AUTHENTICATION OF H.264 SURVEILLANCE VIDEOS BY HIDING TREE-STRUCTURED MACROBLOCK DECOMPOSITION INFORMATION*

[1]*Shu-Hung Hung* (洪菽鴻) and [2]*Wen-Hsiang Tsai* (蔡文祥)
Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
E-mails: [1]july@cs.nctu.edu.tw, [2]whtsai@cis.nctu.edu.tw

ABSTRACT

A method for authentication of H.264 surveillance videos by a new information hiding technique is proposed. The method can verify both temporal and spatial tampering of H.264 videos by using macroblock decomposition information of motion regions in H.264 codes as authentication signals. The tree structured macroblock decomposition information is generated during the video encoding process and is different for different video contents; if a protected video has been tampered with, the authentication signals constructed from such information will be destroyed as well. Therefore, how and where the protected video is tampered with can be inspected accordingly to carry out the authentication work. Experimental results show the feasibility of the proposed method.

1. INTRODUCTION

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. The need of video authentication is especially essential in video surveillance applications because surveillance videos often contain suspicious or unlawful acts and malicious users might want to acquire them in illegal ways and tamper with them for misrepresentation. How to authenticate the integrity and fidelity of surveillance videos so has become a main topic in the research field of video surveillance.

Many different methods have been proposed to solve the problem of video authentication [1-4]. Mobasseri and Raikar [1] proposed an authentication method for H.264 streams by direct watermarking of the CAVLC (context adaptive variable length code) blocks. Zhang and Ho [2] 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. Pröfrock et al. [3] proposed a method using skipped macroblocks of an H.264 video to embed authentication data. Chien and Tsai [4] proposed a method for authentication of MPEG-4 surveillance videos by utilizing the motion vector information in the video frames.

These above-mentioned methods 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 work.

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 video modifications which are manipulated on video frame *contents*, and *temporal tampering* means modifications which are 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. Cropping means deleting some video frames from the original video sequence. Insertion means placing some fake video frames between frames of the original video sequence.

In this study, a method for authentication of H.264 surveillance videos by an information hiding technique using tree-structured macroblock decomposition information as authentication signals is proposed. In the method, a video sequence is divided into several *frame groups*, with each group being composed of some P frames and one I frame. In order to detect spatial and temporal tempering, 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. 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 the remainder of this paper, the proposed data hiding technique used in this paper is introduced first. A scheme for embedding of authentication signals is described in Section 3,

* This work was supported by the NSC project No. 96-2422-H-009-001.

and the proposed authentication process is stated in Section 4. In Section 5, several experimental results of applying the proposed method will be shown. Finally, some discussions and a summary will be made in the last section.

2. DATA HIDING IN H.264 VIDEOS

In this section, the proposed technique for data hiding in H.264 videos will be described. A video in which authentication signals are to be embedded is called a *cover video* in this study, and the embedding result is called a *stego-video*.

2.1 Embedding Data in a Cover Video

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 the video may be lost because the resulting frequency coefficients could be different. A solution proposed in this study is to use a user-selected key to generate intra prediction modes for use in the data hiding process 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. The details are described as an algorithm in the following, where some data are embedded in a 16×16 macroblock in an H.264 video frame composed of I slices only.

Algorithm 1. *Hiding data in a macroblock.*

Input: A secret key R, a sequence of 16-bit data D to be hidden, a 16×16 cover macroblock M_{16}, and a random number generator f.

Output: a stego-macroblock M_{16}'.

Steps.

