AUTOMATIC IDENTIFICATION OF COMPRESSED VIDEO

R. Lancini, F. Mapelli and A. Mucedero

CEFRIEL – Politecnico di Milano
Via R.Fucini, 2 - 20133 – Milano - ITALY
E-mail: {rosa, mapelli}@cefriel.it

ABSTRACT

The goal of the paper is to present a novel technique for automatic identification of videos. This technique is based on a hashing algorithm which analyzes the video in order to extract its fingerprinting or hash value. This fingerprint allows the unambiguous. Two main aspects are considered: the hash extraction process and the database strategy to retrieve information. It is also proposed an analysis of the false positive error probability in order to identify an identification threshold. The proposed technique is tested under different kind of compression using MPEG standard and DivX;-) algorithm. The results show that a reliable identification can be performed.

1. INTRODUCTION

The number of multimedia information that can be accessed is increasing every day. It can be useful to find a way to retrieve automatically information about the data. This can be done in different ways, but two main categories can be identified [2]: techniques that actively modify the signal to add information and techniques which do not modify the data. Watermarking is an example of the first class. It hides data in the original signal in order to identify it. Hashing – or fingerprinting – belongs to the second class [1],[3],[5]. In this case we do not modify the signal but an analysis is performed in order to extract the most important features that allow the unambiguously identification. Some approaches are proposed to use watermarking and fingerprinting together as in [4].

The algorithm proposed in this paper belongs to the class of hashing techniques. The main idea is to extract for each frame or group of frames their distinguishing features. These features are organized and stored as hash value in a database. When it is requested to identify a video we extract its hash value and comparing it with the database we retrieve the needed information. Two main parts can be identified for a hashing algorithm: the hash extraction and the database strategies.

2. HASH VALUE EXTRACTION

The hash value extraction is the first important phase in the hashing technique: we have to choose the best features to represent the video.

This task is split in two parts: *pre-processing* – which converts the original signal to a "standard" signal – and *hash computing* - to extract the hash value using robust features.

2.1. Pre-processing

In this part the original signal is converted to a standard version. Figure 1 shows the main step of the pre-processing algorithm. For each single frame of the sequence the following steps are performed:

- extraction of the luminance component;
- resampling to 360*288;
- low-pass filtering;
- downsampling in both direction of a factor 2.

Finally we have each frame represented by a 180*144 pixels signal.

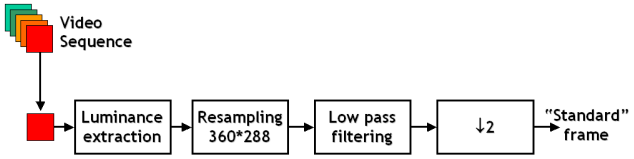

Figure 1 – Pre-processing phase.

2.2. Hash computing

This part performs the extraction of the hash value from the "standard" signal (see Figure 2), the main phases are:

- variance matrix construction;
- block splitting;
- minima selection.

These steps are detailed in the following paragraphs and in Figure 3.

Variance matrix construction

For each pixel a local variance is computed considering a *(2*K+1)* (2*K+1)* square block – where *K* is the

parameter used to identify the square size. A new matrix is built considering the variance value in the same position of the pixel. If a pixel is closer than *K* pixel to the border no variance is computed. So the size of the variance matrix is smaller than the original matrix. In our case we use *K=7*: considering the standard frame of 180*144 pixels we obtain a final size for the matrix of 166*130.

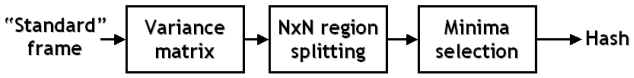

Figure 2 – Hash computing phase.

Block splitting

We perform a block splitting on the variance matrix in order to analyze it. We consider non-overlapped block of size *W*W*. Considering *W*=16 - borders block are bigger in order to cover the whole frame - we have *N*=(166/16)*(130/16)=80 blocks.

Minima selection

This is the last step in the hash extraction algorithm: in each block we extract the position of the minimum – also called star. We can have two different cases:

- minimum on the block border: it is not stored;
- minimum inside the block: it is stored.

We remove the minima on the borders because under attacks they easily move to neighbor block. The obtained matrix – which is full of "0" except in the position of a minimum where we have a "1" - is called constellation and represents our hash.

