Integrity Verification System For Video Content By Using Digital Watermarking

Isao Echizen[1], Stephan Singh[2], Takaaki Yamada[1], Koichi Tanimoto[1], Satoru Tezuka[1], Benoit Huet[2]

[1]System Development Laboratory, Hitachi, Ltd.
890, Kashimada, Saiwai-ku, Kawasaki, 212-8567, Japan
({iechizen, tanimoto, t-yamada}@sdl.hitachi.co.jp)
Ph.:+81-44-549-1467 Fax.:+81-44-549-1640

[2]Eurécom Institute
BP 193, 06904 Sophia Antipolis cedex, France
({singh, huet}@eurecom.fr)
Ph.: +33 (0)4 93 00 26 26 Fax.:+33 (0)4 93 00 26 27

ABSTRACT

An improved system is described for verifying video content integrity using digital watermarking. Current verification systems using the digital signature are unable to distinguish between attacks and regular modifications such as resizing and MPEG encoding and are thus unsuitable countermeasures against actual threats to content kept in storage services. The proposed verification system distinguishes attacks against video content from regular modifications by extracting timecodes and header hash values embedded in the content itself and comparing them with the actual ones, making it well suited for content storage services. Evaluation showed that the system is more effective than a current one using the digital signature scheme and that it can be used by a variety of applications using stored video content.
Keywords: verification service system, video content integrity, digital watermarking

1. INTRODUCTION

Digital video content has become available through various media such as the Internet, digital broadcasting, and DVD because of its advantages over analog video content. It requires less space, is easier to process, and does not degrade over time or with repeated use. A serious problem, however, is that the integrity of digital video content is easily violated because digital video can be easily modified using editing tools such as software on PCs. Systems for verifying the integrity of video content, by detecting any changes in the content, are thus becoming increasingly important.

The unceasing growth in storage capacity, has led to video content storage services being widely used for various applications. Since the video format is subject to regularly re-encoding and transcoding because of version upgrading of the systems devices, the verification system should be able to distinguish between illegal modifications of attacks against the content and regular modifications. Current verification systems using the digital signatures scheme, however, are unable to do this. Moreover, the output is simply Boolean: the content is either changed or unchanged.

We have developed a video content integrity verification system that can distinguish attacks from regular modifications. It extracts timecodes and header hash values embedded in the content itself and compares them with the actual ones, making it well suited for content storage services.

Section 2 describes the use of digital signatures in current methods and summarizes the problems with this scheme. After an overview of watermarking, Section 3 explains the kinds of attacks to be detected and introduces our proposed verification system. Section 4 summarizes our evaluation of the system, and Section 5 introduces an application of it. section 6 concludes the paper with a brief summary.

1-4244-0451-7/06/$20.00

2. DIGITAL SIGNATURE SCHEME AND PROBLEMS

The digital signature scheme is one of the most useful techniques for verifying the integrity of content and is widely used in current systems for verifying the integrity of video content. Digital signatures are generated as follows[1, 2]:

Step G1: Calculate a feature value (hash value) from data values in the video content (e.g. pixel values of picture) by using a one-way hash function.

Step G2: Encrypt the hash value with the private key of a public-key cryptosystem.

Step G3: Add the encrypted hash value into the header fields of the content.

The corresponding process flow of the digital signature verification is as follows:

Step V1: Extract the encrypted hash value from the header field of the content and decrypt the value with the corresponding public key of Step G2.

Step V2: Calculate a hash value of the content in the same manner in Step G1.

Step V3: Compare the value decrypted in Step V1 with the one encrypted in Step V2. If the values match, the content integrity has been maintained. If they do not, it has been broken.

Applying this scheme to all the data values in the video content enables the integrity of the content to be verified and various applications using digital signatures were proposed[3, 4]. However, use of this scheme is problematic.

- It provides no detailed information about the degree of content change. The output is simply Boolean: the content has either been changed or not been changed.
- It cannot distinguish between malicious attacks and regular modifications, so a regular modification is reported as a break in content integrity.

3. USE OF DIGITAL WATERMARKING TO IMPROVE INTEGRITY VERIFICATION SYSTEM

Our proposed verification system solves the two problems mentioned above by using digital watermarking, enabling it to distinguish attacks from regular modifications.

3.1. Digital watermarking

Digital watermarking can be used to embed information (e.g. copyright and copy control information) in digital content in order to prevent illegal copying of the contents and is therefore used in various systems handling digital content such as content-distribution systems [5, 6].

Figure 1 shows an example use of digital watermarking for video pictures [5, 7, 8]. Watermarks representing the copyright information (e.g. the author's identity) are embedded in a digital video by changing some of the pixel values in the frames, but not enough to change the appearance of them. Even if the marked video undergoes image processing, such as MPEG encoding or D/A-A/D conversion, the embedded information can be detected by identifying changes in the pictures.

