 = M_1 + (T_3/2); \end{cases} \tag{2.2}$$

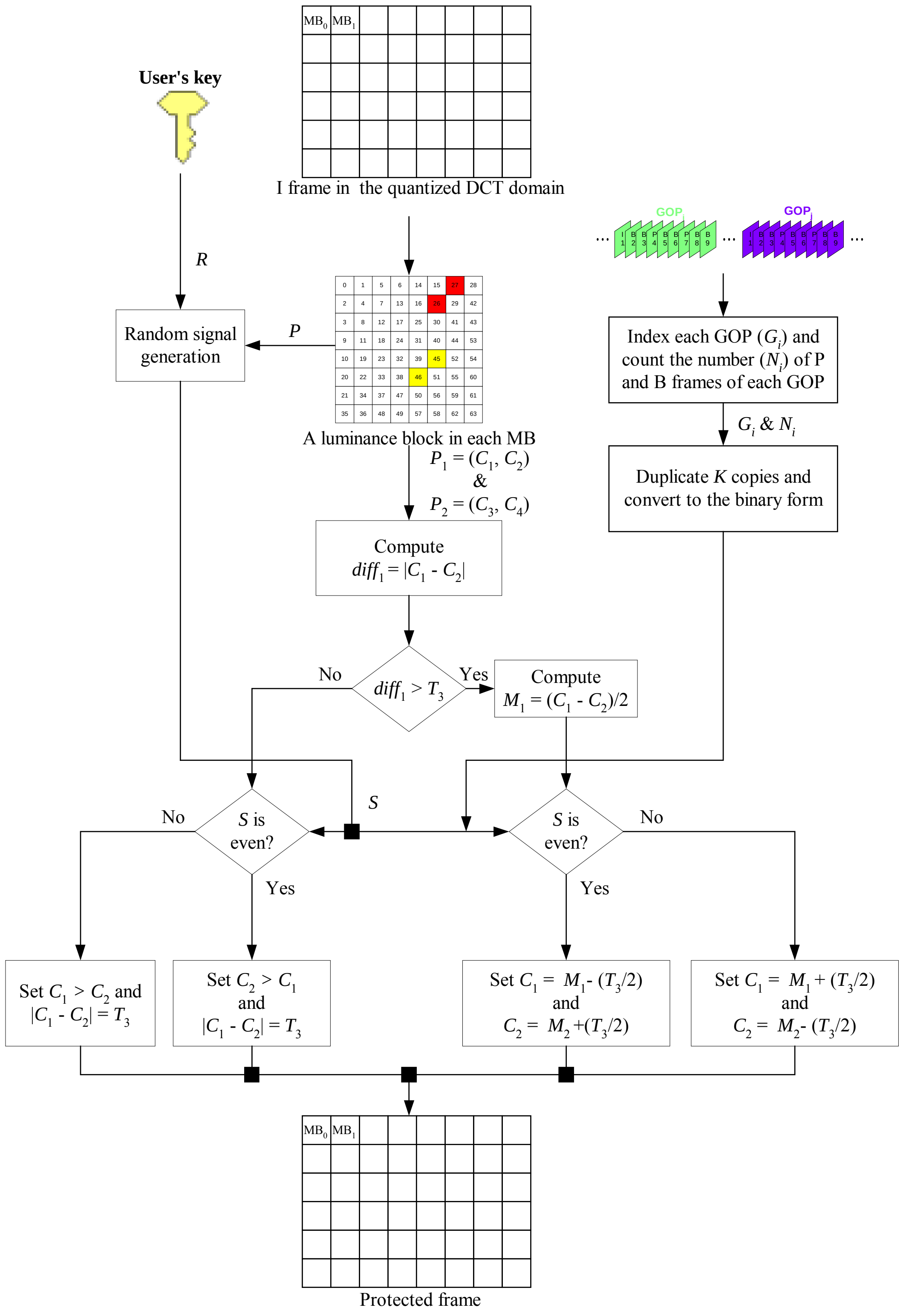
   where M_1 is the mean of C_1 and C_2 calculated as $M_1 = (C_1 + C_2)/2$, and T_3 is a pre-defined threshold value.
5. If the block B is one of the pre-defined blocks selected to embed G_i or N_i, then select the two DCT coefficient located at (4, 5) and (5, 4) as a pair $P_2 = (C_3, C_4)$ to embed a bit b of G'_i or N'_i. Before embedding b, compute $diff_2 = |C_3 - C_4|$. Embed b into P_2 according to the following two types of rules.
   - *When $diff_2 \le T_3$:*

$$\begin{cases} if\ b = 1, then\ set\ C_3 > C_4\ and\ |C_3 - C_4| = T_3; \\ if\ b = 0, then\ set\ C_4 > C_3\ and\ |C_3 - C_4| = T_3. \end{cases} \tag{2.3}$$

   - *When $diff_2 > T_3$:*

$$\begin{cases} if\ b = 1\ and\ C_3 < C_4, then\ set\ C_3 = M_2 + (T_3/2)\ and\ C_4 = M_2 - (T_3/2); \\ if\ b = 0\ and\ C_4 < C_3, then\ set\ C_3 = M_2 - (T_3/2)\ and\ C_4 = M_2 + (T_3/2); \end{cases} \tag{2.4}$$

   where M_2 is the mean of C_3 and C_4 calculated as $M_2 = (C_3 + C_4)/2$.



User's key
I frame in the quantized DCT domain
R
P
Random signal generation
A luminance block in each MB
$P_1 = (C_1, C_2)$ & $P_2 = (C_3, C_4)$
GOP
Index each GOP (G_i) and count the number (N_i) of P and B frames of each GOP
G_i & N_i
Duplicate K copies and convert to the binary form
Compute $diff_1 = |C_1 - C_2|$
$diff_1 > T_3$
No
Yes
Compute $M_1 = (C_1 - C_2)/2$
S
S is even?
Set $C_1 > C_2$ and $|C_1 - C_2| = T_3$
Set $C_2 > C_1$ and $|C_1 - C_2| = T_3$
Set $C_1 = M_1 - (T_3/2)$ and $C_2 = M_2 + (T_3/2)$
Set $C_1 = M_1 + (T_3/2)$ and $C_2 = M_2 - (T_3/2)$
Protected frame

Figure 2.5 Flowchart of the signal embedding process for I frames.

In the above algorithm, the threshold T_3 mentioned in Step 4 is a tradeoff between the robustness and the resulting video quality. The higher it is, the more robust the embedded authentication signals are against the MPEG compression, however, at the expense of degrading of the resulting video quality. Notice that the second set of rules mentioned in Step 4 is proposed to reduce the degradation of the resulting video quality caused by swapping C_1 and C_2 when the difference between C_1 and C_2 is larger than T_3.

2.2.2 Process for Embedding Random Signals in P and B Frames

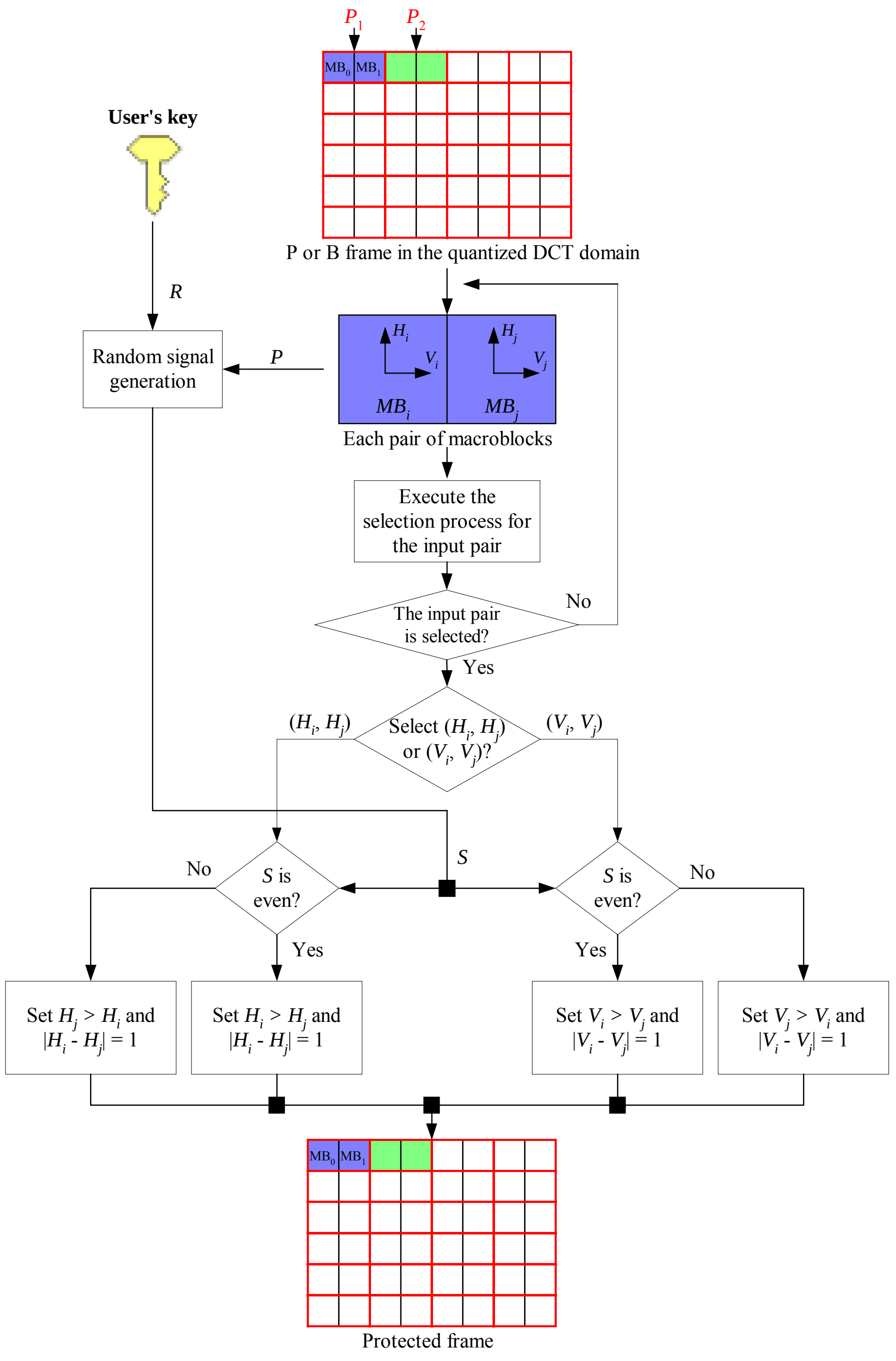
Since inter-coded frames are encoded by motion compensation prediction, embedding authentication signals in the motion vectors for authenticating the fidelity of inter-coded frames can utilize efficiently the information in the video bitstream. A detailed explanation has been stated previously.

In the proposed signal embedding process for each P or B frame of an input video, every two non-overlapping adjacent macroblocks in a P or B frame are selected to form a pair for embedding an authentication signal. However, not each pair is proper for embedding an authentication signal. The principles of selecting proper pairs are presented in the following.

For each pair of macroblocks (MB_i, MB_j) in a P frame, there are two candidates for embedding an authentication signal. One is (H_{fi}, H_{fj}); and the other is (V_{fi}, V_{fj}), where H_{fi} and V_{fi} are the horizontal and vertical components of the forward motion vector in the macroblock MB_i, and H_{fj} and V_{fj} are the horizontal and vertical components of the forward motion vector in the macroblock MB_j. In this study, two principles of how to select a proper pair are proposed. First, motion vectors whose magnitudes are large should be selected. Secondly, the difference between the two components in a candidate pair must be small. The details of the proposed selection process are described in the following.

For each pair of macroblocks (MB_i, MB_j) in a B frame, there are four candidates: (H_{fi}, H_{fj}), (V_{fi}, V_{fj}), (H_{bi}, H_{bj}), and (V_{bi}, V_{bj}), where H_{bi} and V_{bi} are the horizontal and vertical components of the backward motion vector in the macroblock MB_i, and H_{bj} and V_{bj} are the horizontal and vertical components of the backward motion vector in the macroblock MB_j.

A flowchart of the proposed signal embedding process for P and B frames is shown in Figure 2.6 and a corresponding detailed algorithm is described in the following.



P1
P2
MB0
MB1
User's key
P or B frame in the quantized DCT domain
R
Random signal generation
P
Hi
Vi
Hj
Vj
MBi
MBj
Each pair of macroblocks
Execute the selection process for the input pair
The input pair is selected?
No
Yes
(Hi, Hj)
Select (Hi, Hj) or (Vi, Vj)?
(Vi, Vj)
S
S is even?
No
Yes
S is even?
Yes
No
Set Hj > Hi and |Hi - Hj| = 1
Set Hi > Hj and |Hi - Hj| = 1
Set Vi > Vj and |Vi - Vj| = 1
Set Vj > Vi and |Vi - Vj| = 1
MB0
MB1
Protected frame

Figure 2.6 Flowchart of the signal embedding process for P or B frames.

***Algorithm* 2**: Signal embedding process for P and B frames.

Input: a P or B frame F, and a user's key R.

Output: a protected P or B frame F'.

