CONTENT AUTHENTICATION AND TAMPER DETECTION IN DIGITAL VIDEO

Bijan G. Mobasseri
ECE Department
Villanova University
Villanova, PA

Michael J. Sieffert
Computer Sciences Dept
SUNY, Binghamton
Binghamton, NY

Richard J. Simard
Multi-Sensor Exploitation Branch
Air Force Research Laboratory
Rome, NY

ABSTRACT

In this paper we report on the development of a watermarking algorithm designed for video authentication and tamper detection. The objectives are to determine unauthorized cut-and-splice or cut-insert-splice operation and quantify the extent of such editing. We demonstrate that the proposed algorithm can identify cut start and duration down to single frame precision. The approach embeds a watermark with a strong timing content, violation of which can be traced back to the parameters of the editing operation

1. INTRODUCTION

Content authentication is a process by which a user is guaranteed that video content is original and has not been maliciously modified. One example is surveillance and site monitoring footage where incentives exist to remove incriminating material. Another example is chain of evidence scenarios where the authenticity of a piece of video needs to be established for admissibility in legal and other proceedings. Content authentication is an ongoing and constant requirement. As video is transported across networks there is a need for authentication as data changes hand. A broader definition of authentication includes controlled access. Based on the embedding of specific digital signatures, video may be viewed by authorized personnel but unavailable to others.

For the purposes of this work we define two main video tampering scenarios, 1): cut-and-splice and 2): cut-insert-splice. The cut-and-splice operation describes a situations where an unknown number of frames are cut and removed and the two ends spliced. The removed portion may have presumably contained incriminating material. In such a scenario, algorithms are needed to identify two pieces of information, 1): where the cut began and 2): how long it lasted. Cut-insert-splice operation is intended not only to remove material but to insert irrelevant or misleading footage in its place. Of course, it is not a priori known whether we are dealing with the first or the second type or whether in fact there has been any tampering in the first place.

2. WATERMARKING FOR AUTHENTICATION

Watermarking can be used for authentication applications as well as intellectual property rights protection[1]. Both types of watermarks must remain imperceptible, secure and hard to remove. Attacking a watermarked video can be achieved by removal, replacement or mutilation of the watermark. However, in a authentication scenario, any one of the above three operations will result in a failure to authenticate. In some ways, therefore, watermarking for authentication needs to be brittle to attempts at removal or smearing yet robust to benign alterations. Attacks on a watermarked video fall into two distinct categories. The first category consists of intensity attacks such as noise, compression or any other attempt that disrupts the intensity patterns of video. The second category is time-base attack defined as any attempt to disrupt frame sequencing. Examples are frame cuts, foreign frame insertion, frame rate alteration, frame swapping and others. Tamper detection issues of the kind described here fall under time-base attacks. As such, any watermarking algorithm devised for tamper detection must exhibit sensitivity to video sequencing.

A recently proposed watermarking algorithm meets two needed requirements of strong timing content and resistance to intensity attacks[2,3]. The algorithm combines time-hopping and direct sequence spread spectrum in a single unified framework. The watermark is first defined by a 2D binary plane and later spread out by a 3D *pn* spreading code. Another *pn* code serves as the time hopping sequence. This sequence pseudorandomly tags video bitplanes for removal and insertion of the spread watermark.

3. WATERMARK SPREADING AND TIME-HOPPING SEQUENCES

Watermark spreading sequence is generated from a 1D *m*-sequence rearranged in 3D. The period of the 1D sequence relative to video frame size dictates over how many frames the watermark can be spread out. Let w define the single binary watermark plane. Let $c=\{c_0,c_2,...,c_{p-1}\}$ define the 3D spreading sequence of period p. c is derived from a 1D m sequence of period mnp where m and n are video frame rows and columns. The spread watermark is formed by $w_{ss}=\{w.c_0,w.c_2,...,w.c_{p-1}\}$ where $w.c_i$ is the term by term matrix multiplication. In effect, w_{ss} is a spread spectrum signal that is recoverable only if the spreading sequence is available.