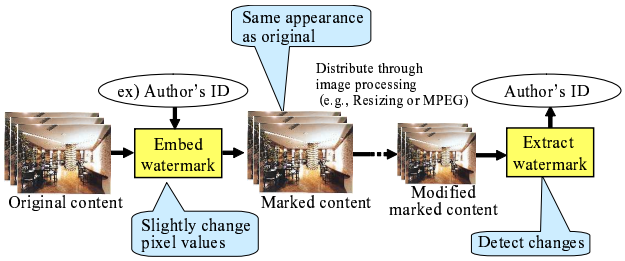


Fig. 1: Digital watermarking

The basic scheme in our verification system using watermarking is as follows: On the encoder side, watermarks representing timecodes and header hash values from the video content are embedded into the video and audio tracks of the content. On the detector side, the values of watermarks extracted from the tracks is compared with the actual ones. If the values are the same, the content integrity was maintained.

This scheme enables our system to detect various types of attacks detailed in the next Section 3.2.

3.2. Definition of attacks

For a system to be robust, it must be able to detect the following types of attacks. We use the nomenclature shown in Figure 2.

A **shift attack** shifts video and/or sound tracks from their original positions, as illustrated in Figure 3. While such attacks are uncommon, they can happen.

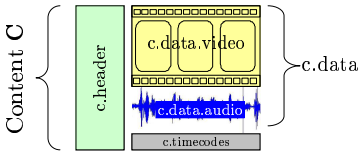


Fig. 2: Structure of video content

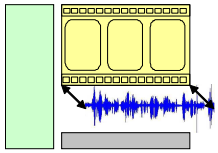


Fig. 3: Shift attack

A **header tamper attack** removes content from the header content or replaces it, as illustrated in Figure 4. For instance, an attacker might replace the copyright information with false information indicating that the attacker is the owner of the content.

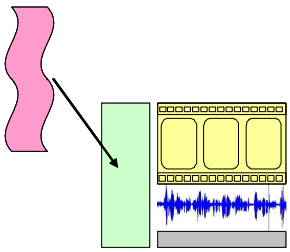


Fig. 4: Header-tampering attack

A **replace attack** replaces some of the video and/or sound tracks with other video/sound tracks, as illustrated in Figure 5. For instance, an attacker might replace a part of the video surveillance content incriminating him with false content.

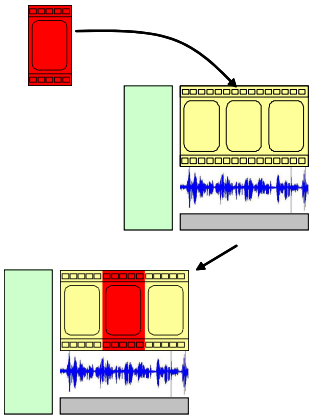


Fig. 5: Replacing attack

A **deletion attack** removes some of the video and/or sound tracks, as illustrated in Figure 6. For instance, an attacker might remove the tracks on a videosurveillance tape showing a car accident to avoid being identified as a suspect.

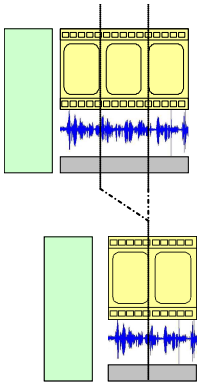


Fig. 6: Deletion attack

Current verification systems are unable to detect all these attacks. Moreover, they are not able to distinguish them; their output is simply Boolean (tampered with or not). Our proposed system with its multiple outputs can distinguish them.

3.3. Improved system

Our proposed verification system has two components: a encoder, used by the client to protect the video content, and a detector, used by the viewer or auditor to check its integrity.

A. The encoder

Description

Video content is first processed by the encoder. The hash values of the header content and the timecodes are embedded in the audio and video data; this is the watermarking action. Both types of data are watermarked to enable content integrity to be checked in the detector. The embedding is done at the same time as encoding. The process flow is illustrated in Figure 7.

Process flow

The process flow of encoding happens as follows:

Step E1: Demultiplex the content into the video content, the audio content, timecodes, and header.

Step E2: Compute the hash values of the header using a one-way hash function.

Step E3: Embed timecodes and the hash values in the video and audio content as a watermark.

Step E4: Multiplex the watermarked content and the header and timecodes to obtain content watermarked.

B. The detector

The verification detector is implemented in the viewer or auditor client. When a video content is received, its content is verified by the detector.

