國立交通大學

資訊科學與工程研究所

碩 士 論 文

利用資訊隱藏技術對 H.264 影片做真偽驗證及內容保護之研究

A Study on Authentication and Protection of H.264 Video Contents by Information Hiding Techniques

研 究 生：洪菽鴻

指導教授：蔡文祥　教授

中 華 民 國　九 十 八 年 六 月

利用資訊隱藏技術對 H.264 影片做真偽驗證及內容保護之研究

A Study on Authentication and Protection of H.264 Video Contents by Information Hiding Techniques

研 究 生：洪菽鴻　　Student：Shu-Hung Hung

指導教授：蔡文祥　　Advisor：Wen-Hsiang Tsai

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

利用資訊隱藏技術對 H.264 影片做真偽驗證及內容保護之研究

研究生：洪菽鴻　　指導教授：蔡文祥 博士

國立交通大學資訊科學與工程研究所

摘要

隨著視訊壓縮及視訊編碼技術的進步，數位影片逐漸在我們的生活中扮演重要角色。本論文針對 H.264 影片，利用資訊隱藏技術，做影片快速檢索、驗證及隱私權保護之研究與應用。在影片快速檢索部份，為了在視訊監控影片中快速搜尋特定的可疑人物，我們提出了一個利用 H.264 編碼特性對影片做快速檢索的方法。在驗證方面，因為視訊監控影片經常成為不法使用者竄改掩蓋犯罪事實的對象，所以我們利用資訊隱藏技術及 H.264 特性提出了一個對影片做完整性及真實性的驗證的方法。在隱私權保護部份，因為視訊監控影片近來在個人隱私權方面經常引起爭議，所以我們提出了一個可將侵犯隱私權的影片部份內容消除及復原的方法。最後我們提出了一些相關的實驗結果，證明了所提方法的可行性。

A Study on Authentication and Protection of H.264 Video Contents by Information Hiding Techniques

Student: Shu-Hung Hung Advisor: Wen-Hsiang Tsai

Institute of Computer Science and Engineering

National Chiao Tung University

ABSTRACT

With the progress of video compression technology and efficient video coding standards, digital videos nowadays have become much more popular than in the past, especially H.264 encoded videos. In this study, we propose methods for video-content search, video authentication, and privacy protection using H.264 videos as cover media. For video-content search, in order to find suspicious people or objects quickly in surveillance videos, a method for quick video-content search by novel uses of H.264 coding features is proposed. Because surveillance videos usually contain suspicious or unlawful acts, malicious users may tamper with videos for misrepresentation. Therefore, we propose a method for authentication of surveillance videos to protect and authenticate video contents. Privacy protection is also an important issue in video surveillance. Since surveillance videos may possibly record some personal information which violates personal privacy, we propose a method for removing and recovering privacy information in H.264 surveillance videos. Good experimental results show the feasibility of the proposed methods.

ACKNOWLEDGEMENTS

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

Thanks are due to Mr. Chih-Jen Wu, Mr. Che-Wei Lee, Mr. Guo-Feng Yang, Miss Mei-Fen Chen, Mr. Yi-Chen Lai, Miss Chiao-Chun Huang, Miss Chin-Ting Yang, Mr. Jian-Yuan Wang, Mr. Chun-Pei Chang, and Miss Yi-Jen Huang for their valuable discussions, suggestions, and encouragement. Appreciation is also given to the colleagues of the Computer Vision Laboratory in the Institute of Computer Science and Engineering at National Chiao Tung University for their suggestions and help during my thesis study.

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

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1
Introduction

1.1 Motivation

With the progress of video compression technology and efficient video coding standards, digital videos nowadays have become much more popular than in the past. H.264 is one of the video coding standards, which contains many innovative features, making the resulting coding performance more efficient than previous standards (MPEG-1, MPEG-2, MPEG-4, H.261, H.263, etc.). Because of the efficiency and good quality yielded by H.264, it has been widely used in many different applications, in which video surveillance is especially an important topic because video fidelity and privacy is often seriously concerned with. Therefore, it is necessary to develop effective methods for authenticating and protecting H.264 surveillance videos.

The crime rate rises along with society development, and the public space needs to be monitored, so the design of environment surveillance systems becomes more and more important. Since a video surveillance system can monitor an environment space for a long period, most frames in the resulting surveillance video are still with the same background. How to find suspicious people or objects quickly in such videos has become a major problem. It is desired to design quick video-content search techniques based on the use of content features, as conducted in this study.

The need of video authentication is especially essential in video surveillance applications. Because surveillance videos usually contain suspicious or unlawful acts,

malicious users may want to acquire the video in an illegal way and tamper with it for misrepresentation. How to protect and authenticate surveillance videos has also become a main topic in the research field of video surveillance. There are two important concerns in the study of video authentication, namely, fidelity and integrity. Embedding invisible authentication signals in a video, which results in a *protected video,* is a good approach to resolving the two concerns mentioned above. If a malicious user tampers with the protected video, the authentication signals hidden in it will be destroyed. By checking the presence of the authentication signals, we can verify the fidelity of the protected video. Furthermore, we may want to understand how and where the protected video has been tampered with; in other words, we may want to validate the integrity of the protected video. From the above reasons, it is necessary to design a good authentication method which not only checks if the video has been tampered with or not, but also shows where and how the tampering occurs. This is also a goal of this study.

Privacy protection is a very important issue in video surveillance. Since a video surveillance system usually monitors a public space for long periods of time, it may possibly record some personal information which violates personal privacy. Hence, it is necessary in some cases to hide the privacy violation parts of the surveillance video content to avoid legal disputes and to protect the personal privacy of non-suspicious people. This is the last of our goals in this study.

1.2 General Review of Related Works

Since digital rights management is more and more important in our daily lives, many works related to video-content search, video authentication, and privacy protection for surveillance videos have been introduced. For video-content search,

many motion detection algorithms such as background subtraction, temporal differencing, etc. used to index videos have been proposed. For video authentication, techniques like watermarking, digital signatures, etc. are widely used for authentication. For privacy protection, many different ways to protect personal information, such as scrambling the area containing sensitive information, removing the authorized person, etc., have been proposed. A detailed review of these techniques mentioned above, which have been developed in recent years, will be presented in Chapter 2. In addition, because the proposed techniques in this thesis are applied to H.264 videos, we will also make a review of the H.264 standard in Chapter 2.

1.3 Overview of Proposed Methods

1.3.1 Terminologies

The definitions of some related terminologies used in this study are described as follows.

1. *Motion region*: a motion region is a detected area which contains motions in an input video after a motion detection process.
2. *Tree structured macroblock decomposition information*: tree structured macroblock decomposition information is the macroblock decomposition information generated in the tree structured motion compensation process of the encoding process of H.264.
3. *Protected video*: a protected video is a video in which authentication signals have been embedded.
4. *Video authentication*: video authentication is a process for verifying the integrity and fidelity of a suspicious surveillance video.
5. *Privacy-covered area*: a privacy-covered area is a part of video content

which might violate privacy after removing personal information.

6. *Recovered Privacy-covered area*: a *recovered privacy-covered area* is a part of video content which might violate privacy obtained after restoring personal information.

1.3.2 Brief Descriptions of Proposed Methods

1.3.2.1 Proposed Method for Video-Content Search in Surveillance Video

A method of motion detection based on the use of content features is proposed for searching surveillance video contents in this study. By using the tree structured macroblock decomposition information in a motion detection process, regions with moving objects can be detected correctly. After the motion detection process, the content features of the detected motion regions are analyzed and embedded into the video. If a user wants to search the video content, the features embedded in the video previously are extracted to index the video content. In this way, the features provide a fast way to search video contents in a surveillance video.

1.3.2.2 Proposed Method for Authentication for Surveillance Video

A method using the tree structured macroblock decomposition information in H.264 codes as authentication signals is proposed for video authentication in this study. The macroblock decomposition information is generated during the encoding process and is different for different video contents, so it is suitable for use as authentication signals. If a protected video has been tampered with, the authentication signals will be destroyed as well. Therefore, how and where the protected video is

tampered with can be inspected to carry out the authentication work.

1.3.2.3 Proposed Method for Protection of Personal Privacy of Surveillance Videos

A method using an information hiding technique is proposed in this study for hiding and recovering video contents containing sensitive personal information. A video can be decoded correctly based on the decoding information including motion vectors and frequency coefficients. Therefore, the original decoding information may be removed from the original video stream and set to some predefined values in order to cover video contents with sensitive privacy information and replace them with background image parts. The removed decoding information is not eliminated but embedded into the video and can be extracted later from the stego-video in case there is a need of retrieving the sensitive contents.

1.4 Contributions

Some contributions made by this thesis are listed in the following.

1. A method of motion detection in videos based on motion information and tree structured macroblock decomposition information in H.264 codes is proposed in this study.
2. A method of data hiding using random encoding mode selection in H.264 videos is proposed.
3. An application of video-content search by scene features is proposed.
4. A video authentication system using tree structured macroblock decomposition information in H.264 codes as authentication signals is proposed.

5. A method is proposed to hide the privacy information of unsuspicious people and restore it in need of retrieving the original video contents.

1.5 Thesis Organization

In the remainder of this thesis, related works about motion detection, video data hiding, video authentication, privacy protection in surveillance videos, and the H.264 standard are reviewed in Chapter 2. In Chapter 3, the proposed method of motion detection and the application of video-content search are described. In Chapter 4, the proposed video authentication system for surveillance videos is described. In Chapter 5, the proposed method of privacy protection of surveillance videos is presented. Finally, conclusions and some suggestions for future works are given in Chapter 6.

Chapter 2
Review of Related Works and H.264 Standard

2.1 Review of Techniques for Motion Detection

A lot of motion detection techniques have been proposed to detect moving objects in a video [1-5]. The techniques can be classified into two categories. One is for use in the pixel domain [1-2]; the other in the compressed domain [3-4]. Generally speaking, the approaches used in the pixel domain need to fully decode a compressed video bitstream first, but they can be employed in videos coded in different video coding standards. On the other hand, each of the approaches used in the compressed domain can perform a motion detection process by partially decoding a compressed video bitstream, but they can only be employed in videos coded in specific standards.

Haritaoglu et al. [1] proposed a motion detection method based on background subtraction in the pixel domain. They built a statistical model for a background scene that allows them to detect moving objects even when the background scene is not completely stationary. Lipton et al. [2] proposed another approach based on temporal differencing in the pixel domain, which means computation of pixel-wise differences between consecutive video frames. The basic idea of the approach is to compare video frames separated by a constant time to find moving objects. Zeng et al. [3] proposed

another approach in the compressed domain. They employed a block-based Markov random field (MRF) model in a field formed with motion vectors to segment moving objects during a decoding process. The methods mentioned above detect motions by common properties of videos, such as pixel values, motion vectors, etc, but they don't use special features of a specific standard as the main clues.

2.2 Review of Techniques for Video Data Hiding

Video data hiding can be used in many applications about videos, such as video watermarking, video authentication, etc. As a result, lots of techniques for video data hiding have been introduced [6-10]. Mobasseri et al. [10] embedded data into the CAVLC code space of an H.264 bitstream, which is one of the existing entropy coding techniques of H.264. This method was directly applied to a bitstream without video decoding or partial decompression. Huang and Tsai [7] proposed a video data hiding method based on the use of prediction modes and tree structured macroblock motion compensation of the H.264 structures. In addition, they also used the Lagrange optimization technique to minimize image distortion yielded by the data hiding process. In Noorkami and Mersereau [6], a robust data hiding algorithm for H.264 was proposed. The basic idea is to embed data by modifying the DC coefficients in luminance residual blocks of the video and employ a human visual model adapted for a 4×4 discrete cosine transform block to increase the payload and robustness while limiting visual distortion. Gong and Lu [8] also proposed a data hiding method in the frequency domain. They employed a texture-masking-based perceptual model to adaptively choose the hiding strength of each block. In an H.264 encoding process, if someone re-encodes a video, the prediction modes in the original video may be

different from the resulting video. As a consequence, if re-encoding is unavoidable, videos using the robust data hiding methods based on the frequency domain will face a critical problem that the frequency coefficients may be different from the original ones with hidden data being modified due to the changes of an intra prediction mode. This problem may cause a loss of hidden information, so we introduce a robust data hiding method which can endure an H.264 re-encoding process.

2.3 Review of Techniques for Video Authentication

Video authentication plays an important role in a digital rights management system, so many different methods have been proposed to solve the problem [10-13]. Zhang and Ho [11] introduced a video authentication method which makes an accurate usage of the tree structured motion compensation, motion estimation, and Lagrange optimization of the H.264 standard. As mentioned in the paper, authentication information is embedded based on the best mode decision strategy in the sense that if a video undergoes any spatial and temporal attacks, the scheme can detect the tampering by the sensitive mode change. Pröfrock et al. [12] proposed a method using skipped macroblocks of a H.264 video to embed authentication data. The data are embedded as a fragile, blind and erasable watermark with low video quality degradations. In contrast with other authentication methods, the embedding process is done *after* an H.264 compression process, while others are done *during* the process. The methods mentioned above usually use additional authentication information to authenticate videos. How to authenticate videos without external information is an interesting research topic and is investigated in this study.

2.4 Review of Techniques for Privacy Protection in Videos

Privacy protection has become more and more important along with the rise of video surveillance systems. Many different approaches have been introduced in recent years [14-16]. Meuel et al. [14] introduced a method to protect faces in surveillance videos. As mentioned in the method, any visible information of faces in a video is deleted and embedded in the video that allows further reconstruction of the faces if needed. In Dufaux et al. [15], the regions containing personal information are scrambled. As a consequence, the scene remains visible, but the privacy-sensitive information is not identifiable. Zhang et al. [16] proposed another method to protect authorized persons, which are not only removed from a surveillance video, but also embedded into the video. The above-mentioned methods are based on a concept which is to protect privacy of authorized persons, so the protected persons must be recognized first by manpower. If there are many persons requiring recognition, it will become a tedious job. We call this kind of methods as *object-based* privacy protection. In order to solve this problem, we introduce a *region-based* privacy protection method to avoid recognizing authorized persons by hand. Besides, an authorized user can define the protected region easily.

2.5 Review of H.264 Standard

2.5.1 Structure of H.264 standard

The H.264 standard defines three *profiles*: *Baseline*, *Main*, and *Extended*, which provide different sets of coding functions and different components required by an

encoder or decoder. Because of the individual features of each profile, there are many different potential applications of these three profiles. The applications of the baseline profile include video telephony, video conferencing, and wireless communications; the applications of the main profile include television broadcasting and video storage; and the extended profile may be particularly useful for streaming media applications. The relation between these profiles is illustrated in Figure 2.1. H.264 videos have a hierarchical structure. A video sequence consists of consecutive video images (frames). A video image is composed of at least one slice. There are five slice types: intra slice (I), predicted slice (P), bi-predicated slice (B), switching P slice (SP), and switching I slice (SI). Generally speaking, the first three slice types are the main slice types which are widely used in H.264 videos. A slice is composed of macroblocks, which can be categorized into four types including I, P, B, and skipped. I macroblocks are predicted by previously encoded and reconstructed blocks in the same slice. P macroblocks are predicted by previously encoded samples before the current frame in temporal order. B macroblocks are predicted by encoded samples before or after the current frame. Skipped macroblocks of the P slice is transmitted with motion vectors but not with frequency coefficients. Skipped macroblocks of the B slice is transmitted without both motion vectors and frequency coefficients. How the macroblocks comprise the slices is illustrated in Table 2.1.

Table 2.1 The way that macroblocks comprise slices

	I macroblocks	P macroblocks	B macroblocks	Skipped macroblocks
I slice	√			
P slice	√	√		√
B slice	√		√	√

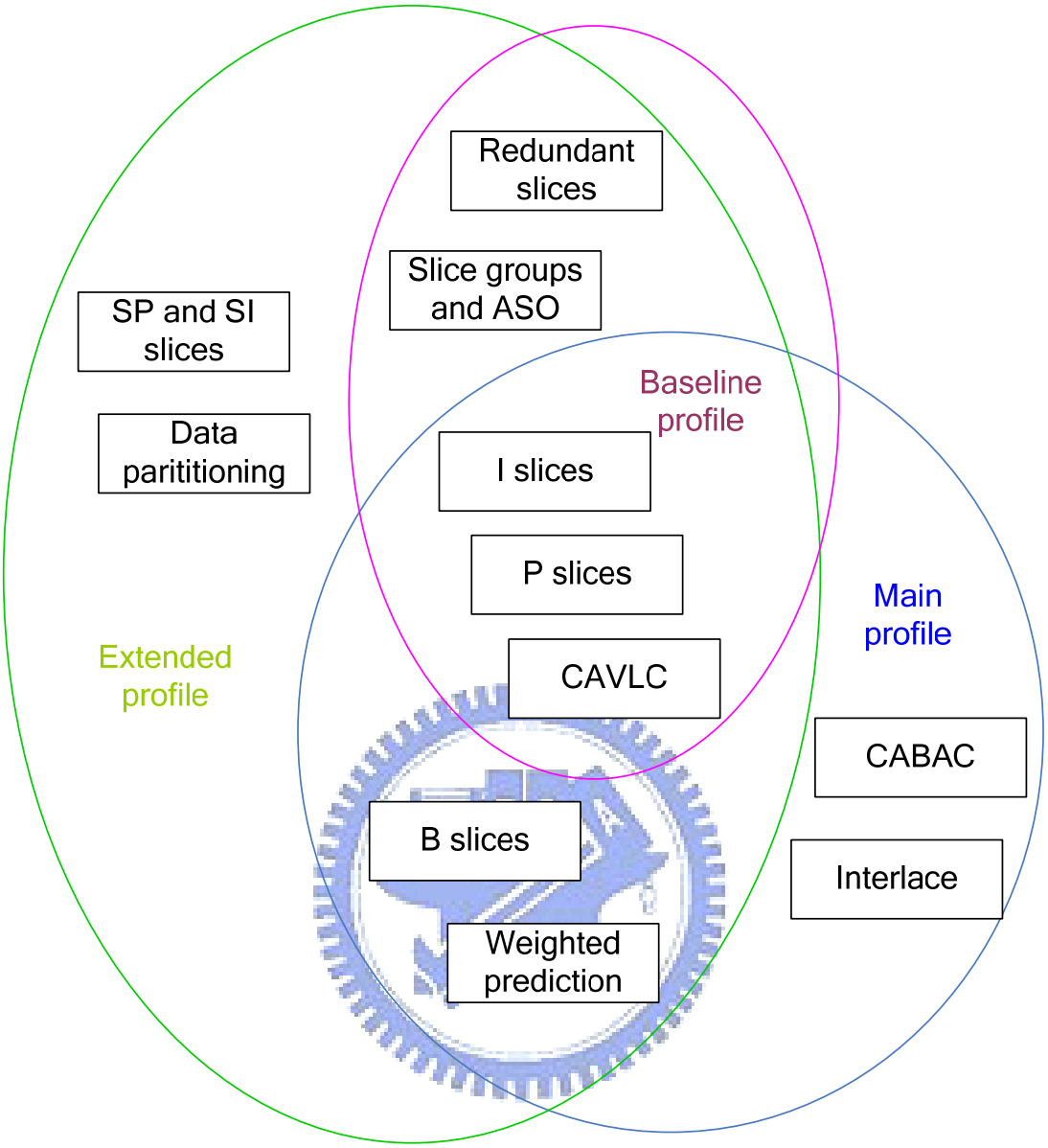


Figure 2.1 Relation between the Baseline, Main, Extended profiles.

2.5.2 Process of Encoding

Before describing the process of encoding of H.264 videos, we introduce the concept "*prediction*" first. Because a video sequence is formed by consecutive similar images, there is a lot of coding redundancy. How to use the high correlation between these similar images to reduce the redundancy has become a main topic in the video compression research field. The basic idea of prediction is to find a block which is the most similar to the current block and to save the difference between the two. There

are two models of prediction: intra mode and inter mode. The intra mode uses the similarity between pixel samples in the same frame and the inter mode uses the similarity between different frames.

The process of encoding of H.264 videos is illustrated in Figure 2.2. There are a forward path and a reconstruction path in the figure. In the forward path, an input frame F_n is processed in units of a macroblock. Each macroblock is encoded in intra or inter mode and can be sub-partitioned into sub-macroblocks. For each sub-macroblock in the macroblock, a prediction P is formed based on previously encoded, decoded, and reconstructed samples. In the intra mode, P is formed from the samples in the current slice. In the inter mode, P is formed by the samples in the past or future frames which can also be called reference frames. The difference between P and the current sub-macroblock is used to produce a residual sub-macroblock that is DCT-based transformed and quantized. The resulting frequency coefficients are reordered and entropy encoded. The coefficients after entropy coding and other information required in a decoding process (prediction modes, motion vectors, etc.) form the compressed bitstream which is passed to a Network Abstraction Layer (NAL) for transmission or storage usage. In the reconstruction path, the encoder decodes (reconstructs) the previously encoded data to provide a reference for further predictions. The encoded data is inverse transformed to produce a difference sub-macroblock and the prediction block P is added to the difference sub-macroblock to create a reconstructed sub-macroblock which is a decoded version of the original sub-macroblock.

2.5.3 Process of Decoding

The decoder receives a compressed bitstream from the NAL and entropy decodes

the data elements to produce quantized coefficients. Then these coefficients are inversely transformed to yield a difference sub-macroblock. Using the decoding information retrieved from the bitstream, the decoder creates a prediction sub-macroblock P which is identical to the original prediction sub-macroblock in the encoder. P is added to the difference sub-macroblock to produce a decoded sub-macroblock. The flow chart of the decoding process is illustrated in Figure 2.3.