0-7803-6297-7/00/$10.00

The spread watermark matrix is inserted in the video stream at bitplane level. The insertion points are decided by a multi-valued *pn* sequence v. v can be defined in a number of ways. Let $u=\{u_0,u_2,...,u_{q-1}\}$ be an m sequence with period q and $T^i(u)$ the ith cyclic shift of u. Then $v=u+Tu+T^2u+...+T^{k-1}u=\{v_o,v_1,...,v_{q-1}\}$ is a multiple-valued *pn*-sequence of period q taking on values in the range {0 to k}. w_{ss} is then shuffled in the video such that the jth spread watermark from w_{ss} is located at bitplane v_j. Although v has the same period as u, it exhibits substantial sidelobes. There is another way to generate a less correlated v . We can sum k m-sequences with the same period but generated by k different primitive polynomials. For illustration purpose we may use 3 different polynomials given by [5,3],[5,2] and [5,4,3,2] all with period 31. Detailed studies of correlation properties of spreading and time-hopping sequences appear elsewhere[4]

4. WATERMARK DETECTOR

The watermarked video now has a number of distinguishing features. A periodic *pn* bitplane has been inserted into the video bitplane stream. The manner of insertion is strictly controlled by the time hopping sequence and is known to authorized decoders only. For example, it is known a priori that the ith spread watermark plane $w.c_i$ is inserted in bitplane v_i. Conversely, bitplane v_i can only contain $w.c_i$. Moreover, the next frame of an authentic video will contain $w.c_{i+1}$ in bitplane position given by v_{i+1}. We have in effect embedded a secure time code in the video. Any and all attempts at removing, replacing or altering video frames will result in the violation of time-hopping and spreading sequence structure. An examination of the recovered indices will not only reveal tampering attempts but will also quantify the extent of such tampering. Watermark detection algorithm below expands upon these points

The locations of spread watermarks along the timeline are known from the markers provided by the positioning sequence v. Given N frames, the watermark is despread by the following operation.

$$\hat{w}=w_{ss}.c=[w.c_1.c_1,w.c_2.c_2,...,w.c_N.c_N] \\ =[w,w,...,w] \quad (1)$$

With no compression or any other disturbances, $\hat{w}$ is simply N replications of the original watermark. The despread watermark $\hat{w}$ is then correlated with w over N frames. A sliding correlator produces N 2D correlation surfaces that are then averaged to arrive at a single indicator for the presence or absence of watermark. In the absence of any disturbances the 2D-correlator response is given by

$$r=w\otimes\hat{W}=[\underbrace{w\otimes w,w\otimes w,...,w\otimes w}_{N}] \\ \bar{r}=\frac{1}{N}\sum w\otimes w\rightarrow\bar{r}(0,0)=mn \quad (2)$$

Where $\otimes$ stands for 2D correlation. With no attack on video, correlator registers the maximum response of mn. It has been shown that the underlying processing gain of spread spectrum receiver provides substantial watermark survivability even after video has undergone MPEG compression/decompression cycle[3].

5. AUTHENTICATION AND TAMPER DETECTION

The relationship between p and q impacts the ability to detect video tampering attempts. We recognize two cases:

§ Equal periods, $p=q$

When both periods are the same, there exists a 1-to-1 correspondence between the element of v and w_{ss},

$$v=\{...,v_1,v_2,v_3,...,v_p,v_1,v_2,v_3,...,v_p,...\} \\ \text{b b .....b b b .....b} \quad (3) \\ w_{ss}=\{...,w_1,w_2,w_3,.,w_p,w_1,w_2,w_3,.,w_p,...\}$$

Define the original watermarked video by frame sequence $F=\{f_1,f_2,f_3,...,f_N\}$. A subset of F, $F'=\{f_if_{i+1},...,f_{i+M}\}$ is chosen and subsequently tampered with by cut and splice operation. Define the tampered video by

$$F^*=\{f_i,f_{i+1},...,f_{i+k-1}f_{i+k+l},f_{i+k+l+1},...,f_{i+M}\} \quad (4)$$

The cut starts from the kth frame in F^* and lasts l frames. There are 3 unknowns here, 1): the beginning frame number i relative to the original video F, 2): the first cut frame position $i+k$ and 3): cut length l in frames We may also face multiple non-contiguous cuts. When $p=q$ the algorithm works as follows

1. For frame 1 compute $\rho=\{f_i(v_j).c_j\}\otimes w, j=1\rightarrow p$ where $f_i(v_j)$ is bitplane v_j
2. Select j for which ρ indicates a match.. We now know that w_j resides in bitplane v_j in f_i.
3. Repeat steps 1 and 2 for f_{i+1} and identify which of the p spread watermarks reside there.
4. If f_{i+1} follows f_i, the extracted watermark in step 3, w_{j+1}, will also follow w_j. Hence no gap is detected

The 1-to-1 correspondence in (3) means any gap detected in the recovered w s corresponds directly to the

gaps existing in the frame sequence. For the example given in (4), the recovered w_{ss} is given by

$$w_{ss}^{*} = \{w_i, w_{i+1}, \ldots, w_{i+k-1} w_{i+k+l}, w_{i+k+l+1}, \ldots, w_{i+M}\} \quad (5)$$

This approach also reveals any tampering attempt that alters the order of frames in any way, such as frame shuffling, reordering or replacement.