Description

The detector verifies video content integrity by extracting the watermark information (timecodes and header hash values) from the video and audio tracks

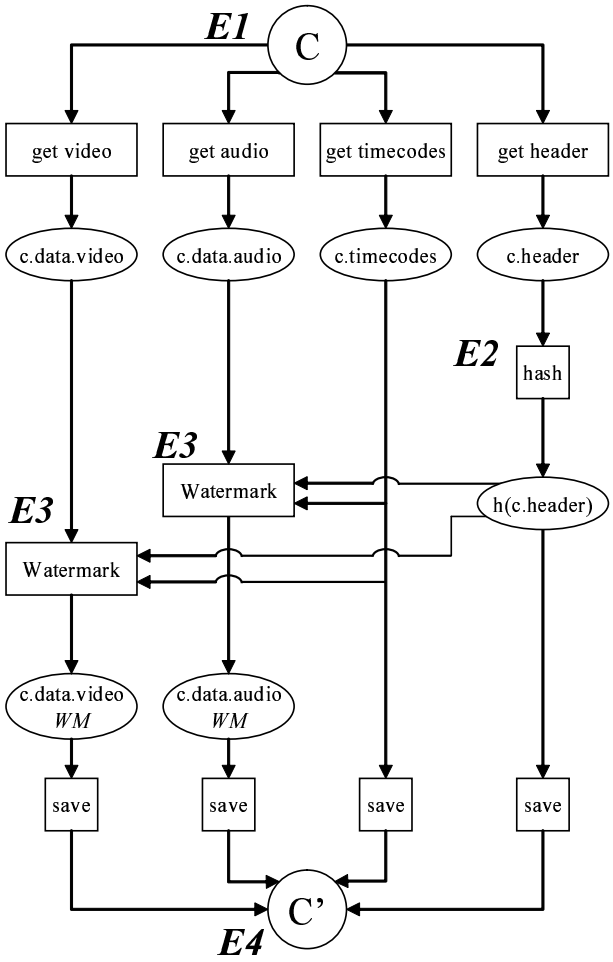


Fig. 7: Encoder block diagram

and comparing the values with the actual ones. If all values are identical, the content has not been corrupted. If at least one is different, the content has been attacked. If the compared elements are equal (i.e. no attack), the output value is 0, and if they are different (i.e. attack), the output value is 1. The process flow is illustrated in Figure 8.

Process flow

The corresponding process flow of the content integrity verification by the detector happens as follows:

Step D1: Demultiplex the watermarked content into the header, timecodes, audio content, and video content.

Step D2: Compute the hash values of the header using the same one-way hash function.

Step D3: Extract the timecodes and hash values from the watermark information.

Step D4: Compute integrity of header's hash values: it compares real hash values and the ones embedded in the audio and video tracks.

Step D5: Compute integrity of timecodes: it compares the actual time values and the ones embedded in the audio and video tracks.

Step D6: Output 0 (no attack) if compared items are equal; output 1 (attack) if compared items are different, as shown by the example outputs in Table 1 (Attacks are represented by their first character: **S**hift,**R**eplace, **D**elete, **H**eader tamper).

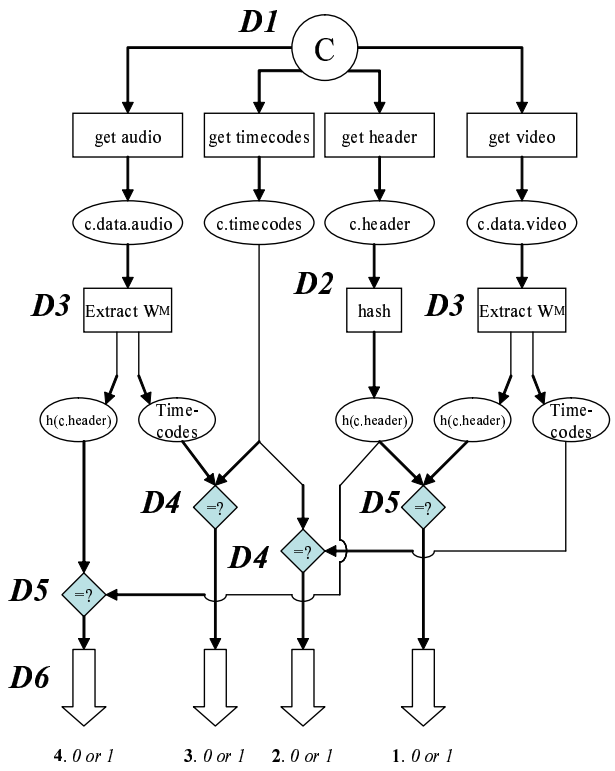


Fig. 8: Detector block diagram

Output	No attack	S	R	D	T
1	0	0	a^1	0	1
2	0	a^1	a^1	a^1	0
3	0	b^1	b^1	b^1	0
4	0	0	b^1	0	1

Tab. 1: Detector outputs

[1] $(a, b) = (0, 1)$ or $(1, 0)$ or $(1, 1)$

Thanks to the multiple outputs, the type of attack can be distinguished.