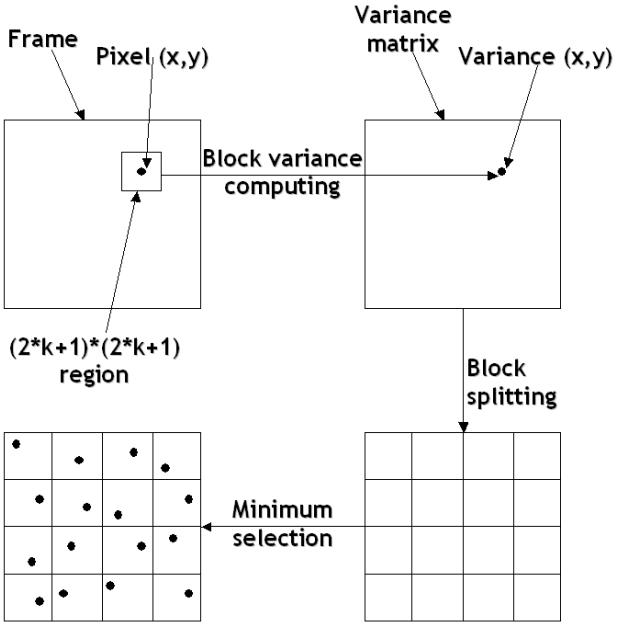

Figure 3 – Detailed hash extraction phase.

2.3. Hash storage

The hash value is stored as a vector containing two kinds of data: the number of minima and their position – see Figure 4. In this way we represents in a very compact way the hash. It is not necessary to store also the constellation size because the algorithm fixes it.

The main advantage of this approach is the very small amount of information that we need to store. Moreover it is simple to restore the original map.

We follow this approach for two reasons:

- storing the constellation size we can easily find out which are the frame with similar amount of minima and comparing only them;
- we need a small amount of memory to store information: the maximum number of minima (and one for the size).

Minima number	1st min	2nd min	3rd min		Nth min

Figure 4 - Hash vector.

3. VIDEO RETRIEVAL

This is the second main phase of hashing algorithm. Once extracted the hash value we need to find it in a database in order to retrieve the related information. The most important issue to take into account is the speed: we need to find this information as fast as possible.

Looking for each single frame hash in the whole database produces a very slow algorithm. In order to improve the efficiency we use some optimizations:

- constellation merging: we put together the constellation of *M* frames and we look for this in the database;
- frame skipping: we consider separated group of *M* frames in the database;
- once identified the group we refine the search to identify the frames, if it is needed.

4. ERROR PROBABILITY ANALYSIS

To analyze the error probability we consider the minima position as a random process i.i.d. To model the error probability we assume that the position in each block can be chosen from a uniform probability. We can choose W^2 different positions for a minimum and the case of no minimum in a block: so we can choose among W^2+1 cases. For a single block the probability to have exactly the same position is:

$$p_T = \frac{1}{W^2+1}.$$

The probability to not choose the same position is:

$$p_F = \left(1 - \frac{1}{W^2+1}\right) = \frac{W^2}{W^2+1}.$$

Considering N blocks the probability to have S=0 errors extracting random position is:

$$P(S=0) = \left(\frac{1}{W^2+1}\right)^N.$$

In general the probability to find S errors in N blocks is:

$$P(S) = \left(\frac{1}{W^2+1}\right)^{N-S}\left(\frac{W^2}{W^2+1}\right)^S = \frac{\left(W^2\right)^S}{\left(W^2+1\right)^N}.$$

Given S errors – i.e. the number of minima in different position between two hashes – and N blocks we assume $P(S)$ as the false positive error probability. This theoretical value is, in practice, quite different due to the fact that the complete independency is not achieved. For this reason we choose a quite high value for S as threshold.

In the tests phase, the results are reported as the percentage of wrong minima, i.e. the value $E=S/N$. This value E represents the parameter used to check the identification. As threshold value we choose E=0.20:

- if E>0.20 we identify the frame;
- if E<0.20 we do not identify the frame.

This threshold is chosen considering two aspects:

- the number of minima in common between the same videos after compression can be quite low: we can have E=30%;
- the number of minima in common between different videos can be quite high: we can have E>10%.

5. TESTS

We performed different tests in order to evaluate the efficiency of our algorithm. In our tests we use different videos in PAL format – i.e. frame size of 720x576 pixels.

The first test verifies the correct identification of frames in uncompressed video. This means that taking a random frame from a video we are able to correctly identify the video sequence and the frame position in it. In this condition each frame is correctly identified in all our tests.

Two issues have to be considered:

- the number of corresponding minima increases for frames close to the considered one (as in Figure 5);
- the number of minima in common between frames of the same sequence is in general higher than the number in common for different sequences (see Figure 6).