There are a number of scenarios where the procedure above may run into problems. For example, the algorithm faces ambiguity in determining the length of the cut if the cut segment crosses one or more periods of w_{ss}. To illustrate this point, consider p=q=31. If the cut begins at frame 10 and lasts 32 frames, the extracted sequence in (5) will look like $\{\ldots, w_8, w_9, w_{12}, w_{13}, \ldots\}$. It appears that the number of cut frames are just two whereas the true number is 33. One way to remove multiple period ambiguity is to increase p, q or both. The ideal period would be long enough to fill the entire length of video. This choice will remove the possibility that cuts could extend across multiple periods. The downside is high computational cost in generation and more importantly detection of watermark. Since p and q can take on only specific values controlled by their generating polynomials, it may not be possible to have them equal N. In that case, we can select q so that $N/q<2$. This condition will assure that there is at most one whole period plus one partial period in N frames. The length of partial period should be kept to a minimum for reasons to be explained shortly. We claim that this arrangement largely removes the multiple period ambiguity. Consider a cut length consisting of l frames beginning with frame k. Then $w_{ss}^{*} = \{\ldots, w_{k-1}, w_k, w_{k'}, w_{k'+1} \ldots\}$. If $k' > k$ then the cut generally resides within the full period and the length of the cut is simply $k' - k - 1$. If cut extends across the period boundary, then $k' < k$. But k is actually given by $k' = (k + l + 1) \bmod(q)$. What it takes to find the length of the cut, is to find a value for l that satisfies the modulo operation.

§ Unequal periods, p>q

The large periods required of the underlying 1D m-sequences to spread the watermark may give rise to a more likely scenario where $p>q$. The following relationship illustrates this idea

$$\begin{Bmatrix} v_1 v_2 \ldots v_q v_{q+1} v_{q+2}, \ldots, v_{2q}, \ldots, v_p \\ w_1 w_2 \ldots w_q w_1 w_2 \ldots w_q \ldots w_1 w_2 \ldots w_q \end{Bmatrix} \quad (6)$$

The most important byproduct is the loss of 1-to-1 correspondence between the spread watermark planes and their insertion point. For example, for p=31 and q=5, the third plane of the spread watermark may be assigned to bitplanes 2, 1, 3, 3, 3 or 4. Therefore, detection of a particular watermark plane does not uniquely identify where we are in the frame sequence.

To resolve this ambiguity, we note the following property. When video undergoes a cut-and-splice operation, the positioning sequence will be similarly affected. The modified sequence v^{*} will have correlation properties that are different from the original. The challenge is to see if we can connect these new properties to the parameters of cut-and-splice. Let the cut begin with frame x+1 and last y frames. What the splice operation does is to cause a phase offset between v and v^{*}.

$$v = \left\{ v_1 v_2 v_3 \ldots \underbrace{v_x v_{x+1} v_{x+2} \ldots v_{x+y}}_{cut} v_{x+y+1} \ldots v_{q-y} \ldots\ldots\ldots v_q \right\}$$

$$v^{*} = \left\{ v_1 v_2 v_3 \ldots \underbrace{v_x v_{x+y+1}}_{gap} v_{x+y+2} \ldots\ldots\ldots\ldots\ldots v_q, v_1, v_2 \ldots v_{q-y} \right\} \quad (7)$$

We now define the cross correlation between v and v^{*} by $R^{*} = E\{v v^{*}\}$. Assume there are α frames before the cut and β frames after. The following relationships have been derived in [4]:

$$\begin{aligned} R^{*}(0) &= \alpha R(0) + R_{q-y}(0) + \beta R(y) \\ R^{*}(y) &= \alpha R(y) + R_{q-y}(y) + \beta R(0) \end{aligned} \quad (8)$$

Cutting y frames from the video generates a corresponding y unit shift in the underlying time-hoping sequence. Computing correlation at lag y simply realigns much of v and v^{*}. Clearly, a pn-sequence registers a much higher value for its autocorrelation at lag 0 than at any other value. Shifting of v^{*} by y positions relative to v takes α periods out of alignment but brings β periods into alignment with v. Therefore, $R^{*}(y)$ will register its peak when y is equal to the number of cut frames. Figures 1 and 2.