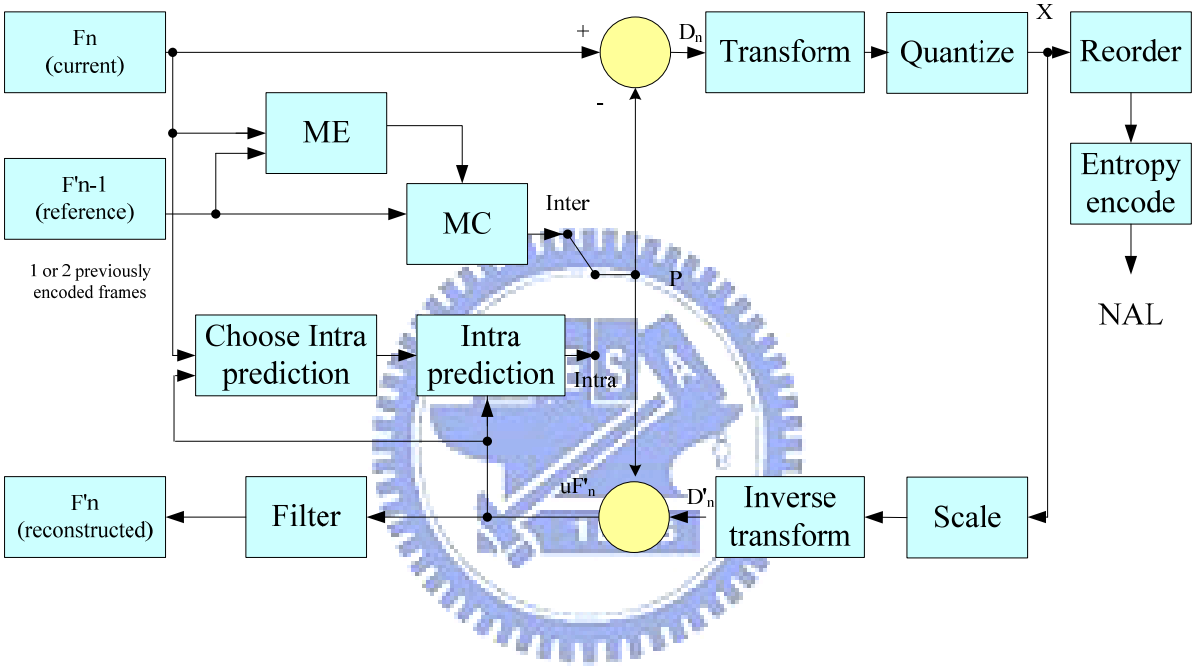


Figure 2.2 Flow chart of H.264/AVC encoding process.

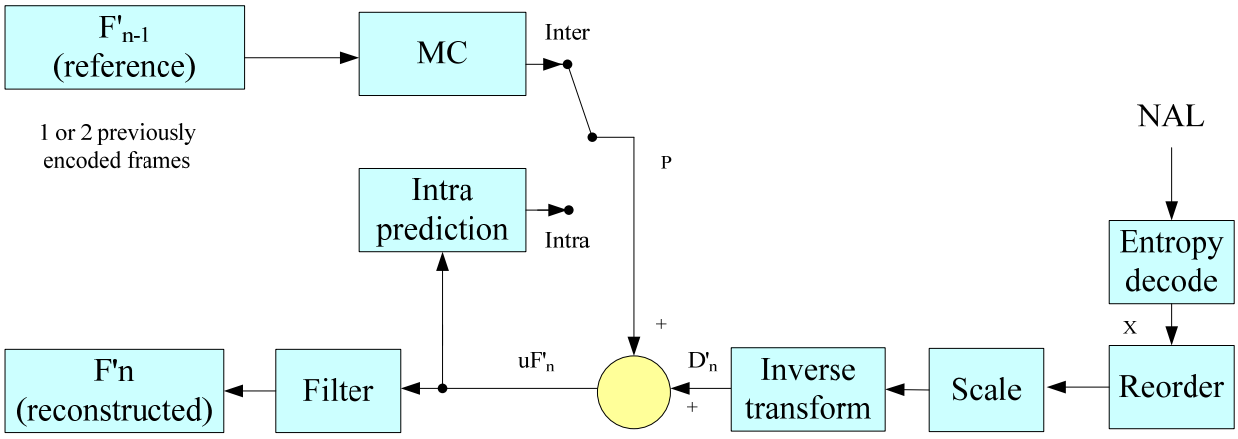


Figure 2.3 Flow chart of H.264/AVC decoding process.

2.5.4 Tree Structured Motion Compensation

Motion compensation is the process of finding the best prediction block in inter mode. In all video standards except H.264, the processing unit of motion compensation is a whole macroblock. H.264 introduces a novel feature: *tree structured motion compensation*. The basic concept is that a macroblock can be divided into sub-macroblocks and each of the sub-macroblocks is motion compensated individually. Macroblocks can be partitioned in different modes for different video contents.

Each 16×16 macroblock may be partitioned and motion compensated by one of the following ways: one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions, and four 8×8 partitions, as illustrated in Figure 2.4. If the 8×8 mode is chosen, each of the four 8×8 sub-macroblocks in the macroblock may be further partitioned in four ways: one 8×8 sub-macroblock partition, two 8×4 sub-macroblock partitions, two 4×8 sub-macroblock partitions, and four 4×4 sub-macroblock partitions, as illustrated in Figure 2.5. This method of partitioning macroblocks into motion compensated sub-macroblocks of varying sizes gives rise to a large number of possible combinations in each macroblock.

Each sub-macroblock requires a separate motion vector. Choosing a large partition size (16×16, 16×8, 8×16) means that a small number of bits are needed to transmit the motion vector(s) and the partition mode(s) but the motion compensated residual may be a large number in detailed frame areas. Choosing a small partition size (8×4, 4×4, etc.) may result in a small residual but needs a larger number of bits to transmit the motion vectors and the partition modes. In general, a large partition size is appropriate for smooth frame areas while a small partition size is suitable for detailed areas, as illustrated in Figure 2.6.

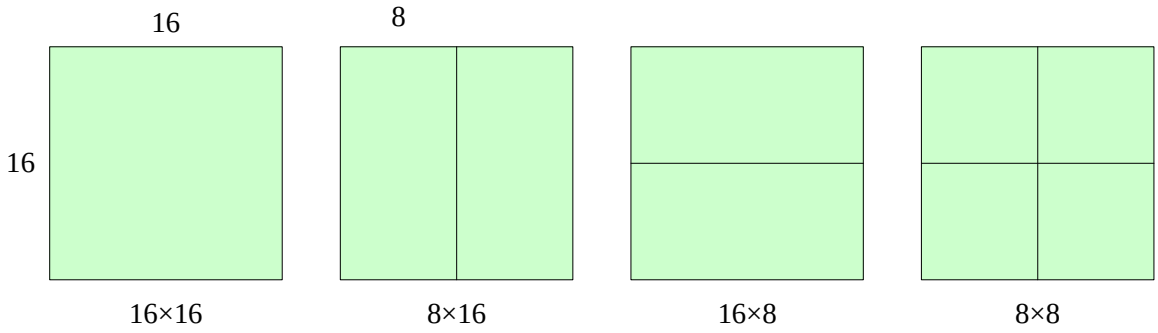


Figure 2.4 Macroblock partitions.

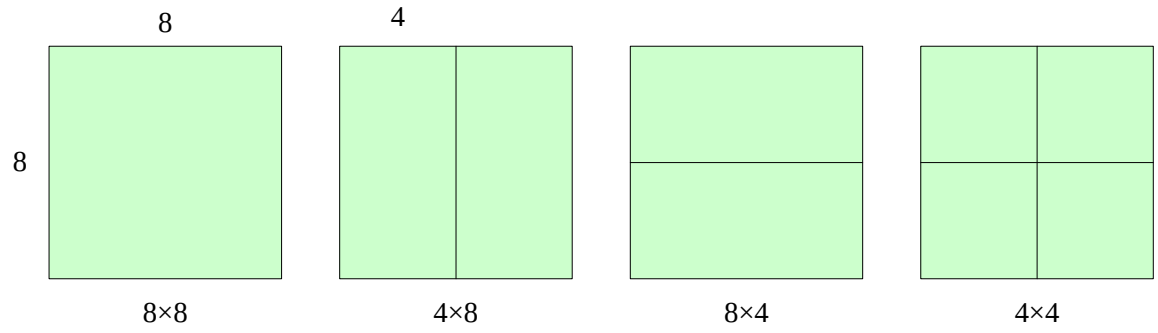


Figure 2.5 Sub-macroblock partitions.

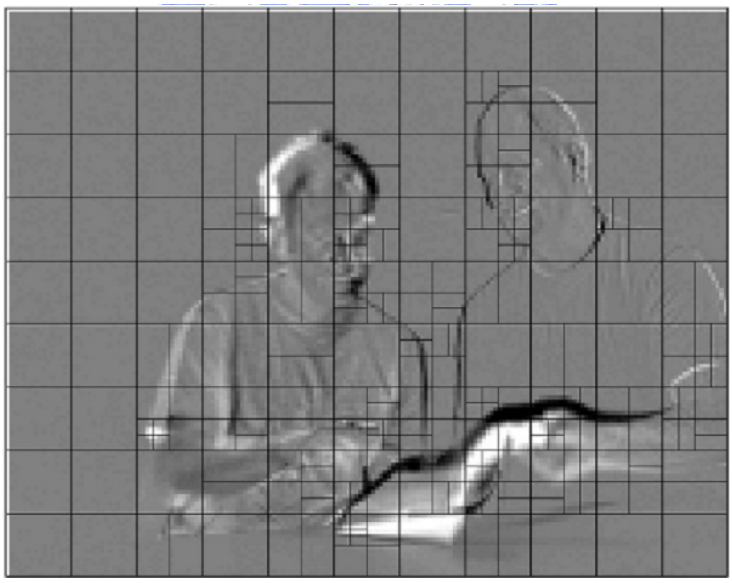


Figure 2.6 An example of tree structured motion compensation.

2.5.5 Intra Prediction Modes

Within an intra macroblock of an H.264 video, a 4×4 sub-macroblock is a unit of processing in intra prediction. Thirteen pixel samples are formed by previously encoded and reconstructed blocks. As illustrated in Figure 2.7, A, B, C and D are from the upper neighboring block; E, F, G and H are from the upper-right neighboring block; I, J, K and L are from left neighboring block, and M is from the upper-left neighboring block. There are nine prediction modes for the thirteen pixel samples to form the prediction block of the current processing 4×4 block. The nine modes of intra prediction are illustrated in Figure 2.8. The prediction block is subtracted from the current block to create the residual block. The encoder selects the best mode of intra prediction that performs the lowest cost of encoding.

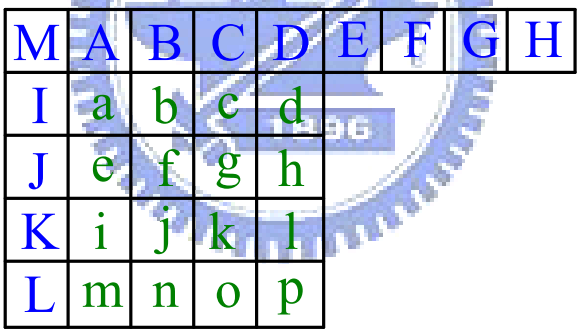


Figure 2.7 The prediction block and the thirteen samples.

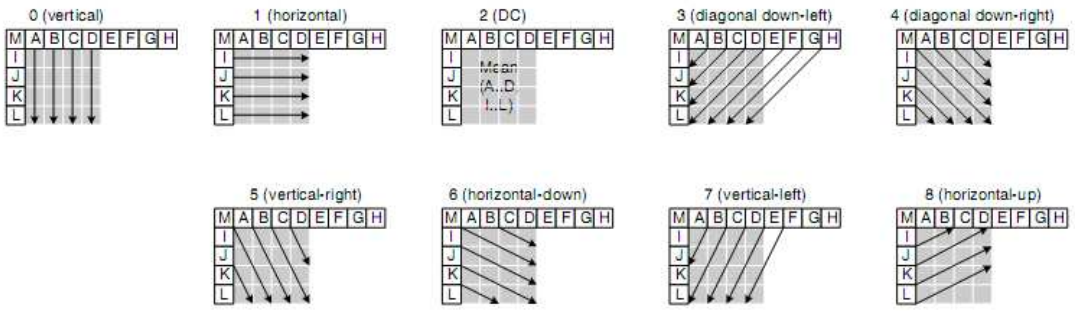


Figure 2.8 The nine modes of intra prediction.

Chapter 3
Searches of Video Contents for Scene Surveillance by Novel Uses of H.264 Coding Features

3.1 Introduction

Since a surveillance video system usually monitors a space for a long period, it may record lots of suspicious people or activities. If someone wants to check whether a surveillance video contains illegal activities, it will often take him/her a very long time to search the whole video for the specific activities or involved people. In this chapter, we describe the proposed method of video-content search by a novel use of H.264 coding features to avoid tedious search on recorded videos. With such a video-content search method, it will become much easier and faster to check any suspicious activity or people in recorded videos.

In Section 3.1.1, some definitions related to the video-content search problem are described, and the proposed idea and system configuration are given in Section 3.1.2. In Section 3.2, a motion detection algorithm based on the proposed idea is introduced. In Section 3.3, the way we use for embedding the motion region information is described. In Section 3.4, the process of extraction of motion region information is presented. Some experiment results are shown in Section 3.5. In Section 3.6, the last section of this chapter, some discussions and summary are given.

3.1.1 Problem Definition

In the video-content search problem dealt with in this study, the activities recorded in an input video are detected and embedded back into the video for later search. Two issues are involved in this problem. The first is how to detect motion regions correctly in a video taken by a real-time surveillance system with a stationary camera. Motion detection is a very popular research topic in video analysis and can be implemented in many different ways as described in Chapter 2. The second issue is how to embed information about the detected motion regions into an H.264 compressed bitstream during an encoding process and how to extract them during a decoding process.

3.1.2 Proposed Idea

In the proposed method, each frame captured from a stationary camera is encoded into a compressed bitstream in the H.264 encoding process. During the encoding process, a novel motion detection technique is used in this study to detect suspicious activities in the currently-processed frame. While the motion regions are detected, the location information of the motion regions is embedded into the quantized frequency domain of the compressed H.264 bitstream. Therefore, if someone wants to know whether a specific region of a video contains suspicious activities or not, the data extraction process in the proposed method can be utilized to search the video contents and output the video clips that the user is interested in.

3.2 Detection of Motion Regions by H.264/AVC Coding Features

In the proposed system, we introduce a novel motion detection technique by use of H.264 coding features. In Section 3.2.1, the idea of the proposed technique is stated. And in Section 3.2.2, the detailed process of the proposed motion detection technique is described.

3.2.1 Proposed Idea of Motion Detection Technique

As mentioned in Chapter 2, in an encoding process of a P or B slice of a compressed video stream, an H.264 encoder needs to find the best partition mode of the currently processed macroblock. Each 16×16 macroblock may be partitioned and motion compensated by one of the following ways: one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions, and four 8×8 partitions, as illustrated in Figure 3.1(a). If the 8×8 mode is chosen, each of the four 8×8 sub-macroblocks in the macroblock may be further partitioned in four ways: one 8×8 sub-macroblock partition, two 8×4 sub-macroblock partitions, two 4×8 sub-macroblock partitions, and four 4×4 sub-macroblock partitions, as illustrated in Figure 3.1(b).

For the same macroblock, different partition modes produce different coding costs to the compressed video stream. If the encoder does not choose the best partition mode for the current macroblock, it will cause more bits to be included in the stream. Therefore, the encoder does so according to the video content of the currently-processed macroblock to get the lowest coding cost. Generally speaking, the partition modes with large partition sizes (16×16, 16×8, 8×16) are suitable for smooth areas, while the modes with small partition sizes (8×8, 8×4, 4×8, 4×4) are appropriate for detailed areas.

Video contents of motion regions usually change greatly both in the time domain and in the spatial domain. In the time domain, the movements in the motion regions result in a lot of information of *changes*, which needs to be described. In the spatial domain, the motion regions may contain some moving *objects* which might be humans, cars, etc. These moving objects might contain lots of details that need to be encoded.

Changes in the time domain generally make the partition sizes of the motion regions to be small ones in order to reduce the coding cost calculated in the motion compensation process. Therefore, the partition modes of the motion regions are mostly with small partition sizes. Moreover, changes in the spatial domain make the partition modes of the macroblocks within the motion regions variable, because video contents between these macroblocks are quite different from each other.

Based on the clues mentioned above, we choose to use small partition sizes and variable partition modes as features of motion regions, and use them and motion vectors to detect motion regions in surveillance videos.

Besides, after detecting motion regions by these features, some noise caused by lights and shadows might be included in the detected motion regions and appears on the fringes of the regions. We call macroblocks containing such noise in the detected motion regions as *noise macroblocks*. Partition modes of these noise macroblocks usually include large partition sizes; Therefore, we also use this characteristic to eliminate the noise. An example of noise macroblocks is illustrated in Figure 3.2.

3.2.2 Process of Detection of Motion Regions

The proposed motion detection method is applied to frames composed of P slices only. During the encoding process of an input frame, the length of every motion

vector of a sub-macroblock of the input frame is compared with a pre-defined threshold in order to filter motion-less sub-macroblocks. The remaining sub-macroblocks are called *motion blocks*.

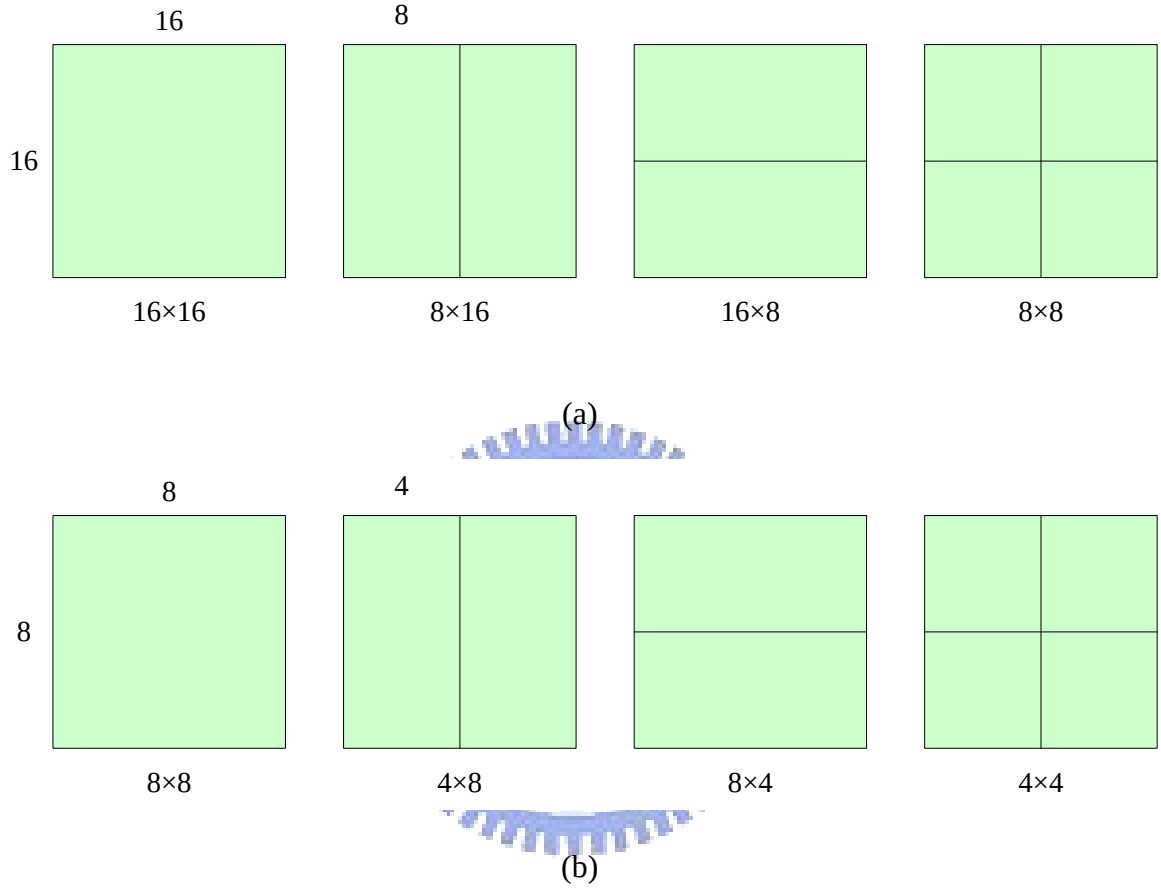


Figure 3.1 Tree structured partition ways. (a) Macroblock partitions. (b) Sub-macroblock partitions.

We obtain candidate motion regions by applying a region growing algorithm to the motion blocks. The basic concept of the region growing algorithm is to check the eight neighboring 16×16 macroblocks M_1 through M_8 around each 16×16 macroblock where the motion blocks are located. If any of M_1 through M_8, say M_i, contains motion blocks, we check further the eight neighboring 16×16 macroblocks of M_i, and so on recursively, until reaching the boundary of the input frame.

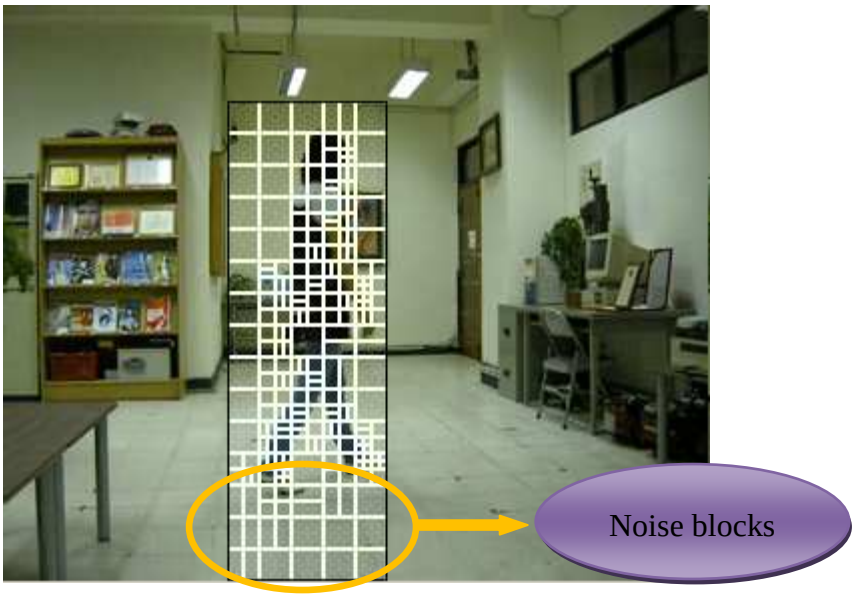


Figure 3.2 An example of noise macroblocks.

