A real-time video watermarking algorithm for streaming media

CHENG Chunling, WAN Teng, WEI Lei, ZHU Hong, ZHANG Dengyin

E-mail: chengcl@njupt.edu.cn

1College of Computer, Nanjing University of Posts and Telecommunications, Nanjing, Jiangsu 210003, China
2Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing, Jiangsu 210003, China
3Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education Jiangsu Province, Nanjing, Jiangsu 210003, China
4Power Dispatching Centre, Nanjing Power Supply Company, Nanjing 210008, China

Abstract

Video watermarking technology is a useful method to track piracy for digital videos. However, most watermarking algorithms cannot be directly applied to H.264 streaming media because of their poor real-time performance, which may cause users to watch the video unsmoothly. In order to solve this problem, a real-time H.264 video watermarking algorithm (RVWA) is proposed. First, the video is segmented before embedding watermark, which makes watermark embedding and video transmission be carried on at the same time. Then, the concept of watermark embedding margin of frame (WEMF) is proposed to shorten the watermark embedding time. By calculating the value of WEMF for each B frame, which is related to its frame bits, suitable B-frames can be selected to embed watermark without decoding it completely. Finally, with the characteristic of variable macro block size in H.264 coding standard, a simplified formula is presented to quickly calculate the just noticeable difference (JND) value of the macro block, and then the macro blocks with larger JND value are selected to embed watermark into their motion vectors, which also provides good invisibility. Experimental results show that the proposed algorithm can find suitable embedding positions and embed watermark quickly with good invisibility and less increase of the bit-rate of video. RVWA algorithm can be used in stream media environment in real time.

Keywords: Video Watermark; Streaming Media; Piracy Tracking; H.264; Motion Vector

1. Introduction

Streaming media greatly facilitates the user to get video resources, but it makes piracy easier. Now copyright is not only an important research subject but also a serious social problem[1]. Digital watermarking technology, as a useful copyright protection method, can effectively protect the interests of business. At present, H.264 is widely used in streaming media. Developed by the ITU-T and ISO/IEC, H.264 videos not only have high compression ratio, but also have high-quality images. Therefore, the study of H.264 piracy track watermark has positive meaning.

In order to track piracy, the merchant needs to embed the user identity information into the video as a watermark when he receives the user's playback request. As the user is uncertain, the user information cannot be embedded into the video in advance. Moreover, the embedding cannot affect the user to watch video smoothly. Therefore, some real-time streaming media watermark algorithms were proposed. In reference [2], feature information of each frame is extracted as a watermark which is embedded into diagonals of DCT coefficients matrix. Because the watermark is embedded in the transform domain, the algorithm has high complexity, and the amount of bits of watermarked video increases significantly, which burdens the load of network transmission [3]. In reference [4], the motion vector of the video was first regarded as watermark carrier and a watermarking algorithm based on motion vector was proposed. It embeds watermark through modifying the motion vector directly. This algorithm is simple, but the invisibility is poor. Based on the idea in reference [4], a H.264 video watermarking
algorithm based on phase angle of motion vector was proposed [5], which realized watermark embedding by dividing \([0, 2\pi]\) into two different angle section sets and rotating the motion vector by a fixed tiny value. The algorithm complexity is low, but the invisibility is poor. Apart from these, a video watermarking scheme based on the region character of motion vectors was proposed [6]. Its invisibility is good, but the watermark embedding must carry on while the video is being encoded, which requires more time. In reference [7], the watermark embedding was performed using the elements of the compressed bit stream. The watermark embedding time is short. But the algorithm has poor visibility, and is likely to cause the video decoding error. In reference [8], the watermark was embedded in a distributed way, which is implemented by hardware. The algorithm can meet the real-time requirements, but it does not reduce the complexity of the algorithm itself. In addition, it requires hardware investment.