Steps:

1. If the input frame F is a P frame, then perform Step 1.1; otherwise, perform Step 1.2.

   1.1 For each pair (MB_i, MB_j) of non-overlapping adjacent macroblocks in F, select (H_{fi}, H_{fj}) and (V_{fi}, V_{fj}) as two candidates for embedding an authentication signal where H_{fi} and V_{fi} are the horizontal and vertical components of the forward motion vector in MB_i, and H_{fj} and V_{fj} are the horizontal and vertical components of the forward motion vector in MB_j. And use the following rule to judge whether (H_{fi}, H_{fj}) is proper to embed an authentication signal:

$$\begin{cases} H_{fi} > T_4, \\ H_{fj} > T_4, \\ \left| H_{fi} - H_{fj} \right| \leq 1, \end{cases} \tag{2.5}$$

   where T_4 is a pre-defined threshold value. If proper, then select (H_{fi}, H_{fj}) to embed an authentication signal; otherwise, use the following rule to judge whether (V_{fi}, V_{fj}) is proper to embed an authentication signal:

$$\begin{cases} V_{fi} > T_4, \\ V_{fj} > T_4, \\ \left| V_{fi} - V_{fj} \right| \leq 1. \end{cases} \tag{2.6}$$

   If proper, then select (V_{fi}, V_{fj}) to embed an authentication signal. If the selected pair B consists of the horizontal components, then denote it as (H_i, H_j). On the contrary, if the selected pair consists of the vertical components, then denote it as (V_i, V_j).

   1.2 For each pair (MB_i, MB_j) of non-overlapping adjacent macroblocks in the input B frame F, there are four candidates: (H_{fi}, H_{fj}), (V_{fi}, V_{fj}), (H_{bi}, H_{bj}), and (V_{bi}, V_{bj}), where H_{bi} and V_{bi} are the horizontal and vertical components of the backward motion vector in MB_i, and H_{bj} and V_{bj} are the horizontal and vertical components of the backward motion vector in MB_j. The selection process for each pair of macroblocks (MB_i, MB_j) in a B frame is similar to the process used for P frames. Just use Equations (2.5) and (2.6), and (2.7) and (2.8) below sequentially to judge which candidate can be selected to embed an authentication signal:

$$\begin{cases} H_{bi} > T_4, \\ H_{bj} > T_4, \\ \left| H_{bi} - H_{bj} \right| \leq 1. \end{cases} \tag{2.7}$$

$$\begin{cases} V_{bi} > T_4, \\ V_{bj} > T_4, \\ \left| V_{bi} - V_{bj} \right| \leq 1. \end{cases} \tag{2.8}$$

If the selected pair *B* consists of the horizontal components, it is denoted as (H_i, H_j). On the contrary, if the selected pair consists of the vertical components, it is denoted as (V_i, V_j).

2. Combine the input key *R* and the position *P* of the selected pair *B* in *F* to form a seed for a random number generator to produce an authentication signal *S*.
3. If *B* is (H_i, H_j) from the first step, use the following rule to embed *S*:

$$\begin{cases} if\ S\ is\ odd, then\ set\ H_i > H_j\ and\ \left| H_i - H_j \right| = 1; \\ if\ S\ is\ even, then\ set\ H_j > H_i\ and\ \left| H_i - H_j \right| = 1. \end{cases} \tag{2.9}$$

On the contrary, if *B* is (V_i, V_j), use the following rule to embed *S*:

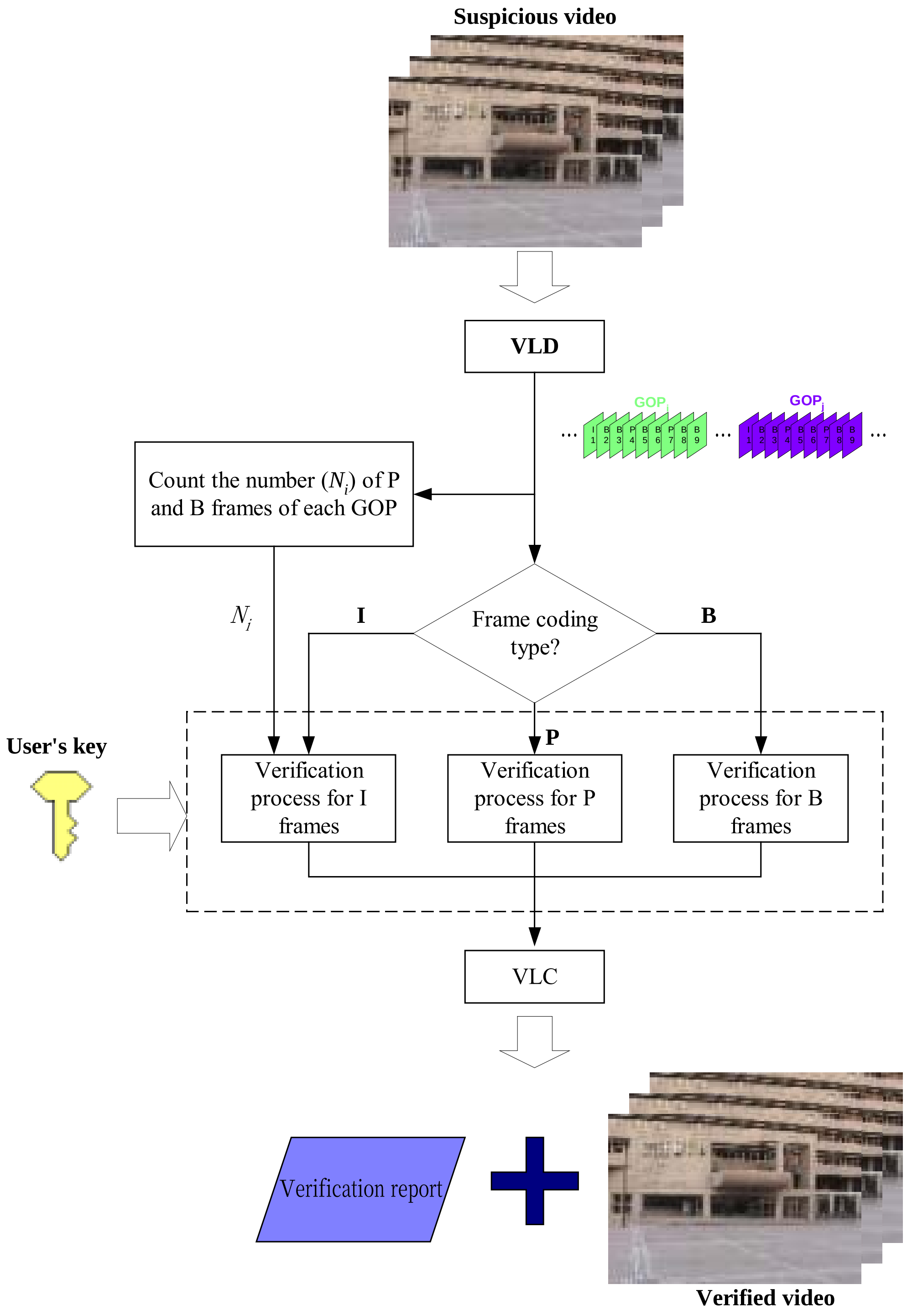
$$\begin{cases} if\ S\ is\ odd, then\ set\ V_i > V_j\ and\ \left| V_i - V_j \right| = 1; \\ if\ S\ is\ even, then\ set\ V_j > V_i\ and\ \left| V_i - V_j \right| = 1. \end{cases} \tag{2.10}$$

2.3 Content Verification

In this section, the proposed video content verification method will be described. An illustration of the method is shown in Figure 2.7. In Section 2.3.1, the process for verifying the integrity and fidelity of an I frame will be described. Next, the process for verifying the fidelity of a P or B frame will be described in Section 2.3.2.

2.3.1 Process for Verification of Integrity and Fidelity of I Frames

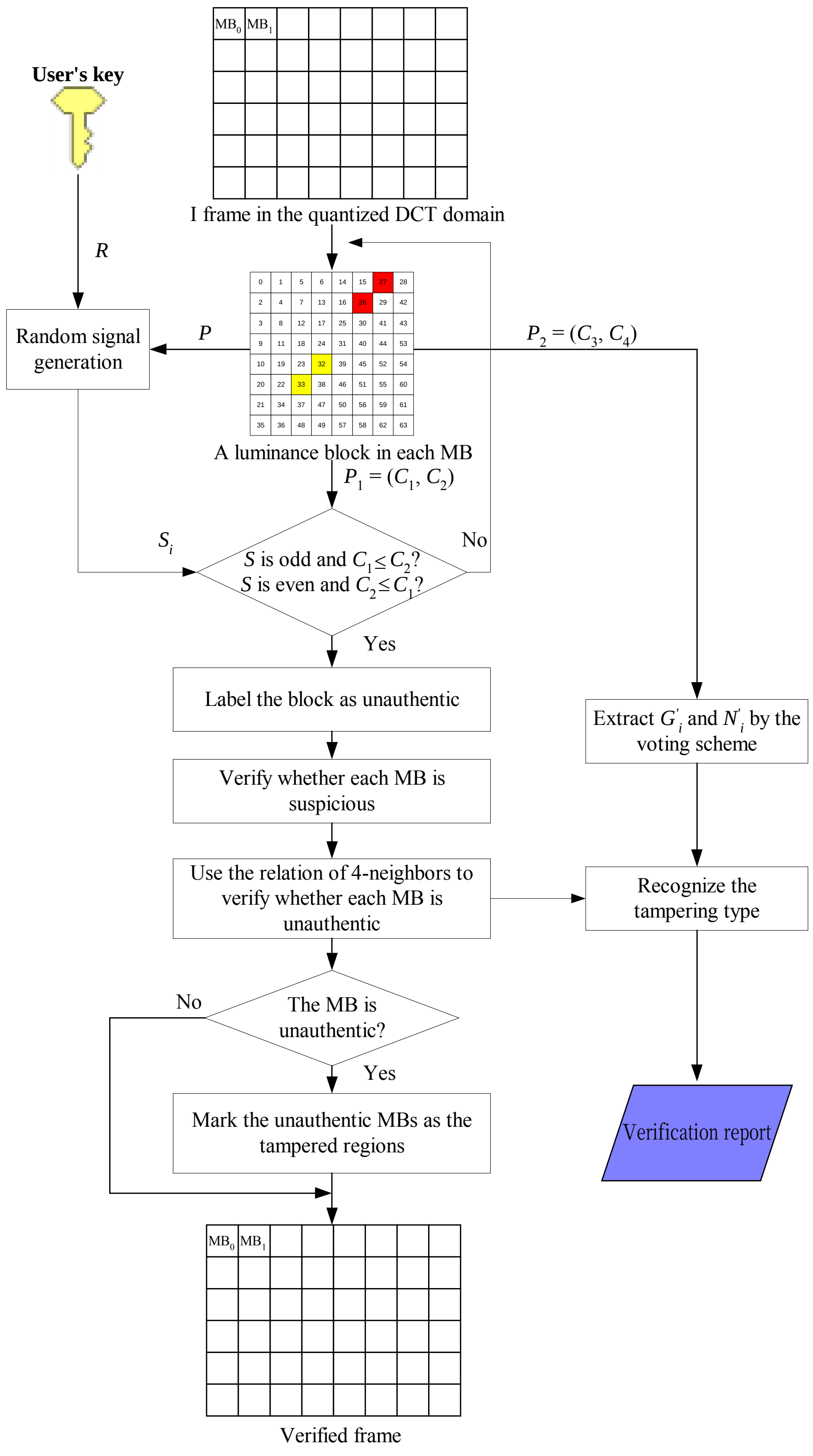
Using the embedded signals as described in the last section, not only the fidelity but also the integrity of each I frame can be verified by the proposed method, since an authentication signal is embedded in each 8×8 luminance block of the I frame. This is useful for detecting spatial tampering. In addition, two pre-defined features can be extracted from each I frames to detect whether *temporal tampering* has been attempted inside the input video. One feature is the index of each GOP, denoted as G'_i; and the other is the number of the P and B frames in each GOP, denoted as N'_i.



Suspicious video
VLD
GOP_i
GOP_j
Count the number (N_i) of P and B frames of each GOP
N_i
I
Frame coding type?
B
P
User's key
Verification process for I frames
Verification process for P frames
Verification process for B frames
VLC
Verification report
Verified video

Figure 2.7 Illustration of the proposed video content verification method.

The content verification process for an I frame is divided into two steps. The first is to verify each macroblock and extract G'_i and N'_i. The second is to recognize the tampering types. A flowchart of the content verification process for I frames is shown in Figure 4.8 and a corresponding detailed algorithm is described in following.



User's key
R
Random signal generation
P
I frame in the quantized DCT domain
MB0 MB1
A luminance block in each MB
P1 = (C1, C2)
P2 = (C3, C4)
Si
S is odd and C1 ≤ C2?
S is even and C2 ≤ C1?
No
Yes
Label the block as unauthentic
Verify whether each MB is suspicious
Use the relation of 4-neighbors to verify whether each MB is unauthentic
The MB is unauthentic?
No
Yes
Mark the unauthentic MBs as the tampered regions
Extract G'i and N'i by the voting scheme
Recognize the tampering type
Verification report
Verified frame

Figure 2.8 Flowchart of the proposed content verification process for I frames.