Since there may be noise caused by lights and shadows on the fringes of each candidate region, a *range reduction algorithm* is introduced to eliminate the noise. The basic idea is that the partition mode of the noise macroblock is usually a large partition size (16×16, 16×8, 8×16) and appears on a fringe. So, we try to shrink most fringes composed of large partition sizes in the range reduction algorithm.

While obtaining these reduced candidate regions, we perform a process to calculate *partition variances* of these regions. To calculate the partition variances, we quantify in advance the partition modes of the motion blocks of these regions by assigning each of them a value called a *quantification value* based on the mode numbers defined in an H.264 encoder and position information illustrated in Figure 3.3. The position information is defined based on transforming the two dimensional coordinates of each 8×8 sub-macroblock within a 16×16 macroblock into one dimensional. We then use the quantification value of each motion block to calculate the partition variance of each candidate motion region. The detail will be described

later. Each candidate region is evaluated finally by the partition variance. The larger the variance, the more possible for the candidate region to be a motion region.

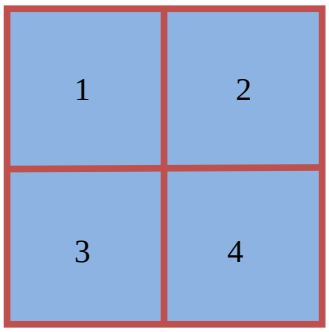


Figure 3.3 The position information of each 8×8 sub-macroblock of a 16×16 macroblock.

Algorithm 3.1. Motion detection process.

Input: a frame F composed of P slices only, a motion vector threshold T_m, and a variance threshold T_v.

Output: position information about the motion regions in F.

Steps:

1. For each sub-macroblock M_s in F, if the length of the motion vector of M_s is larger than T_m, then decide M_s as a motion block.
2. Perform the region growing algorithm to group the motion blocks in F to form several candidate motion regions. Circumscribe each candidate with a rectangle.
3. Reduce the range of each candidate motion region R by shrinking each edge E of the rectangle of R according to the following steps.
   3.1 Define a set S for E, which includes motion sub-macroblocks of 16×16 macroblocks in contact with E.
   3.2 For each motion block B_e in S, compare the length L_e of its motion vector with the mean L_m of the lengths of the motion vectors of all the motion

blocks in S. If L_e is larger than L_m and the partition mode of B_e is a small partition size mode, jump back to Step 3.1 to skip shrinking edge E and to continue processing the next set S.

3.3 After checking all the motion blocks in S in Step 3.2, count the number of motion blocks of large partition sizes of S. If the number is over a half of the total number of motion blocks of S, shrink the edge E for 16 pixels.

3.4 Go to Step 3.1 to process the next edge until all edges of R have been processed.

4. Calculate the partition variance V of each reduced region R' of R according to the following steps.

4.1 Assign each motion block B_R in R' a *quantification value* according to the partition mode number N_{mode} defined in the encoder and the position information Pos of B_R in the following way:

$$\begin{aligned} &\text{if the partition size is small, set quantificaiton value} = N_{mode} + Pos; \\ &\text{if the partition size is large, set quantificaiton value} = N_{mode}. \end{aligned} \tag{3.1}$$

4.2 Use the quantification values of the motion blocks to calculate the variance V of R'.

5. If V is larger than T_v, put R' into a candidate list L; otherwise, find a block B in the region R' which has the largest motion vector length. If the location of B is inside the motion regions of the previous frame, put R' into the list L, too; otherwise, drop R'.

6. Perform a merge process to the remaining regions in L according to the following rules.

6.1 If there are regions which belong to the same motion region in the previous frame, merge these regions.

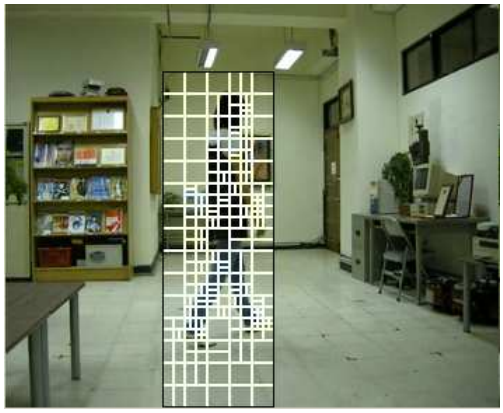
6.2 If there are overlapping regions, merge them.

7. Output the position information of the motion regions remaining in L after the merge process.

The above algorithm shrinks edges by counting the number N_e of motion blocks of large partition sizes in each edge E. If N_e is larger than a half of the total number of motion blocks in E, E is shrunk because the overall composition of E is the large partition size. An example of the results obtained from applying the range reduction algorithm is illustrated in Figure 3.4, in which Figures 3.4(a) through 3.4(d) are the motion detection results of four consecutive P frames without performing the algorithm, and Figures 3.4(e) through 3.4(h) are the results of the same four P frames instead. From Figures 3.4(b) and 3.4(f), we can see that the range reduction algorithm does improve the motion detection result effectively.

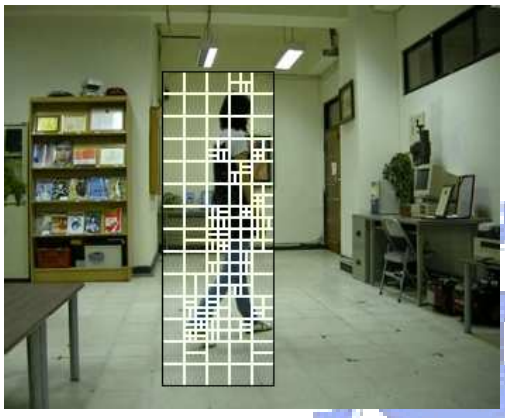
(a) (e)

Figure 3.4 An example of the results of applying the range reduction algorithm. The black rectangle is the motion region and the white rectangles are the sub-macroblocks in the region. (a) The first P frame without reduction. (b) The second P frame without reduction. (c) The third P frame without reduction. (d) The 4th P frame without reduction. (e) The first P frame with reduction. (f) The second P frame with reduction. (g) The third P frame with reduction. (h) The 4th P frame without reduction.



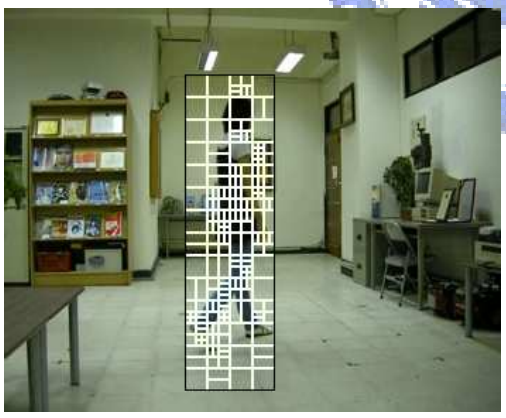
(b)

(f)

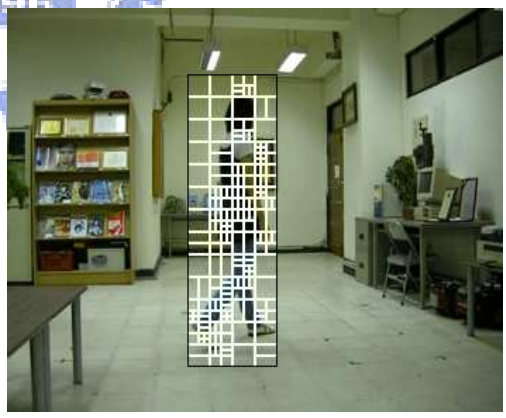


(c)

(g)



(d)



(h)

Figure 3.4 An example of the results of applying the range reduction algorithm. The black rectangle is the motion region and the white rectangles are the sub-macroblocks in the region. (a) The first P frame without reduction. (b) The second P frame without reduction. (c) The third P frame without reduction. (d) The 4th P frame without reduction. (e) The first P frame with reduction. (f) The second P frame with reduction. (g) The third P frame with reduction. (h) The 4th P frame without reduction (continued).

In the process of assigning quantification values to motion blocks, we calculate the quantification values based on the mode numbers which have been defined in the encoder. Moreover, motion regions are mostly composed of small partition sizes (8×8, 8×4, 4×8, 4×4), so a macroblock in a motion region may be partitioned into many small sub-macroblocks. Therefore, for the motion blocks with small sizes, the quantification values are determined not only by mode numbers but also by position information defined in Figure 3.3 in order to distinguish different combinations of these small sub-macroblocks and increase the variety of the quantification values. An example is illustrated in Figure 3.5 in which Figure 3.5(a) is the detection result without utilizing the partition variances, and Figure 3.5(b) is the result instead.

(a) (b)

Figure 3.5 An example of selecting candidate motion regions by partition variances. The black rectangle is the candidate motion region and the white rectangles are the sub-macroblocks in the region. (a) The detection result obtained without using partition variances. (b) The detection result obtained by using partition variances.

3.3 Embedding and Extracting Data in H.264 Videos

DCT-based data hiding methods are commonly used for videos and images. In video standards before H.264 is introduced, the DCT-based methods are the basis of many applications related to videos. However, while re-encoding, the DCT-based methods face a problem that different intra prediction modes cause different sets of frequency coefficients. Therefore, we introduce an improved DCT-based data hiding method suitable for H.264 videos. In Section 3.3.1, the proposed technique of embedding data is introduced. In Section 3.3.2, the proposed technique of extracting data is described.

3.3.1 Process of Embedding Data

In this section, we introduce the method of embedding data. The proposed idea is described in Section 3.3.1.1. In Section 3.3.1.2, the detailed algorithm is stated.

3.3.1.1 Proposed Idea

While an H.264 encoded stego-video is re-encoded, the resulting intra prediction modes may be distinct from the original ones. Hence, the data hidden in the frequency domain in this video may be lost because the resulting frequency coefficients could be different. As a result, we introduce a secret-key-based data hiding method. A secret key designated by a user is used to generate intra prediction modes used in data hiding in order to prevent the hidden data from being lost. The modes generated in the data hiding process will not affect the normal encoding procedure.

3.3.1.2 Detailed Algorithm

The proposed data hiding method deals with frames composed of I slices only. Each 4×4 sub-macroblock of an I frame is used to hide one bit of data. The data hiding process is performed before the encoding process. A secret key is utilized as the seed of a random number generation. The result of the random number generation is used to decide an intra prediction mode P of each 4×4 sub-macroblock. A residual block generated based on P is transformed and quantized. The resulting frequency coefficients are modified to hide one bit of data and inverse transformed to produce a reconstructed block which substitutes the original 4×4 sub-macroblock. The encoder continues to encode this stego 4×4 sub-macroblock. In more details, P will not affect the normal encoding process and the prediction mode decided by the encoder is still the best prediction mode yielding the lowest coding cost.

Algorithm 3.2. The process of hiding a 16×16 macroblock in an H.264 video frame composed of I slices only.

Input: a secret key R, binary data D to be hidden, a 16×16 macroblock M_{16}, and a random number generator f.

Output: a stego macroblock M_{16}'.

Steps:

1. Use the input secret key R as a seed for f to generate a sequence of random numbers.
2. Perform the following steps before M_{16} is encoded.
   2.1 For a 4×4 sub-macroblock M_4 in M_{16}, select one of the available intra prediction modes *Mode* of M_4 according to the result of the random number generation.

2.2 Use *Mode* to produce a prediction block M_p and subtract it from M_4 to generate a residual block M_r.

2.3 Transform M_r into the frequency domain to get the corresponding frequency coefficients of M_4 in the form of a 4×4 block, *Coeff*.

2.4 Modify *Coeff* in order to hide an un-hidden bit B of D according to the following rules.

2.4.1 Select the coefficient pair $C_1(0, 3)$ and $C_2(3, 0)$ in *Coeff*.

2.4.2 Modify C_1 and C_2 according to the following equations:

(1). if $B = 0$:

$$if\ C_1 > C_2,\ swap\ C_2\ and\ C_1; \tag{3.2}$$

(2). if $B = 1$:

$$\begin{aligned} &if\ C_2 = C_1,\ C_1 = C_2 + T; \\ &if\ C_2 > C_1,\ swap\ C_2\ and\ C_1, \end{aligned} \tag{3.3}$$

where T is a pre-defined threshold.

2.5 Inverse transform *Coeff* and add the result to M_p to produce a reconstructed 4×4 sub-macroblock M_c, which is then taken to replace the original 4×4 sub-macroblock.

3. Repeat Step 2 until all 4×4 sub-macroblock in M_{16} are processed or until the data in D to be hidden are all processed.

4. Take the modified macroblock M_{16}' as input to the normal encoding process.

3.3.2 Process of Extracting Data

In this section, we introduce the proposed data extraction technique. The proposed ideas are described in Section 3.3.2.1. In Section 3.3.2.2, the detailed algorithm for it is presented.

3.3.2.1 Proposed Idea

The proposed method extracts data in a decoder. While the decoder performs a decoding process, we can use a secret key to select an intra prediction mode for each 4×4 sub-macroblock of a reconstructed macroblock. These intra prediction modes can produce sets of frequency coefficients. The hidden data can be extracted from these frequency coefficients.

3.3.2.2 Detailed Algorithm

When the video decoder decodes a macroblock to produce a reconstructed macroblock, a prediction sub-macroblock is formed for each 4×4 sub-macroblock of the reconstructed macroblock based on a prediction mode selected by a secret key. The prediction sub-macroblock is subtracted from the reconstructed 4×4 sub-macroblock to produce a residual block which is then DCT-based transformed to a set of frequency coefficients. The hidden data can be extracted from these sets of frequency coefficients. The details are described in the following algorithm.

Algorithm 3.3. The process of data extraction from a 16×16 macroblock in an H.264 video frame composed of I slices only.

Input: a 16×16 macroblock M_{16}, a secret key R, and a random number generator f.

Output: 16 bits of data hidden in M_{16}.

Steps:

1. Use the input secret key R as a seed for f to generate a sequence of random numbers.
2. Perform the following steps after M_{16} is decoded.
   2.1 For each 4×4 sub-macroblock M_{4i} of M_{16}, select an intra prediction mode P

according to the result of the random number generation.

2.2 Use P to produce a prediction block P'.

2.3 Subtract P' from M_{4i} to produce a residual block R.

2.4 Transform R into a set of frequency coefficients *Coeff*.

2.5 Extract the hidden data $bit(i)$ of M_{4i} from *Coeff* according to the following equation:

$$\begin{aligned} &if\ C_2 \geq C_1,\ set\ bit(i) = 0; \\ &else,\ set\ bit(i) = 1, \end{aligned} \tag{3.4}$$

where C_1 is the frequency coefficient $C_1(0, 3)$ and C_2 is $C_2(3, 0)$.

3. Repeat Step 2 until all 4×4 sub-macroblock of M_{16} are processed.

4. Combine all $bit(i)$ of M_{4i} to form the 16 bits of the extracted data as output.

3.4 Embedding of Motion Region Information

In this section, we introduce a technique for embedding motion region information into I frames during an encoding process. In Section 3.4.1, the principle of the proposed embedding process of motion region information is given, and the detailed algorithm of the embedding process is described in Section 3.4.2.

3.4.1 Principle of Proposed Technique

In the proposed system, the motion region information will be taken as input into the H.264 encoding process while there are motions detected. The input information is embedded into the H.264 video by the data hiding method introduced previously. The embedded information is used to index the surveillance video for the subsequent search process. Because the motion of most moving objects will last for several

frames, we divide the video sequence into frame groups formed with several consecutive frames composed of P slices only, called *P frames*, and one frame composed of I slices only, called *I frame*. One of the consecutive P frames is taken as input to the motion detection process and the resulting detection information is embedded into the I frame belonging to the same group if there are motions detected in this group.

3.4.2 Process of Embedding Motion Region Information

We divide the input video into several *frame groups*, and each is composed of some P frames and one I frame. We then apply the motion detection process on these P frames and if there are motions detected in these P frames, the location information of these motion regions forms a string *I*. Since the motion of most moving objects usually last for several frames, we do not hide motion region information of every P frames in the group. We select just one of these P frames, in which the motion regions have covered larger areas in these P frames to represent this group. The string is then transformed into a binary form I_b and embedded into the I frame in the same group. Each bit of I_b is embedded into a 4×4 sub-macroblock in the frequency domain. As mentioned above, we randomly select an intra prediction mode to decide which set of frequency coefficients is used to hide the data.

Algorithm 3.4: The process of embedding motion region information.

Input: a secret key *R*, moving region information *M*, a random number generator *f*, and an I frame *F*.

Output: a stego-I frame F'.

Steps:

1. Denote the binary form of M as $M_b = b_1b_2b_3\ldots b_L$, where L represents the length of M_b.
2. Use the input secret key R as a seed for f to generate a sequence of random numbers.
3. Take out consecutive 16 bits of M_b, which have not been hidden, and denote these data bits as M_{b16}.
4. Perform the following steps before the currently-processed macroblock MB is encoded.

   4.1 For a 4×4 sub-macroblock MB_4 in MB, select one of the available intra prediction modes $Mode$ of MB_4 according to the result of the random number generation.

   4.2 Use $Mode$ to produce a prediction block MB_p and subtract it from MB_4 to generate a residual block MB_r.

   4.3 Transform MB_r into the frequency domain to get the corresponding frequency coefficients of MB_4 in the form of a 4×4 block, $Coeff$.

   4.4 Hide an un-hidden bit B of M_{b16} into $Coeff$ according to the following rules.

      4.4.1 Select the coefficient pair C_1 (0, 3) and C_2 (3, 0).

      4.4.2 Modify C_1 and C_2 according to the following rule:

      (1) if $B = 0$:

$$if\ C_1 > C_2, swap\ C_2\ and\ C_1; \tag{3.5}$$

      (2) if $B = 1$:

$$\begin{aligned} &if\ C_2 = C_1,\ C_1 = C_2 + T; \\ &if\ C_2 > C_1, swap\ C_2\ and\ C_1, \end{aligned} \tag{3.6}$$

      where T is a pre-defined threshold.

   4.5 Inverse transform $Coeff$ and add the result to MB_p to produce a

reconstructed 4×4 sub-macroblock M_c, which is then taken to replace the original 4×4 sub-macroblock.

5. Repeat Step 4 until all 4×4 sub-macroblock in *MB* are processed or until the data in M_{b16} to be hidden are all processed.
6. Jump to Step 3 and repeat the following steps until reaching the last macroblock of *F* or the end of *M*.

3.5 Extraction of Motion Region Information

In this section, we introduce a method for extracting motion region information during a decoding process. In Section 3.5.1, the principle of the proposed process of extracting motion region information is given, and the detailed algorithm of the extraction process is described in Section 3.5.2.

3.5.1 Principle of Proposed Technique

A compressed stego-video is recorded by the proposed system. Before searching motion motions in the resulting surveillance video, the motion region information must be extracted first. A secret key is used to form a seed of a random number generation. Extraction of the information hidden in the frequency domain is performed based on the result of the random number generation.

3.5.2 Process of Extraction of Motion Region Information

We extract motion region information during a decoding process. For each 4×4

sub-macroblock of the current I frame, an intra prediction mode chosen by the result of the random number generation produces a set of frequency coefficients. Each bit of hidden motion region information is extracted from the set of frequency coefficients. The extraction process continues until reaching the end mark of the motion region information or the last macroblock of the current I frame.

Algorithm 3.5. The process of extracting motion region information.

Input: a stego-I frame *F*, a secret key *R*, and a random number generator *f*.

Output: motion region information of the frame group *F* belonging to.

Steps:

1. Use the input secret key *R* as a seed for *f* to generate a sequence of random numbers.
2. Perform the following steps after the currently-processed 16×16 macroblock M_{16} is decoded in the decoding process.
   - 2.1 For a 4×4 sub-macroblock M_{4i} of M_{16}, select an intra prediction mode *P* according to the result of the random number generation.
   - 2.2 Use *P* to produce a prediction block *P′*.
   - 2.3 Subtract *P′* from M_{4i} to produce a residual block *R*.
   - 2.4 Transform *R* to a set of frequency coefficients *Coeff*.
   - 2.5 Extract the hidden data *bit*(*i*) from *Coeff* according to the following equation:

$$\begin{aligned} &if\ C_2 \geq C_1, bit(i) = 0; \\ &else, bit(i) = 1, \end{aligned} \tag{3.7}$$

   where C_1 is the frequency coefficient $C_1(0, 3)$ and C_2 is $C_2(3, 0)$.
3. Repeat Step 2 until all 4×4 sub-macroblocks of the current macroblock are processed or until the end mark of motion region information is reached.

4. Continue processing the next macroblock until reaching the end mark of the motion region information or the last macroblock of this frame.
5. Combine each $bit(i)$ of M_{4i} to get the motion region information and output.