Above H.264 video watermark algorithms cannot be directly applied to streaming media environment because of the long embedding time and the obvious increase of the bit-rate of video. To solve this problem, a real-time video watermarking algorithm for H.264 video (RVWA for short) is presented in this paper. To ensure users watch online video smoothly, the video is segmented before embedding. Considering the fact that part of the video frames are not suitable for watermark embedding because of high compression ratio of H.264 video, this paper presents the concept of watermark embedding margin of frame (WEMF) and the formula to calculate its value of B-frames. According to the WEMF value, suitable frames can be selected to embed watermark without complete decoding the video. Furthermore, combining with the H.264 coding standard, a simplified formula of calculating value of the just noticeable difference (JND) is given to select macro blocks quickly.

2. Preprocess of Video

In order to ensure that users watch online video smoothly, watermark embedding and video transmission should be carried on at the same time. Therefore, the video is segmented. The first segment is transmitted to the user directly and the watermark is embedded into the remaining segments, so that the watermark is embedded as the previous segments are transmitted. In view of the fact that some users may not watch the full video, the segments are numbered in the playing sequence and then organized as a binary tree structure as Figure 1. While the user watches segments in one layer, the watermark is embedded into the segments in the next layer, instead of all the video segments.

![Figure 1. Structure of the video segments](image)

3. Selection of The Video Frame

Because P-frames of H.264 video contain more motion vectors and the quantity of them is large [9], most of the existing H.264 video algorithms embed watermarking into P frames based on motion vectors. However, P-frame can be used as reference frame, in which the watermark embedding will cause a larger inter-frame error, so the quality of the video may decline seriously [10]. B-frame is not used as a reference frame in the streaming media, so in this paper, B-frame is used to embed watermark, which has the advantage of little degrading of the video quality. However, comparing with P-frames, B-frames have less motion vectors, so some B frames are not suitable for embedding watermark. Therefore B-frames should be filtered before embedding watermark.

According to the H.264 coding rules, if a macro block is identical to its reference macro block, it is encoded into a skip macro block, in which both motion vector residuals and pixel
residuals are 0. The more skip macro blocks one B-frame contains, the less motion vectors B-frame contains and the less suitable for embedding watermark.

As only the number and the type of the reference macro block of the skip macro block macro block are written into bit stream, which make the bits of it smaller than the normal macro block. Therefore, the B-frame can be quickly judged whether it is suitable for watermark embedding according to its bits, without decoding it completely. Accordingly, the concept of watermark embedding margin of frame is proposed in this paper.

Definition 1: Watermark Embedding Margin of Frame: the ratio of B-frame bits and I-frame bits which are in the same GOP, denoted by $WEMF(i)$.

$$WEMF(i) = \frac{Bf_{bit}(i)}{If_{bit}(i)}$$  \tag{1}

Where $i$ is the B-frame number, $Bf_{bit}(i)$ is the bits of the $i$-th B-frame, $If_{bit}(i)$ is the bits of I-frames which is in the same GOP as the B-frame. Let $T$ be the embedding threshold. If $FJND(i) > T$, the B-frames is suitable for watermark embedding. Embedding Threshold $T$ is calculated according to formula 2 before embedding.

$$T = \alpha \frac{1}{n} \sum_{i=1}^{n} \frac{Bf_{bit}(i)}{If_{bit}(i)}$$  \tag{2}

Where $n$ is the number of B-frames, $\alpha$ is the watermark embedding strength factor, the greater its value is, the less watermark contains in the video. In our experiments, the value of $\alpha$ is 1, which is to ensure that half of the B frames are embedded watermark.

4. Selection of Macro Block

To improve the invisibility of the watermark algorithm, the proposed algorithm selects appropriate macro blocks to embed watermark adaptively according to the video content. According to the reference [11], if the just noticeable difference ($JND$) of macro block is big, the watermarked macro block is less likely to be noticed. In order to achieve better invisibility, algorithm embeds watermark into macro blocks with larger $JND$ value.