Algorithm 3: Content verification process for I frames.

Input: the *i*-th I frame *F* and a user's key *R*.

Output: the verified I frame and a verification report.

Steps:

1. For each 8×8 luminance block *B* of the input I frame *F*, combine the input key *R* and the position *P* of *B* in *F* to form a seed for a random number generator to produce a random signal *S*.
2. Use the pre-defined pair $P_1 = (C_1, C_2)$ of the DCT coefficients to verify the existence of an embedded authentication signal according to the following rule:

$$\begin{cases} \text{if } S \text{ is odd and } C_1 \le C_2, \text{ then label this block as unauthentic;} \\ \text{if } S \text{ is even and } C_2 \le C_1, \text{ then label this block as unauthentic.} \end{cases} \tag{2.11}$$

3. If the block *B* is one of the pre-defined blocks selected to embed the index G'_i of each GOP, then use the pre-defined pair $P_2 = (C_3, C_4)$ of the DCT coefficients to extract a bit $g'(j)$ as part of the bitstream g' according the following rule:

$$g'(j) = \begin{cases} 1, & \text{if } C_3 > C_4; \\ 0, & \text{otherwise}; \end{cases} \tag{2.12}$$

   where $1 \le j \le L_1 \times K$, and L_1 is the length of the binary form of G'_i, and *K* is the number of copies used originally.

4. If the block *B* is one of the pre-defined blocks selected to embed the number N'_i of the P and B frames in each GOP, then use the pre-defined pair $P_2 = (C_3, C_4)$ of the DCT coefficients to extract a bit $n'(j)$ as part of the bitstream n' according the following rule:

$$n'(j) = \begin{cases} 1, & \text{if } C_3 > C_4; \\ 0, & \text{otherwise}; \end{cases} \tag{2.13}$$

   where $1 \le j \le L_2 \times K$, and L_2 is the length of the binary form of N'_i.

5. After processing each luminance block, employ the following three steps to verify each macroblock *MB*.
   5.1 If two or more of the four luminance blocks of *MB* are considered unauthentic, then consider *MB* as suspicious.
   5.2 If two or more of the 4-neighbors of a suspicious *MB* are considered suspicious, then consider *MB* as unauthentic. The 4-neighbor relationship is illustrated in Figure 2.9.
   5.3 Mark unauthentic macroblocks as tampered regions.
6. After extracting all bits of g', perform majority voting to get a result as follows:

$$V(m) = \sum_{a=0}^{K-1} g'(a \times L_1 + m), \tag{2.14}$$

where $1 \le m \le L_1$. Then, reconstruct $g(m)$ by the following rule:

$$g(m) = \begin{cases} 1, & \text{if } V(m) > \dfrac{K}{2}; \\ 0, & \text{otherwise,} \end{cases} \tag{2.15}$$

where $1 \le m \le L_1$. And convert the binary string $g(m)$ into a decimal value G'_i.

7. After extracting all bits of n', perform majority voting to get a result as follows:

$$V(m) = \sum_{a=0}^{K-1} n'(a \times L_2 + m), \tag{2.16}$$

where $1 \le m \le L_2$. Then reconstruct $n(m)$ by the following rule:

$$n(m) = \begin{cases} 1, & \text{if } V(m) > \dfrac{K}{2}; \\ 0, & \text{otherwise,} \end{cases} \tag{2.17}$$

where $1 \le m \le L_2$. And convert the binary string $n(m)$ into a decimal value N'_i.

8. If some macroblocks of F are considered unauthentic, decide that spatial tampering has been attempted inside the video. Then use the following rule to determine the temporal tampering type TTT:

$$\begin{cases} \mathit{if}\ (G'_i - G'_{i-1}) \ne 1\ \mathit{and}\ E_i = 0, \mathit{then}\ TTT\ \mathit{is\ cropping}; \\ \mathit{if}\ (G'_i - G'_{i-1}) \ne 1\ \mathit{and}\ E_i \ne 0, \mathit{then}\ TTT\ \mathit{is\ replacement}; \\ \mathit{if}\ (G'_i - G'_{i-1}) = 1\ \mathit{and}\ E_i \ne 0, \mathit{then}\ TTT\ \mathit{is\ insertion}; \\ \mathit{if}\ (G'_i - G'_{i-1}) = 1\ \mathit{and}\ E_i = 0\ \mathit{and}\ (N_i - N'_i) \ne 0, \mathit{then}\ TTT\ \mathit{is\ cropping}; \end{cases} \tag{2.18}$$

where G'_i and $G'_{i\text{-}1}$ are the GOP indexes extracted from the i-th and (i-1)-th I frame, respectively; N'_i is the number of the P and B frames extracted from the i-th I frame; N_i is the number of the P and B frames in the (i-1)-th GOP of the input video; and E_i is the number of the unauthenticated frames before the i-th I frame.

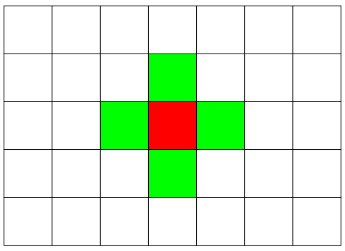


Figure 2.9 Illustration of 4-neighbor relationship (The four green macroblocks are the 4-neighbors of the red one).

2.3.2 Process for Verification of Fidelity of P and B Frames

In the content verification process of a P or B frame, a characteristic of inter-coded frames in the MPEG standard can be utilized to verify the fidelity of a frame, that is, the property that the number of the intra-coded macroblocks in P or B frames is usually small. If a P or B frame is manipulated illegally, then most of the macroblocks in the frame will become intra-coded ones, resulting in a great increase of the number of such macroblocks, because the illegal manipulation will cause the frame to become quite different from its reference frame. Therefore, the proportion of the number N_{intra} of the intra-coded macroblocks to the number N_{all} of the total macroblocks in the frame can be utilized for verifying the fidelity of the frame first. More specifically, if the proportion of N_{intra} to N_{all} is high, then this frame is decided to be unauthentic.

In the second verification step, the number N_{sel} of the pairs of macroblocks which satisfy the conditions specified by Equations (2.5) through (2.8) presented previously is checked. If N_{sel} is greater than a pre-selected threshold value, it means that the number of the authentication signals embedded in the frame is large enough, which may be used to verify the fidelity of the frame. On the contrary, if N_{sel} is smaller than a pre-selected threshold value, it indicates that the authentication signals embedded in the frame are insufficient for reliable fidelity verification. In this case, a method based upon a *temporal reference relation* is proposed in this study for verifying the frame. The method is presented as follows. For a P frame, if its forward reference frame is authentic, then the current frame is decided to be authentic; otherwise, the frame is decided to be unauthentic, and then the proposed verification system will mark as tampered those regions in the current frame whose corresponding regions in its forward reference frame were marked tampered.

For a B frame, the number N_f of the forward-coded macroblocks of the frame and the number N_b of the backward-coded macroblocks of the frame must be compared first to decide whether the frame is similar to its forward reference frame or to its backward reference frame. If N_f is greater than N_b, it means the frame is similar to its forward reference frame. In this case, if its forward reference is authentic, then the frame is decided to be authentic; otherwise, the frame is decided to be unauthentic, and then the proposed verification system will mark as tampered those regions in the current frame whose corresponding regions in its forward reference frame were marked tampered. On the contrary, if N_f is smaller than N_b, it means the frame is similar to its backward reference frame. In this case, if its backward reference frame is authentic, then the frame is decided to be authentic; otherwise, the frame is decided to be unauthentic, and then the proposed verification system will mark as tampered those regions in the current frame whose corresponding regions in its backward reference frame were marked tampered. A reason for using this method is that tampering, taking place in a brief amount of time, should occur within neighboring regions in a series of frames that are similar to each other. If not so, tampering should be easily detected.

A flowchart of the content verification process for P and B frames is shown in Figure 2.10 and a corresponding detailed algorithm is described in following.

***Algorithm* 4**: Content verification process for P and B frames.

Input: a P or B frame F and a user's key R.

Output: the verified P or B frame.

Steps:

1. Count the number N_{intra}, N_f, and N_b of the intra-coded, forward-coded, and backward-coded macroblocks of the input P or B frame F, respectively. In the mean time, count the number N_{all} of all the macroblocks of F. Judge whether the input frame is authentic by the following rule:

$$\begin{cases} if\ \dfrac{N_{\text{int}ra}}{N_{all}} > T_5, & then\ the\ input\ \ frame\ is\ unauthentic; \\ otherwise, & continue\ the\ next\ step; \end{cases} \tag{2.19}$$

   where T_5 is a pre-defined threshold value.

2. For each pair of non-overlapping adjacent macroblocks of F, use the selection process presented in Section 2.2.2 to select the pairs used to embed authentication signals. If the selected pair consists of the horizontal components, it is denoted as (H_i, H_j). On the other hand, if the selected pair consists of the vertical components, it is denoted as (V_i, V_j). And then count the number N_{sel} of the selected pairs. If N_{sel} is larger than a pre-defined threshold value T_6, perform Step 3; otherwise, perform Step 4.

3. For each selected pair B, combine the input key R and the position P of B in F to form a seed for a random number generator to produce a random signal S. If the selected pair B in Step 2 is (H_i, H_j), then examine the relation between S and (H_i, H_j) and count the number N_w of unauthentic pairs according to the following rule:

$$\begin{cases} \text{if } S \text{ is odd and } H_i \le H_j, \text{ then } N_w = N_w + 1; \\ \text{if } S \text{ is even and } H_j \le H_i, \text{ then } N_w = N_w + 1. \end{cases} \tag{2.20}$$

If the selected pair in Step 2 is (V_i, V_j), then examine the relation between S and (V_i, V_j) and count the number N_w of unauthentic pairs according to the following rule:

$$\begin{cases} \text{if } S \text{ is odd and } V_i \le V_j, \text{ then } N_w = N_w + 1; \\ \text{if } S \text{ is even and } V_j \le V_i, \text{ then } N_w = N_w + 1. \end{cases} \tag{2.21}$$

After verifying each selected pair, decide whether the frame F is authentic by the following rule:

$$\begin{cases} \text{if } \dfrac{N_w}{N_{sel}} > T_7, & \text{then the input frame is unauthentic;} \\ \text{otherwise,} & \text{this input frame is authentic;} \end{cases} \tag{2.22}$$

where T_7 is a pre-defined threshold value.

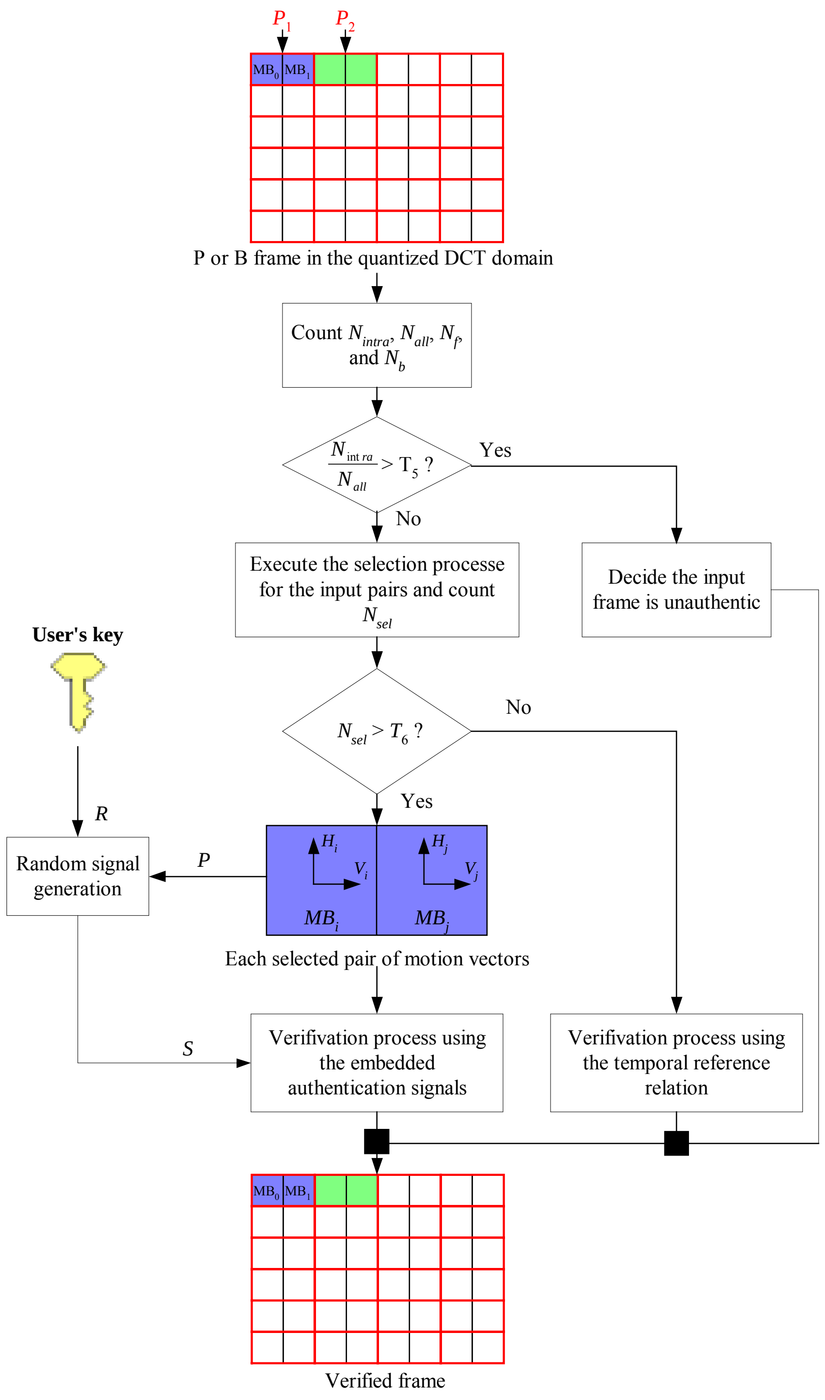
4. If the input frame F is a P frame, then perform Step 4.1; on the other hand, if F is a B frame, then perform Step 4.2.