3.6 Experimental Results

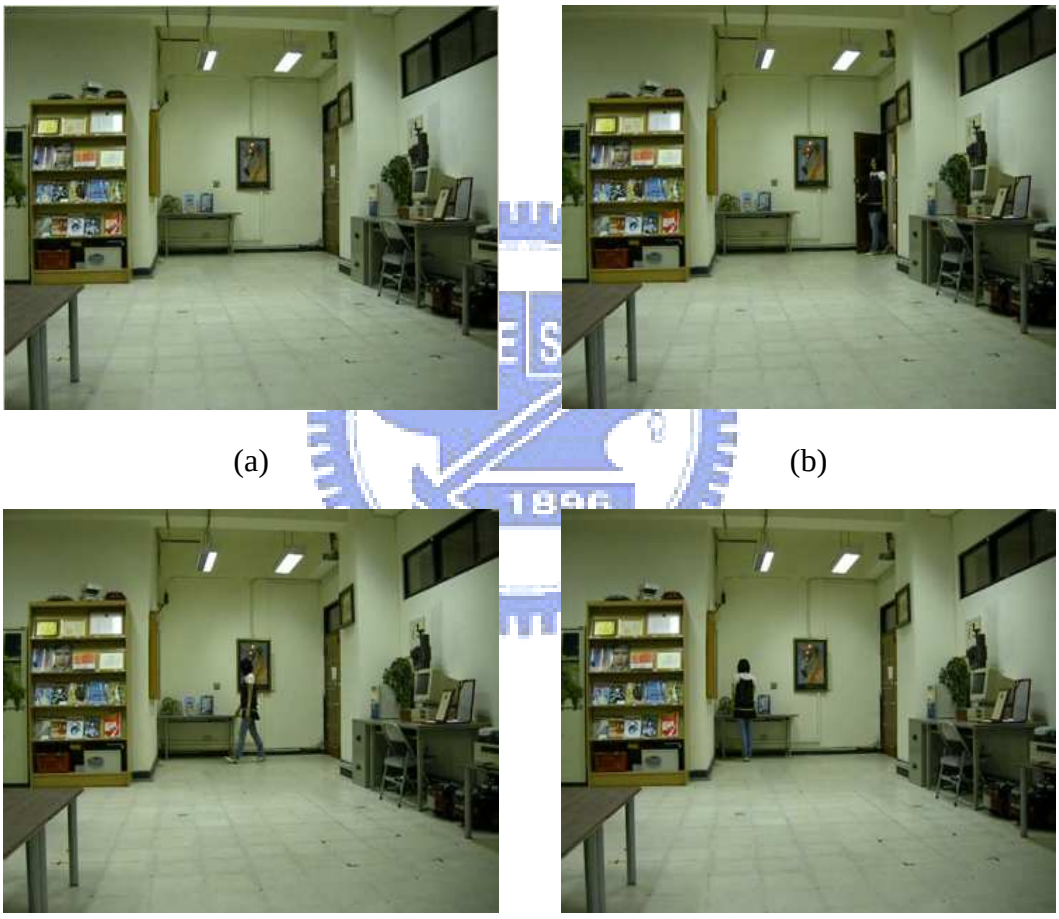
In our experiments, each image captured by a video camera is encoded by an H.264 encoder to form an H.264 compressed surveillance video with frame size 352×288. We simulate a surveillance system composed of a video camera and a notebook computer. A surveillance video of the computer vision lab is taken as an input video. The contents of this video are that a person enters the lab and walks around. Ten representing frames of the resulting stego-video are shown in Figure 3.6. The proposed user interface for searching suspicious activities in a surveillance video is shown in Figure 3.7. If we want to know whether the person has been appeared around the bookshelf, we can specify a region containing the bookshelf and press the search button. Then, the related video clips will be shown in a few milliseconds. The search result of the specific region is shown in Figure 3.8. Some video clips of the result are shown in Figure 3.9.

This experiment shows that the proposed system can help a user search video clips that he or she may be interested in quickly and prevent him or her from go through the whole video to find a short video clip.

3.7 Discussions and Summary

In this chapter, we have proposed a motion detection method using tree structured macroblock decomposition information, a data hiding method suitable for H.264 videos, and a surveillance video search system. The proposed motion detection

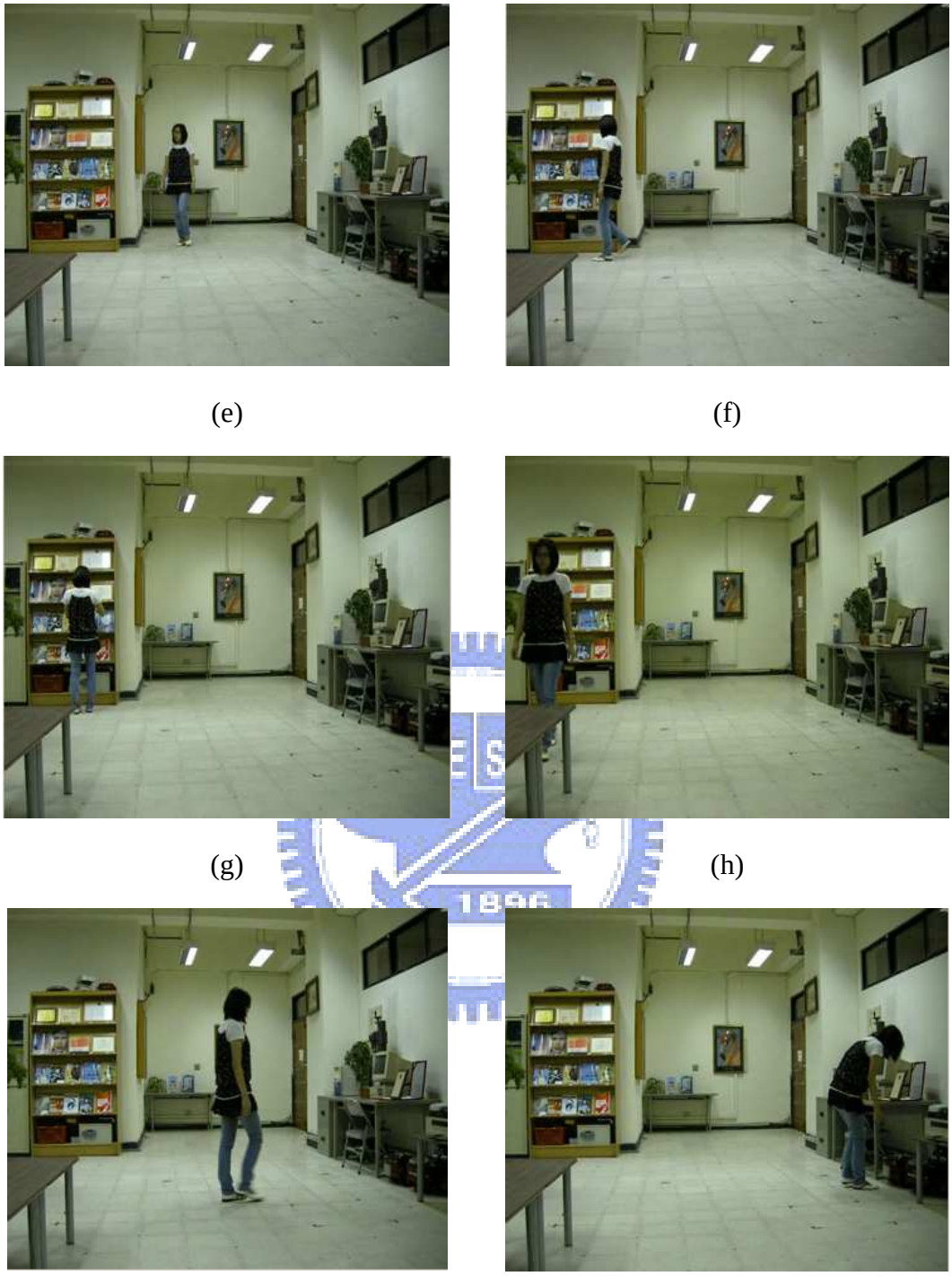
method fully utilizes the encoding information generated during the encoding process. Thus, the method can detect motion regions quickly. The proposed data hiding method avoids losing hidden data due to changes of intra prediction modes. The proposed surveillance video search system provides an easy way to search activities in a surveillance video based on the techniques mentioned above. Experimental results show the feasibility of the proposed method.



(a) (b)

(c) (d)

Figure 3.6 Ten representing frames of the resulting stego-video. (a) The first frame. (b) The second frame. (c) The third frame. (d) The 4th frame. (e) The 5th frame. (f) The 6th frame. (g) The 7th frame. (h) The 8th frame. (i) The 9th frame. (j) The 10th frame.



(e) (f)

(g) (h)

(i) (j)

Figure 3.6 Ten representing frames of the resulting stego-video. (a) The first frame. (b) The second frame. (c) The third frame. (d) The 4th frame. (e) The 5th frame. (f) The 6th frame. (g) The 7th frame. (h) The 8th frame. (i) The 9th frame. (j) The 10th frame (continued).

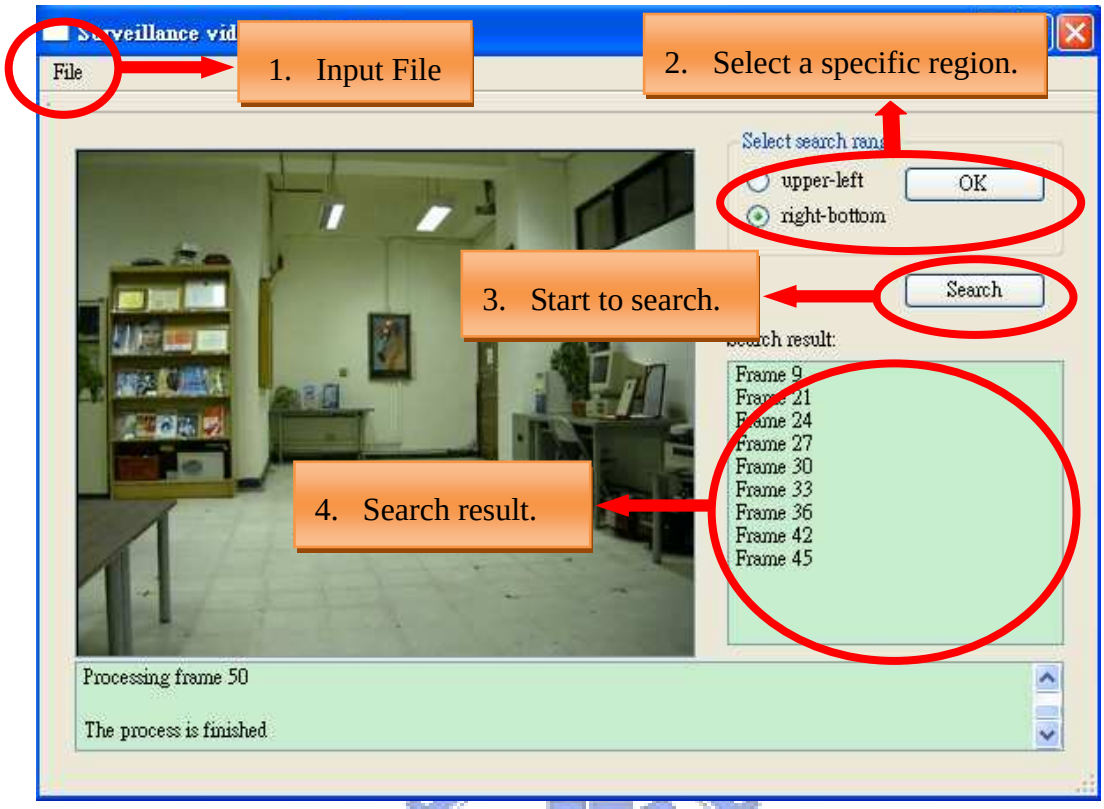


Figure 3.7 The proposed user interface for searching suspicious activities in a surveillance video.

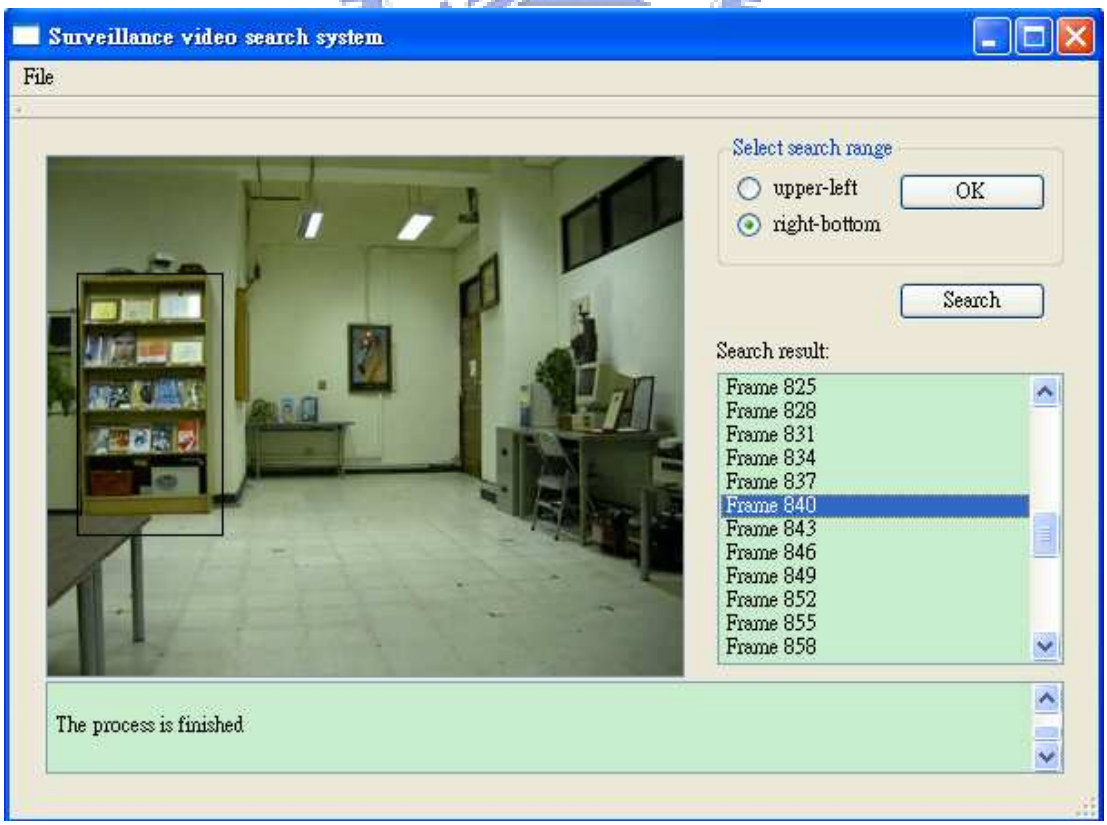
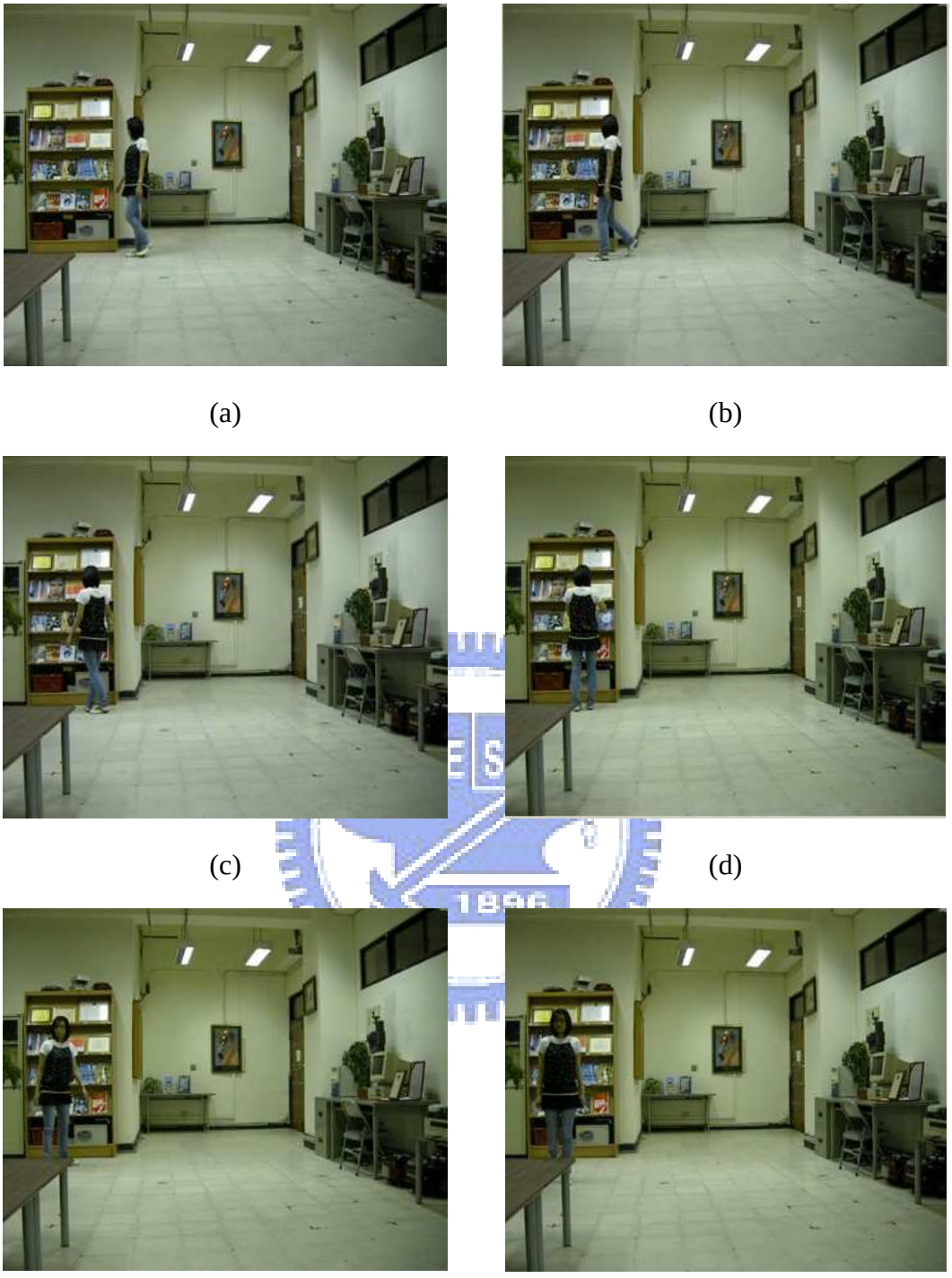


Figure 3.8 The search result of the bookshelf which is specified by a black rectangle.



(a) (b)

(c) (d)

(e) (f)

Figure 3.9 Some resulting video clips of the search in Figure 3.8. (a) The first video clip. (b) The second video clip. (c) The third video clip. (d) The 4th video clip. (e) The 5th video clip. (f) The 6th video clip.

Chapter 4
Authentication of Surveillance Videos by Hiding Tree-Structured Macroblock Decomposition Information

4.1 Introduction

With the progress of video compression technology and efficient video coding standards, digital videos nowadays have become more and more popular than in the past. The H.264 standard is especially used widely in many applications related to videos such as surveillance video systems. This convenience raises a problem that digital videos are easier to be modified through lots of video editing software than traditional ones recorded in tapes. Moreover, along with the progress of the Internet technology, digital videos are often transmitted on the Internet. As a consequence, some malicious users may acquire and tamper with the videos easily. Especially in the case of using a surveillance video system, if the videos stored in the system are tampered with, it may cause lots of serious legal disputes. Therefore, it is necessary to authenticate the integrity and fidelity of surveillance videos. In this chapter, a method for authentication of surveillance video sequences and their contents is proposed.

In Section 4.1.1, the related problem definitions are given. In Section 4.1.2, the idea of the proposed method is presented. In Section 4.2, the proposed process for

generating authentication signals is described. In Section 4.3, the proposed process for embedding authentication signals in surveillance videos is described. The proposed process for authentication of video sequences and contents is stated in the Section 4.4. Some experimental results are shown in Section 4.5. Finally, some discussions and a summary will be given in the last section of this chapter.

4.1.1 Problem Definition

The main task of a video authentication system is to verify whether a video has been tampered with or not. Tampering operations can be categorized into two types: *spatial* and *temporal*. Spatial tampering means modifications manipulated on video frame *contents*, and temporal tampering means modifications manipulated on video frame *sequences*.

Temporal tampering can be categorized further into three types: *replacement*, *cropping*, and *insertion*. Replacement means substituting fake video frames for some of the original video frames, respectively. In this way, the number of video frames will not change and the difference of the size between the original video and the fake one will be too tiny to detect. For example, a malicious user may want to replace a frame including a suspicious activity with a non-suspicious one. An illustration of replacement is shown in Figure 4.1.

Cropping means deleting some video frames from the original video sequence. For instance, a malicious user may want to eliminate his or her criminal fact by cropping some video frames in the original video sequence. An illustration of cropping is shown in Figure 4.2.

Insertion means placing some fake video frames between frames of the original video sequence. If a malicious user wants to impute his or her criminal activities to

someone who is innocent, he/she may try to insert fake video frames into the original video sequence. An illustration of insertion is shown in Figure 4.3.

The main task of the proposed authentication system is not only to detect if a surveillance video has been tampered with, but also to recognize the tampering type and mark further the altered frames.

4.1.2 Proposed Idea

In the proposed method, we divide a video sequence into several *frame groups*, with each group being composed of some P frames and one I frame. In order to detect spatial and temporal temperings, *authentication signals* are generated for each frame group *G* and hidden into the DCT coefficients of each macroblock within the I frame in *G*. The authentication signals of *G* are composed of two types of features, as proposed in this study. The first is the *tree structured macroblock decomposition information* of a P frame in *G*, which can be used to detect spatial tempering. The second is the index of *G*, which can be used to detect temporal tampering.

In Chapter 2, we have reviewed the tree structured motion compensation technique used in the H.264 standard. The basic idea is that a macroblock in a P slice can be partitioned into sub-macroblocks and each of these sub-macroblocks is motion compensated individually. The way of partitioning the macroblock is usually adapted to the video content. Because different video contents result in different sub-macroblock partitions, the partition modes of the sub-macroblocks, called *tree structured macroblock decomposition information*, are suitable for use to generate authentication signals.

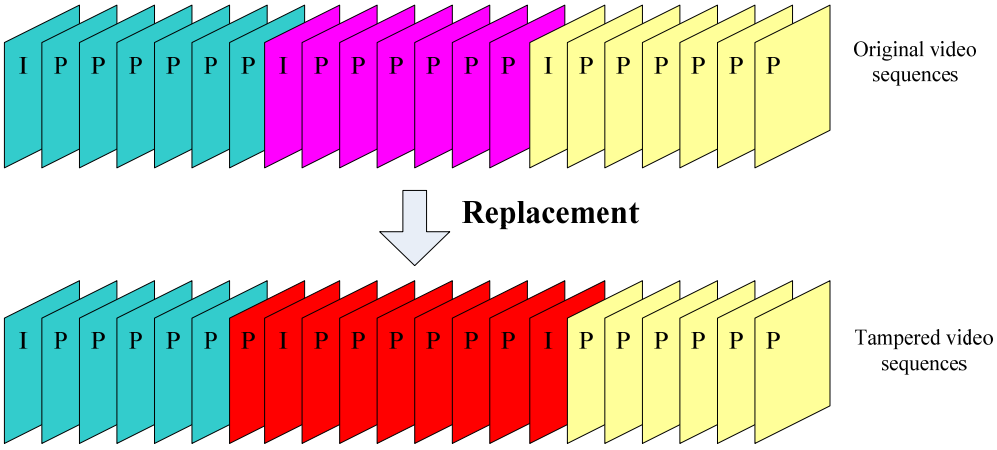


Figure 4.1 An illustration of replacement.