4. EVALUATION

4.1. Outputs of detector of the verification system

As shown in Table 2, each type of attack was correctly represented by the outputs. The outputs are shown in the same order as in Figure 7. The Audio/Video column shows which type of content was attacked and which attack occurred. For instance, $(\text{A \& V})_S$ means that audio and video tracks were shifted. Outputs which do not represent attacks are not presented on the table.

Outputs	Attack type	Audio / Video
0 0 0 0	None	-
0 0 1 0	*S* or D^2	**A**
0 0 1 1	*R* or *R* & (*S* or D^2)	**A**
0 1 0 0	*S* or D^2	**V**
0 1 1 0	*S* or D^2	**A** & **V**
0 1 1 1	(*S* or D^2) & *R*	$(\mathbf{A}\ \&\ \mathbf{V})_{S\ or\ D}$ *and* $\mathbf{A}_R$
1 0 0 1	*T*	**A** & **V**
1 0 1 1	(*R* or *S* or D^2) & *T*	$\mathbf{A}_{(S\ or\ D\ or\ R)\&T}$ & $\mathbf{V}_T$
1 1 0 0	*R* or *R* & (*S* or D^2) & *R*	**V**
1 1 0 1	(*R* or *S* or D^2) & *T*	$\mathbf{V}_{(S\ or\ D\ or\ R)\&T}$ & $\mathbf{A}_T$
1 1 1 0	(*S* or D^2) & *R*	(**A** & **V**) *or* (**A** & $\mathbf{V}_R$)
1 1 1 1	*R* or *R* & (*S* or *T* or *R*) or *T* & (*S* or *D*)	**A** & **V**

Tab. 2: Evaluation outputs

[2] See Section 4.2 for differentiation

4.2. Frames consistency verification

The detector also checks the order of the frames (included in timecodes) and their integrity, enabling shift and deletion attacks to be differentiated. If the detector detects a shift or a deletion and that some frames are missing (i.e. the numbering is not consistent), a deletion attack has occurred. If the detector detects a shift or a deletion but no missing fames, a shift attack has occurred. Using this same technique, the detector can differentiate a replace attack and a shift or deletion attack. If the detector detects a replacement and a shift or deletion, and that the frame numbering is not consistent, a deletion attack has occurred. If the detector detects a replacement and a shift or deletion, and if the frame numbering is consistent, a shift attack has occurred.

4.3. Comparison with the previous method

We compared the performance of the proposed system with that of a current one using the digital signature scheme described in Section 2. Table 3 shows the performance of both systems for three different video-content conditions; without modification, regular modifications (e.g. MPEG encoding, resizing, filtering, D/A-A/D conversion), and maliciously attacked (defined in Section 3.2). It shows that the current system can determine only whether the video content has changed, while the proposed one can differentiate between regular modifications and malicious attacks. Moreover, it can specify the type of attack. The proposed system is thus more effective.

5. APPLICATION

Buildings requiring high security such as banks, casinos, and airport terminals generally use video surveillance systems. The video files are stored on servers and periodically compressed to reduce the storage requirements [9, 10]. If video content has been stored for a long time, it is difficult

	Without modification	Regular modifications	Maliciously attacked
Previous	OK (no change)	NG (change)	NG (change)
Proposed	OK (no attack)	OK (no attack)	Specify attack

Tab. 3: Performance of current and proposed methods

for current verification systems to determine whether the content has been attacked or corrupted because they cannot distinguish between regular modifications and attacks.

Figure 9 shows an example video surveillance system. All the video content captured by the surveillance cameras in the banks other high-security buildings is stored and compressed in storage units, typically operated by a third party such as a storage service center. The stored video is periodically re-encoded or transcoded: When the video is quite new, the compression is light; as it becomes older, the compression becomes stronger. If the video is stored by a third party, it has a greater exposure to the threat of illegal modification or attack because the third party could destroy or change some of the content for self-gain or due to outside pressure. The proposed verification system is an effective countermeasure against these threats.

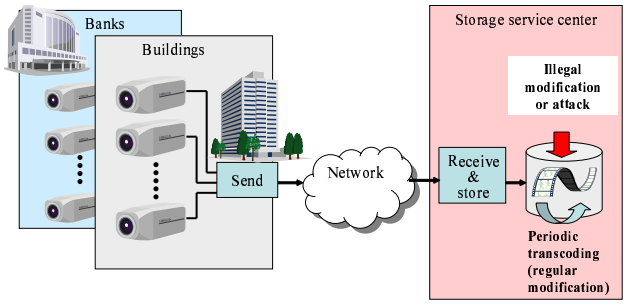


Fig. 9: Example video surveillance system

Figure 10 shows the same system with our verification system implemented. The encoder embeds watermarks in the video content, and the watermarked videos are stored at the storage service center. At the request of an auditing service, all the video content stored at the storage service center is periodically send to the auditing service. The auditing service checks the integrity of the content using the proposed detector; if the content has been maliciously modified or attacked, the detector detects the corresponding modification or attack. If the content has simply been re-encoded or transcoded, the detector confirms its integrity.