According to the H.264 coding standard, larger macro blocks are mainly used for encoding flat areas of the frame, whose $JND$ value is smaller. On the contrary, smaller macro blocks are mainly used for encoding dramatic changing areas, whose $JND$ value is larger. In reference [12], a formula is given to calculate the $JND$ of each macro block, but it needs to calculate the deformation of the macro block which is complex. Therefore, a simplified formula for rapid calculation of the $JND$ value of each macro block is presented:

$$JND = \frac{JSIZE}{4} \arccos \frac{mvp(V_x, V_y) \cdot mv(V_x, V_y)}{|mvp(V_x, V_y)| \times |mv(V_x, V_y)|} \sqrt{V_x^2 + V_y^2}$$  \tag{3}

Where $V_x$ and $V_y$ represent the horizontal and vertical components respectively; $mv(V_x, V_y)$ represents the motion vector of the macro block; $mvp(V_x, V_y)$ represents the motion vector of the predicted macro block; the value of $JSIZE$ relates to the size of the macro block:

$$JSIZE = \begin{cases} 
1 & block \ 4 \times \ 4, \\
2 & block \ 4 \times \ 8, \ 8 \times \ 4, \ 8 \times \ 8 \\
3 & block \ 8 \times \ 16, \ 16 \times \ 8 \\
4 & block \ 16 \times \ 16 
\end{cases}$$  \tag{4}
After calculating the JND of the macro blocks, the proposed algorithm selects all the macro blocks whose JND is larger than $T_1$, sorts them in descending and embeds watermark to the motion vector of these macro blocks in sequence. $T_1$ is the selection threshold which is used to select the appropriate macro block. As the H.264-encoded pixel precision is 1/4 pixel, 2-bit watermark information will be embedded into each motion vector. Taking into account the difference of the horizontal and vertical component of the motion vector, different strategies are adopted while watermarking embedding: If $||V_x|-|V_y|| \leq T_2$, one bit watermark information is embedded to both motion vector component; otherwise, 2 bit watermark information is embedded to the motion vector component whose absolute value is larger. $T_2$ is the disparity threshold which is used to select the appropriate motion vector.

5. Steps of the Algorithm

If there are several motion vectors in a macro block in B frame, the motion vector with the largest motion amplitude is called as the motion vector of the macro block. The user identity information is encoded 32-bit binary number as watermark, denoted by $W$; each video segment contains $n$ frames. Steps of the proposed algorithm are as follows:

Step1: Divide the video into segments which have the same length, i.e., 1 minute long.

Step2: $i=1$.

Step3: Decode the video segments: supposing that the $i$-th frame is to be decoded, it will be determined whether it is a B frame according to its head information. If it is B frame, only its head is decoded to get the quantity of bit of the frame; otherwise, the whole frame should be decoded completely. And then calculate the \( WEMF(i) \) of B-frames Using formula 1; if \( WEMF(i) > T \), go to step 4 to embed watermark into the $i$-th frame; otherwise, jump over the frame, $i = i + 1$. Repeat Step 3 until $i = n$.

Step4: Calculate the JND of each macro block in B-frames embedding watermark according to formula 3.

Step5: Sort the macro blocks whose JND value is larger than $T_1$ by their JND value in descending order using the quick sort algorithm, and select the first 16 macro blocks to embed watermark.

Step6: Embed watermark into the macro blocks in sorted order. After one macro block complete the watermark embedding, $j=j+1$, the $j$ is 1 at the beginning. When $j=16$, the watermark embedding is over. Goto the step 2.