6. EXPERIMENTAL RESULTS

The above algorithm was implemented on moonshot clip consisting of 145 120x160 frames with color and audio. The time hopping sequence was generated from three 5th order polynomials given by [1 0 0 1 0], [1 1 1 1 0], and [1 0 0 0 0 1]. This sequence has a length of 31, selected for its prime property and small computational demand. The actual time hopping sequence v and its alignment with spread watermark planes w_{ss} for p=q is as follows

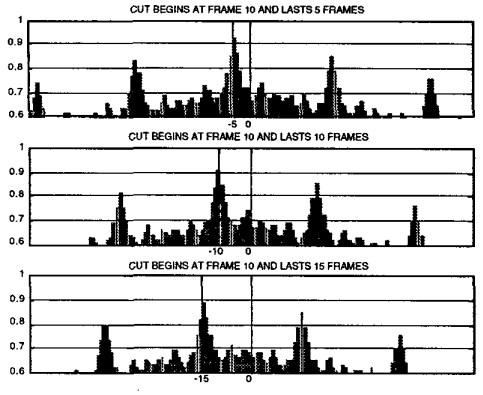

Figure 2- The separation between $R^*(0)$ and the first major peak is precisely the number of cut frames

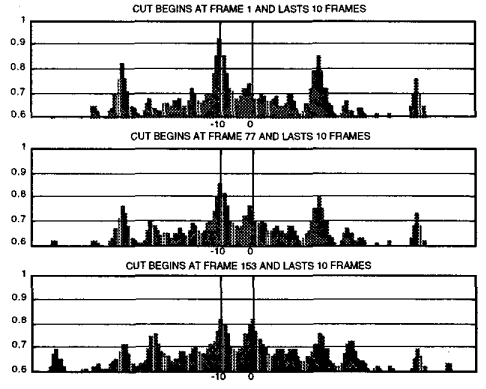

Figure 3- Cut length is fixed but cut start position varies. The difference between the two peak values measures cut start beginning.

4	2	3	2	4	3	3	2	..	3	1	4	1
w	w	w	w	w	w	w	w		w	w	w	w
1	2	3	4	5	6	7	8		28	29	30	31

To spread the watermark over 31 video frames, a 22nd order *m*-sequence will suffice. Figure 3 illustrates the watermark as binary logo and its spread spectrum spread. When presented with a potentially tampered video, synchronization must first be established. The first frame may start with any of the 31 spread watermark planes. An exhaustive search, however, is not necessary. To search for w_1 only the 4th bitplane needs to be searched because w_1 cannot be anywhere else. The number of correlations per frame are therefore bounded by 31, frequently much less because a match will be found without going through all 31 watermarks. If tampering has not occurred, each watermark will be detected in the order they are expected. If tampering has occurred, watermarks are found out of order, or not found at all. In this simulation the software generated the following results from a tampered video

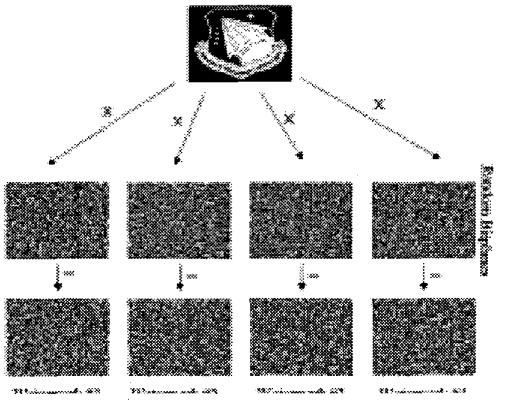

Figure 1- watermark is encrypted by the 2D spreading sequence over multiple planes

wm# 21 22 23 6 7 8 9 10 11 12 13 14 x x x x 3 4 5 6
frame# 64 65 66 67 68 69 70 71 72 73 74 75 80 81 82

The algorithm reveals that the first frame of the sample video is actually the 64th frame of the original and there are 13 frames missing beginning with the 3rd (66th) frame. x s indicate that no watermark was found indicating a possible cut-insert-splice of foreign material of length 4

7. CONCLUSIONS

A digital video authentication algorithm has been presented. The algorithm is capable of identifying cuts and splices both in length and duration. Still many questions need further investigation including more effective ambiguity resolution and countering attempts at false authentication of bogus video.

8. ACKNOWLEDGMENTS

This work was performed in summer of 99 at the Air Force Research Lab, Rome , NY.

9. REFERENCES

[1] F. Hartung, B. Girod, Watermarking of uncompressed and Compressed Video, , *Signal Processing* 66 (1998), pp.283-301

[2] B. Mobasseri, Direct Sequence Watermarking of Digital Video using *m*-frames, *Proc. IEEE ICIP 98*, October 4-7, Chicago.

[3].B. Mobasseri, A spatial video watermark that survives MPEG , *IEEE International Conference on Information Technology: Coding and Computing*, March 27-29, 2000, Las Vegas

[4] B. Mobasseri, M. Sieffert, Content authentication of digital video by direct sequence spread spectrum watermarking, AFRL/Rome, Final Report, August 1999