6. CONCLUSION

Integrity verification systems are essential for applications such as video surveillance. Current verification systems using the digital signature scheme are unable to distinguish attacks from regular modifications and are thus not effective countermeasures against actual threats to stored video content. Moreover, current verification systems are unable to identify the type of attack because their output is simply Boolean (content changed or not changed). The proposed verification system can distinguish between attacks and regular modifications by extracting header timecodes

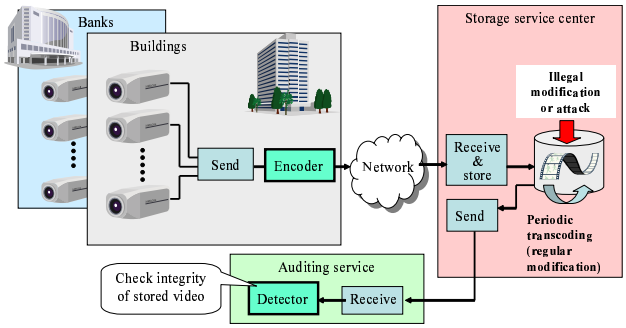


Fig. 10: Expected application using proposed verification system

and hash values embedded in the content and comparing them with the actual ones. These characteristics make it well suited for actual content storage services. Evaluation showed that the proposed system is more effective than a current one using the digital signature scheme and that it can be used by a variety of applications using video content storage.

REFERENCES

[1] B. Schneier, "Applied Cryptography, Second Edition", *John Wiley & Sons*, Inc., Sect. 2.6, pp. 34–41, 1996

[2] A. Menezes, P. Oorschot, and S. Vanstone, "Handbook of Applied Cryptography", *CRC Press*, Sect. 1.6, pp. 22–23, 1996.

[3] H. Morito, K. Anzai, D. Watanabe, H. Yoshiura, and Y. Seto, "Digital Camera for Taking Evidential Photographic Images", *Proceedings, IEEE International Conference on Consumer Electronics (ICCE'01)*, pp. 118–119, 2001.

[4] A. Pramateftakis, T. Oelbaum, and K. Diepold, "Authentication of MPEG-4-based surveillance video", *Proceedings, International Conference on Image Processing (ICIP'04)*, vol. 1, pp. 33–37, 2004.

[5] I. Cox, M. Miller, and J. Bloom, "Digital Watermarking", *Morgan Kaufmann Publishers*, 2002.

[6] W. Bender, D. Gruhl, and N. Morimoto, "Techniques for Data Hiding", *IBM Syst. J.*, vol. 35, pp. 313–336, 1996.

[7] I. Echizen, H. Yoshiura, T. Arai, H. Kimura, T. Takeuchi, "General Quality Maintenance Module for Motion Picture Watermarking", *IEEE Trans. Consumer Electronics* , Vol. 45, No. 4, pp. 1150 – 1158, 1999.

[8] I. Echizen, Y. Fujii, T. Yamada, S. Tezuka, H. Yoshiura, "Perceptually Adaptive Video Watermarking Using Motion Estimation", *International Journal of Image and Graphics*, Vol. 5, No. 1, pp. 89 – 109, World Scientific, 2005.

[9] D. Nicholson, J. Meessen, "Technologies for multimedia and video surveillance convergence", *Proceedings of SPIE – Image and Video Communications and Processing 2005, Amir Said, John G. Apostolopoulos, Editors*, vol. 5685, pp. 1–13, 2005.

[10] M. Abdel-Maguid, M.Moniri, “Video combiner for multi-channel video surveillance based on finite state methods”, *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance*, pp. 599–603, 2005.

Improved Video Verification Method Using Digital Watermarking

Isao Echizen, Takaaki Yamada, Satoru Tezuka
Hitachi Ltd. SDL, 890, Kashimada, Saiwai-ku
Kawasaki, 212-8567, Japan
isao.echizen.vs@hitachi.com

Stephan Singh
Swiss Institute of Technology - Eurécom
BP 193, 06904 Sophia Antipolis cedex, France
stephan.singh@eurecom.fr

Hiroshi Yoshiura
Faculty of Electro-Communication
University of Electro-Communications
1-5-1, Chofugaoka, Chofu, 182-8585, Japan
yoshiura@hc.uec.ac.jp

Abstract

A method is described that verifies video content integrity by checking the continuity of embedded timecodes used as digital watermarks. Conventional verification methods using digital signatures and fragile watermarking are unable to distinguish between attacks and regular modifications due and thus are unable to protect against threats to content. The proposed verification method distinguishes attacks against video content from regular modifications by extracting timecodes embedded in consecutive frames of the content and then checking their continuity. A prototype implementation showed that the method is more effective than conventional ones and that it can be used by a variety of applications using video content.