1. Use the 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 randomly according to the random number sequence one of the available intra prediction modes of M_4, denoted as p.
   - 2.2 Use p 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 F.
   - 2.4 Embed the first un-hidden bit B of D in F by modifying the values C_1 and C_2 of the coefficient pairs $F_1(0, 3)$ and $F_2(3, 0)$ in F in the following way:
     - (1) *if* $B = 0$, *then*
       if $C_1 > C_2$, *then swap* C_1 *and* C_2;
     - (2) *if* $B = 1$, *then*
       if $C_2 = C_1$, *then set* $C_1 = C_2 + T$;
       if $C_2 > C_1$, *then swap* C_2 *and* C_1,
       where T is a pre-defined threshold.
   - 2.5 Inversely transform F and add the result to M_p to yield a reconstructed 4×4 sub-macroblock M_c.
   - 2.6 Use M_c to replace the 4×4 sub-macroblock M_4.
3. Repeat Step 2 until all 4×4 sub-macroblocks in M_{16} are processed, or until the data in D to be hidden are all exhausted.
4. Take the resulting macroblock M_{16}' as input to the normal encoding process.

2.2 Extracting Data from a Stego-video

The secret key mentioned previously may be utilized to re-generate the intra prediction modes used in the data hiding process described by Algorithm 1, so that the hidden data can be extracted correctly based on the re-generated modes, as described in the following.

Algorithm 2. *Extracting data from a macroblock.*

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

Output: a sequence of 16-bit data D hidden in M_{16}.

Steps.

1. Use the 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 generated random number sequence.
   - 2.2 Use p to produce a prediction block M_p.
   - 2.3 Subtract M_p from M_4 to produce a residual block M_r.
   - 2.4 Transform M_r into a set of frequency coefficients in the form of a 4×4 block F.
   - 2.5 Extract the hidden data bit b in M_4 from F according to the following rule:

     if $C_2 \geq C_1$, *set* $b = 0$; *else*, *set* $b = 1$.

     where C_1 and C_2 are the values of the coefficient pairs $F_1(0, 3)$ and $F_2(3, 0)$ in F.
3. Repeat Step 2 until all the 16 4×4 sub-macroblocks of M_{16} are processed.
4. Concatenate all the extracted 16 bits in order to form the desired sequence of hidden data D as output.

3. EMBEDDING OF AUTHENTICATION SIGNALS IN SURVEILLANCE VIDEOS

In this section, the method for embedding authentication signals will be described.

3.1 Embedding of Authentication Signals

Since surveillance videos usually contain some suspicious activities in motion regions, we use the information of the motion regions to generate the authentication signals. A frame group is treated as a *unit for authentication* in this study. Accordingly, each frame group G of an input video has its own authentication signals which are composed of *region signals*. Each region signal is generated by the use of a motion region in G, and is composed of the index I of G and the tree structured macroblock decomposition information T of the region. I is used to detect temporal tampering. T is used to detect spatial tampering.

Also, a scheme is designed in this study to extract the authentication signals more precisely based on a *voting* technique. For this, the authentication signals are duplicated for several copies and then embedded into the I frame in G for authentication use. More details are described in the following.

3.2 Generation of Authentication Signals

Besides the index of a frame group G, we also use the tree structured macroblock decomposition information of a P frame within G to generate authentication signals. This information is generated in a way which we describe now. First, we perform a motion detection algorithm proposed in this study to every P frame in G and randomly select one of the P frames, F_p, in G. Then, we find out the motion regions in F_p as a set R. Each region R_i of R is used to generate a *region signal*. For each 16×16 macroblock M of R_i, denote its macroblock partition mode as P_m. If P_m is of the mode of 16×16, 16×8, or 8×16, we use the bit 1 to represent M. If P_m is the 8×8 macroblock partition and each sub-macroblock partition mode of M is of the mode 8×4, 4×8, or 4×4, we use the bit 0 to represent M. If P_m is of the 8×8 macroblock partition mode and all the sub-macroblock partition modes of M are the 8×8 sub-macroblock partition modes, then the bit for representing M is decided according to the arrangement condition of the neighboring macroblocks. The index of G, the tree structured macroblock decomposition information of R_i, and the coordinate information of R_i comprise the region signal of R_i. More details will be described later.