   4.1 If the forward reference frame R_f of F is considered authentic in the previous verification process, then decide F to be authentic; otherwise, unauthentic and mark as tampered those regions in F whose corresponding regions in R_f were marked tampered.

   4.2 First, compare N_f with N_b. If N_f is larger than N_b, perform Step 4.2.1; otherwise, Step 4.2.2.

      4.2.1 If the forward reference frame R_f of F is considered authentic in the previous verification process, then decide F to be authentic; otherwise, unauthentic and mark as tampered those regions in F whose corresponding regions in R_f were marked tampered.

      4.2.2 If the backward reference frame R_b of F is considered authentic in the previous verification process, then decide F to be authentic; otherwise, unauthentic and mark as tampered those regions in F whose corresponding regions in R_b were marked tampered.

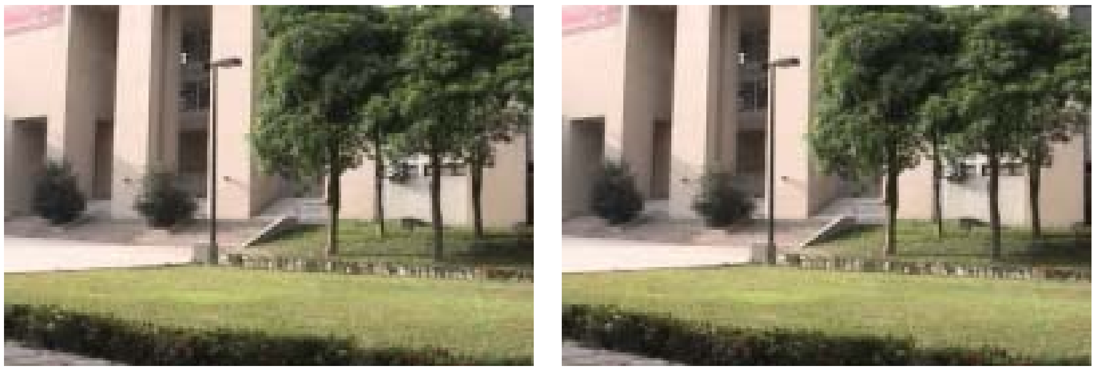


P_1
P_2
MB_0
MB_1
P or B frame in the quantized DCT domain
Count N_{intra}, N_{all}, N_f, and N_b
$\frac{N_{intra}}{N_{all}} > T_5$?
Yes
No
Execute the selection processe for the input pairs and count N_{sel}
Decide the input frame is unauthentic
User's key
$N_{sel} > T_6$?
No
Yes
R
Random signal generation
P
H_i
V_i
MB_i
H_j
V_j
MB_j
Each selected pair of motion vectors
S
Verifivation process using the embedded authentication signals
Verifivation process using the temporal reference relation
MB_0
MB_1
Verified frame

Figure 2.10 Flowchart of proposed content verification process for P and B frames

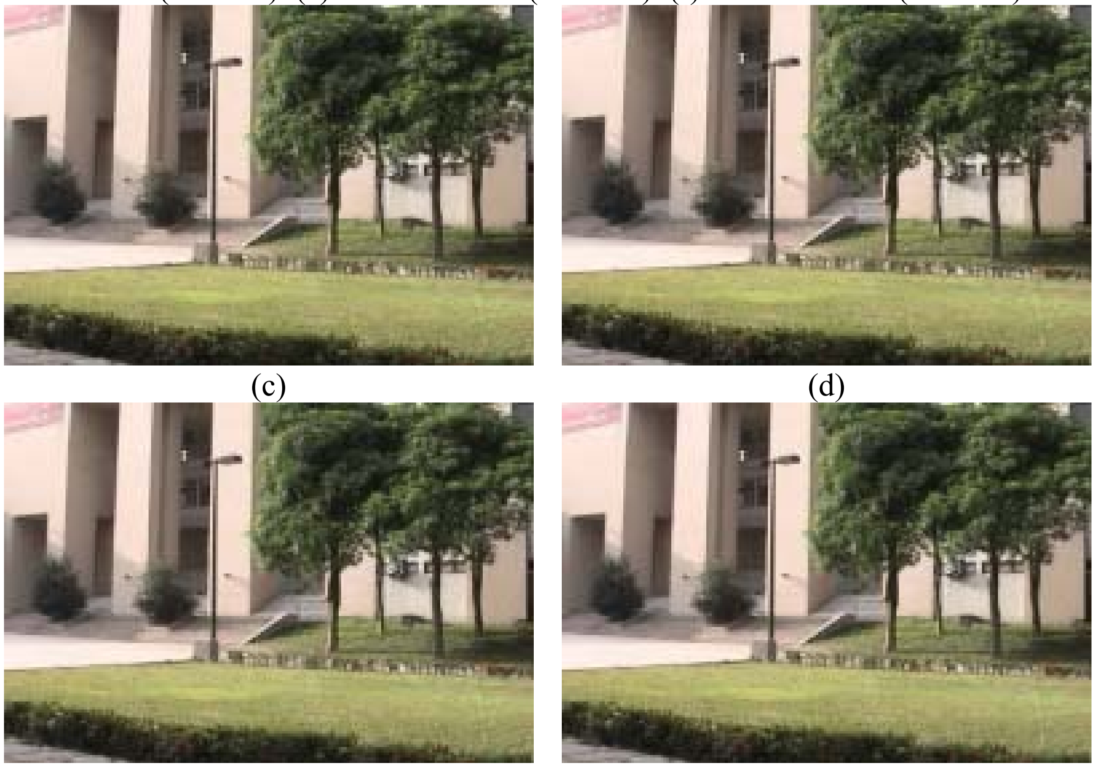
2.4 Experimental Results

In our experiments, a video with frame size 352╳240 was used as the input. Six frames of the input video are shown in Figure 2.11. The six corresponding frames of the resulting video after the proposed signal embedding process was performed are shown in Figure 2.12. The PSNR values of them are shown in Table 2.2, from which we can see that the authentication signals can be embedded into MPEG videos imperceptibly by applying the proposed method. Figure 2.13 shows certain modification results of the six frames by commercial software. Figure 2.14 is the verification result of these modified frames, in which the yellow regions represent the tampered regions. Moreover, the tampering was recognized to be of the type of spatial tampering. From these figures we can see that the tampered regions can be identified efficiently and the tampering types can be recognized correctly by the proposed method.



(a) (b)

Figure 2.11 Six frames of the original video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).



(c) (d)

(e) (f)

Figure 2.11 Six frames of the original video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame) (continued).

Table 2.2 The PSNR values of the resulting video.

	f_0 (I)	f_1 (B)	f_2 (B)	f_3 (P)	f_4 (B)	f_5 (B)
PSNR	34.0	34.1	34.1	34.0	36.0	36.1

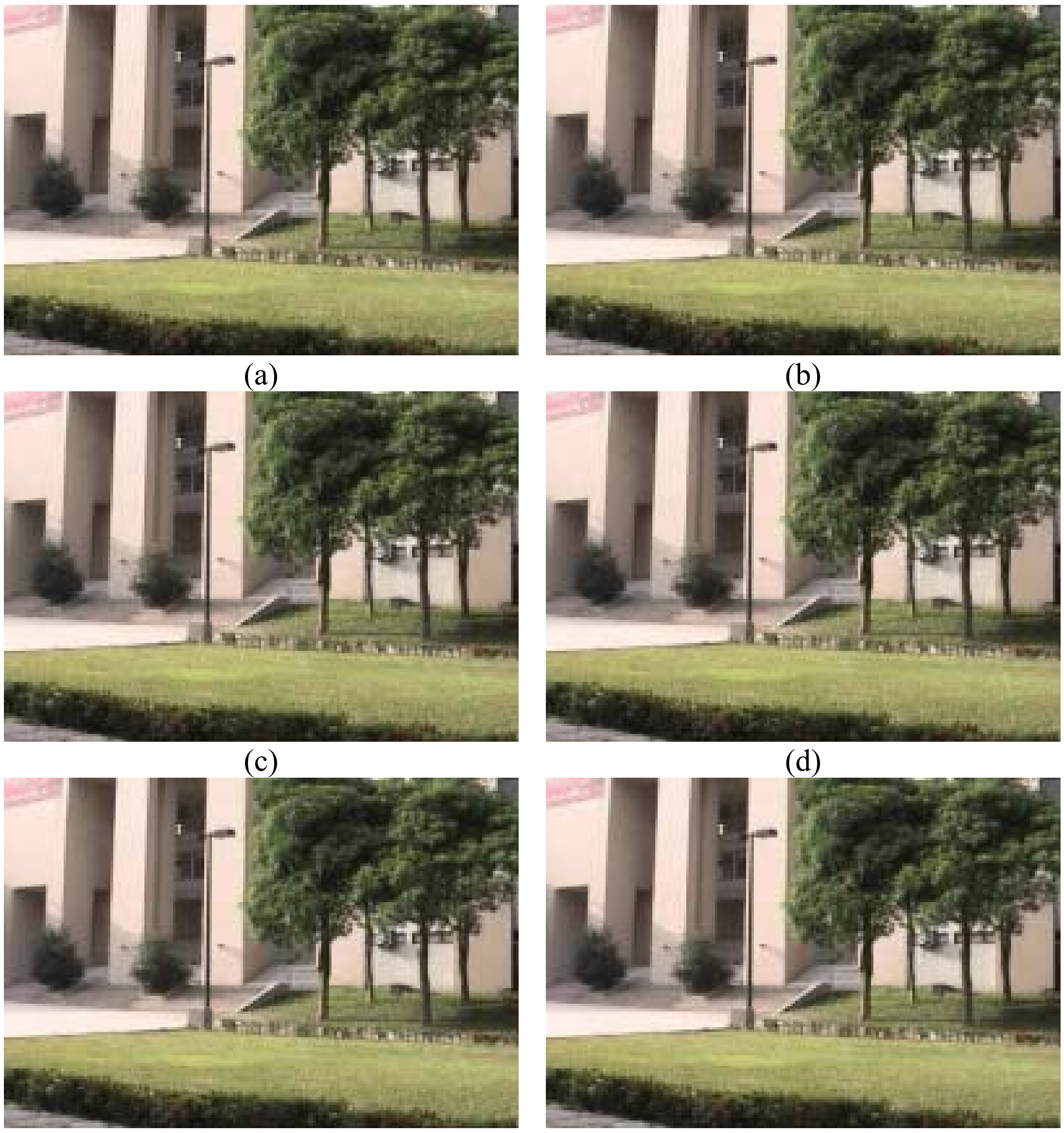
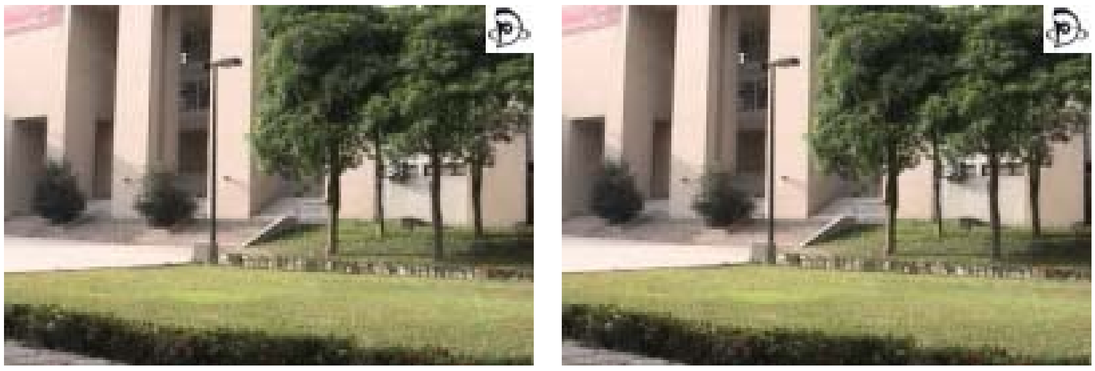
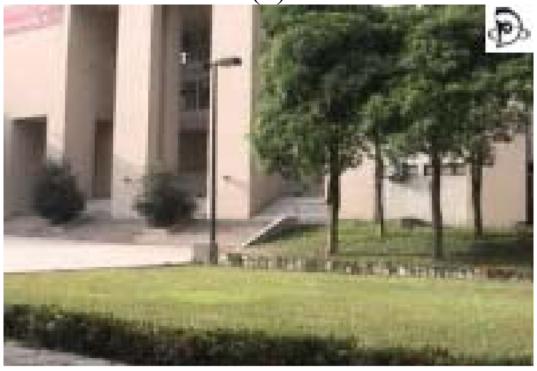


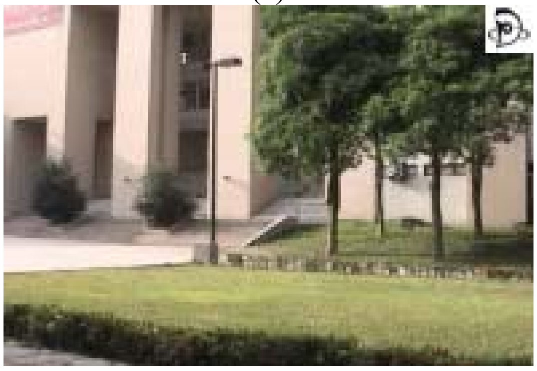
Figure 2.12 Six frames of the resulting video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).