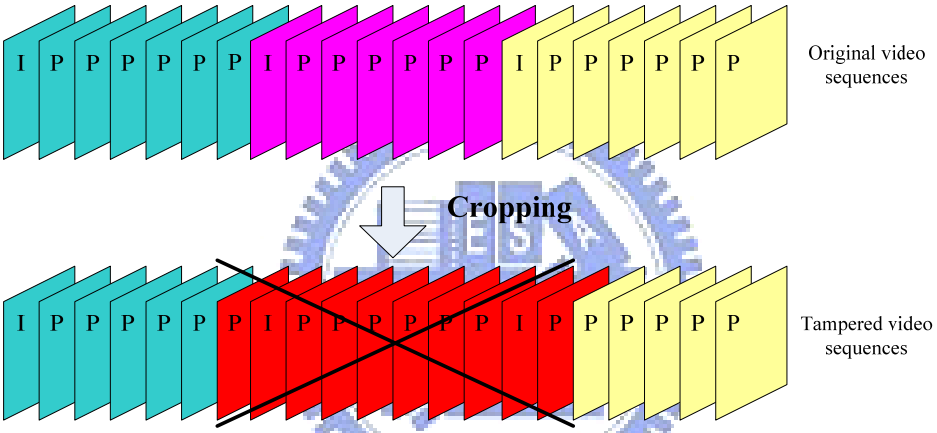


Figure 4.2 An illustration of cropping.

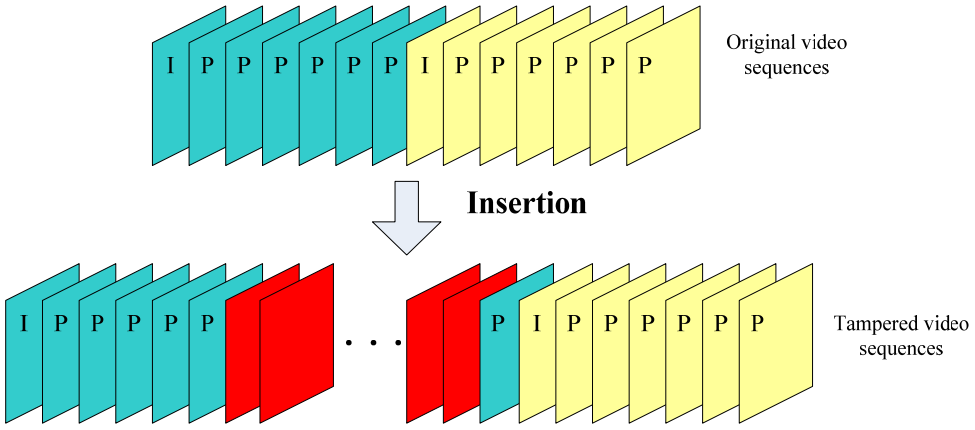


Figure 4.3 An illustration of insertion.

4.2 Generation of Authentication Signals

In this section, the proposed technique for composition of authentication signals is described. In Section 4.2.1, the principle is described first, and in Section 4.2.2, the proposed process for generation of authentication signals is presented.

4.2.1 Principle of Authentication Signal Generation

In this study, a frame group is treated as a *unit for authentication*. Therefore, each frame group G of an input video has its own authentication signals which comprise two parts. The first part is the index of G which is used to detect temporal tampering. The second part is tree structured macroblock decomposition information T of the motion regions of a P frame within G, which is selected randomly by a key. Since surveillance videos usually contain some suspicious activities in motion regions, we use the motion regions to generate the authentication signals. More specifically, we record T and quantify it to form a string I. Then, I together with the index of G comprise the authentication signals S_b. Finally, S_b is embedded into the I frame of G for the authentication use.

4.2.2 Process for Generation of Authentication Signals

In Chapter 2, we have reviewed the tree structured motion compensation technique of the H.264 standard. A 16×16 macroblock can be partitioned by one of the following ways: one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions, and four 8×8 partitions. If the 8×8 macroblock partition mode is chosen,

each of the four 8×8 sub-macroblocks in the macroblock may be further partitioned in four ways: one 8×8 sub-macroblock partition, two 8×4 sub-macroblock partitions, two 4×8 sub-macroblock partitions, and four 4×4 sub-macroblock partitions. Therefore, a macroblock partition mode and four sub-macroblock partition modes are used to describe the partition of a 16×16 macroblock in the H.264 standard.

Besides the index of a frame group G, we also need the tree structured macroblock decomposition information T of a P frame within G, which is selected randomly, to generate the authentication signals. Specifically, we select one of the P frames F_p in G and perform the motion detection algorithm introduced in Chapter 3 to F_p to obtain a set of the motion regions, R, in it. For each 16×16 macroblock M of each region R_i of R, denote its macroblock partition mode as P_m. If P_m is a large partition mode except for the 8×8 macroblock partition, we use the bit 0 to represent M. If P_m is the 8×8 macroblock partition and each sub-macroblock partition mode of M is the 8×4, 4×8 or 4×4 mode, we use the bit 1 to represent M. If P_m is the 8×8 macroblock partition mode and all the sub-macroblock partition modes of M are the 8×8 sub-macroblock partition modes, then we treat M as a special case which will be described later.

Next, we form a binary string S' with a pre-defined length L_T to represent the tree structured macroblock decomposition information of R_i by assigning each macroblock a representing bit in the above-mentioned way. We then combine S' with the binary form G_b of the index of G and the binary form R_{ib} of the coordinates of R_i to compose a binary string S_i with length L_R which is the sum of the value of L_T, the length of G_b and the length of R_{ib}. Moreover, we call S_i the *region signal* of R_i. Finally, we combine all region signals of R to produce the authentication signals S_b of G. The following algorithm describes the details of the above-mentioned process.

Algorithm 4.1. Process for generating authentication signals.

Input: a frame group G in a video, a secret key K, and a random number generator f.

Output: authentication signals S_b to be embedded.

Steps:

1. Use the input secret key K as a seed for f and use f to generate a sequence of random numbers, Q.
2. Select randomly a P frame F_p in G according to Q.
3. Perform the motion detection process (Algorithm 3.1) to F_p to obtain a set of the motion regions R in F_p.
4. For each region R_i within R, perform the following steps.
   - 4.1 For each 16×16 macroblock M in R_i, perform the following steps.
     - 4.1.1 Denote the macroblock partition mode of M as P_m and the sub-macroblock partition mode as P_s.
     - 4.1.2 If P_m is the 16×16, 16×8, or 8×16 mode, Mark M as a *large partition macroblock*.
     - 4.1.3 If P_m is the 8×8 mode and each P_s of M is 8×4, 4×8, or 4×4 mode, Mark M as a *small partition macroblock*.
     - 4.1.4 For the case that both P_m and Ps are the 8×8 mode, decide that M is a large partition macroblock or a small partition macroblock according to the following rules.
       - 4.1.4.1 Evaluate the *partition score* of M according to the following rules.
         - 4.1.4.1.1 Name the eight neighboring macroblocks as A through H, as depicted in Figure 4.4.
         - 4.1.4.1.2 Define the *macroblock gain* G_i for each of A through H in the following way.

(1) For *A*, *B*, *C*, and *D*, if P_i is the 8×8 mode, set the value of G_i to 1; otherwise, to 0.

(2) For *D*, *E*, *F*, and *H*, if P_i is the 8×8 mode, set the value of G_i to 0.5; otherwise, to 0.

4.1.4.1.3 Calculate the partition score according to the following equation:

$$partition\ score = \sum_{i=A}^{H} G_i \tag{4.1}$$

4.1.4.2 If the partition score is larger than a pre-defined threshold *T*, mark *M* as a large partition macroblock; otherwise, as small.

4.1.5 If *M* is a large partition macroblock, set *B(M)* to 1; otherwise, to 0.

4.2 For each R_i, select L_T 16×16 macroblocks M_1 through M_L, each denoted as M_i, and combine all $B(M_i)$ to form a binary string S', where L_T is a pre-defined length of signals. If the total number of macroblocks in R_i is smaller than L_T, allow repetition of using macroblocks in R_i.

4.3 Transform the coordinate information of R_i into the binary form and combine it with S' to form a new binary string S_i.

5. Combine the string S_i of every region R_i within *R* to form the desired authentication signals S_b.

The meaning of the rules mentioned in Step 4.1.4 is explained here. The 8×8 partition mode is treated as a special case where the macroblock *M* can be either a large partition macroblock or a small partition macroblock, depending on the eight neighboring macroblocks. The basic idea for deciding this is that if most of the neighboring macroblocks are large partition cases, regard *M* as a large partition macroblock; otherwise, a small one. A partition score is calculated for *M* based on the

eight neighboring macroblocks to decide which case *M* belongs to. The macroblocks which are in direct contact with *M* (macroblocks *A*, *B*, *C*, and *D* in Figure 4.4) have much influence on *M* than macroblocks *E* through *H*.

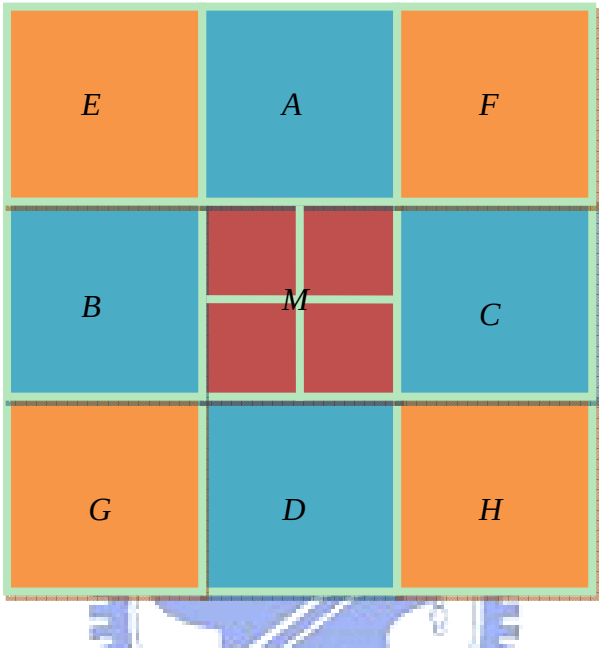


Figure 4.4 The notations of the eight neighboring macroblocks of *M*.

4.3 Embedding and Extracting of Authentication Signals in Surveillance Videos

In this section, the proposed methods of embedding and extracting authentication signals are introduced. In Section 4.3.1, the proposed technique of embedding authentication signals is described, and in Section 4.3.2 the proposed technique of extracting authentication signals is presented.

4.3.1 Embedding of Authentication Signals

In this section, the proposed technique of embedding authentication signals is described. In Section 4.3.1.1, the proposed idea is presented. In Section 4.3.1.2, the detail steps of the embedding process are described.

4.3.1.1 Proposed Idea

We divide a video into several frame groups, and each of them is treated as a unit of authentication, as mentioned previously. After generating the authentication signals S_b for each group G, we embed S_b into the only I frame in G for authentication use, resulting in a *protected video*.

A protected video V_p might be tampered with and recompressed by a malicious user. Therefore, if the method used for embedding authentication signals is not robust enough to be recompression-resilient, the authentication signals hidden in V_p may get lost and the protected video will not be authenticable any more. As a result, the data hiding method applied to the authentication signal embedding process needs to be robust with respect to H.264 recompression, so the robust data hiding method introduced in Chapter 3 is utilized in the proposed embedding process described next.

4.3.1.2 Process for Embedding Authentication Signals in I Frames

After obtaining the authentication signals S_b, we duplicate S_b into several copies, where the total length of these copies is set smaller than the capacity of an I frame. The main purpose of this duplication process is to facilitate extracting authentication signals more precisely using a *voting technique* in the later authentication process in

order to reduce the probability of misrepresentation. Then, the signals are embedded into the I frame using the secret-key-based data hiding method introduced previously in Chapter 3. The details are described as an algorithm as follows.

Algorithm 4.2. Process for embedding authentication signals.

Input: authentication signals S_b, an I frame F, a secret key K, and a random number generator f.

Output: a protected I frame F'.

Steps:

1. Denote the length of S_b as $L(S_b)$. Duplicate S_b k times and concatenate them in order to form a new binary string S_b', where k is such that $L(S_b)\times k$ is smaller than the capacity of an I frame.
2. For each 16×16 macroblock M in F, perform the following steps before M is encoded.
   2.1 Take out the first consecutive 16 bits of S_b', which have not been hidden, and denote these data bits as S_{b16}'.
   2.2 Take K, S_{b16}', M, and f as the input to the data hiding method (Algorithm 3.2) introduced in Chapter 3, and perform the data hiding process.
3. Repeat Step 2 until all macroblocks in F are processed.

An example of embedding authentication signals is illustrated in Figure 4.5. Figures 4.5(a) through 4.5(c) comprise the first frame group G_1 in an input video. Figures 4.5(a) and 4.5(b) are two P frames of G_1, and Figure 4.5(c) is the I frame of G_1. We selected one frame F from the first P frame in Figures 4.5(a) or the second P frame 4.5(b) to construct the authentication signals of G_1. More specifically, there are two motion regions, R_1 and R_2, within G_1, and the region signals of R_1 and R_2 are

combined to comprise the authentication signals of G_1 and embedded into the I frame of G_1. Figures 4.5(d) through 4.5(f) comprise the second frame group G_2 in the video, and the embedding process for is the same as for G_1. A comparison between the original I frame and the stego-I frame is illustrated in Figure 4.6. Figure 4.6(a) is the original I frame and Figure 4.6(b) is the stego-I frame. The comparison shows that the data hiding process does not result in many noises in the stego-I frame.

4.3.2 Extraction of Authentication Signals

In this section, the proposed technique of extracting authentication signals is described. In Section 4.3.2.1, the proposed idea is presented, and in Section 4.3.2.2, the detail steps of the extraction process are described.

4.3.2.1 Proposed Idea

If a protected video is re-encoded, the original DCT coefficients may be slightly changed. Some of the authentication signals hidden in the video may also be changed due to the recompression process. For this reason, we duplicate the signals several times and embed all of them, as mentioned previously. We extract them in the data extraction process by the voting technique in order to increase the precision of the extracted authentication signals. Furthermore, if the protected video is tampered with, we can still extract the correct signals while the non-suspicious area is larger than the suspicious area in an I frame spatially.

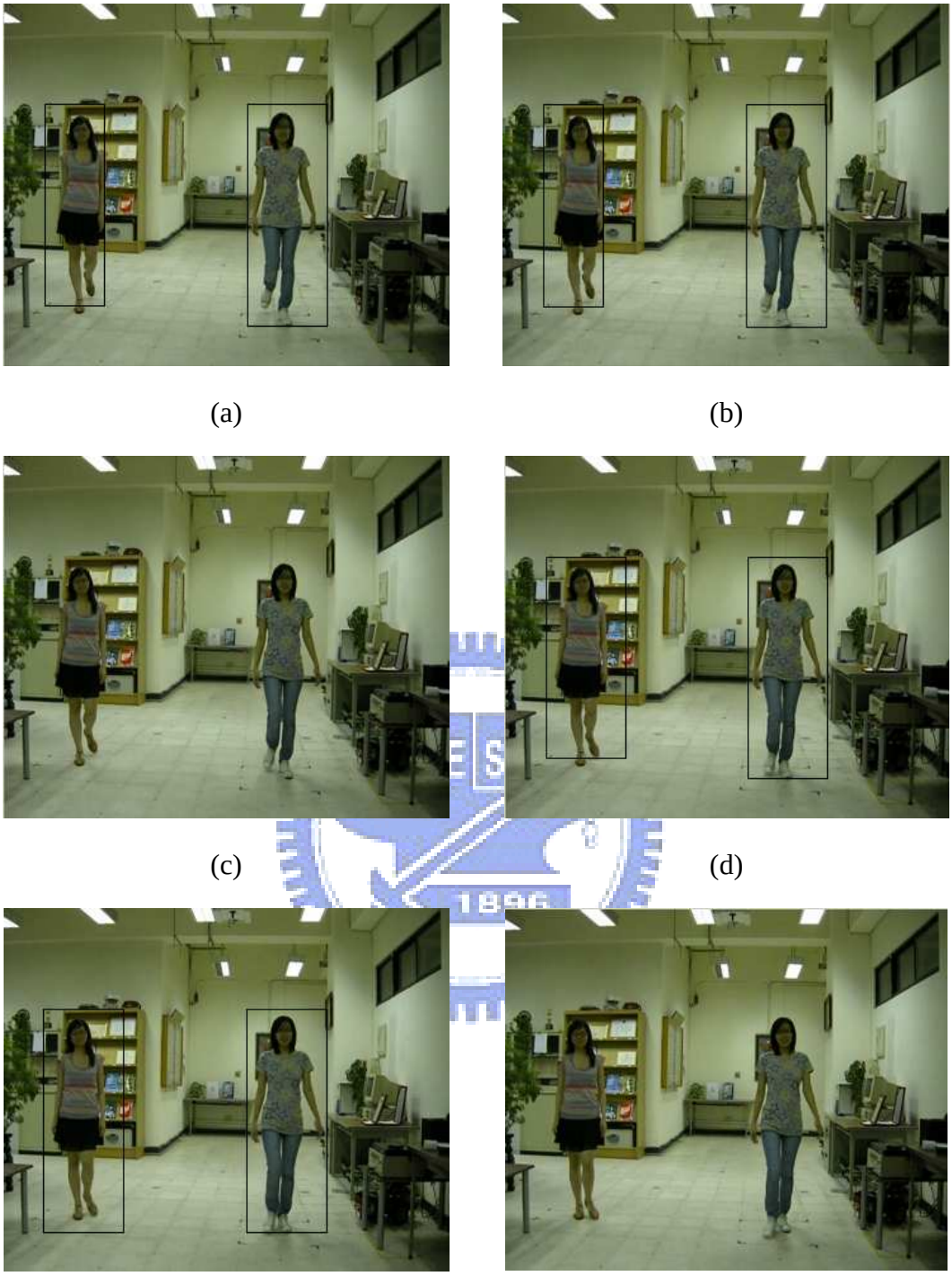


Figure 4.5 An example of embedding authentication signals. The region signals of one of the two P frames form authentication signals, and the authentication signals are hidden into the following I frame. (a) The first P frame of the first frame group. (b) The second P frame of the first frame group. (c) The I frame of the first frame group. (d) The first P frame of the second frame group. (e) The second P frame of the second frame group. (f) The I frame of the second frame group.

(a) (b)

Figure 4.6 A comparison between the original I frame and the stego-I frame. (a) The original I frame. (b) The stego-I frame.

4.3.2.2 Process for Extracting Authentication Signals by Voting Technique

We extract signals hidden in a video by the data extracting method mentioned previously in Chapter 3, and get authentication signals based on the use of the voting technique proposed later. In Section 4.2, we have introduced the process for generation of authentication signals. For each frame group G, the signals are produced based on the motion detection result of the P frame in G. Since there may be several detected motion regions and since a specific segment of authentication signals in the authentication signals S_b is produced for each region, the length of S_b is *not fixed* for every frame group in the video. As a result, the length L_s of S_b needs to be decided first, so that the voting process can be performed based on L_s.

In the proposed voting process, each bit of the extracted data may be either of the two possible values, 0 and 1, so every bit B_i of S_b is associated with two scores: Score-0 and Score-1. If the value of B_i is 0, one vote is added to Score-0; otherwise, to Score-1. Then the value with the higher vote score will be regarded as the correct value of B_i.

Algorithm 4.3. Process for extracting authentication signals.

Input: a protected I frame F, a secret key K, and a random number generator f.

Output: authentication signals S_b.

Steps:

1. For each macroblock M_i of F, take M_i, K, and f as the input to the data extraction method (Algorithm 3.3) to get the hidden data D_i of M_i.
2. Combine all D_i to form a binary string S.
3. Perform the following steps on S to get the authentication signals S_b.
   - 3.1 Denote the length of S as L.
   - 3.2 Divide L into segments of lengths L_R, and denote the number of segments as T, where L_R is the length of a region signal as mentioned in Section 4.2.2.
   - 3.3 Generate T candidate authentication signals according to the following steps and denote them as S_1 through S_T.
     - 3.3.1 Denote the currently-processed candidate authentication signal as S_j, where $1 \le j \le T$.
     - 3.3.2 Divide S into several segments of signals, with each of them being of the length $L_R \times j$.
     - 3.3.3 Transform each segment S' of S into the binary form as $S' = b_1 b_2 b_3 \ldots b_l$, where l is the length of S'. Associate each bit of S' with two vote scores $V_0[m]$ and $V_1[m]$, where $1 \le m \le l$. Calculate the score of each bit of the candidate authentication signal S_j to be constructed in the next step according to the following rule, where $1 \le j \le T$:

$$\begin{aligned} &\text{if } b_m = 0, \text{ then set } V_0[m] = V_0[m] + 1; \\ &\text{if } b_m = 1, \text{ then set } V_1[m] = V_1[m] + 1, \end{aligned} \tag{4.2}$$

       where $1 \le m \le l$.

     - 3.3.4 Denote the binary form of S_j as $S_j = s_1 s_2 s_3 \ldots s_l$. Construct S_j by

comparing the two scores of each bit of S' according to the following rule:

$$\begin{aligned} &\text{if } V_0[m] > V_1[m], \text{ then set } s_m=0; \\ &\text{if } V_1[m] > V_0[m], \text{ then set } s_m=1, \end{aligned} \tag{4.3}$$

where $1 \le m \le l$.

3.3.5 Calculate the *distribution rate* p_m of each bit s_m of S_j by the following rule:

$$\begin{aligned} &\text{if } s_m=0, \ \ p_m=\frac{V_0[m]}{(V_0[m]+V_1[m])}; \\ &\text{if } s_m=1, \ \ p_m=\frac{V_1[m]}{(V_0[m]+V_1[m])}, \end{aligned} \tag{4.4}$$

where $1 \le m \le l$.