The motion detection method proposed in this study is utilized to detect motion regions used for authentication. Two features of motion regions are used in the method. The first is *small partition size*, and the second is *variable partition mode*. Because motion regions usually contain some moving objects, video contents of the regions change a lot both in time and in space. Therefore, tree structured macroblock decomposition information of the motion regions is often *variable* and composed of *small* blocks. The first feature is used to filter *noise* around motion regions. And the second feature is utilized to drop *unstable motion regions* caused by background disturbance. The details of the methods are described as an algorithm below.

Algorithm 3. *Detection of motion regions*.

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 region growing to group the motion blocks in F to form several candidate motion regions, and 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 (16×16, 16×8, and 8×16) 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.
     - (1) If the block size is smaller or equal to 8×8:
       set *quantification value* = $N_{mode} + Pos$;
     - (2) otherwise:
       set *quantification value* = N_{mode},

       where N_{mode} is a mode number defined in the video encoder, and Pos is the coordinate information of a motion block.
   - 4.2 Use the quantification values of all the motion blocks to calculate the variance V of them for 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, too.
7. Output the position information of the motion regions remaining in L after the above merge process.

We are now ready to describe the proposed authentication signal generation process.

Algorithm 4. *Generation of authentication signals.*
Input: A frame group G in a video, a secret key K, and a random number generator f.
Output: a set of authentication signals, S_b, to be embedded.
Steps.
1. Use the key K as a seed for f to generate a sequence of random numbers, denoted as Q.
2. Perform motion detection using Algorithm 3 to every P frame in G.
3. Select randomly a P frame F_p in G according to Q, and 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 16×16, 16×8, or 8×16, mark M as a *large partition macroblock*.
      4.1.3 If P_m is 8×8 and each P_s of M is 8×4, 4×8, or 4×4, mark M as a *small partition macroblock*.
      4.1.4 For the case that both P_m and P_s are 8×8, decide M as a large or small partition macroblock by the following rules.
         (A) Evaluate the *partition score* of M in the following way.
            (a) Name the eight neighboring macroblocks as A through H, as depicted in Figure 1, and each of the corresponding macroblock partition modes as P_i.
            (b) 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.
            (c) Calculate the partition score as:

$$\textit{partition score} = \sum_{i=A}^{H} G_i \, .$$

         (B) If the partition score is smaller than a pre-defined threshold T, mark M as a large partition macroblock; else, as small.
      4.1.5 If M is a large partition macroblock, set the representing bit $B(M)$ 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 set of 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.

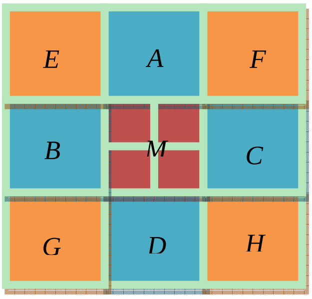

Figure 1. eight neighboring macroblocks of M.

3.3 Embedding of Authentication Signals

The proposed process for embedding the generated authentication signals is described as an algorithm below.

Algorithm 5. *Embedding authentication signals.*
Input: a set of authentication signals S_b which each of is in length L, an I frame F, a secret key K, and a random number generator f.
Output: a protected I frame F'.
Steps.
1. Duplicate S_b k times and concatenate them in a sequential order to form a new binary string S_b', where k is the *largest* integer such that L×k is just smaller than the data hiding capacity of an I frame.
2. Take an unprocessed 16×16 macroblock M from F and 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 them as S_{b16}'.

2.2 Take K, S_{b16}', M, and f as input to perform the data hiding process described by Algorithm 1.

3. Repeat Step 2 until all macroblocks in F are processed.

4. AUTHENTICATION OF H.264 VIDEOS

In this section, the proposed method for authentication of H.264 surveillance videos will be described.

4.1. Extraction of Authentication Signals

The voting technique is applied in the process of extracting duplicated authentication signals to obtain correct authentication signals, as mentioned previously. The details are described in the following algorithm.