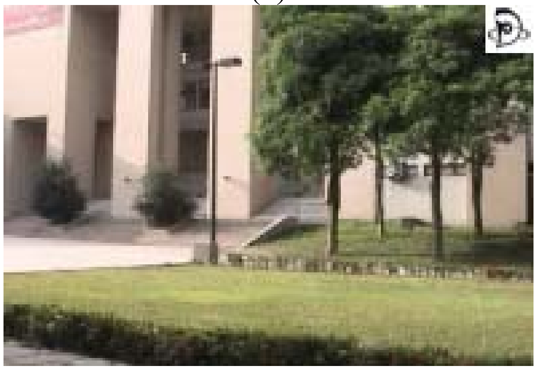


(a) (b)



(c) (d)





(e) (f)

Figure 2.13 Six frames of the resulting video that has been modified. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).

(a) (b)

(c) (d)

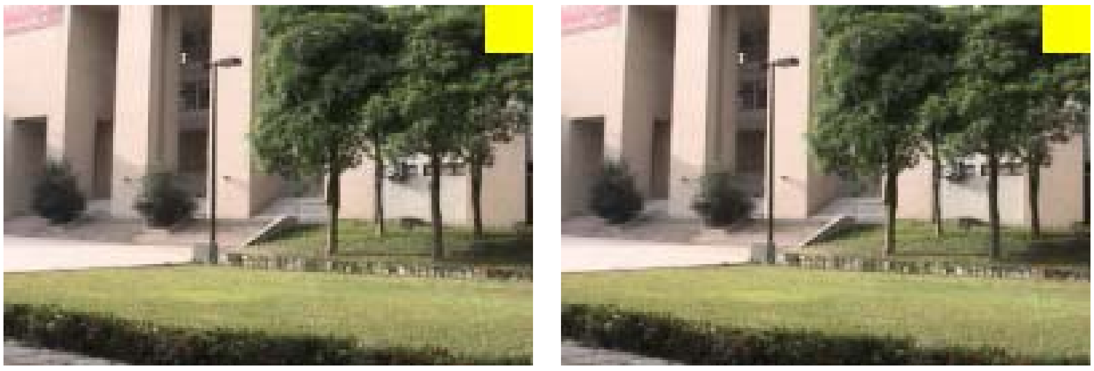

(e) (f)

Figure 2.14 Six frames of the verified video of a modified video. (a) The first frame (I frame). (b) The second frame (B frame). (c) The third frame (B frame). (d) The 4th frame (P frame). (e) The 5th frame (B frame). (f) The 6th frame (B frame).

Chapter 3

Multiple Authorizations for Images, Videos or Software Packages

In this chapter, when an owner distributes a product to authorized users, it is a problem to deter authorized users from releasing the authorized product and to trace back to the releasing user if the authorized product was released. Some methods based on fingerprinting techniques are proposed in this study.

The remainder of this chapter is organized as follows. In Section 3.1, an introduction is given first. In Section 3.2, multiple authorizations for images or software packages mechanisms are presented briefly.

3.1 Introduction

Digital watermarking has been proposed for the purposes of copyright and integrity protection of digital media. Many watermarking techniques have been applied to images. The main idea is to insert a unique watermark and some authentication signals into images. When suspicious images are discovered, with the help of the extracted watermark, the image ownership can be protected. With the help of the extracted authentication signals, the image integrity and fidelity can be verified.

An image authentication center can provide a software package for each owner to protect image copyrights. When the image copyright is unambiguous, the image authentication center can judge whether the image is tampered with and to whom the image copyright belongs. So, it is an important issue to design the software package such that the image authentication center can claim the image ownership correctly. Besides, an image owner can distribute an image to authorized users. It is also an important issue to trace back to the releasing user if the authorized product was released. Fingerprinting and watermarking techniques can be applied to solve problems in this issue. We will introduce how to apply the concept of the fingerprinting and watermarking techniques in images and software packages in Section 3.2.

3.2 Multiple Authorization Mechanisms

Fingerprinting means the process of adding fingerprints to an object or identifying fingerprints that are already intrinsic to an object. In this section, the idea of fingerprinting and watermarking techniques is used in three cases. One is in images, another is in software packages and the other is in videos. We will discuss the three cases, respectively.

A. Case A: Images

Sometimes, an image owner will distribute an image to authorized users. If the image is released, the owner wishes to capture the traitor. So, the concept of fingerprinting and watermarking can be employed in this situation. We take a watermark as a fingerprint. The watermark contains the identity of the recipient of the protected content.

For the authorized image, we use watermarking techniques to embed a watermark in each copy of the image to create a stego-image with an identifiable watermark each time. That is, each authorized user will get an image with an identifiable watermark. When a copyright violation is detected, the watermark will be extracted to allow tracking back to the infringing user. By identifying the watermark, the traitor can be caught.

B. Case B: Software Packages

The image authentication center (IAC) can offer a software package for image copyright protection to many image owners. Consider the following situation. One of the image owners utilizes the software package to embed his or her ownership in images and get watermarked images. And then, other image owners may obtain watermarked images by some ways like browsing on the Internet. If they utilize the software package to embed their ownerships in these images, the image copyright will belong to the latter, not the former. That is, the earliest ownership of the stego-image is replaced with the latest one. It is an important problem to design the software package. So, two approaches are proposed in this study to solve this problem. They are described subsequently.

The first is that after dealing with images, the owner goes to the IAC and registers these images in the IAC. When the suspicious image is discovered, the owner can bring the image to the IAC. The IAC can umpire a dispute by the timestamp of the registration. The image ownership can be verified via the IAC. That is, all judgments are based on the registration time. But this is not effective. People with the software package can still embed information in the image. And if the registration time of the owner is later than that of the misused user, the image copyright will not be judged to belong to the owner again.

So, another approach is proposed. The main idea of the proposed approach is to detect whether the image is processed. If the image has been processed, other user cannot embed in it. Two important signals may be used in the embedding process. One signal is designated as P. It can determine whether the image is processed or not. The other signal designated as S is the *identifiable code* of the software package. The identifiable code of each authorized user is different and unique. The IAC gives a unique identifiable code with respect to each authorized user in each software package. When an owner wants to embed ownership in the image, the embedding process may be designed to insert automatically the signals P and S into the image. So, if the image has not been processed, the signal P and S will not exist. And any user can embed information in this image. Otherwise, it means that the signal P and S exists in this image. So, when an owner wants to embed his copyright in the image, the embedding process will check S in the image, called s. If the s is equal to the identifiable code S of the software package, it means that the user is the same as the owner of the image and the information can be embedded into this image. In the meantime, the earlier information will be replaced with the last one. If the *s* is not equal to the identifiable code *S* of the software package, no one can hide information in this image. Therefore, the image copyright can be protected. The advantage of this approach is that other users will not hide information in the stego-image. The ownership of the stego-image will not be replaced. Figure 3.1 shows the procedure about the embedding process of the second approach.

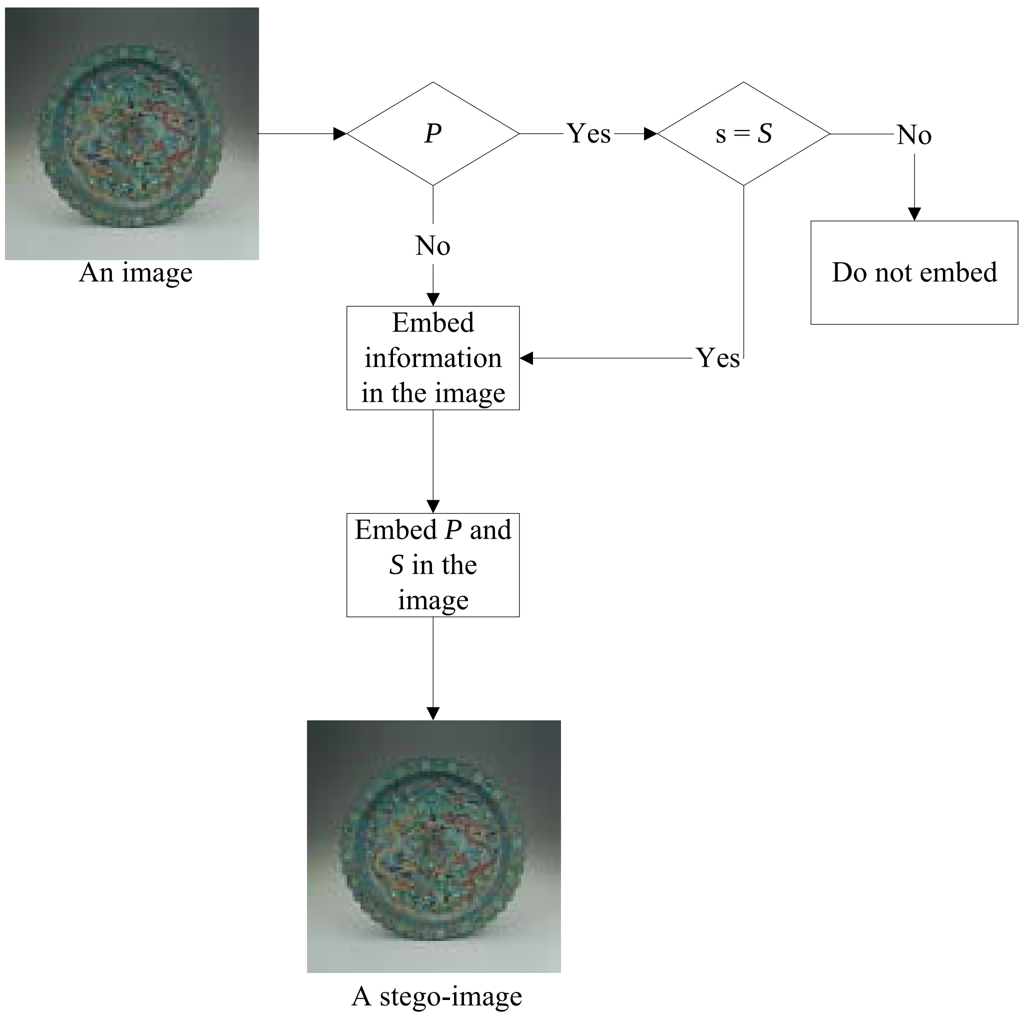


Figure 3.1 Procedure about the embedding process of the second approach.

In addition, in the second approach, we can take identifiable codes as fingerprints. So, if the software package is released, we can trace back to the releasing user by detecting the identifiable code of the software package.

C. Case C: Videos

When an image authentication center (IAC) provides a software package to multiple video owners, it must prevent the problem we have discussed in the introduction carefully. There are three approaches proposed.

The first is that after dealing with videos, the owner goes to the IAC and registers these videos in the IAC. When the suspicious video is discovered, the owner can bring the video to the IAC. The IAC can umpire a dispute by the timestamp of the registration. The video ownership can be verified via the IAC. That is, all judgments are based on the registration time. But this is not effective. People with the software package can still embed information in the video. And if the registration time of the owner is later than that of the misused user, the video copyright will not be judged to belong to the owner again.

So, another approach is proposed. The main idea is when we insert ownership into a video, the embedding process also insert two signals S and P at the same time. P determines whether the video is processed or not. *S* is the *identifiable code* of the software package. The identifiable code of each authorized user is different and unique. If P does not exist in a video, it shows that this video has not been processed. So, we can insert watermarks and authentication signals along with P and S into this video. But if we find signal P during the embedding process (means that this video has been processed before), we will extract S from the video and check whether S equals to the identifiable code s of this user or not. If it does, it means that the extracted authentication signals are inserted by the current user. Hence, we will replace the original authentication signals with newer ones. If $S \neq s$, it means that the authentication signals are inserted into this video by another user. In order to protect the rights of the one who inserted the authentication signals. The embedding process will deny the request of embedding new authentication signals. In other words, the inserted authentication signals will be protected from being replaced by someone else. Figure 3.2 Procedure about the embedding process of the third approach.