In the following paragraph we show the results obtained in identification of videos compressed with different techniques: MPEG-1, MPEG-2, MPEG-4 and DivX;-). In each test we compress videos at different bit-rates then we extract the hash for each video and retrieve the information from the database. The results are obtained extracting the hash from the compressed video and comparing it with all hashes from all videos. In the tables of the following paragraphs we report the results as the percentage of common minima: $E=S/N$, as explained in paragraph 4.

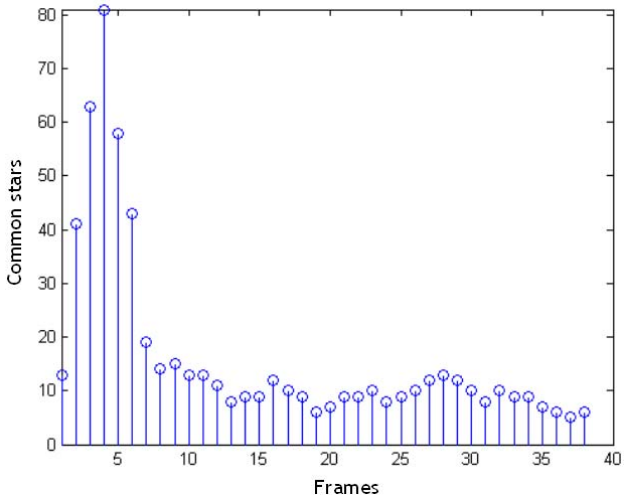

Figure 5 – Comparison between the hash of selected frames and hash of all frames of the video.

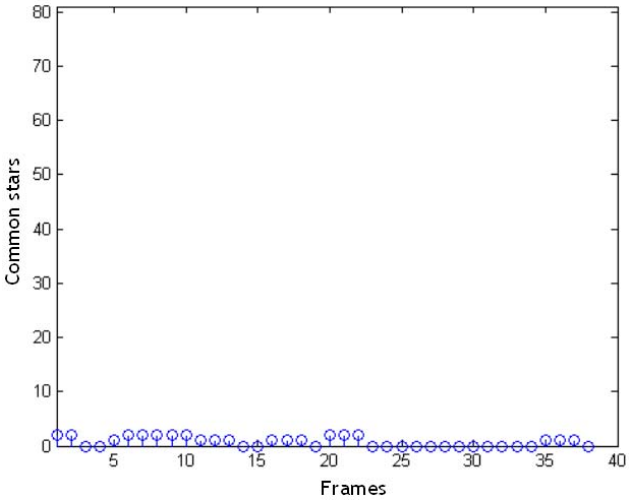

Figure 6 – Comparison between the hash of selected frames and hash of all frames of a different video.

5.1. MPEG-1

The video is tested for compression ranging from 500kbps to 2000kbps. In Table 1 we show the comparison between different videos for compression at 500 kbps. V1,…,V9 are 9 different original videos, while CV1,…,CV9 are the compressed version.

The highlighted values show the comparison between the original and compressed versions of the same video.

The minimum value in this case is about 28%. The comparison between different videos– value out of the diagonal - always results in very low value, below 12%.

These values compared with the chosen threshold provide good results: all videos are correctly retrieved and there are no false positive cases.

	V1	V2	V3	V4	V5	V6	V7	V8	V9
CV1	**46,1**	6,5	7,8	3,0	5,8	5,1	6,0	4,8	2,5
CV2	5,9	**27,9**	6,4	2,0	5,7	5,0	4,1	4,1	3,4
CV3	8,6	7,0	**35,8**	2,6	6,2	5,5	5,7	4,4	2,0
CV4	5,4	3,6	4,2	**78,0**	4,8	4,2	3,6	6,1	3,0
CV5	7,0	5,1	5,6	3,0	**66,1**	5,8	6,0	4,6	1,6
CV6	4,9	4,9	4,2	4,2	5,2	**35,8**	4,2	2,9	2,9
CV7	11,8	7,9	7,5	4,7	10,2	5,1	**64,8**	8,7	2,3
CV8	8,9	6,9	7,3	5,9	7,9	4,3	9,9	**66,4**	1,6
CV9	5,4	8,1	5,9	4,8	3,7	7,5	3,2	2,1	**44,8**

Table 1 - Comparison results obtained for MPEG-1 compression at 500kbps.