3.3.6 Calculate the *average distribution rate* P_j of S_j by the following rule:

$$P_j=\frac{\sum_{m=1}^{l} p_m}{l} \tag{4.5}$$

where l is the length of S_j, and $1 \le j \le T$.

3.4 Select one candidate from S_1 through S_T, which has the highest average distribution rate, as the output authentication signals S_b.

In the above process for extracting authentication signals, we first divide the extracted signals S by L_R in order to know how many region signals S can hold, and the division result is denoted as T. Since we do not know how many region signals comprise the desired authentication signals, we check each possible number N of region signals, where N cannot be greater than T, and construct the corresponding candidate authentication signals S_j by the voting technique, where $1 \le j \le T$. Based on the voting result which yields S_j, we calculate the distribution rate of each bit of S_j. The candidate with the highest average distribution rate of all bits is recognized as the desired authentication signals.

4.4 Authentication of Surveillance Videos

In this section, the proposed detection and verification techniques for spatial and temporal tamperings are introduced. In Section 4.4.1, the process for detection and verification of spatial tampering is described. And the process for detection and verification of temporal tampering is presented in Section 4.4.2.

4.4.1 Detection and Verification of Spatial Tampering

A frame group *G* is treated as a unit of authentication. The authentication signals for *G* are embedded in the I frame of *G*. The first step of authentication of spatial tampering is to perform authentication on the I frame. If any region in the I frame is marked as a suspected region, *G* is also marked as a suspected frame group. For a suspected frame group *G′*, we perform authentication on P frames of *G′* to get more information about the tampering, which will be described later.

In Section 4.4.1.1, the proposed process for authentication of I frames is described. The proposed process for authentication of P frames is described in Section 4.4.1.2.

4.4.1.1 Authentication of I frames

Because usually most frames of a surveillance video are still background without moving objects, a malicious user may try to cover suspicious activities by the background image. They may cut some regions *R* from the background image, and replace the regions containing suspicious activities in other frames with *R*. While the

area which is tampered with is usually smaller than the area which is not tampered with, we can extract the correct authentication signals by the previously-mentioned voting technique, and use the signals to detect and verify the area which is tampered with within the corresponding I frame.

Algorithm 4.4. Process for spatial tampering detection in an I frame.

Input: a protected I frame F, authentication signals S extracted from F, a secret key K, and a random number generator f.

Output: an authenticated I frame F'.

Steps:

1. For each 16×16 macroblock M in F, take M, K, and f as input to the data extraction method (Algorithm 3.3) to get the hidden data D in M.
2. Denote the sixteen 4×4 sub-macroblocks in M as M_1 through M_{16}, say M_i, and the corresponding hidden bit of M_i as D_i.
3. Denote the number of suspected 4×4 sub-macroblocks in M as $N_{4\times4}$.
4. Compare D_i with the corresponding bit s_i in S to determine whether M_i is suspicious or not. If D_i is not equal to s_i, regard M_i to be suspected.
5. If M_i is a suspected sub-macroblock, set $N_{4\times4}= N_{4\times4}+1$.
6. After processing each M_i, name the eight neighboring macroblocks of M as A through H, as depicted in Figure 4.4., and verify each macroblock M according to the following rules:
    (1) If $N_{4\times4}$ is larger than 5, regard M to be *content-unauthentic*.
    (2) If $N_{4\times4}$ is larger than 3 and one of the neighboring macroblocks, A through D, is content unauthentic, regard M to be *neighbor-unauthentic*.
    (3) If $N_{4\times4}$ is larger than 0 and one of the neighboring macroblocks, A

through *H*, is content unauthentic, regard *M* to be *neighbor-unauthentic*.

7. Mark all the content-unauthentic and neighbor-unauthentic macroblocks as suspected regions.

In the above proposed algorithm, $N_{4\times4}$ is the number of suspected 4×4 sub-macroblocks in a macroblock *M*. Therefore, an $N_{4\times4}$ of a macroblock *M*, which is larger than a pre-defined threshold, means that most video contents within *M* are attacked, so that *M* is regarded to be *content-unauthentic*. If *M* is not content-unauthentic, but one of the eight neighboring macroblocks of *M* is content-unauthentic, then we decide that if *M* is *neighbor-unauthentic* according to the rules (2) and (3) in Step 6 of the proposed algorithm.

4.4.1.2 Authentication of P frames

During the process for generation of authentication signals of a frame group *G*, the authentication signals are composed of region signals in *G*. These region signals are formed based on the tree structured macroblock decomposition information of the corresponding motion regions in *G*; therefore, the region signals contain some information about the moving objects.

If a frame group *G′* is marked as a suspected frame group, *G′* is decided to be tampered with, and there may be some moving objects missing in *G′*. Hence, we can extract the tree structured macroblock decomposition information of the motion regions from the authentication signals to get more information about the missing moving objects.

Each region signal is composed of the index of *G*, the coordinates of the

corresponding motion region R, and the tree structured macroblock decomposition information of R. In the proposed process of authentication of the P frames in a frame group G, we first divide the authentication signals of G into several region signals with length L_R. A region signal represents a motion region. For each motion region R in G, the corresponding region signal is analyzed to get the coordinates R_c of R, and the tree structured macroblock decomposition information R_T of R. R_c is used to locate the motion region, and R_T is used to describe the information of the missing moving objects in R.

Algorithm 4.5: Process for detection of spatial tampering of P frames in a frame group.

Input: a frame group G and authentication signals S of G.

Output: authenticated P frames with information of missing moving objects.

Steps:

1. Divided S into several region signals with length L_R, where L_R is the length of a region signal. Denote these region signals as S_1 through S_N, and the corresponding regions as R_1 through R_N, where N is the number of region signals.
2. For each P frame F in G, perform the following steps.
   2.1 For each region signal S_i, where $1 \le i \le N$, perform the following steps.
      2.1.1 Transform the binary form of the coordinate information in S_i back to decimal numbers. Denote the decimal form of coordinate information as R_c.
      2.1.2 Extract the tree structured macroblock decomposition information from S_i, and denote it as R_T.
      2.1.3 Use R_c to locate the corresponding motion region R_i in F, where $1 \le i \le N$. Denote the corresponding rectangle as *Rec*.

2.1.4 Denote R_T as $R_T = r_1 r_2 r_3 \ldots r_l$, where l is a pre-defined length of the tree structured macroblock decomposition information part in a region signal.

2.1.5 For each bit r_j of R_T, if $r_j = 0$, mark the corresponding macroblock M in R_i as a small partition macroblock. Otherwise, mark M as a large partition macroblock, where $1 \leq j \leq l$.

2.1.6 Regard the partition information of macroblocks of R_i and the coordinate information of R_i as information of *missing* moving objects of R_i.

3. Output the authenticated P frames in G with information of missing moving objects.

Based on the information of missing objects, I, we can get more information about the missing objects that may be covered by a malicious user. The coordinate information part of I can locate which area in these P frames of the frame group may contain suspicious activities. Moreover, the partition information can describe moving objects appearing in the area in more details, such as the moving direction, the shape of a moving object, etc.

4.4.2 Detection and Verification of Temporal Tampering

In the proposed method, temporal tampering is categorized into three types: replacement, cropping, and insertion. We not only can detect three types of tampering, but also can detect the starting frame and the end frame for each type of tampering.

We utilize the extracted index I_i' of a frame group G_i obtained from the

corresponding authentication signals to verify the correctness of a video sequence. We compare I_i' with the index of G which is denoted as I_i to detect the temporal tampering.

Algorithm 4.5: Process for temporal tampering detection of a video sequence.

Input: a video sequence V, and authentication signals S of each frame group of V.

Output: a report R of the detection result.

Steps:

1. Denote the total number of frame groups in V as N, each frame group as G_i, and the index of each frame group as I_i, where $1 \le i \le N$.
2. For each frame group G_i in V, extract the index I_i' hidden in the corresponding authentication signals, where $1 \le i \le N$.
3. Create a flag bit B to indicate the occurrence of tampering, and initialize B to 0.
4. Create a flag bit F to indicate the occurrence of replacement, and initialize F to 0.
5. Subtract I_i from I_i', and denote the result as D_i.
6. If $D_i \neq 0$, perform the following steps.
   - 6.1 If B is equal to 0, set B to 1, and record the index n_s of the I frame in G_i.
   - 6.2 If B is equal to 1 and D_i is equal to D_{i-1}, set F to 1.
7. If $D_i = 0$, perform the following steps.
   - 7.1 If B is equal to 1, record the index n_f of the I frame in G_i, and perform the following steps.
     - 7.1.1 If F is equal to 1, decide the tampering type as replacement
     - 7.1.2 If F is equal to 0, decide the tampering type as cropping and insertion.
     - 7.1.3 Store the tampering type, n_s and n_f, into R.
     - 7.1.4 Set B, n_s, and n_f to 0.
8. Repeat Steps 5 through 7 for each frame group until reaching the end of V.

9. If B is equal to 1, perform the following steps.

9.1 If F is equal to 1, recognize the tampering type as replacement.

9.2 If F is equal to 0, perform the following steps.

9.2.1 If $D_N > 0$, decide the tampering type as cropping.

9.2.2 If $D_N < 0$, decide the tampering type as insertion.

9.2.3 Store the tampering type, n_s and the index of the last I frame of V into R.

The meanings of some of the steps of the algorithm are explained here. The basic idea of the above proposed method is to detect tampering based on the difference between the real index and the extracted index of a frame group. All the differences of the frame groups in a video sequence without temporal tampering should be zero. If a cropping operation occurs in frames before a frame group G, the difference of G is larger than zero. If an insertion operation occurs in frames before a frame group G, the difference of G is smaller than zero.

Once there is found a frame group G_i with non-zero difference D_i in Step 6, we mark G_i as the start n_s of the tampering. Then, if there is a frame group G_j after G_i in the video sequence and the difference D_j of G_j is zero, G_j is marked as the end n_f of the tampering in Step 7.1. For convenience of use, we call the tampering T.

Next, in Steps 7.1.1 through 7.1.4, we decide the tampering type of T based on the difference sequence of the frame groups between G_i and G_j. If the differences in the sequence are all equal, then the type of T is marked as cropping and insertion, which means there is a cropping operation which crops a number of frames as well as an insertion operation which inserts the same number of frames as the cropping operation. If the difference sequence includes non-consecutive numbers, then the frame groups between G_i and G_j are not the original frame groups of the video. Therefore, we mark the type of T as replacement.

If the frame group G_i with non-zero index difference is found, but the frame group G_j with zero index difference is not found even when reaching the end of the video, we decide the tampering type of T based on the following rules in Step 9. If the difference sequence includes non-consecutive numbers, the type is regarded as replacement. If the differences in the sequence are all equal and larger than zero, the type is regarded as cropping. Otherwise, the type is regarded as insertion.

4.5 Experimental Results

In our experiments, the size of each video frame is 352×288. The input video is a surveillance video of the Computer Vision Lab at National Chiao Tung University, where this study was conducted. In this video, a person wants to take a book on the table, and a malicious user try to cover this person and crops the part of the person in all frames of the input video. Each row of the figures in Figure 4.7 through Figure 4.10 is a frame group G_i of the input video. The left figure of the row is a representing P frame of G_i, and the right figure of the row is the I frame of G_i. Three consecutive frame groups of an original video are shown in Figure 4.7. Three consecutive frame groups of the protected video yielded by the proposed method are shown in Figure 4.8. Three consecutive frame groups of an attacked version of the video are shown in Figure 4.9. The malicious user crops the area containing the person in each frame and replaces it with the background image. The three corresponding consecutive frame groups of the authenticated video are shown in Figure 4.10.

In Figure 4.10, the green areas in the right figures are the suspicious areas of the I frames, which are attacked. The black rectangles in the left figures are the results of authentication on P frames. These rectangles reveal the information of the original video contents in the attacked areas, and the tree structured macroblock

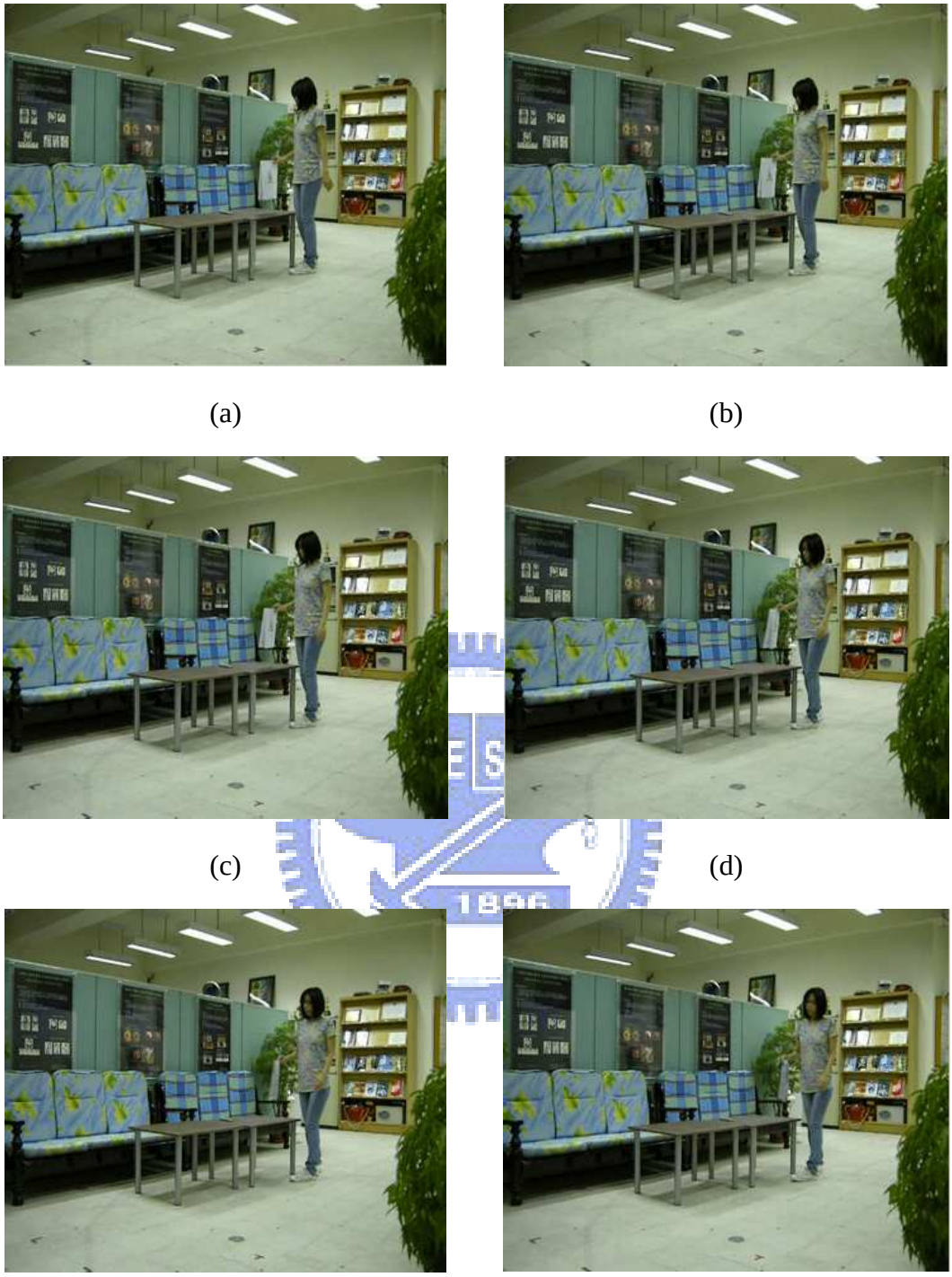
decomposition information of the contents. Based on the concept of tree structured motion compensation, the areas with small rectangles may contain some moving objects. The areas with small rectangles are distributed around the table. If we compare the areas with the background image, we may guess that the book on the table is moved by someone.

This experiment shows that the proposed authentication method not only can detect whether a video has been tampered with or not, but also can specify which part of the image frame is tampered with.

4.6 Discussions and Summary

In this chapter, we have proposed an authentication method that can detect and verify tamperings in a suspicious video. The proposed method uses the tree structured macroblock decomposition information in H.264 codes as authentication signals and embeds the authentication signals into the I frames of the input video. In order to extract the authentication signals more precisely, we use the voting technique to make sure we can still extract the correct signal while most regions of a suspicious frame are not tampered with. The correct signals can detect both temporal tampering and spatial tampering and verify the suspicious regions and frames.

Therefore, the proposed authentication system not only checks if a protected video has been tampered with or not, but also further shows where and how the tampering occurs.

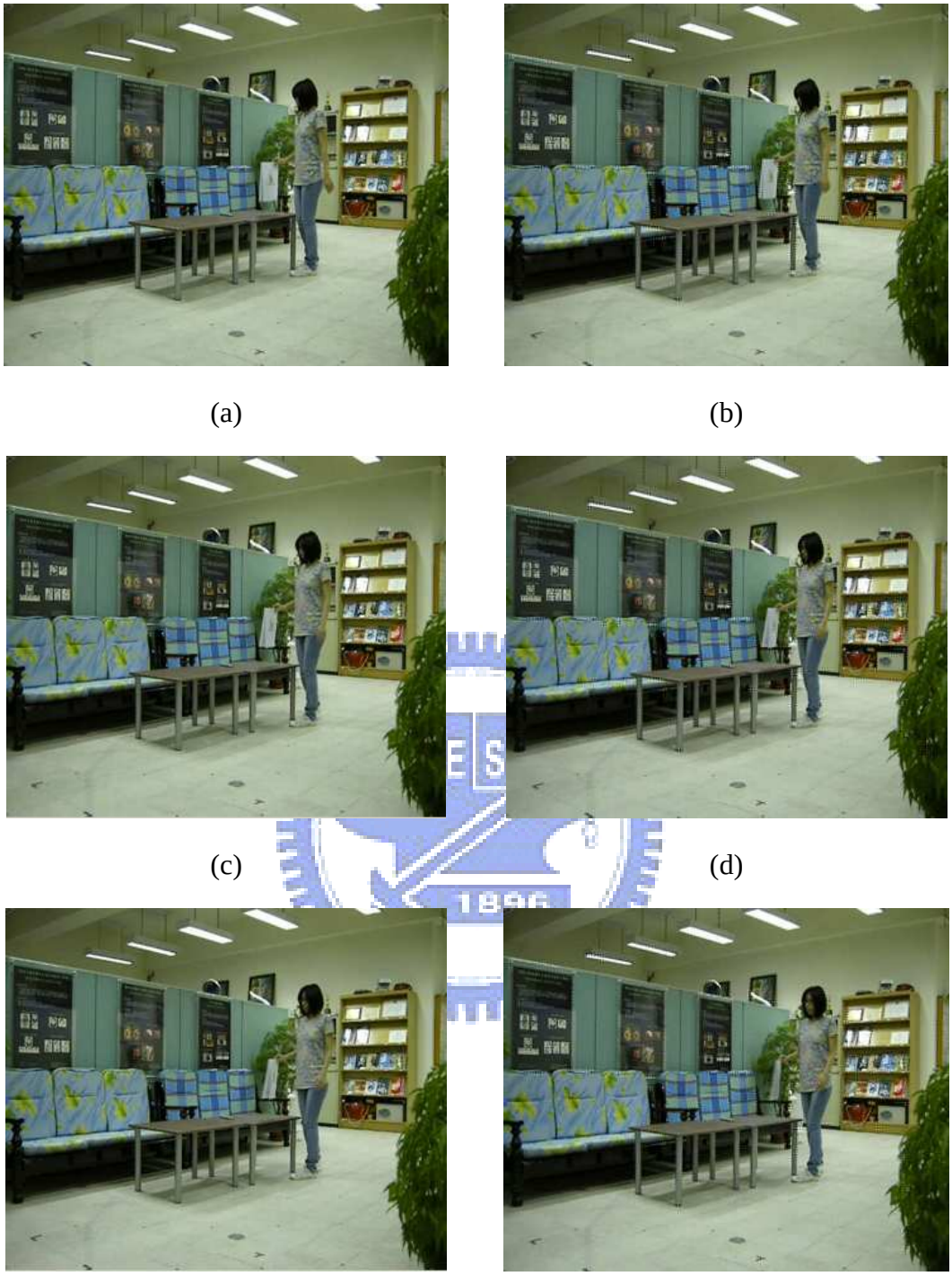


(a) (b)

(c) (d)

(e) (f)

Figure 4.7 Three consecutive frame groups of the original video. (a) A representing P frame of $G_{1.}$ (b) The I frame of G_1. (c) A representing P frame of $G_{2.}$ (d) The I frame of G_2. (e) A representing P frame of $G_{3.}$ (f) The I frame of G_3.