Algorithm 6. *Extraction of authentication signals*.
Input: a protected I frame F, a secret key K, and a random number generator f.
Output: a set of 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 process described by Algorithm 2 to get the hidden data D_i of M_i.
2. Combine all D_i to form a binary string S with length L.
3. Perform the following steps on S to get the set of authentication signals S_b.

3.1 Divide the length L of S into segments of an equal length L_R, and denote the number of segments as T, where L_R is the length of a region signal.

3.2 Generate T candidate authentication signals according to the following steps and denote them as S_1 through S_T.

(a) Denote the currently-processed candidate authentication signal as S_j, where $1 \leq j \leq T$.

(b) Divide S into several segments of signals, with each of them being of the length $L_R \times j$.

(c) Transform each segment S' of S into the binary form as $S' = b_1b_2b_3\ldots b_l$, where l is the length of S'.

(d) Associate each bit of S' with two vote scores $V_0[m]$ and $V_1[m]$, where $1 \leq m \leq l$. 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 $1 \leq j \leq T$:

$$\text{if } b_m = 0, \text{ then set } V_0[m] = V_0[m] + 1;$$

$$\text{if } b_m = 1, \text{ then set } V_1[m] = V_1[m] + 1,$$

where $1 \leq m \leq l$.

(e) Denote the binary form of S_j as $S_j = s_1s_2s_3\ldots s_l$. Construct S_j by comparing the two scores of each bit of S' according to the following rule:

$$\text{if } V_0[m] > V_1[m], \text{ then set } s_m = 0;$$

$$\text{if } V_1[m] > V_0[m], \text{ then set } s_m = 1,$$

where $1 \leq m \leq l$.

(f) Calculate the *distribution rate* p_m of each bit s_m of S_j by the following rule:

$$\text{if } s_m = 0,\ \ p_m = \frac{V_0[m]}{\left(V_0[m] + V_1[m]\right)};$$

$$\text{if } s_m = 1,\ \ p_m = \frac{V_1[m]}{\left(V_0[m] + V_1[m]\right)},$$

where $1 \leq m \leq l$.

(g) Calculate the *average distribution rate* P_j of S_j by the following rule:

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

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

3.3 (*Selection by voting*) Select one candidate from each of S_1 through S_T, which has the highest average distribution rate, and collect them as the desired output set of authentication signals, S_b.

4.2. Authentication of Spatial Tampering

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. Because 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. The details are described as an algorithm below.

Algorithm 7. *Authentication 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 process described by Algorithm 2 to get the hidden data D in M.
2. Denote the sixteen 4×4 sub-macroblocks in M as M_1 through M_{16}, and for each of them, M_i,

denote the corresponding hidden bit of M_i as D_i.

3. Denote the number of suspected 4×4 sub-macroblocks in M as $N_{4\times4}$ and set its initial value to be zero.
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 as *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, 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, if the $N_{4\times4}$ of a macroblock M is larger than a pre-defined threshold, it means that most video contents within M are attacked, and so 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 M to be *neighbor-unauthentic* according to Rules (2) and (3) in Step 6 of the above algorithm.

4.3. Authentication of Spatial Tampering

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 according to 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 might 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 objects. More details of spatial tampering authentication are described in the following algorithm.

Algorithm 8: *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, and for each region signal S_i, where $1 \le i \le N$, perform the following steps.
   2.1 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.
   2.2 Extract the tree structured macroblock decomposition information from S_i, and denote it as R_T.
   2.3 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*.
   2.4 Denote R_T as $R_T = r_1r_2r_3\ldots r_l$, where l is a pre-defined length of the tree structured macroblock decomposition information part in a region signal.
   2.5 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 $1 \le j \le l$.
   2.6 Regard the partition information of macroblocks of R_i and the coordinate information of R_i as the *information of missing moving objects* of R_i.
3. Output the authenticated P frames in G with respective 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 using replacement tampering. The coordinate information part of I can be used to locate those areas in these P frames of the frame group which may contain suspicious activities. Moreover, the partition information can be used to describe the moving objects appearing in the area in more details, such as the moving direction, the shape of a moving object, etc.