The last of three approaches is proposed with the idea of a secret key. Three significant signals will be hidden in the video. Two of them are the same as those mentioned above, namely, *P* and *S*. The other is a key signal *K*, which is an arbitrary code assigned by the user. When an owner wants to embed ownership in the video, the embedding process will ask the owner to input a key *K* and insert automatically the signals *P*, *S*, and *K* into the video. If a user wants to embed another copyright in the stego-video, first the embedding process will request the user to input a key *k*. If *k* is different from *K*, the user cannot embed because the embedding process regards the action as a tort. Otherwise, the user can hide information in the stego-video, and the embedding process will extract the embedded original identifiable code *s* from the stego-video and embed *s* again. Besides, the embedding process will also embed the identifiable code *S* of the user. An advantage of this approach is that the owner can control the embedding operation by a key. Even if a misappropriating user guesses the correct key to embed information in the stego-video, both the earliest identifiable code and the identifiable code of a misappropriating user will be kept in the stego-video. When the video is unambiguous, the IAC can extract the earliest identifiable code to judge the ownership of the suspicious video and extract another identifiable code to know which user has misappropriated the video.

In addition, in the second and the third approaches, we can take identifiable codes as fingerprints. So, if the software package is released, we can trace back to the releasing user by detecting the identifiable code of the software package.

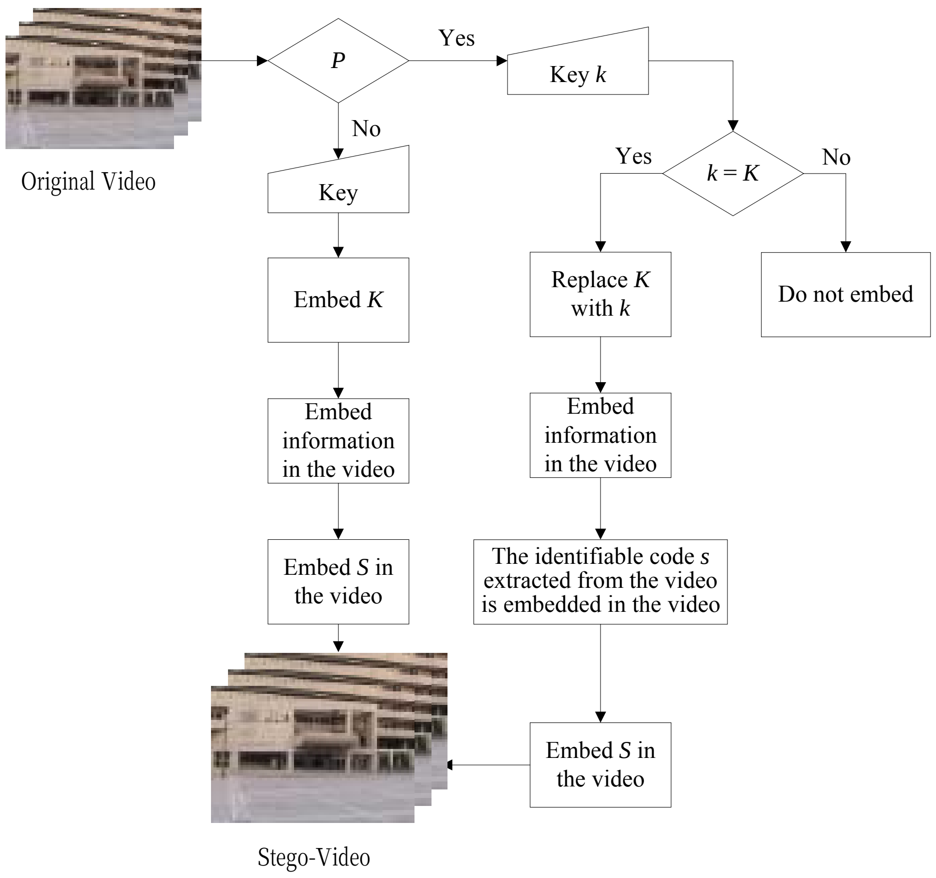

Original Video
P
Yes
No
Key k
Key
k = K
Yes
No
Embed K
Replace K with k
Do not embed
Embed information in the video
Embed information in the video
Embed S in the video
The identifiable code s extracted from the video is embedded in the video
Embed S in the video
Stego-Video

Figure 3.2 Procedure about the embedding process of the third approach.

D. Discussion:

In this chapter, a study about multiple authentications for videos or software packages and video is stated. In the multiple authentications for videos or software packages, we can apply the idea of fingerprinting and watermarking techniques in videos and software packages. For videos, each watermark with respect to each authorized user is embedded each time in a video to create a stego-video. By checking the embedded watermark, we can catch the traitor. For software packages, three approaches have been proposed. The first approach is a passive method. Any authorized owner with the software package can hide image copyrights in videos. The owner must register the video in the video authentication center. Via the video authentication center, the copyright of the video can be proved. The second approach is an active method. No one, except the owner, can embed copyrights in videos. That is, the video will not be replaced with other copyrights. The copyright of the video will exist until the owner embeds a new one in it. The last approach is a compromised method. The concept of the secret key is utilized. The user cannot embed information in videos without the correct key. Even if the user obtains the correct key to embed his copyright, the owner can send the video to the video authentication center. With the help of the extracted identifiable code in videos, the copyright of the video owner can still be proved by the video authentication center and the misused user will be caught. Video authentication centers thus can support video authentication for copyright claim.

The structure of a video authentication center may be divided into two parts from our viewpoint. One is the central video authentication center. The other is the local image authentication center. Besides, video authentication centers are suggested to fulfill several service functions, namely, video integrity authentication and video copyright verification. With the capability of multiple authentications and the video authentication center, the ownership of the video will can be protected well.

Chapter 4

Video Authentication for Copyright Claim

4.1 Introduction:

Embedding a unique image (such as a watermark) and some authentication signals into videos is the core issue of applying digital watermark techniques to host videos. At the beginning, we embed a unique watermark and authentication signals into a video. Then we can authenticate the ownership and integrity of a video according to the watermarks and authentication signals embedded in it. But sometimes, malicious users will dispute the results of the video authentication. So, an authentication center for arbitration of the dispute is needed. Through a fair third party, we can get a judgment on the ownership and integrity of the debatable video.

4.2 Proposed mechanism for authentication center:

The proposed video authentication mechanism includes a central authentication center, abbreviated as CAC, and several local authentication centers, abbreviated as LAC. The CAC is at the top level of the proposed mechanism. And LAC's, such as movie studios, multimedia studios, and TV stations, are at the second level of the proposed mechanism. Here, the CAC plays the role of a third party. When people other than those in the CAC and LAC's doubt the ownership of a video, they can ask the CAC to make a fair judgment on the ownership of that video. Of course, in order to make the CAC function well, the LAC's must register their watermarked video to the CAC before the judgments. Figure 4.1 shows the two-level video authentication center.

Embedding information into the host video requires heavy system resources. In order to alleviate the load of the CAC, we move the embedding procedure form the CAC to the LAC's. After embedding information into the host video, the software will register the video to the CAC through the Internet. And the software will leave a copy of stego-video with the LAC. If the LAC finds a tort from the Internet, It can use the copy (which can prove the ownership of that video) to ask the malicious user to retract the video form the Internet. But sometimes, malicious users will dispute the results of the copy. Thus, we can ask the CAC to give a fair judgment on the ownership of that video.

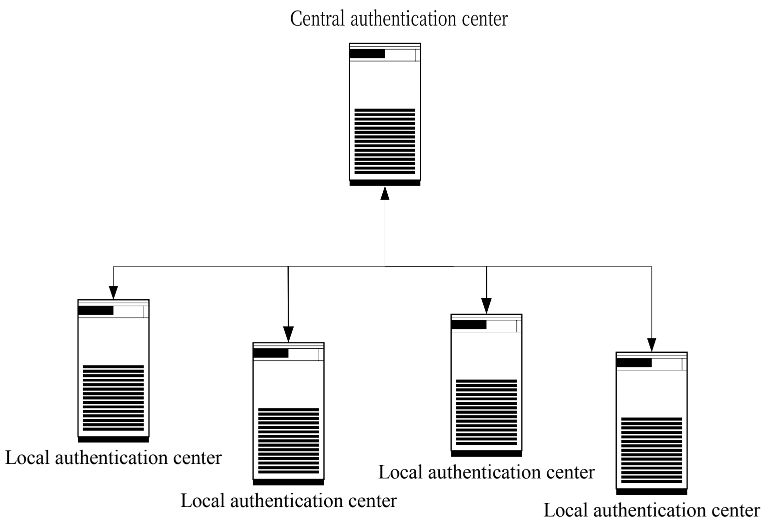

Central authentication center
Local authentication center
Local authentication center
Local authentication center
Local authentication center

Figure 4.1 Two-level video authentication center

Multiple authorization mechanisms are added to functions of Image Authentication Center (IACs). That is, the IAC will contain two other service functions, called authorization testimony and authorization tracking. They are described as follows:

1. Authorization testimony: When an owner distributes an authorized product to some authorized users, according to the aforementioned situation, the owner will embed an identifiable fingerprint each time in the product to create a stego-product. Then, the owner distributes each stego-product to each authorized user. The IAC can play a witness to prove the authorized procedure and ask the authorized user to sign a contract whose content is that he/she cannot release the authorized product. And the IAC can record the identifiable fingerprints and their associated authorized users.
2. Authorization tracking: If a contract violation is detected, the IAC can trace back to the violating authorized user. It can extract the fingerprint from the detected product and capture the traitor according to the records in the IAC. Besides, for the authorization of a software package, if users attempt to replace the original copyright with their copyright in a video, the IAC also can trace back to the original owner and the violating authorized user. When a copyright is embedded into a video by a user, the embedding process will extract the original identifiable code form the video and simultaneously embed it together with the user's identifiable code into the video. That is, the owner's identifiable code and the user's one will be in the video. The IAC can extract the owner's identifiable code from the stego-videos to judge who is the owner of the video and who is the cheater.

4.3 Discussion and Summary

The structure of a video authentication center may be divided into two parts from our viewpoint. One is the central video authentication center. The other is the local video authentication center. Besides, video authentication centers are suggested to fulfill several service functions, namely, video integrity authentication, video copyright verification, authorization testimony, and authorization tracking. With the capability of multiple authentications and the video authentication center, the ownership of the video will can be protected well.

附件一

科學博物館所提建議之回覆

問題	解決方式	軟體現況

將圖檔翻轉，就無法偵測浮水印	已解決	目前可處理轉 90 度，180 度等圖檔
縮小影像，就無法偵測浮水印	已解決	針對 JPEG 圖檔，在一定範圍內縮小後，可再放大為原圖大小，還是可抽取得出浮水印
更改圖檔副檔名，就無法偵測浮水印	已解決	請見下圖 A 之說明
JPEG 圖檔無法加入 logo	已解決	已加入資訊保護家 2.0 版之功能中
加入不可視浮水印，但列印時能印出藏入的浮水印圖像	研究中	此為可作為碩、博士論文之學術問題
希望程式可提供可視與不可視浮水印兩種選擇，並可選擇可視浮水印的位置	已解決	已加入資訊保護家 2.0 版之功能中
在 JPEG 圖檔，是否同樣可以讓使用者選擇藏入可視的浮水印位子與大小	解決中	將加入資訊保護家 3.0 版之功能中
藏入看不見浮水印，製成出版品後再重新掃瞄是否仍然可偵測到浮水印	研究中	此為可作為碩、博士論文之學術問題
TIFF 檔案格式的處理	已解決	已加入資訊保護家 2.0 版之功能中
當典藏單位授權影像或視訊給不同廠商時，當廠商有侵權的行為發生時，是否可以追蹤的出來	已解決	使用者可以依授權廠商的不同，藏入不同的註解，當侵權行為發生時，可對其做影像認證，判斷是哪一家廠商違反合約

圖 A：資訊保護家 2.0 版抵抗轉檔之能力

使用其他軟體轉檔				
	BMP	JPEG	GIF	TIFF
BMP	可	否（註解 1）	否（註解 2）	可
JPEG	可（註解 3）	可	可（註解 4）	可
GIF	可	否（註解 5）	可	可
TIFF	可	否（註解 6）	否（註解 7）	可