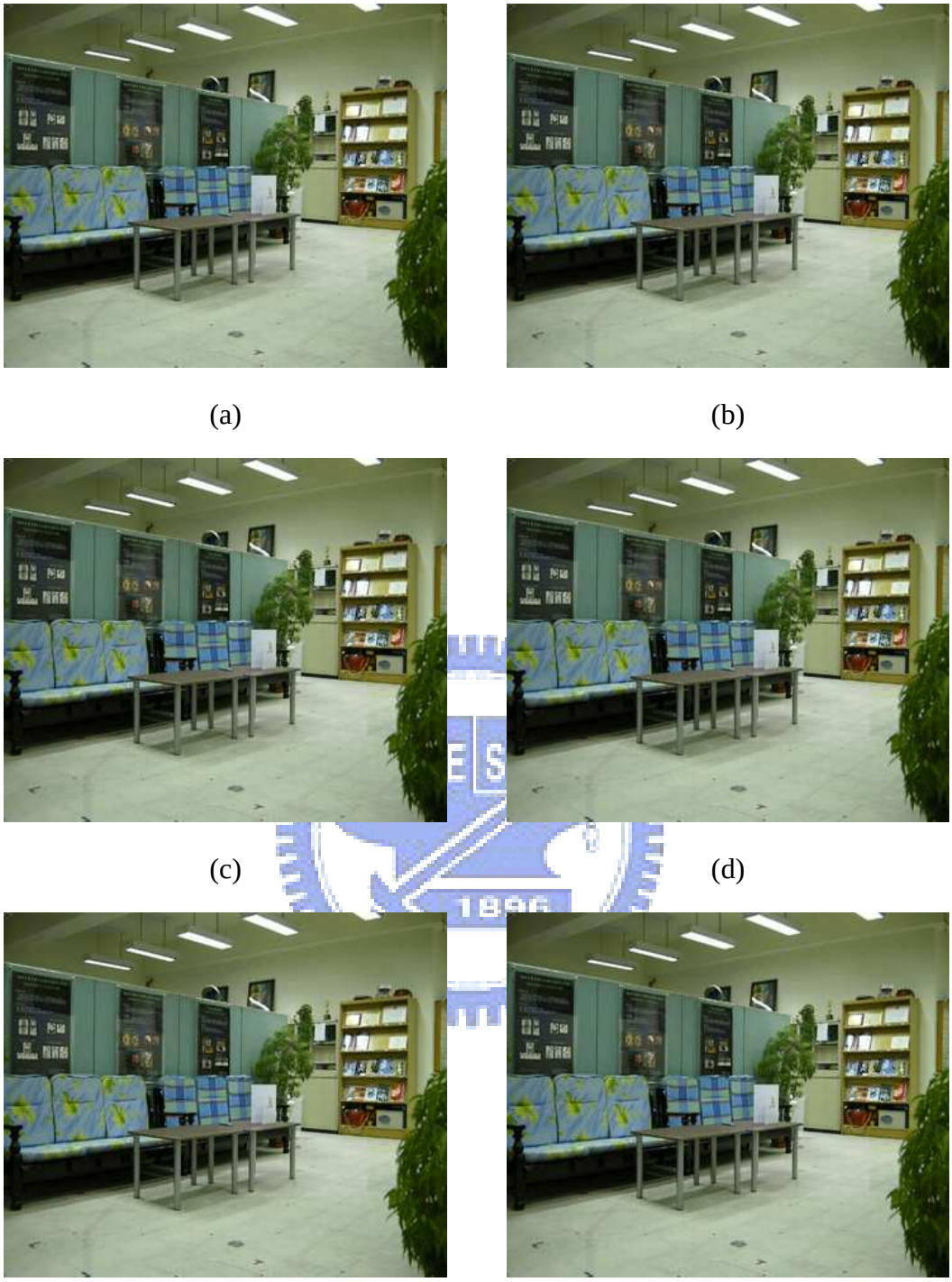


(a) (b)

(c) (d)

(e) (f)

Figure 4.8 Three consecutive frame groups of the protected video. (a) A representing P frame of $G_{1.}$ (b) The I frame of G_1. (c) A representing P frame of $G_{2.}$ (d) The I frame of G_2. (e) A representing P frame of $G_{3.}$ (f) The I frame of G_3.

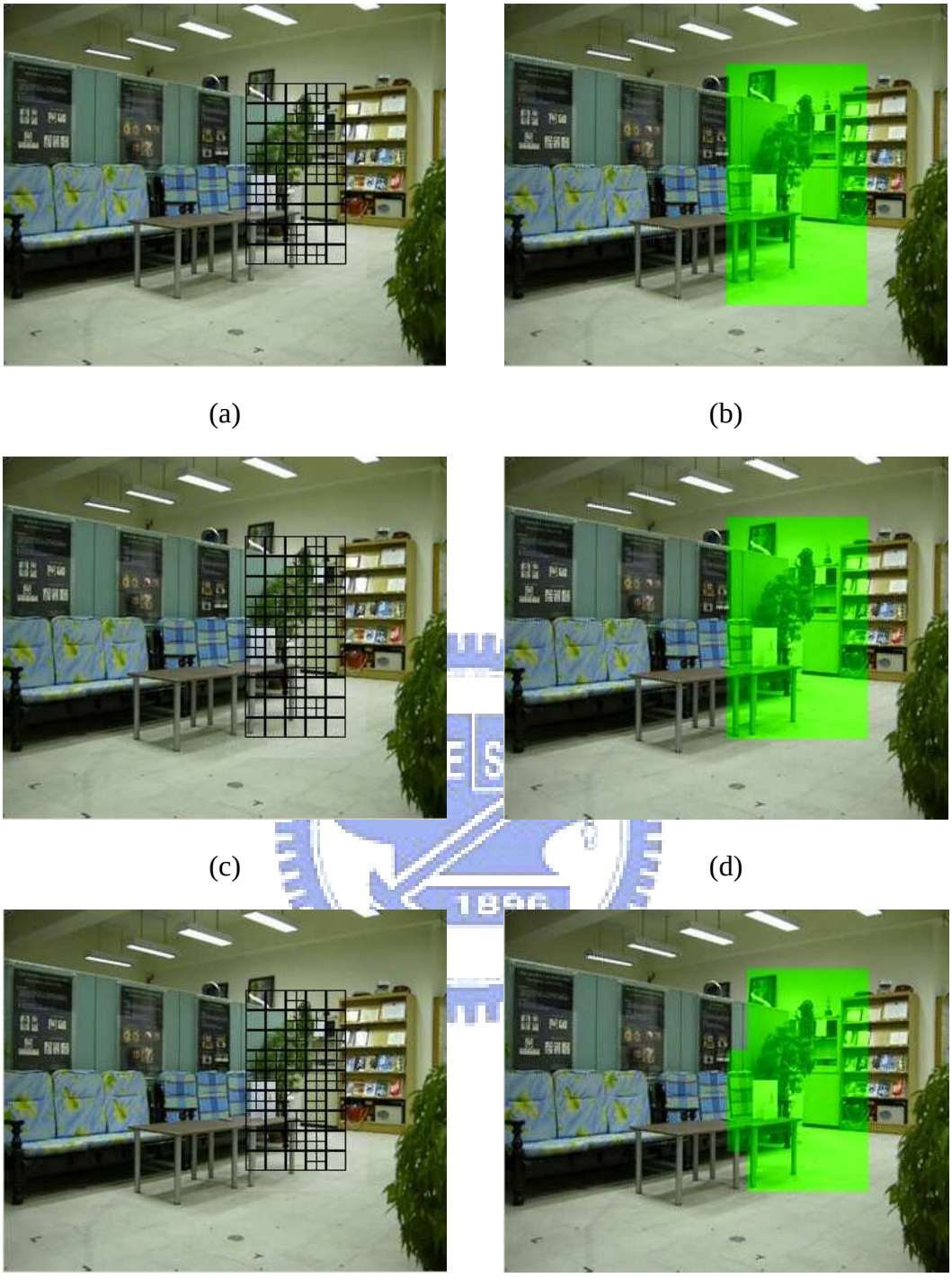


(a) (b)

(c) (d)

(e) (f)

Figure 4.9 Three consecutive frame groups of the tampered video. (a) A representing P frame of $G_{1.}$ (b) The I frame of G_1. (c) A representing P frame of $G_{2.}$ (d) The I frame of G_2. (e) A representing P frame of $G_{3.}$ (f) The I frame of G_3.



(a) (b)

(c) (d)

(e) (f)

Figure 4.10 Three consecutive frame groups of the authenticated video. The green areas in the right figures are suspicious areas of the I frame. The black rectangles in the left figures are the tree structured macroblock decomposition information of the suspicious areas. (a) A representing P frame of G_1. (b) The I frame of G_1. (c) A representing P frame of G_2. (d) The I frame of G_2. (e) A representing P frame of G_3. (f) The I frame of G_3.

Chapter 5
Protection of Personal Privacy in Surveillance Videos

5.1 Introduction

Surveillance systems rise along with the development of society, so lots of issues have to be considered. Privacy protection is one of these issues in video surveillance. Since a video surveillance system usually monitors a public space for long periods of time, it may possibly record some information which violates personal privacy. Therefore, we propose a method for privacy protection to solve this issue and the method is described in this chapter.

In Section 5.1.1, the related problem definitions are given. In Section 5.1.2, the idea of the proposed method is described. In Section 5.2, the proposed process for embedding decoding information into videos is presented. In Section 5.3, the proposed process for extracting decoding information from videos is presented. Some experimental results are shown in the Section 5.4. Finally, some discussions and a summary will be given in the last section of this chapter.

5.1.1 Problem Definition

In the privacy protection problem dealt with in this study, an authorized user can specify a protected region R in an input video. The video contents in R then are removed and replaced with the background image in order not to reveal sensitive

privacy information in *R*. Also, the privacy information of *R* is hidden into the video to produce a *privacy video*. Thereafter, once the privacy information needs to be recovered, the data hidden in the privacy video is extracted and used to recover *R*.

Two main issues are involved in this problem. The first is how to replace the video contents in *R* with the background image and to embed the information about the contents in *R* into the video. The second is how to extract the data from the privacy video and to recover the original contents of the protected region.

5.1.2 Proposed Idea

A video can be decoded correctly based on the decoding information generated during the encoding process. Therefore, in order to remove sensitive video contents of a region *R*, which is specified by an authorized user, in an input video, we set the decoding information of *R* to some pre-defined values, so that the video contents are removed and replaced with the background image. The decoding information of *R* is then hidden into the input video. If the video contents of *R* need to be recovered, the decoding information of *R* hidden in the video is extracted and used to recover the contents of *R*.

5.2 Hiding of Privacy Information

In this section, the proposed process for hiding privacy information is introduced. In Section 5.2.1, the proposed idea of the process is stated. In Section 5.2.2, the proposed process for hiding privacy information is described.

5.2.1 Proposed Idea

In Chapter 2, we have reviewed the concept of motion compensation. Motion compensation is the process of finding the best prediction block in inter mode. A *motion vector* is used to indicate the location of the best prediction block. The difference between the best prediction block and the currently-processed block is DCT-based transformed into a set of *frequency coefficients*. Motion vectors and frequency coefficients are used in the decoding process to decode the corresponding block.

A P frame can be decoded correctly based on correct decoding information which includes motion vectors, frequency coefficients, partition modes, etc. In order to remove the privacy information in the user-specified region R and replace the privacy information with the background image, we first use the proposed motion detection method to detect if there are any activities in R. If any motion region is detected in R, we denote the resulting motion region as a *replaced region*. We modify motion vectors of the replaced region R' in order to change the original prediction blocks into the corresponding blocks in the background image, and set all frequency coefficients of R' to zero. Therefore, the video contents in R' turn into the corresponding part of the background image. The original motion vectors and frequency coefficients of R' are embedded into the input video for recovery use.

5.2.2 Process for Hiding Privacy Information

The proposed process is applied on P frames of input H.264 videos. The proposed motion detection method introduced in Chapter 3 is applied on the P frame to detect motions in a user-specified region R and get the replaced region R'. When encoding macroblocks within R', the motion vectors and frequency coefficients of the

currently-processed macroblock M is all set to zero. Therefore, the video contents of R' become the corresponding part of the background image which has appeared in the previous frames of the input video. It also places a restriction on this proposed process that the first frame of the input video must be a background frame.

The values of the original motion vectors and frequency coefficients of macroblocks of R' are then hidden into the remaining region of the P frame. We use a secret key to randomize the hiding order of macroblocks for the security protection purpose.

In Chapter 2, we have reviewed the process for encoding an H.264 video. In the prediction procedure of an H.264 encoding process, all sample values of a prediction block are computed by those of previously encoded and reconstructed blocks. Therefore, if we modify the motion vectors and frequency coefficients of R' during a traditional encoding process, then it will cause prediction errors on macroblocks which have referenced the macroblocks in R'. In more details, assume that some macroblocks M of the following frames have referenced the modified macroblocks in R'. Once the video contents within R' are recovered, then the prediction blocks of M used in the decoding process will be different from the prediction blocks computed in the previous encoding process. It causes decoding errors on M. Therefore, we introduce the use of *multiple slice groups* to solve this problem. The details are described in the following algorithm.

Algorithm 5.1. Process for removing and hiding privacy information with a user-specified region.

Input: an H.264 video V, a secret key K, a random number generator f, and a region R specified by an authorized user.

Output: an H.264 video V' with privacy information in R removed and hidden.

Steps:

1. Use explicit mapping, which is Type 6 of multiple slice groups maps to set the slice group number N_{ij} of each macroblock M_{ij} of each frame of the input H.264 video V according to the following rule:

$$\begin{aligned} &\text{if } M_{ij} \text{ is in } R, \text{ set } N_{ij} = 0; \\ &\text{otherwise}, \quad \text{set } N_{ij} = 1. \end{aligned} \tag{5.1}$$

2. For each P frame F of V, perform the following steps.

   2.1 Take the currently-processed P frame F as input to the proposed motion detection algorithm (Algorithm 3.1), denote the resulting motion region as R', and regard R' as a replaced region for removing privacy information.

   2.2 For each macroblock M_{ij} in R' of F, if the corresponding N_{ij} is equal to 0, store the motion vector and the frequency coefficients of M_{ij} in a report E and set the motion vector and frequency coefficients of M_{ij} to zero.

   2.3 Denote the total number of macroblocks in R' as N.

   2.4 Use the input secret key K as a seed for f and use f to generate a sequence of random numbers $Q = \{i_1, i_2, \ldots, i_N\}$ in the range of $\{1, 2, \ldots, N\}$ without repetitive values.

   2.5 For each number in Q, get the motion vector and the DC coefficient of the frequency coefficients of the corresponding macroblock in R', and transform them into a binary string S_k.

   2.6 Combine all S_k and the binary form of the coordinate information of R' to form a binary string S, and denote it as $S = s_1 s_2 s_3 \ldots s_L$, where L is the length of S.

   2.7 For each macroblock M_{ij} of F, if the corresponding N_{ij} is equal to 1, perform the following steps.

      2.7.1 For each 4×4 sub-macroblock M_4 of M_{ij}, denote the corresponding

frequency coefficients as *Coeff*.

2.7.2 Modify *Coeff* in order to hide an un-hidden bit B of S according to the following rules.

2.7.2.1 Select the coefficient pair $C_1(0, 3)$ and $C_2(3, 0)$ in *Coeff*.

2.7.2.2 Modify C_1 and C_2 according to the following equations.

(1) if $B = 0$:

$$if\ C_1 > C_2,\ swap\ C_2\ and\ C_1; \tag{5.2}$$

(2) if $B = 1$:

$$\begin{aligned} &if\ C_2 = C_1,\ C_1 = C_2 + T; \\ &if\ C_2 > C_1,\ swap\ C_2\ and\ C_1, \end{aligned} \tag{5.3}$$

where T is a pre-defined threshold.

2.8 Repeat Step 2.6 until reaching the end of S or the last macroblock of F.

3. Repeat Step 2 until reaching the end of V.

We have mentioned the concept “slice” in Chapter 2. A slice is composed of macroblocks. During an H.264 encoding process, macroblocks are predicted from samples in the same slice. In other words, macroblocks in different slice will not refer to each other. It also implies that an H.264 encoder processes the next slice until all macroblocks in the currently-processed slice are encoded.

We solve the prediction problem by the use of multiple slice groups, which are introduced in the H.264 standard. A slice group may contain one or more slices. Multiple slice groups define a number of flexible ways to map coded macroblocks to slices groups. There are totally seven types of multiple slice groups maps. The first six types are illustrated in Figure 5.1. The last type called *explicit mapping* is entirely user-defined.

Since macroblocks in different slices will not refer to each other, macroblocks in

different slice groups will not refer to each other, either. Therefore, we can solve the prediction error problem by using the explicit mapping to set the user-specified region and the remaining region in different slice groups. Then, the decoding of macroblocks of the remaining region will not be affected by the modified macroblocks of the user-specified region.

Another benefit that the multiple slice groups bring about is that we can control the encoding order of each slice groups by setting the slice group identifier. As a consequence, the process of removing privacy information and the process of embedding privacy information can be done in the mean time. In other words, we do not have to perform the encoding process two times. Without multiple slice groups, the macroblocks are encoded in a raster scan order. Then, we have to perform an encoding process to remove the privacy information and get the decoding information, and then perform another encoding process to hide the decoding information into the video. In this situation, it results in another problem that the decoding information stored in the first encoding process may not be the same as the one generated in the second encoding process. The mismatching decoding information may result in decoding errors and cause the recovery process which will be introduced later to fail.

That is why the slice group identifier of R is set to 0, and the remaining region is set to 1 in Step 1 of the proposed algorithm. We can remove the privacy information in the encoding of the first slice group and hide it into the video in the second slice group during the same encoding process.

5.3 Recovery of Privacy Information

In this section, the proposed process for recovery of privacy information is introduced. In Section 5.3.1, the proposed idea is described, and the process for

recovery of privacy information is presented in Section 5.3.2.

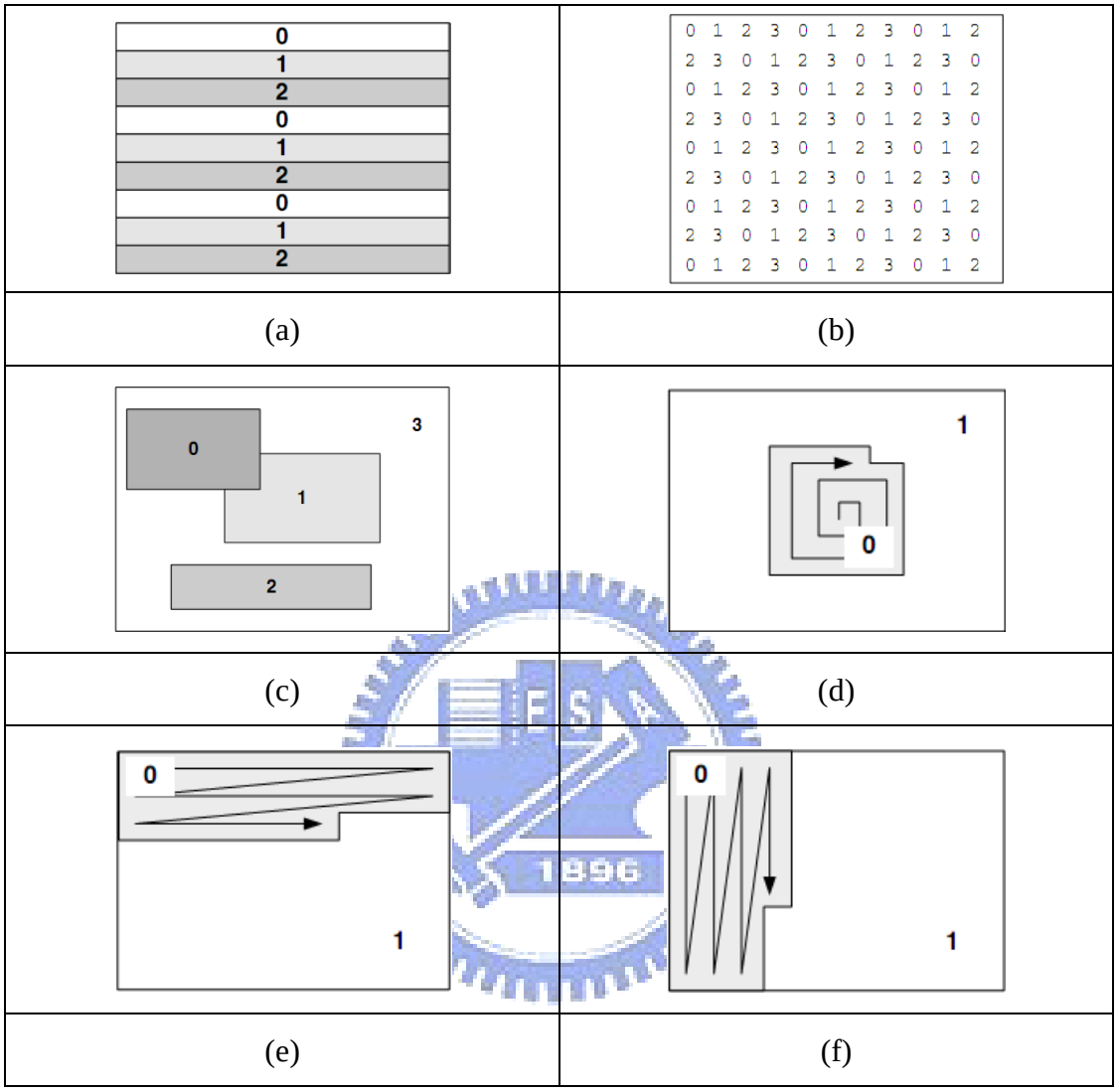


Figure 5.1 Types of multiple slice groups. The numbers in these figures are the slice group identifiers. There is another type, Type 6 - explicit mapping which is entirely user-defined. (a) Type 0 - interleaved mapping (three slice groups). (b) Type 1 - dispersed mapping (three slice groups). (c) Type 2 - foreground and background mapping (four slice groups). (d) Type 3 - box-out mapping (two slice groups). (e) Type 4 - raster mapping (two slice groups). (f) Type 5 - wipe mapping (two slice groups).

5.3.1 Proposed Idea

We use a secret key to extract the coordinate information, the motion vectors,

and the frequency coefficients of each macroblock of a replaced region R' from an input privacy video. Once the privacy information in R' needs to be recovered, the extracted information is used to recover the video contents in R'.

5.3.2 Process for Recovery of Privacy Information

In the proposed process for removing privacy information, we have mentioned that the replaced region and the remaining region are in different slice groups. We call the slice group of the replaced region *privacy slice*, and the slice group of the remaining region *remaining slice*. Because of the slice group identifier, during an H.264 decoding process, the privacy slice is decoded first and then the remaining slice.

Therefore, there are two phases in the proposed process for recovery of privacy information in an input video. The first is to extract the decoding information of the replaced region in the input video from the remaining slice. The second is to replace the decoding information of the replaced region, which is stored in the input video stream, with the extracted one.

Algorithm 5.2. Process for extraction of privacy information of an H.264 privacy video.

Input: an H.264 privacy video *V*.

Output: a report *E* of the privacy information of the replaced region in *V*.

Steps:

1. For each P frame *F* in *V*, perform the following steps.

   1.1 For each macroblock M_i in *F*, perform the following step.

      1.1.1 For each 4×4 sub-macroblock M_{ij} of M_i, perform the following steps.

1.1.1.1 Denote the frequency coefficients of M_{ij} as *Coeff*.

1.1.1.2 Extract the hidden bit *Bit*(*ij*) of M_{ij} from *Coeff* according to the following equation:

$$\begin{aligned} &if\ C_2 \geq C_1,\ set\ Bit(ij) = 0; \\ &else,\ set\ Bit(ij) = 1, \end{aligned} \tag{5.4}$$

where C_1 is the frequency coefficient $C_1(0, 3)$ and C_2 is $C_2(3, 0)$.