5.2. MPEG-2

The tests are performed for compressions ranging from 500kbps to 10000kbps. Table 2 shows the results for 500kbps MPEG2 compression.

	V1	V2	V3	V4	V5	V6	V7	V8	V9
CV1	**27,9**	6,8	6,3	0,0	6,1	4,2	4,6	5,1	0,0
CV2	6,2	**23,0**	5,7	0,0	5,0	4,1	3,0	3,2	0,0
CV3	7,2	5,7	**26,1**	0,0	5,7	3,7	4,7	3,7	1,5
CV4	0,0	1,7	2,3	**63,0**	2,3	1,7	2,8	4,0	2,3
CV5	7,6	5,4	6,1	0,0	**53,5**	5,7	6,1	5,2	0,0
CV6	3,8	2,8	4,1	2,5	4,8	**30,7**	3,5	2,8	2,2
CV7	5,6	6,0	7,4	4,6	9,2	4,6	**54,4**	9,2	2,4
CV8	4,3	4,7	5,8	6,2	8,0	4,3	8,3	**55,1**	2,1
CV9	0,0	2,0	4,5	2,5	2,0	4,0	2,0	2,5	**33,5**

Table 2 - Comparison results obtained for MPEG-2 compression at 500kbps.

In this case the minimum value for a comparison between the original and compressed versions of the same video is 23% which is above the threshold. In case of comparison between different videos the maximum value is below 10%, which is heavily below the threshold.

5.3. MPEG-4 and DivX;-)

These tests are discussed together because the results are almost similar. We test the algorithm mainly for low bit-rates coding, ranging from 250kbps to 1000kbps. Table 3 shows the results in case of compression at 250kbps.

In this case we have the minimum value comparing the same video that is about 23%, and the maximum for different videos comparison below 10%. Even in this case the identification can be performed without any false positive errors.

	V1	V2	V3	V4	V5	V6	V7	V8	V9
CV1	**29,3**	5,5	6,3	2,5	6,0	4,3	6,0	4,8	1,2
CV2	5,5	**27,1**	5,5	0,0	4,6	3,6	3,6	3,1	0,0
CV3	8,2	7,1	**23,5**	2,0	6,1	3,8	4,3	4,3	2,0
CV4	0,0	0,0	1,9	**62,6**	1,9	0,0	2,5	4,4	3,1
CV5	6,5	4,9	5,1	0,0	**55,1**	6,0	6,3	4,6	0,0
CV6	2,4	2,8	5,3	3,2	6,1	**33,8**	5,3	3,2	3,6
CV7	3,2	7,4	5,3	4,5	8,2	4,5	**51,8**	8,6	1,6
CV8	3,4	3,8	3,4	5,3	8,0	4,2	6,9	**58,8**	2,3
CV9	0,0	0,0	0,6	3,4	2,0	4,1	3,4	2,7	**25,0**

Table 3 - Comparison results obtained for DivX compression at 250kbps.

6. CONCLUSIONS

In this paper we presented a novel technique for automatic identification of digital videos that can be compressed by different technologies. The algorithm is based on a hashing technique that identifies the video extracting its fingerprint. The results showed the efficiency of the algorithm at different bit-rates. We were able to identify the video even for high compression like DivX;-) down to 250kbps.

It is difficult a numerical comparison with other techniques due to the fact that each algorithm has its own parameter. From a practical point of view our approach has a simpler approach than most of the techniques but it provides similar results and robustness.

Our present works is focused on two main aspects to improve the algorithm: the error probability analysis and the database strategies.

7. REFERENCES

[1] V.Fotopoulos and A.N. Skodras, "A new fingerprinting method for digital images," *Proc. 1st IEEE Balkan Conference on Signal Processing, Communications, Circuits and Systems*, Istanbul, Turkey, June 1-3, 2000.

[2] T.Kalker, J.Haitsma and J.Oostveen "Issues with Digital Watermarking and Perceptual Hashing", *SPIE Conference on Multimedia Systems & Applications*, August 2001.

[3] T.Kalker, J.Haitsma and J.Oostveen "Visual Hashing of Digital Video: applications and techniques", *SPIE Applications of Digital Image Processing,* San Diego, USA, August 2001.

[4] D.Kirovski, H.Malvar, and Y.Yacobi "Multimedia Content Screening using a Dual Watermarking and Fingerprinting System", *ACM Multimedia* 2002.

[5] R.Venkatesan, S-M.Koon, M.H.Jakubowski and P.Moulin "Robust Image Hashing", *ACM Multimedia* 2002.