

Speech Content Integrity Verification Integrated with ITU G.723.1 Speech Coding

Chung-Ping Wu and C.-C. Jay Kuo
Department of Electrical Engineering-Systems
University of Southern California, Los Angeles, CA 90089-2564
E-mail: {chungpin, cckuo}@sipi.usc.edu

Abstract

A speech content integrity verification scheme integrated with ITU G.723.1 speech coding to minimize the total computational cost is proposed in this research. Speech features relevant to the semantic meaning are extracted, encrypted and attached as the header information. This scheme is not only much faster than cryptographic bitstream integrity algorithms, but also more compatible with a variety of applications. The speech signal could go through re-compression, amplification, transcoding, re-sampling, D/A and A/D conversion and minor white noise pollution without triggering the verification alarm.

1. Introduction

Traditionally, tamper detection in speech communication could be accomplished by the human recognition of the voice characteristics (sometimes called the bio-signature) of the speech sender. However, modern digital media editing and processing technology allows high quality forgery to be created at a relatively low cost, and low bit-rate speech coders used in network environments discard certain voice characteristics. Thus, judging the authenticity of speech data by human perception alone is not enough anymore. Computer-aided methods are increasingly needed for fail-proof message authentication.

Traditional cryptographic integrity protection is designed to ensure that every bit in the data stream is unmodified. While these algorithms could be applied to speech data as well, speech data integrity should be aimed at the protection of the *content* (or called the *semantic meaning*), not the bitstream itself. We use *content integrity verification* to describe integrity verification algorithms that focus on the perceptual content of data. They allow *content preserving* operations to be performed without triggering the alarm. Such ability makes content integrity verification sys-

tems compatible with applications such as Internet servers that automatically adjust audio signal volume and perform transcoding. False alarm caused by random noise in the transmission channel should also be avoided.

2. Requirements for Content Integrity Systems

Audiovisual content integrity is a very new research field with 90 percent of papers in this field published in the last 2 years. There is currently a lot of different opinions about what content integrity should achieve. Numerous requirements have been suggested, but it is quite obvious that no system could achieve them all because some of them are mutually contradictory. In this section, we give a summary of all proposed requirements, and argue that some of them are mandatory and some are optional towards the goal of content integrity protection.

1. Compatible with lossy compression before transmission

Some tamper detection systems embed watermarks in the LSB of spatial domain samples, and demand that the image must be transmitted with lossless compression. This would impose an undesirable waste on the network bandwidth because lossless compression has much larger compressed data size than lossy compression. Therefore, new algorithms should meet this requirement.

2. To be able to perform blind detection

Audiovisual content integrity schemes that require the original data may only be suitable for proving the authenticity of data in a courtroom environment.

3. Introduced noise should be imperceptible

Obvious artifacts are not acceptable and the smaller the noise the better the method. In fact, traditional cryptographic hash function methods and feature extraction schemes extract signatures rather than embed watermarks.

4. To be able to tolerate content preserving operations in the transmission channel

Examples of content preserving operations include lossy

compression, amplification (or brightness adjusting), random noise and A/D-D/A conversion. The traditional cryptographic hash function method can detect any single bit of modification, but cannot tolerate content preserving operations in the transmission channel. Therefore, one of the main purposes of developing the new breed of content integrity verification schemes is to overcome this problem.

5. *Authentication data should be significantly smaller than carrying audiovisual data*

Authentication data such as the hash code or extracted content features have to be encrypted and attached to the carrying audiovisual data. They should be kept as small as possible to save the encryption/decryption time and the network bandwidth.

6. *To be able to detect localized malicious modifications*

Localized malicious modifications often dramatically change the semantic meaning of the whole content. For example, adding or dropping a single word “not” could reverse the meaning of a whole sentence. Thus, the ability to trigger the alarm when merely a small portion of audiovisual data is modified is crucial to a serious authentication system.