1. Introduction

Digital video content has become widely available through various media such as the Internet and digital broadcasting because of its advantages over analog video content. It requires less space, is easier to process, and does not degrade over time or with repeated use. A serious problem, however, is that the integrity of digital video content is easily violated because the content can be easily modified using software editing tools. Methods for verifying the integrity of video content by detecting changes in the content are thus becoming increasingly important. Since the video format is regularly encoded and transcoded in many ways, the verification methods should be able to distinguish between illegal modifications, i.e., attacks against the content, and regular modifications. Conventional verification methods using digital signatures and fragile watermarking schemes cannot do this.

We previously investigated the technical requirements for verifying and protecting the integrity of video and proposed a system concept [1]. We have now developed a method for implementing this concept. Testing of a prototype system using the proposed algorithms showed that the method can fully verify video content integrity

2. Conventional methods

There are two types of conventional methods for verifying the integrity of video content:

(a) Methods using digital signatures are widely used for content verification [2, 3]. Digital signatures are generated by calculating a hash value from data values of the content, encrypting the value, and adding it to the content header. Verification is done by recalculating the hash value from the content, decrypting the one in the header, and comparing them. If the values match, content integrity has been maintained. If they do not, it has been broken.

(b) Methods using fragile or semi-fragile watermarks are also widely used [4, 5, 6]. Watermarks are embedded in each frame and are easily broken by a change in the content. Semi-fragile watermarks are likely to survive against JPEG and MPEG compression at high bit rates, while fragile ones are not. Verification is done by checking for broken watermarks. If any are found, content integrity is assumed to have been broken.

The first type is well suited for small-sized content, such as text and document files, that are not modified by an application. The second type is well suited for still images that are not modified or restrictively modified. Neither type is well suited for video content because video content is regularly encoded, transcoded, and converted in various ways

such as MPEG encoding, resizing, filtering, and D/A-A/D conversion depending on the application. A method for verifying video content should therefore be able to distinguish between regular modifications and irregular modifications, i.e., attacks. Our proposed method has this capability.

3. Proposed method

Our proposed integrity verification method can identify attacks against video content. It can also distinguish between attacks and regular modifications such as video encoding and transcoding.

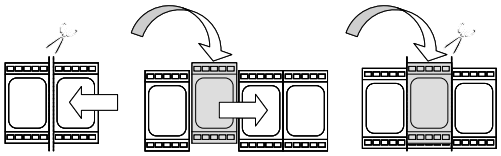
3.1. Example target applications

Our verification method has many target applications. Here we describe two of them.

Medical operations
Operations in hospitals are now commonly video recorded. If the surgeon makes a mistake, he or she might be tempted to later edit the recording to excise any damaging evidence. An auditor checking the consistency of the altered recording using our method could determine that it had been changed.
Public Works
Public works projects often use video recording to create a visual record of the progress of construction. If progress falls behind schedule, the site manager can, using a simple PC editing tool, replace some of the content with a recording of work completed elsewhere. Again, an auditor checking the consistency of the altered recording using our method could determine that it had been changed.

3.2. Attack Types

We consider three types of attacks against video recordings: deletion, addition, and replacement. A *deletion attack* removes some of the content, as described in the medical operations example. An *addition attack* adds content between frames. A *replacement attack* is a combination of deletion and addition, resulting in the same number of added and deleted frames at the same position. This is the type of attack described in the public works example.



(a) deletion (b) addition (c) replacement

Figure 1. Attack types

3.3. Process flow

Our method uses an *encoder*, which embeds watermarks, and a *detector*, which extracts the watermarks and checks content integrity.

The encoder is implemented in a video camera system and embeds watermarks and encodes the frames at the same time the data is recorded. The watermarks are timecodes equal to the actual time (*hh:mm:ss*). The same watermark is embedded in N consecutive frames, as shown in Figure 2. The watermark for each N-frame segment is the timecode corresponding to the encoding time (beginning at $t = t_1$) of the segment's first frame.