Detailed steps of watermark embedding are as follows: Supposing the $j$-th macro block is to embed watermark, the $V_x$ and $V_y$ of the macro block will be compared first. If $||V_x|-|V_y|| \leq T_2$, then respectively embed 1 bit watermark information into either component of its motion vector. In other word, embed the $2j-1$-th bit of $W$ to $V_x$ and the $2j$-th bit to $V_y$, which means modify the value of $|v|$ according to formula 5. Where $v$ is $V_x$ or $V_y$, $\lambda$ is the value of the embedded watermark information; mod is the modulo operator. If $v$ is negative, $v=-|v|$; otherwise $v=|v|$.

\[
|v| = \begin{cases} 
|v| & (\lambda - (v \times 4 \mod 2) \mod 2 = 0) \\
|v| + 0.25 & (\lambda - (v \times 4 \mod 2) \mod 2 = 1)
\end{cases}
\] (5)

If $||V_x|-|V_y|| > T_2$, embed the $2j-1$-th and the $2j$-th bit of $W$ to the component whose absolute value is larger, which means modify the value of $|v|$ according to formula 6. Where $v$ is the component whose absolute value is larger, $|v|=\max(|V_x|,|V_y|)/\lambda$ is the value of the embedded watermark information; mod is the modulo operator. If $v$ is negative, the final $v= -|v|$; otherwise $v=|v|$.
The watermark algorithm described above is a blind watermark algorithm, which means that the watermark extraction does not require the original video. Watermark extraction process is similar to the embedding process. After decoding video, the watermarking can be extracted from the motion vectors according to the reverse derivation of the formula 5 and formula 6.

6. Experiments and Results Analysis

Several experiments are performed to evaluate the watermarking algorithm from the real-time performance, invisibility and increase of the video bits. The watermark embedding is implemented by C language. The H.264 coding is implemented by H.264 JM-8.6 reference software [13]. The experiments are carried out on the computer which has dual-core 1.66GHz CPU and 1GB memory.

Three standard video sequences (foreman, carphone, coastguard [5]) were used for our experiments. All sequences are encoded to H.264 video where there are 199 frames, in which 99 frames are B-frames. The GOP structure of video is "IBPBP ..."; and each GOP contains 25 frames, in which the first frame is the IDR frame; watermark embedding strength factor $\alpha$ is 1, the threshold value $T_1$ is 4 and $T_2$ is 3.

6.1. Watermark Embedding Time

Jordan’s algorithm is one of the typical watermarking algorithms and the embedding time of it is short enough. In order to verify the proposed algorithm can shorten the watermark embedding time, it is compared with Jordan’s algorithm. Table 1 shows the time that the two algorithms spend in embedding watermark in 199 frames. It can be seen that watermark embedding time of RVWA is shorter because it shortens the time to find the location of the watermark embedding.

<table>
<thead>
<tr>
<th>Video</th>
<th>Jordan (s)</th>
<th>RVWA (s)</th>
<th>Reduced time ratio (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>foreman</td>
<td>17.172</td>
<td>15.621</td>
<td>9.03</td>
</tr>
<tr>
<td>carphone</td>
<td>18.427</td>
<td>16.576</td>
<td>10.04</td>
</tr>
<tr>
<td>coastguard</td>
<td>18.685</td>
<td>16.782</td>
<td>10.18</td>
</tr>
</tbody>
</table>

In addition, a typical streaming media scenario is taken to analyze whether RVWA algorithm is suitable for stream media environment. Suppose that the length of the streaming video is 100 minutes and the video size is 320M. The video is divided into 100 segments, each of which is 1 minute long; generally, streaming media server limit the user's download speed according to the video quality. 200Kb/s download speed is used for analysis: Generally, video frame ratio is 25fps, which means 25 frames per second. If the watermark is embedded to each frame and the total time of watermark embedding and transmission of each frame is less than 1000ms/25 = 40ms, the embedding algorithm would be thought to satisfy the real-time requirement. In above experiments, the average embedded time is 100ms per frame. Because the streaming media server has dedicated decoder chip and its CPU is more powerful than the computer which is used in our experiments, the watermark embedding time will be shorter. The average watermark embedding time of each frame is roughly as: $T_{\text{embed}} = 20$ms, the average transmission time of each frame is: $T_{\text{transmit}} = 10.9$ms. It can be seen that the total time of $T_{\text{embed}}$ and $T_{\text{transmit}}$ is less
than 40ms. Therefore, RVWA algorithm can meet real-time requirement and does not affect the video playing.