7. *Low computational cost at both the sender and receiver*

A high processing speed is vital to real-time audio and video applications. The current trend of mobile computing and the server computing also requires algorithms of a low computational cost.

8. *To be able to indicate the exact location of tampering*

Locating the exact location of modifications is a valuable function that often comes with a price. For some applications which only need to know whether the received data have been tampered, this requirement may be optional.

9. *Integrating authentication data with host media seamlessly*

Most modern multimedia compression standards such as MPEG audio include user-defined fields, and authentication data may be placed in these fields and integrated with the transmitted audiovisual data. With this approach, if audiovisual data are transcoded in the transmission channel, the transcoding mechanism must be able to copy authentication data into user-defined fields of the new compression format. To avoid this constraint, it is preferable to embed authentication data into the host media with watermarking techniques. However, previous work in watermark embedding often sacrifices some of the requirements discussed earlier. As a result, the choice between “integration with the user-defined field” and “watermark embedding” should be made based on the nature of the underlying application.

Based on arguments given above, it is concluded that Requirements 1 ~ 6 are mandatory for a reasonable content integrity verification system. Requirements 7 ~ 9 are desirable but not absolutely essential, and they should be accomplished only when Requirements 1 ~ 6 are not sacrificed.

3. Previous Work on Content Integrity Verification

3.1. Feature Extraction

The feature extraction method for integrity verification evolves from the traditional cryptographic hash function method. The hash function takes the large-size message as the input and produces a small fixed-size *hash code*. Changing any bit in the message would completely alter the hash code so that it is practically impossible to find a way to modify the message yet keeping the hash code remain the same. The hash code is encrypted and sent to the receiver along with the message. The receiver uses the same hash function on the received data and compares the result with decrypted hash code. The feature extraction method replaces the hash function with feature extraction, and the bit-wise comparison unit with threshold comparison. Depending on the application, the encryption unit in the diagram could be secret-key encryption, public-key encryption or other cryptographic methods.

One popular choice of the content feature for image authentication is the mean intensity of image blocks [1, 2, 3, 4]. The block size can be either fixed [1, 2] or variable [3]. This feature is insensitive to common content preserving operations, and calculation of this feature can be done very fast. The minor problem caused by image brightness adjustment can be solved by recording the intensity difference between blocks instead of the intensity of blocks [4]. However, the block mean intensity only covers low frequency information while high frequency features such as edges are very important to the semantic meaning of an image. Some image content tampering could be done without disturbing the block mean intensity feature. Another popular choice of the content feature is the edge information [5, 6, 7]. While the edge information does cover most semantic meanings of the image, it is still possible to modify an image without a significant change of the edge pattern. For example, we can add a mole to a face image and take special care of the mole boundary so that transition from white to black occurs smoothly. The gradient can be set to be just below the threshold of the edge detector.

The block mean intensity represents the low frequency content while the edge information covers the high frequency content. Thus, a secure content integrity verification scheme may be formed by combining both of them. However, the size of the combined features may become too large so that the total efficiency of the scheme is lowered.

The technique of *self-embedding* is a branch of feature extraction research, in which one tries to embed extracted feature values back into the audiovisual data with high data capacity watermarking. The purpose of self-embedding is to eliminate the need for a field in the header dedicated to

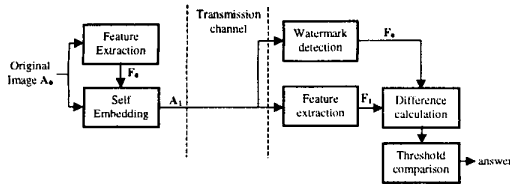


Figure 1. The self-embedding scheme

authentication data. This could be beneficial when the audiovisual data are transcoded or D/A-A/D converted in the transmission channel. However, due to the following two reasons we choose to avoid using this technique.