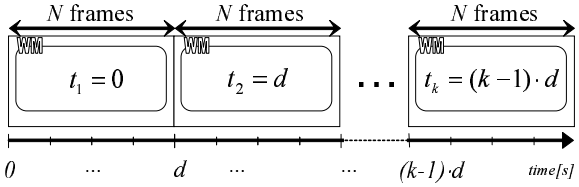


Figure 2. Watermark embedding

The detector's aim to locate and identify any attack on the video content. It works as follow: Using a detection window ($n = \frac{N}{2}$ frames sized), it first take the timecode extracting the watermark (WM) for each n-frame segment. It then checks the consistency between the timecodes by verifying their order. It also counts the number of windows where the WM has not been detected. In this way, it can detect an attack, determine the type of attack, and determine how many frames are affected (with n-frame precision). The parameters used are listed in Table 1.

Definition of parameters	
TC_{cur}	current detected timecode
TC_{pre}	previous detected timecode
TC_{old}	timecode detected before TC_{pre}
i	ordering number of the detection windows
nD	number of the detection windows where WMs are not detected
N_{max}	Maximal number of non-detected WM in succession

Table 1. Parameters used in detection

There are four steps in the detection process.

Step D1: Set the initial values of the parameters, TC_{cur}, $TC_{\text{pre}}, TC_{\text{old}}, i, nD$:

- $TC_{\text{cur}} = TC_{\text{pre}} = TC_{\text{old}} = i = nD = 0$,

Step D2: For the ith detection window, accumulate n frames and extract their WMs.

1. If $nD > N_{max}$, end the process.
2. If WMs are not detected, increment nD and i ($nD = nD + 1$; $i = i + 1$).
3. If WMs are detected, set $TC_{\rm old}$, $TC_{\rm pre}$, and $TC_{\rm cur}$:
   - $TC_{\rm old} = TC_{\rm pre}$
   - $TC_{\rm pre} = TC_{\rm cur}$
   - $TC_{\rm cur}$ = "detected timecode"
   - $nD = 0$

Step D3: Check the consistency between the timecodes $TC_{\rm old}$, $TC_{\rm pre}$ and $TC_{\rm cur}$, and check the value of nD. If the values satisfy the specified conditions, the content has been attacked.

Step D4: Increment i ($i = i + 1$) and retry Step D2.

4. Attack identification method

As described above, the detector uses $TC_{\rm cur}$, $TC_{\rm pre}$, and $TC_{\rm old}$ and the value of nD to identify the type of attack. The detector first determines whether three distinct timecodes following themselves appear ($TC_{old} + d = TC_{pre}$ and $TC_{pre} + d = TC_{cur}$). If they do, less than N frames have been deleted. If they do not ($TC_{old} = TC_{pre}$ or $TC_{cur} = TC_{pre}$), the type of attack is identified using the values of $TC_{\rm pre}$, $TC_{\rm cur}$, and nD ($\alpha, \beta > 1$), as shown in Table 2. A replacement attack is when addition and deletion attacks occur in succession. Such an attack has occurred if $\beta = \alpha$ or $\beta = \alpha + 1$, and it is detected after the timecode for the next window is extracted.

	$TC_{cur} - TC_{pre}$	nD
No attack	d or 0	0 or 1
Addition	d or 0	α
Deletion	$\alpha \cdot d$	0 or 1
Combination	$\beta \cdot d$	α

Table 2. Correspondence between parameter values and type of attack

Figure 3 illustrates how an addition attack is identified. By detecting consecutive windows without a WM ($nD = 3$ in the example shown) and detecting no gaps between the preceding and the current timecodes (t_2 and t_3), the detector identifies an addition attack.[1]

[1]Note that the 3^{rd} window is not identified as an attacked one but simply as one without a WM.

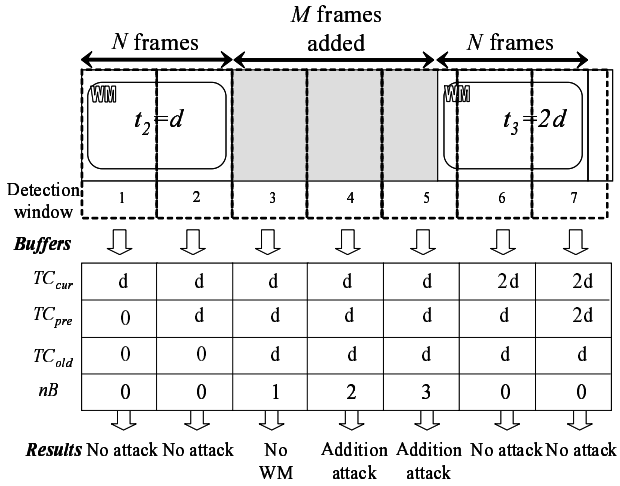


Figure 3. Addition attack

5. Prototype system

5.1. Description

We developed a prototype system of our proposed method for verifying video content integrity.

First, to measure the WM encoding strength, we watermarked a video sample by using the method described in paper [7] and encoded it in H.264 at a low bit rate. The detection rate of the highly compressed video content was 100% with 30 frames accumulation.