6.2. Invisibility Test

PSNR (peak signal to noise ratio) is used to analysis the invisibility of the algorithm from an objective point of view. PSNR is defined as follows[5]:

$$\text{PSNR} = \frac{255^2}{\text{MSE}}$$  \hspace{1cm} (7)

Where the MSE means square error, calculated as follows:

$$\text{MSE} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} (f_{ij} - f_{ij}^w)^2}{M \times N}$$  \hspace{1cm} (8)

Where $f_{ij}$ and $f_{ij}^w$ respectively represent the pixel value of the original image and watermarked image in the position $(i,j)$, $M, N$ are the height and width of the image respectively, in pixels.

Figure 2 gives the PSNR of three original frames and their watermarked frames, where blue squares represent the original frames and red points represent the watermarked frames. It can be seen that most PSNR of frames change little except a few frames. This is caused by the fact that WEMF value of these frames are at the zero boundary point, which means a watermark can be just embedded in these frames. Watermark information is embedded into the smaller JND value of the macro blocks. By adjusting the threshold value $T$, watermark can not to be embedded in these frames.

![Figure 2. Comparison of PSNR](image)

Figure 3 shows the subjective comparison of the Invisibility. In figure 3, (a) is original video sequences of foreman, carphone and coastguard, while (b) is the corresponding watermarked images. It can be seen that they cannot be distinguished from the subjectivity. The proposed algorithm can meet the requirement of invisibility.

![Figure 3. Comparison of Subjectivity](image)
6.3. Increase of The Video Bits

Table 2 shows the bits of all the B frames of original video and the watermarked video. The second column of the table 2 shows the bits of the 99 B frames in original videos. Similarly, the third column shows the bits of them in watermarked videos. The column of “Increased ratio” shows that the bits of watermarked video increase little, because the RVWA chooses the motion vector to embed watermark. So it has little influence on the transmission of streaming video.

<table>
<thead>
<tr>
<th>Video</th>
<th>Original bits (bit)</th>
<th>Watermarked bits (bit)</th>
<th>Increase ratio (%)</th>
<th>Number of watermark</th>
</tr>
</thead>
<tbody>
<tr>
<td>foreman</td>
<td>128528</td>
<td>129040</td>
<td>0.39</td>
<td>54</td>
</tr>
<tr>
<td>carphone</td>
<td>101528</td>
<td>101752</td>
<td>0.22</td>
<td>45</td>
</tr>
<tr>
<td>coastguard</td>
<td>66408</td>
<td>66840</td>
<td>0.65</td>
<td>49</td>
</tr>
</tbody>
</table>

7. Conclusions

In order to solve the problem that existing H.264 video watermark algorithms cannot be directly applied to streaming media because of the long embedding time and the obvious increase of the bit-rate of video, a real-time video watermarking algorithm for H.264 streaming media is presented. Combining with the characteristics of the transmission of streaming media, the video is segmented before watermark embedding, which can ensure that users watch online video smoothly. By calculating the value of $WEMF$ for each B frame, suitable B-frames can be selected to embed watermark without decoding it completely, which shortens the embedding time. Furthermore, by calculating the $JND$ value of the macro block according to the simplified formula 3, the appropriate watermark embedding positions can be quickly selected and the visibility is also good. All these realize the real-time watermark embedding, which ensure the proposed algorithm be suitable for streaming video.

The algorithm in this paper focuses on the efficiency and invisibility of watermark algorithm, but considers less on the robustness. So the future work is to improve the security of algorithm.

8. Acknowledgement

The work was partially supported by Swedish Research Links [No.348-2008-6212], the National Natural Science Foundation of China [61071093], China’s Project 863 [2010AA701202], SRF for ROCS, SEM [NJ209002], and the Priority Academic Program Development of Jiangsu Higher Education Institutions [yx002001].

9. References

A real-time video watermarking algorithm for streaming media
CHENG Chunling, WAN Teng, WEI Lei, ZHU Hong, ZHANG Dengyin