1. *Most self-embedding methods are not compatible with content preserving operations in the transmission channel. (violate requirement no. 4)*

In order to embed the content features back into the host data, the watermarking algorithm must have a very high data capacity. Therefore, almost all proposed work use some form of LSB embedding. Early work [8, 9] embedded content features directly into the LSB of spatial domain samples. Since lossy compression would completely destroy data embedded in LSB, the image could not even be compressed before transmission. This problem could be solved by embedding content features into the LSB of DCT coefficients during JPEG compression [10, 4]. However, most content preserving operations in the transmission channel, such as brightness adjusting, transcoding and D/A-A/D conversion, will alter the LSB of DCT coefficients and hence destroy embedded data.

2. *Most self-embedding methods have a high computational cost. (violate requirement no. 7)*

The blockdiagram of a self-embedding system is shown in Figure 1. The feature extracted from the original image A_0 is denoted as F_0 . After self-embedding F_0 into A_0 , the resulting image A_1 is transmitted. At the receiver side, the feature extraction algorithm obtains feature F_1 from A_1 , but watermark detection still obtains F_0 . Thus, there could be some mismatch between F_0 and F_1 , even though no attack of any kind occurred in the transmission channel. The difference between F_0 and F_1 might exceed the threshold and induce a false detection of tampering. In order to solve this problem, a method was proposed [2] to repeatedly extract features and embed them back into A_0 until F_i and F_{i+1} converges. According to [2], about 4 feature extraction and 4 self-embedding operations are needed for F_i to converge. Therefore, the computational cost is high.

3.2. Fragile Watermarking

Fragile watermarking embeds a secret sequence into the host audiovisual data. If the host data are tampered, the

secret sequence is also modified. The receiver calculates the correlation between the watermark sequence and the received data. If the correlation is beneath a threshold, it indicates that the data has been modified.

If the whole image is used to calculate one correlation value, a localized small modification cannot be noticed because it would not have much effect on the correlation value. Therefore, the image should be divided into small blocks, and one correlation value is calculated for each block [11, 12]. Since the original image is not used in the detection process to be subtracted from watermarked image, the block size must be large enough to avoid a high false detection rate. Lin *et al.* [12] used a block size of 8×8 so that localized modifications could be detected. The resulting false detection rate is around 10%. Fridrich [11] used a block size of 64×64 to decrease the false detection rate, but small modifications may not be detected. There is clearly a contradiction between a low false detection rate and the ability to detect localized tampering. Thus, some researchers feel that watermarking is intrinsically not well suited in protecting authenticity [3].

4. Proposed Algorithm

In Section 2, we listed 9 requirements for a reasonable content integrity verification in a descending order of importance. Since it is virtually impossible for one system to satisfy all of the requirements, we decide to fulfill the first eight of them. According to requirement no. 6, our system should determine the integrity of each 0.5 second of speech data, which is about the length of a single word in speech. Since most modern speech coders has a data rate lower than 800 bytes/sec, 0.5 second of speech is equivalent to less than 400 bytes of compressed data. It is very difficult to embed and detect a fragile watermark in such a small amount of host data while achieving reasonable false detection rate. Therefore, we choose the feature extraction method to implement speech integrity verification.

4.1. Speech Feature Extraction

The popular low-level features for audio/speech analysis include short-time energy function, average zero-crossing rate, short-time fundamental frequency and spectral peak track, etc. Extracting these features requires low computation, and most of them are fairly stable under content preserving operations. However, the large size of these features does not fit our requirement no. 5 and thus could not be directly used for content integrity verification.

In order to achieve smaller feature data size, we need to extract speech features that are more focused on the meaning of speech rather than the signal characteristics. We have

chosen 3 kinds of higher level features that are relevant to speech semantic meaning.

1. Pitch information

The pitch information represents the intonation of a sentence and the emphasized syllable of each word. The emphasized syllable in each word has a higher pitch than others.

2. The changing shape of the vocal tract

The shape of the vocal tract determines vowels, which are important to speech semantic meaning.

3. Energy envelope

The energy envelope of a speech signal controls the temporal location of each syllable and the number of syllables in each time frame. Controlling the energy envelope can detect whether any syllable has been deleted or added.