4.4 Authentication of Temporal 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 9: *temporal tampering detection of a* *video sequence*.

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

Output: a report R of the detection result.

Steps.

1. For each frame group G_i in V with index I_i, extract the index I_i' hidden in the corresponding authentication signals, where $1 \le i \le N$.
2. Create a flag bit B to indicate the occurrence of tampering, and initialize B to 0.
3. Create a flag bit F to indicate the occurrence of replacement, and initialize F to 0.
4. Subtract I_i from I_i', and denote the result as D_i.
5. If $D_i \ne 0$, perform the following steps.
   5.1 If B equals 0, set B to 1, and record the index n_s of the I frame in G_i.
   5.2 If B equals 1 and D_i does not equal D_{i-1}, set F to 1.
6. If $D_i = 0$, perform the following steps.
   6.1 If B equals 1, record the index n_f of the I frame in G_i, and perform the following steps.
   (a) If F equals 1, decide the tampering type as replacement
   (b) If F equals 0, decide the tampering type as cropping and insertion.
   (c) Save the tampering type, n_s and n_f, into R.
   (d) Set B, n_s, and n_f to 0.
7. Repeat Steps 5 through 6 for each frame group until reaching the end of V.
8. If B equals 1, perform the following steps.
   8.1 If F equals 1, recognize the tampering type as replacement.
   8.2 If F equals 0, perform the following steps.
   (a) If $D_N > 0$, decide the tampering type as cropping.
   (b) If $D_N < 0$, decide the tampering type as insertion.
   (c) Save the tampering type, n_s and the index of the last I frame of V into R.

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. In this video, a malicious user tries to cover a person who wants to take a book on the table and crops the part of the person in all frames of the input video. Two consecutive frame groups of an original video are shown in Figure 2. Two consecutive frame groups of the protected video yielded by the proposed method are shown in Figure 3. The malicious user crops the area containing the person in each frame and replaces it with the background image. Two consecutive frame groups of an attacked version of the video are shown in Figure 4. The two corresponding consecutive frame groups of the authenticated video are shown in Figure 5.

In Figure 5, the green areas in the I frames are the suspicious areas which are attacked. The black rectangles in the representative P frames are the results of authentication on P frames. These rectangles reveal the content information and motion information of the original video in the attacked areas. Based on the concepts 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.

6. CONCLUSIONS

In this paper, 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 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, but also shows further where and how the tampering occurs. Experimental results show the feasibility of the proposed method.

REFERENCES

[1] 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.

[2] J. Zhang and A. T. S. Ho, "Efficient Video Authentication for H.264/AVC," *1st International Conference on Innovative Computing, Information and Control*, Beijing, China, vol. 3, pp. 46-49, Aug. 2006.

[3] D. Pröfrock, H. Richter, M. Schlauweg, and 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, July 2005.

[4] 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.

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

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

Figure 2 Two consecutive frame groups of the original video. (a) A representative P frame of G_1. (b) The I frame of G_1. (c) A representative P frame of G_2. (d) The I frame of G_2.

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

Figure 3 Two consecutive frame groups of the protected video. (a) A representative P frame of G_1. (b) The I frame of G_1. (c) A representative P frame of G_2. (d) The I frame of G_2.

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

Figure 4 Two consecutive frame groups of the tampered video. (a) A representative P frame of G_1. (b) The I frame of G_1. (c) A representative P frame of G_2. (d) The I frame of G_2.

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

Figure 5 Three consecutive frame groups of the authenticated video. The green areas in the figures are suspicious areas of the I frame. The black rectangles in the figures are the tree structured macroblock decomposition information of the suspicious areas. (a) A representative P frame of G_1. (b) The I frame of G_1. (c) A representative P frame of G_2. (d) The I frame of G_2.