註解 1：由於 BMP 是用 LSB 的方法，因此不夠強韌，無法操作。

註解 2：同註解 1。

註解 3：轉成 BMP 檔之後，還是須用 JPEG 的格式，才能抽得出不可視浮水印可視浮水印、註解。

註解 4：作法與上面相同。

註解 5：由於轉成 JPEG 檔為具嚴重性的破壞壓縮。

註解 6：TIFF 目前發展 TIFF-BMP 格式檔案，故情形與註解 1 相同。

註解 7：同註解 6。

附件二

交大與台中自然科學博物館會議記錄

時間：九十二年十一月十四日(星期五) 上午十時

地點：台中自然科學博物館

參加人員：交大—吳大鈞教授、邱彥中、羅楠焜、高薇婷、洪世結、許琇萍

　　科博館—

議程

一、資訊保護家 2.0 版 Demo。（約 10 分鐘）

二、視訊之資訊隱藏技術之 Demo。（約 5 分鐘）

三、前次科博館數位典藏人員所提意見之說明。（約 30 分鐘）

四、意見交流與建議。（約 70 分鐘）

討論問題

1.一旦資訊保護家以後變為商業軟體之後，是否大家都可以使用軟體來破解浮水印？

回應：本計畫團隊發展出軟體多家授權的機制，也就是不同的單位所拿到不同的軟體彼此之間無法互相操作。

2.連註解、不可視浮水印都可有回復的功能嗎？

回應：回復的功能僅包括可視浮水印。

3.以 BMP 圖檔來說，若進行裁切的部分為可視浮水印，那麼剩下的圖還可偵測得出可視浮水印嗎？

回應：若已經可視浮水印裁切，當然不可能再偵測得出可視浮水印，至於不可視浮水印則應該還是可以偵測得到。

4.若僅裁切一張圖的上半部或者更小，那麼還是可偵測得到浮水印嗎？

回應：理論上應該可以。

5.目前科博館的作業方式為只要將 TIFF 檔案圖檔輸入，即可自動產生 BMP、JPEG、GIF 等圖檔，藏入浮水印的機制也希望可以在剛開始的作業平台上處理，如此一來，由 TIFF 轉檔後，所有的浮水印皆無法偵測？如何解決這樣的問題？

科博館建議（徐典裕）：是否告訴你們如何轉檔，會不會對本計畫團隊抵抗轉檔的能力有所幫助。

6.若 JPEG 轉 TIFF 之後，再轉回 JPEG 格式，是否還是可以偵測得到浮水印？

回應：可以，但要使用 JPEG 的方法來抽取浮水印。

7.我們直接使用數位相機拍攝下來的圖片，若處理 1MB 的 JPEG 格式圖檔（2560x1920）大約

需要八分鐘。（電腦配備為 P4、ram 512MB）
回應：本計畫團隊對於大圖檔的處理主要是針對 TIFF 格式的檔案，目前我們可處理到約 20MB，時間大約為一分鐘。至於 JPEG 格式部分，由於運算比較複雜，所以還需要再研究。

8.延伸上一問題，若開啟大圖檔，程式是否可自動將畫面縮小，讓使用者可看得到整張圖檔以符合畫面大小。
回應：應該可以做到。

9.是否可在程式中多加一項功能，顯示目前系統開的檔案有哪些？否則常會有未隱藏的檔案、隱藏過的檔案等，使用者會混淆？
回應：應該可以做到。

10.如果更改圖檔的明暗，是否還使可以偵測得出浮水印？
回應：還是可以。

11.若圖檔被縮小之後，原典藏單位必須回覆為原來大小才可偵測得出浮水印，但典藏單位的圖檔十分繁雜，且每張標本圖幾乎都長得差不多，要再去找出原始圖檔的大小，似乎太困難。
回應：可再研究。

12.延伸上一個問題，原始圖的檔名、大小等資訊，是否也可以藏入圖檔之中。
回應：可再研究。

13.藏完浮水印之後，背景的顏色似乎有一點改變？
回應：假如背景太單調的話，較容易看得出來顏色的改變。

14.若使用 Print Screen 的話，就無法偵測得到浮水印？
回應：假如螢幕上還是以 1:1 呈現的話，還是可以偵測得到浮水印。但前題是要裁切得剛剛好。

附件三

参加「第二屆數位典藏研討會」發表之論文

Hiding Authenticable General Digital Information behind Binary Images with Reduced Distortion✧

Chih-Hsuan Tzeng and Wen-Hsiang Tsai

Department of Computer and Information Science, National Chiao Tung University
Hsinchu, Taiwan 300, Republic of China, Tel: +886-3-5728368 Fax: +886-3-5734935
E-mails: {chtzeng, whtsai}@cis.nctu.edu.tw

Abstract

A new approach to information hiding in binary images with the capabilities of hidden data authentication and image distortion reduction is proposed. The hidden information may be of any data form. Based on a new feature called surrounding edge count for measuring the structural randomness in an image block, pixel embeddability is defined from the viewpoint of minimizing image distortion. Accordingly, embeddable image pixels suitable for hiding secret data are selected. Furthermore, an error-correcting scheme is used both for hidden data authentication and for image distortion reduction. Finally, to increase the security of embedded data, a secret key and a random number generator are employed to randomize the locations of selected pixels into which secret data are embedded. Experimental results show the feasibility of the approach for real applications.

Key words: secret hiding, secret recovery, secret authentication, binary images, error-correcting schemes, distortion reduction, surrounding edge count, pixel embeddability.

EDICS: 2. Application areas; 2. IMMD Image and multidimensional signal processing; 2. COMM Signal processing for communications.

✧ This work is supported partially by the MOE Program for Promoting Academic Excellency of Universities under Grant No. 89-1-FA04-1-4.

I. Introduction

Information hiding behind digital images has many applications, including covert communication, copyright protection, annotation association, etc. However, it is generally difficult to hide information behind binary images. There are at least three reasons for this problem. First, embedding data in a binary image will cause obvious image content changes because of the binary (black and white) nature of the image. This indicates that reduction of image distortion due to data embedding (called *embedding distortion* in the sequel) should be taken as a major consideration in designing algorithms for data hiding in binary images. Next, binary images are more fragile to disturbances or attacks like channel noise or image operations. Such a characteristic makes authentication of recovered hidden information a required work. Finally, with the widespread use of color images, binary images are used today mainly for conveying text or graphic based document images in which color information is not important, and so the semantics of binary image contents are very vulnerable to pixel value changes due to data hiding. This means that a more careful selection of image pixels for data hiding is required; pixel value changes leading to obvious destruction of image contents should be avoided. In this paper, we propose an information hiding method which takes all of the above three requirements into consideration. Moreover, digital information that can be hidden by the method is general in type, and is assumed to be *bit streams* in the sequel.

There were only a few studies in the past about information hiding behind binary images, possibly due to the difficulty mentioned above. Wu, et al. [1] embedded bits in image blocks selected by pattern matching. The method can be used both for data hiding and for image authentication. Tseng, et al. [2] changed pixel values in image blocks and mapped block contents into the data to be hidden. In [3, 4], word or line spaces in textural document images are utilized to embed watermarks for copyright protection. In [5, 6], secret information is embedded into dithered images by manipulating dithering patterns. And Koch and Zhao [7] embedded a bit 0 or 1 in a block by enforcing the ratio of the number of black pixels in the block to that of white ones to be larger or smaller than the value 1, respectively. The method proposed in this study may be used for data hiding as well as data authentication. We will compare our method with [1] and [2] in more details later in this paper.

More specifically, in the proposed method we define a measure of pixel embeddability by which we can select suitable pixels from a given binary image, called the *cover image*, for embedding given secret data.

The measure is defined in such a way that not only embedding distortion in the resulting image, called *stego-image*, can be reduced, but also the pixels selected for data embedding can be identified correctly subsequently for secret recovery. In addition, we employ an error-correcting scheme to encode the secret data before they are embedded, for the purposes of hidden data authentication as well as further embedding distortion reduction. At last, we propose the use of a secret key as well as a random number generator to randomize the locations of the selected pixels for data embedding. This enhances the security of the hidden data from being attacked or accessed illicitly. Based on these measures of distortion reduction and safety protection, processes for secret hiding and recovering are proposed. Some experimental results are also included to show the effectiveness the proposed method.

In the remainder of this paper, we first describe the proposed secret hiding and recovering processes in Section II, followed by the descriptions of the involved measures for distortion reduction and security protection in Section III. Some experimental results are given in Section IV, followed by a conclusion in Section V.

II. Proposed Secret Embedding and Recovering Processes

In the proposed method, we hide a given secret bit stream behind a binary image in a random fashion controlled by a secret key and a random number generator. The proposed secret hiding process is described first. Only basic ideas are included; the details of the involved terms and techniques will be explained in the next section. In the sequel, by *embedding a value v into a pixel p*, we mean to replace the value of p with v; and by *extracting a value v from p*, we mean to take v to be the value of p.

Algorithm 1. ***Secret hiding process***.

Input: a secret bit stream S, a cover image I, a secret key K, a random number generator g, and three pre-selected positive integer numbers m, n, and t.

Output: a stego-image I' in which S is embedded.

Steps:

1. Take sequentially m bits of S and encode them, using a t-error-correcting scheme, to form an n-bit substream s.
2. Create a set C of n-bit streams from s by changing at most t bits in s in all possible ways.
3. Select an *ordered sequence E* of *n embeddable pixels* in I randomly using g with K as the seed.

4. Select from C a substream s_{opt}, which causes minimum distortion, after being embedded into the pixels of E.
5. Embed the bits of s_{opt} sequentially into E.
6. Repeat Steps 1 through 5 until all bits in S are processed.

For convenience, in the sequel each pixel selected to be included in E in Step 3 above is said to *have been visited*. The proposed secret recovering process (including secret bit stream extraction and authentication) is described as follows.

Algorithm 2. ***Secret recovering process***.

Input: a stego-image I' presumably including a secret bit stream S; and the secret key K, the random number generator g, as well as the positive integer numbers m, n, and t all used in Algorithm 1.

Output: the secret bit stream S or a report of failure to recover the secret.

Steps:

1. Select an ordered sequence E of n embeddable pixels in I' using g with K as the seed.
2. Extract a bit b' from each pixel p in E, and compose all the n extracted bits sequentially to form a bit stream s'.
3. Decode s' by the t-error-correcting scheme used in Algorithm 1 to recover an m-bit secret stream s''. If more than t errors are found in s' during the decoding process, decide the bits of s'' to be *unauthentic*, yield a report of failure to recover the secret, and exit; otherwise, take s'' as part of the desired secret bit stream S.
4. Repeat the above steps to extract other m-bit substreams sequentially to compose the remaining part of S until done.

The ordered sequence E of pixels selected in Step 1 above presumably should be identical to that yielded in Step 3 of Algorithm 1 to ensure that the secret bit stream can be extracted correctly. For this to be true, in addition to requiring the use of the same random number generator g and the same secret key K in the two processes as already done, an extra condition is that the embeddability of the selected pixels must be *preserved* after the secret hiding process, and not be changed before the secret recovering process. We satisfy this condition by proposing a proper definition of pixel embeddability, as described in the next section.

III. Proposed Pixel Embeddability And Distortion Reduction Measures

A. Pixel Embeddability based on surrounding edge count

We define pixel embeddability from the viewpoint of reducing embedding distortion. First, we propose a new type of feature, called *surrounding edge count* and abbreviated as SEC. Let B be a 3×3 block in a cover image I with pixel p being its center and p_1, p_2, …, and p_8 being the eight surrounding neighbors of p in B. The SEC of p, denoted as SEC_p, is defined as the number of *existing edges* between p and its eight neighbors in B. Since I is binary, the existence of an edge between p with value v and one of its neighbors, say p_i with value v_i, means that $|v_i - v| = 1$, and the reverse situation means that $|v_i - v| = 0$. This in turn means that SEC_p may be computed by

$$SEC_p = \sum_{i=1}^{8} |v_i - v|.$$

The SEC value of p is a measure of the structural randomness in the block from the centralized viewpoint of p. By definition, the SEC value of a fully black or white block is 0 (no edge exists), that of a block filled with a checker pattern is 4 (four edges exist), and that of a white (or black) pixel surrounded by eight black (or white) neighbors is 8 (eight edges exist).

Next, we define a *measure of distortion* resulting from complementing the value v of p by:

$$\Delta SEC_p = |SEC_p - SEC_p'|$$

where SEC_p and SEC_p' denote the SEC values before and after the complementation operation, respectively. It is not difficult to figure out that the above measure of distortion is just the amount of the resulting change of the numbers of edges in B. As an example, if the central pixel p of a fully black block is changed to be a white one, the above distortion value ΔSEC_p will have the largest possible value 8, which cannot be endured, because then the new white central pixel is too contrastive to its eight black neighbors.

Finally, we define a pixel p in a block B to be *embeddable* (i.e., suitable for embedding a bit value) if the following two conditions are satisfied:

(a) $\Delta SEC_p < T_d$; and

(b) p and its eight neighbors in B have not been visited yet,

where T_d is a pre-selected threshold value. Condition (a) above restricts the distortion introduced by the complementation of p's value to be *sufficiently small*, so that the resulting image quality will not be affected too much. And Condition (b) requires that embeddable pixels be *disconnected* from one another (by at least one pixel in distance), so that pixel value changes due to secret embedding will not be clustered or propagated to cause obvious larger-sized visual artifacts.

It is pointed out that pixel embeddability defined as above can be *preserved* indeed after the secret hiding process. The existence of this embeddability preserving property is briefly explained as follows. First, the pixel p and its eight neighbors are required by Condition (b) to be unvisited yet, and this means that the neighbors' values will not be altered after p is visited and labeled to be embeddable. This in turn means that the value ΔSEC_p will be fixed, and so Condition (a) will hold after the secret hiding process. As a result, after a secret bit is embedded into an embeddable pixel p in the secret hiding process, p will still be embeddable in the secret recovering process, guaranteeing that the embedded secret bit stream can be extracted correctly.