2. Combine all *Bit*(*ij*) to form a report of the privacy information of the replaced region in *V*.

Algorithm 5.3. Process for recovery of privacy information of an H.264 privacy video.

Input: an H.264 privacy video *V*, a report *E* of the privacy information of the replaced region in *V*, a secret key *K*, and a random number generator *f*.

Output: an H.264 video *V*′ with recovered privacy information.

Steps:

1. Extract the coordinate information of the replaced region *R*′ from *E*.
2. For each P frame *F* in *V*, perform the following steps.
   2.1 Locate the position of *R*′ in *F* based on the coordinate information, calculate the total number of macroblocks in *R*′, and denote the result as *N*.
   2.2 Use the input secret key *K* as a seed for *f* and use *f* to generate a sequence of random numbers, $Q = \{i_1, i_2, \ldots, i_N\}$, in the range of $\{1, 2, \ldots, N\}$ without repetitive values.
   2.3 Extract decoding information of macroblocks of *R*′ from *E*, which includes motion vectors and DC coefficients, and denote each set of a motion vector and a DC coefficient of a macroblock as D_i, where $1 \leq i \leq N$.
   2.4 For each set of decoding information D_i, perform the following steps.

2.4.1 Select a macroblock M of R' according to Q.

2.4.2 Replace the motion vector and the DC coefficient of M with D_i.

3. Repeat Step 2 until reaching the end of V.

In phase 1 of the proposed process of recovery of privacy information, Algorithm 5.2 is used to extract the decoding information including motion vectors and DC coefficients of the replaced region R' in the input video. In phase 2, Algorithm 5.3 is utilized to recover the privacy information of R' based on the extracted information in phase 1. In Step 2.2 of Algorithm 5.3, the secret key is used to generate a sequence of random numbers to map the extracted motion vector and the extracted DC coefficient to the corresponding macroblock in R'. Therefore, the privacy information of R' can be recovered.

Since we use the motion vector and DC coefficient of each macroblock to restore the original contents, some details described by the AC coefficients may be lost. Therefore, it may result in some distortion in the recovered video.

5.4 Experimental Results

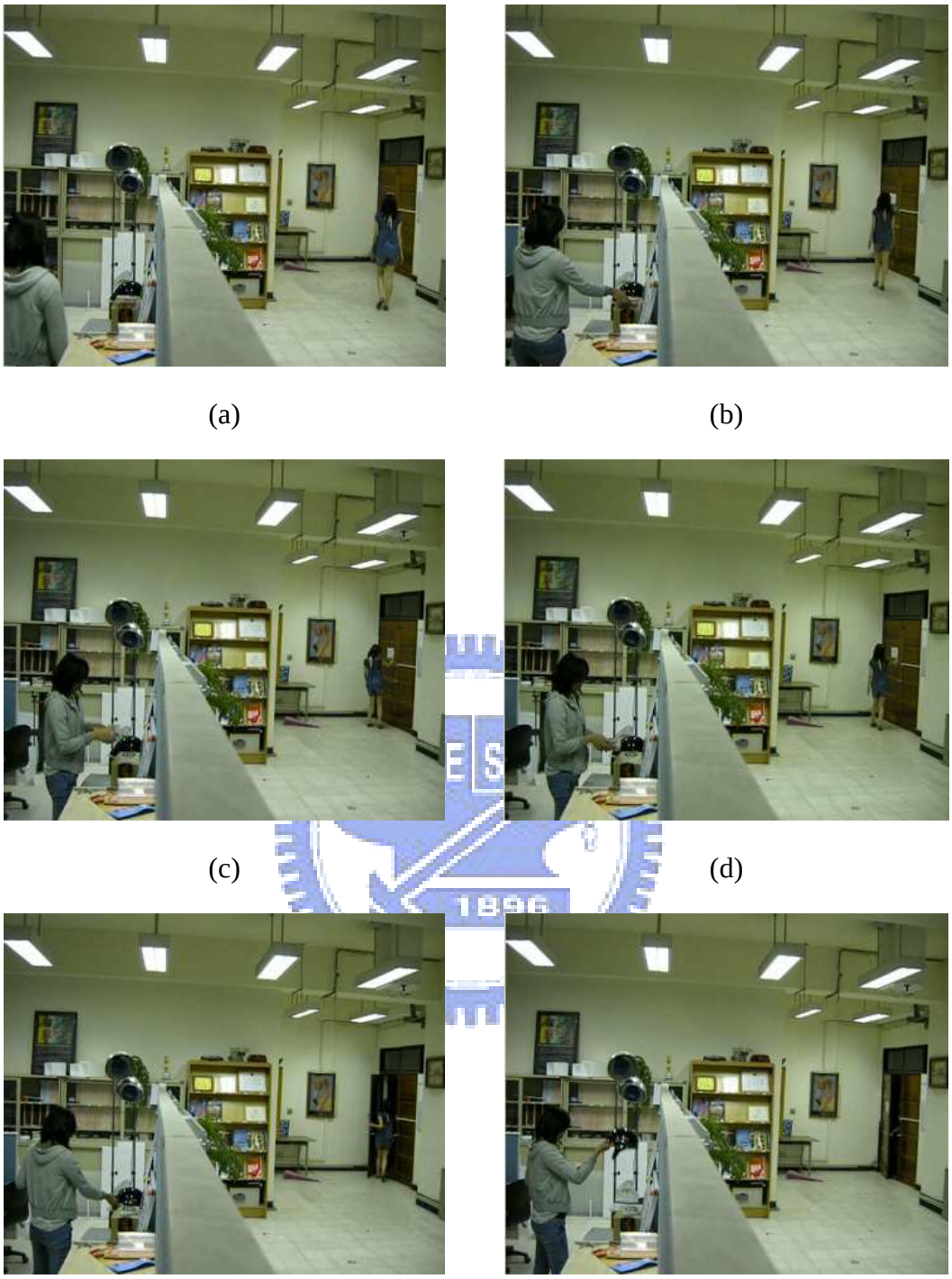
In this experiment, the input video is a surveillance video of the Computer Vision Lab at National Chiao Tung University. Six representative frames of an original video are illustrated in Figure 5.2. In this surveillance video, we want to monitor activities around the door, but we hope that the personal information within the left-bottom corner will not be revealed. Therefore, we utilized the proposed process for removing privacy information to remove the personal information. Six representative frames of a privacy video yielded by the proposed method of removing privacy information are shown in Figure 5.3. Six representative frames of a recovered video yielded by the

proposed method of recovery of privacy information are shown in Figure 5.4. Comparison between an original image and the corresponding recovered image is illustrated in Figure 5.5. From Figure 5.5(b), we can see that some details of the clothes of the protected person are lost. The experiment shows that the privacy information of regions selected by authorized users can be protected while surveillance videos still keep the surveillance information.

5.5 Discussions and Summary

In this chapter, we have proposed a method for removing selected privacy information in an H.264 surveillance video, and hiding them into the video. Based on multiple slice groups, we can modify the motion vectors and the frequency coefficients to remove sensitive information, and embed them into the surveillance video for recovery use. We have also proposed a method of recovering privacy information of a privacy video. The privacy information hidden in the privacy video can be extracted to recover the sensitive privacy information of the input video. Since the AC coefficients are not embedded into the privacy video for recovery use, some details of the recovered area may be lost. To solve this problem, we can also embed the AC coefficients into the privacy video, but it may cause an increase on the hidden data. This will be discussed in Chapter 6.

With this proposed privacy protection system, we can hide the privacy violation parts of surveillance video contents to avoid legal disputes and to protect the personal privacy of non-suspicious people. Moreover, we introduce the concept: region-based privacy protection, to avoid recognizing authorized persons by hand, which is different from traditional object-based privacy protection that needs to recognize protected persons by manpower.

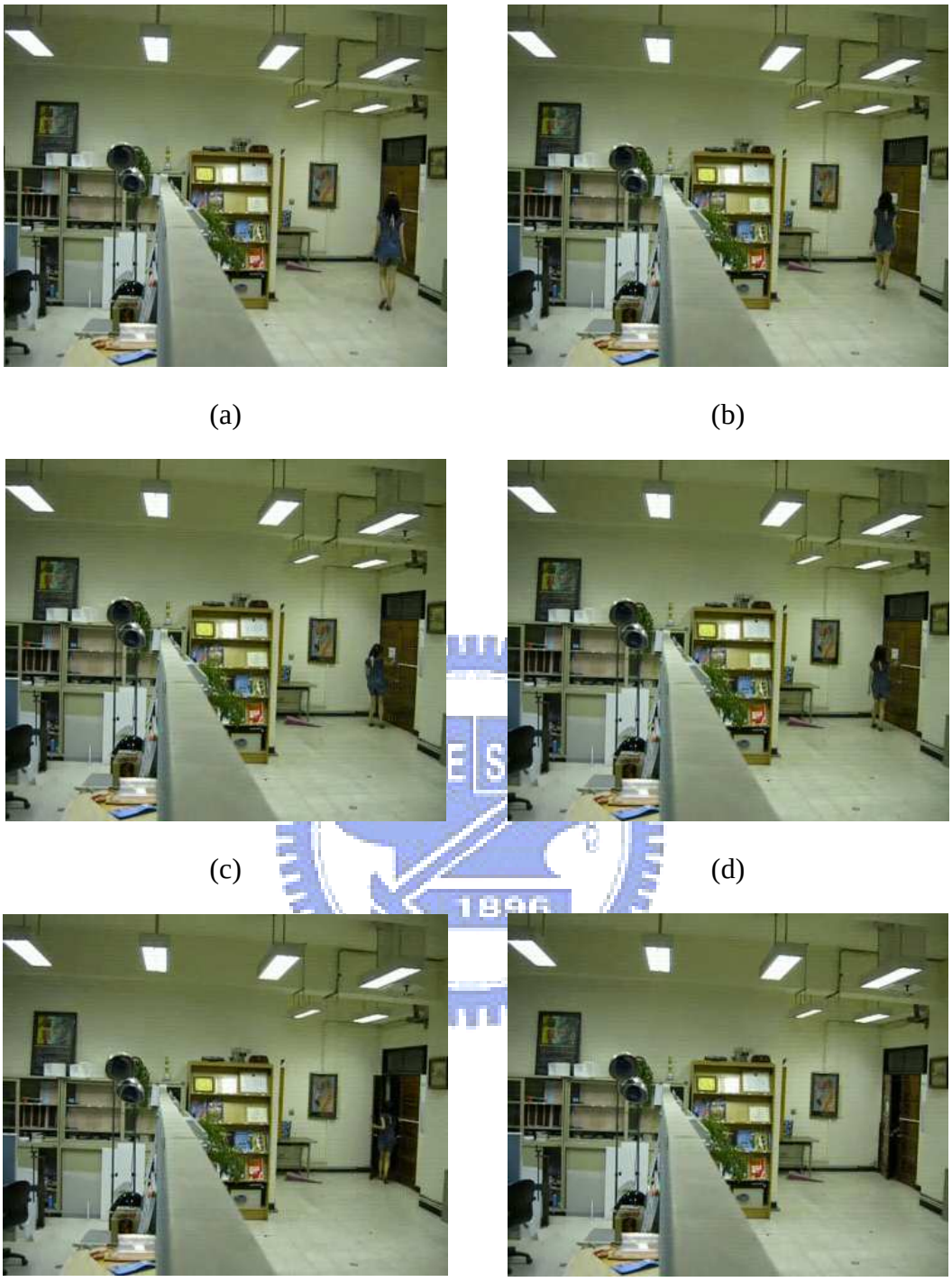


(a) (b)

(c) (d)

(e) (f)

Figure 5.2 Six representative frames of an original video. (a) The first frame. (b) The second frame. (c) The third frame. (d) The 4th frame. (e) The 5th frame. (f) The 6th frame.

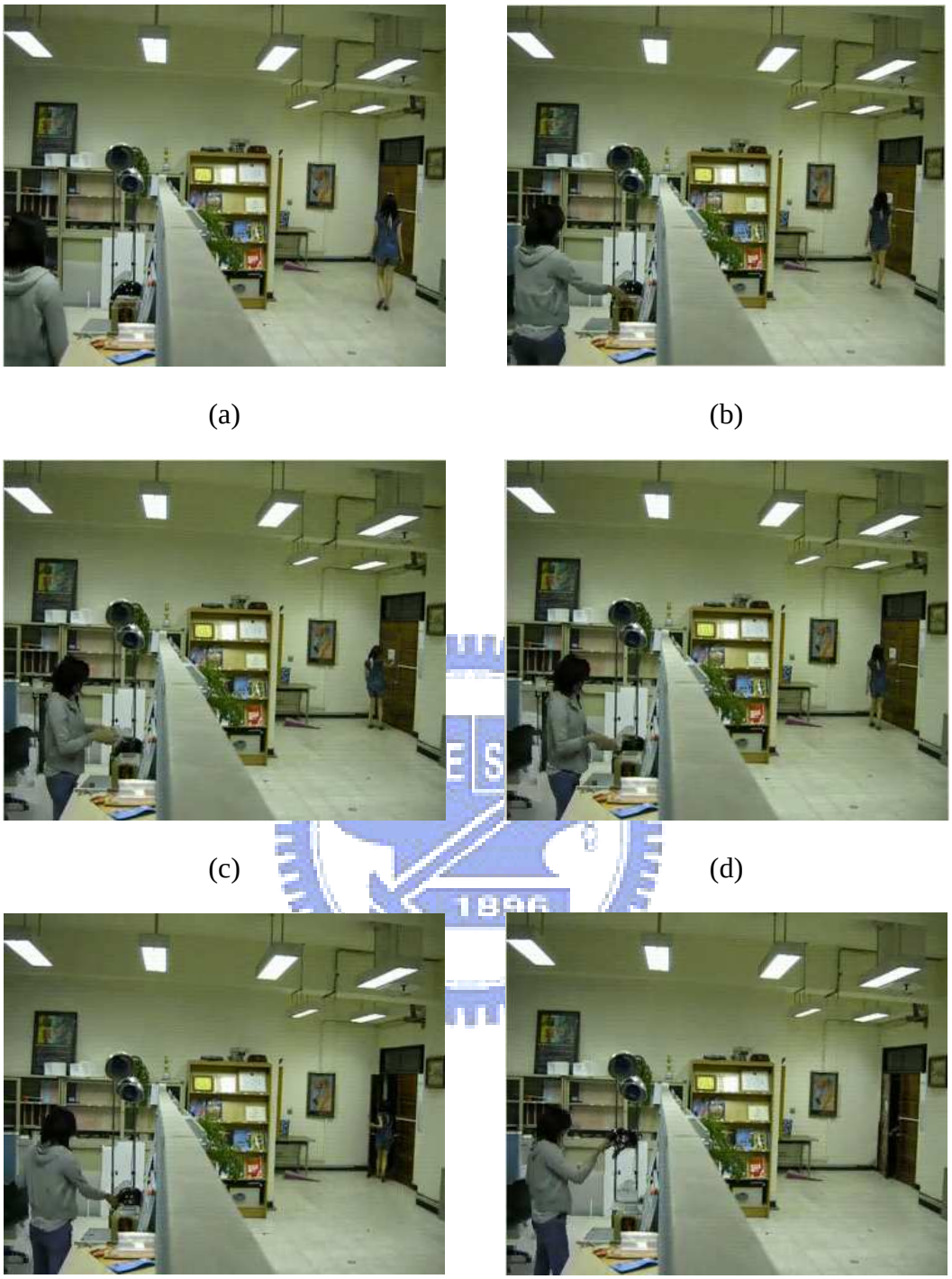


(a) (b)

(c) (d)

(e) (f)

Figure 5.3 Six representative frames of a privacy video. (a) The first frame. (b) The second frame. (c) The third frame. (d) The 4th frame. (e) The 5th frame. (f) The 6th frame.



(a) (b)

(c) (d)

(e) (f)

Figure 5.4 Six representative frames of a recovered video. (a) The first frame. (b) The second frame. (c) The third frame. (d) The 4th frame. (e) The 5th frame. (f) The 6th frame.

(a) (b)

Figure 5.5 Comparison between an original image and the corresponding recovered image. (a) The original image. (b) The recovered image.

Chapter 6
Conclusions and Suggestions for Future Works

6.1 Conclusions

In this study, we have proposed several methods for a variety of information hiding applications, such as video-content search, authentication, privacy protection, etc.

For video-content search, a method for quick search of video contents by novel uses of H.264 coding features has been proposed. A motion detection method using the tree structured macroblock decomposition information was utilized to detect the motion regions in an input video, and a data hiding method suitable for H.264 videos is applied to hide the motion region information into the video for search use. By use of the techniques mentioned above, the proposed surveillance video search system provides an easy way to search activities in a surveillance video.

For video authentication, a method for authentication of surveillance videos by hiding tree-structured macroblock decomposition information has been proposed. Tree structured macroblock decomposition information and frame indexes of an input video are used to form authentication signals. The authentication signals can be utilized to detect and to verify both spatial and temporal tamperings in a suspicious video.

For privacy protection in surveillance videos, a method for removing privacy

information of an H.264 surveillance video has been proposed. In the proposed method, an authorized user can specify a protected region in an input video, and personal information of the protected region can be removed to protect the privacy of persons within the protected region. The removed personal information is not eliminated but embedded into the video in case there is a need of retrieving the sensitive contents.

6.2 Suggestions for Future Works

Several suggestions for future works are listed in the following.

1. The content features used to search video contents can be extended to include more types, such as color information, user information, etc.
2. The way to modify DCT coefficients in the proposed data hiding method for H.264 videos can be modified to reduce distortion.
3. The proposed idea of the authentication method applied on P frames can be developed to get more details about the missing moving objects in a suspicious H.264 video.
4. The decoding information used to recover privacy information can be expanded to include AC coefficients in order to recover more details of the privacy information.
5. It is interesting to expand these video applications to handle videos of other profiles of the H.264 standard.
6. It is interesting to integrate the authentication method and the privacy protection method to authenticate video contents of surveillance areas and privacy-protected areas.

References

[1] I. Haritaoglu, D. Harwood, and L. S. Davis, "W^4: real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 809–830, Aug. 2000.

[2] A. J. Lipton, H. Fujiyoshi, and R. S. Patil, "Moving target classification and tracking from real-time video," *Proceedings of IEEE Workshop Applications of Computer Vision*, Princeton, USA, pp. 8–14, Oct. 1998.

[3] W. Zeng, J. Du, W. Gao and Q.M. Huang, "Robust moving object segmentation on H.264/AVC compressed video using the block-based MRF model," *Real-Time Imaging*, vol. 11, pp. 290–299, Aug. 2005.

[4] C. J. Li, and S. J. Wang, "Detection and tracking of a single deformable object on an active surveillance camera," *Proceedings of IPPR Conference on Computer Vision, Graphics and Image Processing*, Kinmen, Taiwan, pp. 16–18, Aug. 2003.

[5] W. Hu, T. Tan, L. Wang and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, vol. 34, pp. 334–352, Aug. 2004.

[6] M. Noorkami and R. M. Mersereau "A framework for robust watermarking of H.264-encoded video with controllable detection performance," *IEEE Transactions Information Forensics and Security*, vol. 2, pp. 14–23, Mar. 2007.

[7] G. L. Huang and W. H. Tsai, "Optimal Data Hiding in H.264/AVC Videos for Covert Communication," *Proceedings of 2008 Conference on Computer Vision, Graphics and Image Processing*, Ilan, Taiwan , Aug. 2008.

[8] X. Gong, and H. M. Lu, “Towards fast and robust watermarking scheme for H.264 video,” *IEEE International Symposium on Multimedia*, California, USA, pp. 649–653, Dec. 2008.

[9] G. Wu, Y. Wang, and W. Hsu, ‘‘Robust watermark embedding/detection algorithm for H.264 video,’’ *Journal of Electronic Imaging*, vol. 14, pp. 1–9, Mar. 2005.

[10] B. G. Mobasseri and Y. N. Raikar, “Authentication of H.264 streams by direct watermarking of CAVLC blocks,” *SPIE Conference on Security, Steganography and Watermarking of Multimedia Contents IX*, San Jose, USA, vol. 6505, pp. 664–669, Feb. 2007.

[11] J. Zhang and A.T.S. Ho, ‘‘Efficient Video Authentication for H.264/AVC,’’ *First International Conference on Innovative Computing, Information and Control*, Beijing, China, vol. 3, pp. 46-49, Aug. 2006.

[12] D. Pröfrock, H. Richter, M. Schlauweg, E. Müller, “H.264/AVC video authentication using skipped macroblocks for an erasable watermark,” *Proceedings of Visual Communication and Image Processing*, Beijing, China, vol. 5960, pp. 1480–1489, Jul. 2005.

[13] K. F. Chien and W. H. Tsai, “Authentication of surveillance video sequences and contents by hiding motion vector information,” *Proceedings of 2006 Conference on Computer Vision, Graphics and Image Processing*, Taoyuan, Taiwan, Aug. 2006.

[14] P. Meuel, M. Chaumont, and W. Puech “Data Hiding in H. 264 Video for Lossless Reconstruction of Region of Interest,” *European Signal Processing Conference*, Poznań, Poland, Sep. 2007.

[15] F. Dufaux, T. Ebrahimi, Emitall SA, “Smart video Surveillance System Preserving Privacy,” *Proceedings of SPIE Image and Video Communications and*

Processing, San Jose, USA, vol. 5685, pp. 54–63, Jan. 2005.

[16] W. Zhang, S.-C. S. Cheung, and M. Chen, "Hiding privacy information in video surveillance system," *Proceedings of IEEE International Conference on Image Processing*, Genova, Italy, vol. 3, pp. 868–871, Sep. 2005.

[17] H.264/AVC JVT reference software JM14.2, http://iphome.hhi.de/suehring/tml/download/

[18] I. Richardson, *H.264 and MPEG-4 video compression video coding for next-generation multimedia*, John Wiley & Sons, Hoboken, USA, 2003.