The computation cost of extracting these three features is higher than that of the low-level features, but we could dramatically reduce this cost by integrating feature extraction with speech coding.

4.2. Integration with ITU G.723.1 coder

ITU-T Recommendation G.723.1 “Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 kbit/s” is a very-low bit rate compression standard for telephone bandwidth speech signal. It is part of ITU H.324, and also recommended for ITU H.323. After its approval in 1996, ITU G.723.1 won wide support from the industry and is without doubt the dominant low bit-rate speech coder for Internet today.

The cost of our speech content feature extraction is greatly reduced by obtaining “pitch information” and “shape of the vocal tract” from G.723.1 coefficients instead of the raw signal. The block diagram for the proposed speech content integrity verification system is shown in Fig. 2. The G.723.1 coder and encryption/decryption units are standard modules, and the feature extraction, difference calculation and threshold comparison units are detailed below.

A. Feature extraction

Since the G.723.1 speech coder was designed base on a model of human speech generation mechanism, its coefficients capture the semantic content of speech pretty well. The “lag of pitch predictors” control the pitch information of speech, and the “LSP codebook indices” model the changing shape of the vocal tract. Among the 10 LSP coefficients in each frame, we only extract the first 3 as content features because they contribute the most to the model. The third feature, the energy envelope, is extracted as the ratio of average energy between adjacent frames of speech. Each frame in G.723.1 is equivalent to 30 ms. A summary of extracted features and their size are given in Table 1. The total size of the features is 27 bits per frame, which is about 14%

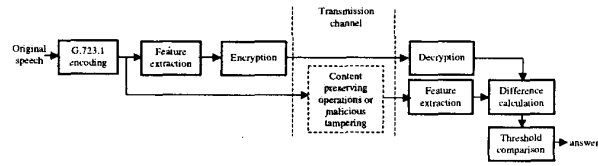


Figure 2. The blockdiagram of the proposed audio content integrity verification scheme

Feature name	Feature size (bits per frame)
LSP coefficients	8
Pitch information	14
Frame average energy	5

Table 1. Speech content features and their sizes.

of the compressed speech. The extraction of the first two features, i.e. the LSP coefficients and pitch information, from the G.723.1 coded bitstream requires no computation at all. The calculation of the frame average energy requires one multiplication operation (to obtain the sample energy) and one addition operation (to perform summation) for each speech sample.

B. Feature difference calculation

The feature difference calculation of the three features are done independently. For LSP coefficients, we take the average of the 3 LSP coefficients and then compute the difference between decrypted and extracted results. For pitch information and frame average energy, we also compute the difference between decrypted and extracted features.

C. Threshold comparison

Before differences are compared to the threshold, a low-pass filter is applied to the difference sequence. This step ensures that random burst-type errors do not trigger the false alarm. The low-pass filter is implemented with a moving averaging window.

5. Experimental Results

We test the ability of our system to distinguish between content preserving operations and malicious content modification in the transmission channel using a speech segment containing voice from six different persons with different gender and voice characteristics. The content preserving operations in our experiment include re-compression, amplifying, transcoding into MP3, resampling and D/A-A/D conversion. The detailed scenario of these operations are shown in Figure 3.

Figures 4(a)-(e) show the effect of these content preserving operations on the difference sequence of LSP co-

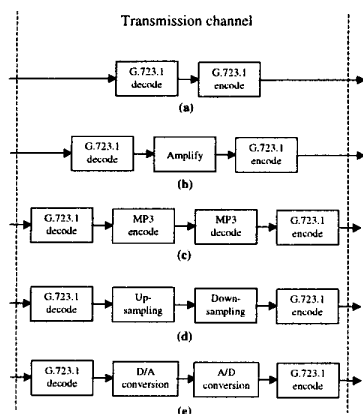


Figure 3. Illustration of various content preserving operations: (a)re-compression, (b)amplifying, (c)transcoding, (d)resampling, (e)D/A-A/D conversion.

efficient, and Figure 4(f) shows the effect of replacing the speech with another segment of speech. The horizontal axis in the charts represent the frame (30 ms/frame) number, and the vertical axis is the difference value after applying a moving averaging window. Note that the smallest value in (f) is still greater than any value in any other charts. The threshold could be set at 500 to clearly distinguish the two cases.