On the basis of this accumulation value, we set the window size, n, to 30 frames and the number of frames per segment, N, to 60 ($2 \cdot 30$). The timecodes used are the actual ones for the sample video file.

The encoder embed, as a WM, the timecode for the first frame in the first 60 frames; that is, they all have the same WM. The timecode for the 61^{st} frame are embedded as a WM in frames 61-120 and so on until all the frames had been watermarked.

The detector's display is shown in Figure 4. 18 seconds of the video content is represented on the first line. Each character represents 15 frames (two characters per window). There were no attacks in the first six seconds (as shown by the $\star$ at the bottom). A deletion attack (|) then occurred, followed by seven clean seconds. A five-second addition attack (A) then occurred.

5.2. Evaluation

First we compared the performance of our system with those of conventional ones using digital signatures, fragile watermarking, and semi-fragile watermarking (see Section 2). Table 3 shows the performances for three different

```
*: no attack
|: deletion attack
A: addition attack
R: replacement attack

      !!! VERIFICATION SYSTEM !!!
0---------o---------o---------o------18
|---------|---------|---------|------|
************|************AAAAAAAAAA**
```

Figure 4. Portion of System Display Showing Deletion and Addition Attacks

video-content conditions: without modification, with regular modifications (e.g., MPEG encoding, resizing, filtering, D/A-A/D conversion), and attacked. It shows that conventional systems using digital signatures and fragile watermarking cannot differentiate regular modifications and attacks because they can determine only whether the video content has changed. The one using the semi-fragile watermarking can differentiate them but only for particular modifications and attacks. The proposed system can differentiate various regular modifications and the attacks described in Section 3.2. Moreover, it can identify the type of attack. The proposed system is thus more effective.

	Without modification	Regular modifications	Maliciously attacked
Digital signatures	OK	NG	NG
Fragile WM	OK	NG	NG
Semi-fragile WM	OK	OK for limited modifications	OK for limited attacks
Proposed	OK	OK	OK

Table 3. Performance of conventional and proposed systems

We then evaluated the performance of our system by using the following standard video samples [8] (450 frames of 720×480 pixels) having different motion properties: "Square" having little movement and "Whale" having a great deal of movement. To measure our system's detection reliability, we first watermarked a sample video file and compressed it in H.264 ($bitrate = 1Mbps$). We then applied deletion, addition, and replacement attacks to it. Next we checked the attacked file with our detection system. Each attack was detected and identified.

Then we applied two attacks on different part of the same content. Each arrangement has been tested: two deletions, additions, and replacements, deletion-addition, deletion-replacing, and finally replacing-addition. Our detector identified every attack, and also determined the position where they happened.

6. Conclusion

Conventional video content integrity verification systems using digital signatures and fragile watermarking schemes are unable to distinguish attacks from regular modifications and are thus not effective countermeasures against threats to video content. Moreover, they are unable to identify the type of attack because their output is simply Boolean (content changed or not changed). The proposed verification method distinguishes attacks and regular modifications by extracting the timecodes embedded as watermarks in consecutive frames of the content and checking their continuity. Evaluation using a prototype showed that the proposed method is more effective than conventional ones. It can detect and identify attacks on video content, even if the content has suffered multiple types of attacks. It is thus usable by various types of applications using video content as evidence.

References

[1] I. Echizen *et al.*, "Integrity Verification System For Video Content By Using Digital Watermarking", *International Conference on Service Systems and Service Management (ICSSM'06)*, October, 2006.

[2] M. Pramateftakis *et al.*, "Authentication of MPEG-4-based surveillance video", *Proceedings, International Conference on Image Processing (ICIP'04)*, vol. 1, pp. 33–37, 2004.

[3] H. Morito *et al.*, "Digital Camera for Taking Evidential Photographic Images", *Proceedings, IEEE International Conference on Consumer Electronics (ICCE'01)*, pp. 118–119, 2001.

[4] M. Wu *et al.*, "Watermarking for Image Authentication", *IEEE International Conference on Image Processing*, vol. 2, pp. 437–441, 1998.

[5] C-Y. Lin *et al.*, "Robust Image Authentication Method Surviving JPEG Lossy Compression", *Storage and Retrieval for Image and Video Databases (SPIE)*, vol. 3312, pp. 296-307, 1998.

[6] C-Y. Lin *et al.*, "Issues and Solutions for Authenticating MPEG Video", *Security and Watermarking of Multimedia Contents (SPIE)*, vol. 3657, pp. 54-65, 1999.

[7] I. Echizen *et al.*, "Perceptually Adaptive Video Watermarking Using Motion Estimation", *International Journal of Image and Graphics, World Scientific*, vol. 5(1), pp. 89-109, 2005.

[8] "Evaluation video sample (standard definition)", *The Institute of Image Information and Television Engineers*.