B. Authentication of extracted secret bit streams

In Step 3 of Algorithm 2, we recover secret substreams and authenticate them simultaneously using a t-error-correcting scheme described in [8]. The authentication capability of the scheme is explained here. In the encoding stage, the scheme appends several extra bits, forming a *redundant checking part*, to a bit sequence, called the *message part*. Each extra bit in the redundant checking part is a parity bit computed from a certain number of bits at certain specific positions in the message part. Then, in the decoding stage, if some bits in the two parts are changed, after the parity bits are computed, their values will be found to mismatch those in the redundant checking part, and errors can thus be detected. In this way, authentication of the message part, which, in our case here, is the secret data stream, can be achieved.

C. Further reduction of embedding distortion

By using the error-correcting scheme, errors in the extracted secret data not only can be detected, but also can be corrected. In this study, we adopt the BCH method [8] to achieve such an error-correction function in the scheme. And this error-correcting capability is utilized in Step 4 in Algorithm 1 to select a so-called optimal substream s_{opt} for further reduction of embedding distortion from a more global view. The details are described in the following.

After embedding an n-bit secret substream $s = b_1b_2\ldots b_n$ into n pixels p_1, p_2, …, p_n in a selected embeddable pixel sequence E, we compute a measure of the *total embedding distortion*, denoted by $D(s)$, as

$$D(s) = \sum_{i=1}^{n} \Delta SEC_{p_i} .$$

Also, as described in Step 2 in Algorithm 1, we create from s a set C of n-bit streams by changing at most t bits in s in all possible ways. For each stream s_i in C, we compute similarly another total embedding distortion value $D(s_i)$. Then, we choose as s_{opt} the stream s_i in C with the smallest $D(s_i)$, and embed s_{opt} into E as done in Step 3 of Algorithm 1. Thereafter, even though somehow erroneous bits occur in s_{opt} before secret recovering is conducted, if the number of errors is not larger than t, then by the t-error-correcting scheme, s can be still correctly recovered from s_{opt}, as done in Step 3 of Algorithm 2. Because of the freedom of the choice of s_{opt} from totally $\sum_{r=0}^{t}\binom{n}{r}$ candidate streams in C, it may be expected that the embedding distortion can be reduced further.

IV. Experimental Results

A large number of binary images, including several typical ones used for testing binary image compression standards, were used in our experiments. An example of the experimental results is shown in Figure 1. Figure 1(a) shows a 256×256 cover image with machine-printed as well as handwritten characters, and line drawings. And Figure 1(b) shows the stego-image resulting from embedding 630 secret bits into Figure 1(a) using Algorithm 1 with m, n, t, and T_{d} being 4, 15, 5, 3, respectively. The difference between Figure 1(a) and Figure 1(b) is illustrated in Figure 1(c) as black pixels. The gray pixels are included just for the purpose of comparison and are not part of the difference. It can be seen that Figure 1(a) and Figure 1(b) are visually close to each other; no obvious distortion can be observed. Another example of the results is shown in Figure 2, which demonstrates the effectiveness of the proposed technique for reducing embedding distortion. Figure 2(a) shows a 64×192 cover image. Two stego-images resulting from embedding a 60-bit secret stream into Figure 2(a) without and with the use of the t-error-correcting scheme are shown in Figure 2(b) and Figure 2(c), respectively. It can be noted that the visual quality of Figure 2(c) is better than that of Figure 2(b). Figure 3 shows the effect of distortion reduction using the error-correcting scheme with different values of t. The upper curve in the figure specifies the average ΔSEC yielded in Algorithm 1, and the lower curve specifies the *average number of changed pixels*, computed as the ratio of the number of changed pixels to that of the embedded secret bits. Both curves show that the distortion is decreased with the increase of the error-correcting capability specified by t.

Furthermore, we have implemented the methods proposed in [1] and [2] by programs in C++ to compare them with ours in the aspects of embedding capacity, robustness, and image quality. The comparison result is shown in Table 1. From the table, we can see that although the method of [2] has the maximum data embedding capability in the best case, it does not consider reduction of embedding distortions and might generate isolated spots in the stego-images. On the other hand, the method of [1] does consider distortion reduction just like ours, and if we compare image quality reduction by counting the total number of changed pixels in the embedding process, then our method is as good as [1]. However, our method can embed data more efficiently than [1] for most images by finding embeddable pixels in the image instead of embedding just a bit in every block. In short, our method maintains a good tradeoff between data embedding capacity and stego-image quality.

As to robustness, it was neither considered in [2] nor in our method because both methods were designed for steganography. Though, we still conducted some experiments to investigate the robustness of our method by introducing some random noise on the resulting stego-images, simulating possible attacks on the images. It is found that when the embeddability at certain pixels, which include an error-correcting code, is not destroyed by the noise, the proposed method will correct errors (no fewer than *t* ones) found in these pixels, thus achieving a certain degree of robustness. On the other hand, though [1] has mentioned how to achieve robustness against noise, no experimental data were shown. At last, it is emphasized that our method has the new capability of hidden data authentication, which is not found in any other method dealing with binary images, including [1] and [2].

V. Conclusion

A new approach to data hiding in binary images has been proposed, which may be employed to hide general secret data behind binary images with the additional capability of hidden data authentication. Reduction of embedding distortion is the major consideration in the approach. The first measure for this goal is the proposal of pixel embeddability based on the new feature of SEC, which makes data hidden in embeddable pixels less noticeable. Computation of SEC values does not require excessive works like pattern matching, and so is efficient. Another measure proposed for distortion reduction from a more global view is the use of the error-correcting scheme for creating a distortion-minimizing secret stream from the original one. This merit is not found in other approaches dealing with data hiding in binary images. Our experimental results also reveal its effectiveness. In addition, the use of the error-correcting scheme also provides the ability to verify the authenticity of hidden data. Finally, because of the randomization policy employed for selecting embeddable pixels as well as the nature of the proposed pixel embeddability, embedded values are spread in the entire image and disconnected from one another, so that pixel changes will not be clustered and thus less hints for the embedded secret will be revealed.

References

[1] M. Wu, E. Tang, and B. Liu, "Data hiding in digital binary images," presented at the *IEEE International Conference on Multimedia and Expositions*, New York, 2000.

[2] Y.-C. Tseng, Y.-Y. Chen, and H.-K. Pan, "A secure data hiding scheme for binary images," *IEEE Transactions on Communications*, vol. 50, no. 8, pp. 1227-1231, August 2002.

[3] S. H. Low, N. F. Maxemchuk, and A. M. Lapone, "Document identification for copyright protection using centroid detection," *IEEE Transactions on Communications*, vol. 46, no. 3, pp. 372-383, March 1998.

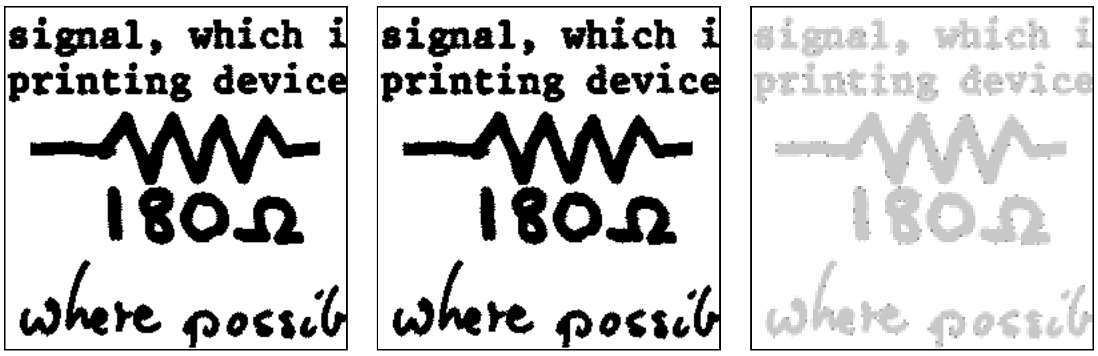
[4] D. Huang and H. Yan, "Interword distance changes represented by sine waves for watermarking text images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 12, pp. 1237-1245, December 2001.

[5] K. Matsui and K. Tanaka, "Video-steganography: how to secretly embed a signature in a picture," *Proceedings of IMA Intellectual Property Project*, vol. 1, no. 1, 1994.

[6] H. C. Wang, "Data hiding techniques for printed binary images," *Proceedings of the IEEE International Conference on Information Technology: Coding and Computing*, Las Vegas, NV, USA, April 2001, pp. 55-59.

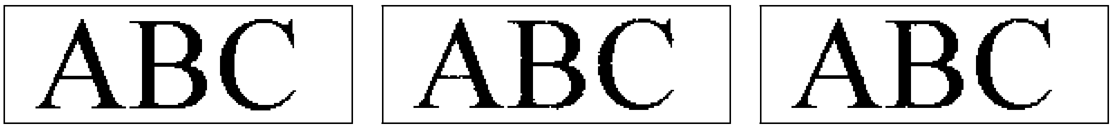
[7] E. Koch and J. Zhao, "Embedding robust labels into images for copyright protection," *Proceedings of International Congress on Intellectual Property Rights for Specialized Information, Knowledge and New Techniques*, Munich, Germany, 1995, pp. 242-251.

[8] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1983.



(a) (b) (c)

Figure 1. An experimental result of the proposed method: (a) a cover image; (b) the stego-image resulting from embedding 630 secret bits into (a); (c) the difference between the cover image (a) and the stego-image (b) shown as black pixels.



(a) (b) (c)

Figure 2. Embedding results yielded with and without proposed embedding distortion reduction: (a) the cover image; (b) the stego-image yielded without distortion reduction; (c) the stego-image yielded with distortion reduction.

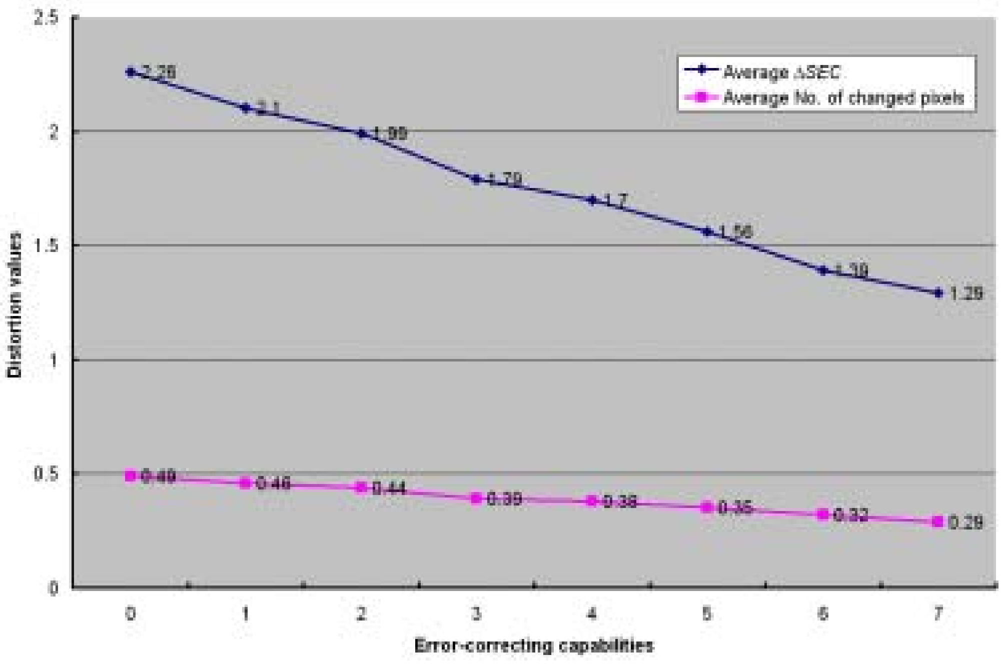


Figure 3. Curves showing the decrease of embedding distortion with the increase of the error-correcting capability specified by the number *t* of corrected bits.

Table 1. Comparison of characteristics of proposed method with those in [1] and [2].

	Wu and Liu's method [1]	Tseng, Chen, and Pan's method [2]	Proposed method
Processing manner	Block-based	Block-based	Pixel-based
Maximum embedding capacity of an $M\times N$ image	$(M/m) \times (N/n)$	$(M/m) \times (N/n) \times \lfloor \log_2(mn+1) \rfloor$	$\lceil (M-2)/2 \rceil \times \lceil (N-2)/2 \rceil$
Reduction of embedding distortions	Yes	No	Yes
Generation of isolated spots in stego-images	No	Yes	No
Robustness to image manipulations	unknown	No	No
Authentication of hidden data	No	No	Yes

附件四

交通大學影像認證中心

註冊證明書

本影像認證中心於民國___年___月___日，茲收到_____________單位註冊之圖檔共______幾張，以及添加浮水印之軟體名稱_____________。

經過註冊程序後，本中心於民國____年____月____日寄還貴單位圖檔，以及註冊後圖檔之編號，請貴單位務必留存圖檔編號，往後若遇到圖檔版權糾紛情事，貴單位將可攜帶此證明書與註冊圖檔編號來本中心予以認證，證明圖檔版權。

本聲明書一式兩份，由本影像認證中心及註冊單位各留存一份為憑。

單位/計畫名稱			
主持人姓名			
聯絡電話			
傳真			
地址			
電子郵件信箱			
添加浮水印之軟體名稱			
BMP 圖檔格式		張數	
TIFF 圖檔格式		張數	
JPEG 圖檔格式		張數	
GIF 圖檔格式		張數	