Similarly, the effect of content preserving operations on the pitch information and the difference sequence of frame average energy is relatively small and stable, and content changing operations cause a much higher difference value.

References

- [1] D.-C. Lou and J.-L. Liu. Fault resilient and compression tolerant digital signature for image authentication. *IEEE Transactions on Consumer Electronics*, 46(1):31–39, February 2000.
- [2] C. Rey and J.-L. Dugelay. Blind detection of malicious alterations on still images using robust watermarks. *IEE Seminar on Secure Images and Image Authentication*, pages 7/1 –7/6, April 2000.
- [3] M. Schneider and S. F. Chang. A robust content based digital signature for image authentication. *Proceedings of IEEE International Conference on Image Processing (ICIP'96)*, pages 227–230, 1996.
- [4] M. Wu and B. Liu. Watermarking for image authentication. *Proceedings of International Conference on Image Processing*, 2:437–441, October 1998.
- [5] J. Dittmann, A. Steinmetz, and R. Steinmetz. Content-based digital signature for motion pictures authentication and content-fragile watermarking. *IEEE International Conference on Multimedia Computing and Systems*, 2:209–213, June 1999.

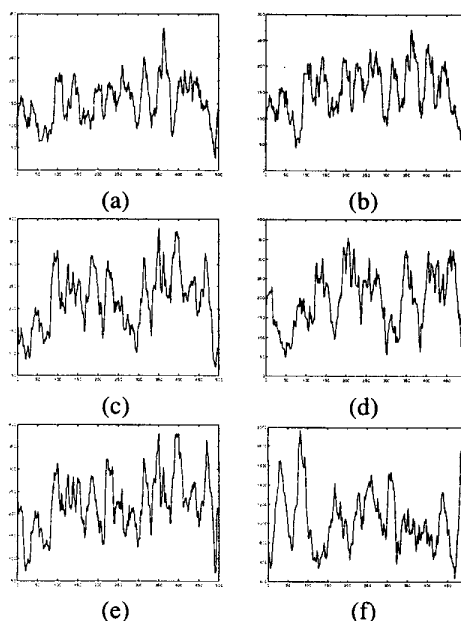


Figure 4. The effect of various operations on the LSP feature difference calculation due to (a) re-compression, (b) amplifying, (c) transcoding, (d) resampling, (e) D/A-A/D conversion, and (f) malicious content changing

- [6] M. P. Queluz. Towards robust, content based techniques for image authentication. *Proceedings of IEEE Signal Processing Society 1998 Workshop on Multimedia Signal Processing*, December 1998.
- [7] M. Steinder, S. Iren, and P. D. Amer. Progressively authenticated image transmission. *MILCOM 1999*, 1:641–645, November 1999.
- [8] J. Lee and C. S. Won. A watermarking sequence using parities of error control coding for image authentication and correction. *IEEE Transactions on Consumer Electronics*, 46(2):313–317, May 2000.
- [9] P. W. Wong. A public key watermark for image verification and authentication. *Proceedings of IEEE International Conf. on Image Processing*, pages 455–459, October 1998.
- [10] J. Fridrich and M. Golijan. Images with self-correcting capabilities. *Proceedings of International Conference on Image Processing*, 3:792–796, 1999.
- [11] J. Fridrich. Image watermarking for tamper detection. *Proc. ICIP '98*, October 1998.
- [12] E. T. Lin, C. I. Podilchuk, and E. J. Delp. Detection of image alterations using semi-fragile watermarks. *SPIE International Conf. on Security and Watermarking of Multimedia Contents II*, 3971(14), January 2000.