Robust real world video fingerprinting system

Daniel Pereira, Luis Loyola
Research and Development Department
SkillUpJapan Corporation
Tokyo, Japan
{d.pereira,loyola}@skillupjapan.co.jp

Abstract— **Video fingerprinting is one of the most promising technologies to uniquely identify video content in the Internet in a totally format-independent way. The logic behind the creation of the video fingerprints or perceptual hashes is a key factor for the performance of the system, especially in terms of storage and search efficiency. Although several video fingerprinting methods have been proposed, most of them largely focus on the robustness of the perceptual hashes to distortions, giving much less importance to the required storage as well as the efficiency of the database structure for fast and reliable searches of video content. In this paper we propose a video fingerprinting system with an excellent robustness to distortions but also offering lower storage requirements and a more efficient database structure than prior art.**

Keywords-video processing; pattern recognition; information storage; video fingerprinting;

I. INTRODUCTION

The growth of video-related services in the Internet is increasingly creating needs for uniquely identifying video data in the Internet using video fingerprints. Applications that may profit from such technology include automatic tracking of copyrighted titles in user generated content sites, tagging movies for reducing the cost of its distributed storage and improving video search engines in a totally video-codec or video-format independent way.

Video fingerprints are perceptual hashes that allow us to discriminate one video from all the rest in a totally binary-independent way, i.e. irrespective of the codec or format in which they have been compressed. The main requirements for such a system are given below:

- Uniqueness: a piece of video content must be associated to a unique hash value.
- Storage: the amount of bits generated per unit of time from the hashing system should be as low as possible.
- Database: the necessary time to find a specific piece of content inside a huge pool of contents should be as low as possible. To do that the logic behind the creation of the hash is quite important since it defines the efficiency of the database structure.
- Robustness: the perceptual hash should be robust to malicious or intentional distortions including chromatic alterations, addition of watermark and subtitles and rotations.

Several video fingerprinting methods have been proposed. Most of them relate to pixel-level statistical analysis due to their low computational complexity.

Lee and Yoo [1] propose an approach based on centroid of gradient orientation (CGO), while Lee and Suh [2] proposal focus on the orientation of luminance centroid and Lowe [3] focus on the histogram of gradient orientation.

Other approaches propose histogram similarity calculations[4], ordinal measures of block means [5] and differential block means [6], however the dimensionality of these solution are problematic.

In this work we propose a video fingerprinting generation system as well as its associated database structure. The proposed system is very robust to video distortions including rotations, chromatic alterations, frame rate change, blurring, subtitles and watermarks. Furthermore, the structure of the perceptual hash allows to significantly reducing the amount of storage required for the hash values as well as to decreasing the search time in the database.

The paper is organized as follows: some background is given in section II. The related work is covered in section III. Section IV describes in detail our proposed fingerprinting system. Experimental results and comparisons with previous art are shown in Section V. Finally, Section VI concludes the paper and point out some directions for future work.

II. BACKGROUND

Movies, or motion pictures, pose some interesting obstacles that need to be properly addressed in the digital domain. The amount of information present in movies is so large that easily becomes a problem in areas such as information extraction and processing, data storage and the subsequent search inside the stored information.

While information extraction can be as straightforward as decoding a movie, the subsequent processing cannot only be a simple, instantaneous, analysis of values of each frame. Data needs to be processed as a whole so that more meaningful information can be obtained from a specific movie, or part of it.

Furthermore, the analysis of the data must produce results that are independent of the format on which the data has been encoded. That is, rather than a binary analysis of the data, a perceptual analysis of it is necessary. For instance, five different files corresponding to a 30-second scene from a movie encoded with mpeg2, mpeg4, vp-6 and vc-1 using different values of width, height and frame rate must originate exactly the same unique fingerprint. Thus, that fingerprint is intrinsically associated to the perceptual content and not to the binary format in which it is presented or stored.Development in data storage has been enormous, however the improvement of quality of motion pictures have expanded even further. Storing, for analysis, a movie in its original raw format has an extremely high cost in any solution and negatively impacts its success.

Moreover, searching for information in a continuous, time based, structure is very inneficient. Efficient algorithms need to know how to discard irrelevant information and where to start searching for similar content. The increasing number of content creators, with contents distributed over several mediums, need to have a solution to efficiently search, analyze and take actions according to the modality of the distribution of their contents by third parties. This is another field where the utilization of perceptual fingerprints has a great potential.

It is therefore essential to provide tools that allow a fast and efficient search of information that cannot be organized as a simple liaison of facts, keywords or occurences. New search engines and data storage systems can take a lot of advantage from perceptual fingeprinting.

III. RELATED WORK

Up to this time, many video fingerprinting algorithms have been working at the pixel level. Working directly with pixels is, nowadays, computationally feasible and accurate.

However, those solutions do not address the magnitude of the resources needed for such a system and little analysis is provided in order to use a video fingerprinting solution in practical cases.

Many of the solutions use no real databases and store the content in memory. Others provide no effective way to index the video contents in an effective and scalable manner, that can leading to large increases in system resources.

Additionally, most recent approaches [1][2] rely on a heavy extraction process, comprising video partitioning, frame-rate conversion and image resizing.

Out of the analyzed solutions, the proposals based on CGO[1] are deemed the most interesting, incorporating some unaddressed practical point, while providing a challenging algorithm. It will, therefore, be the base of our comparison.

IV. PROPOSED METHOD

Several characteristics can be analyzed in movies. While a plethora of information is readily available, only some of them are of interest to us, more specifically the Luminance (or Luma) values for each frame.

A. Database indexing

The database indexing relies on an inherent aspect of the measured Luma. The values were found to be similar for a contiguous, varying, amount of time. That way, in order to reduce storage footprint, we aggregate information that is similar up to a threshold *t* percent. This is our Database Threshold. When the value surpasses the threshold a new index, that is, a reference in time for the start of the current cluster of information, is created until all the information is processed for that movie. With that, it is not necessary to store all the information of the movie initially gathered. The following figure can more clearly show how this process is achieved.

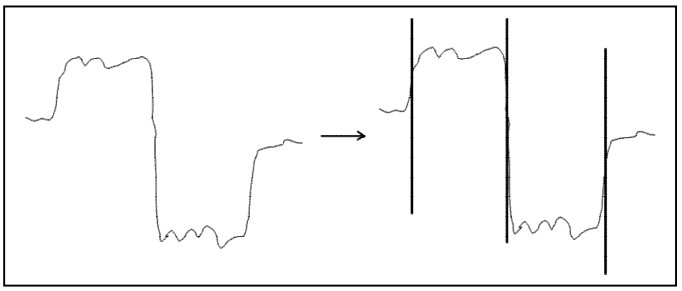

Figure 1. Detection of indexes

Figure 1 shows on the left the original measurements of the Luma values of a video file. On the right, the black vertical bars depict where the indexes are detected.

However, the threshold value depends on the expected usage for the information and the storage infrastructure and needs to be carefully chosen. A strict, lower, threshold will aggregate fewer values and generate more indexes that will in turn increase the system accuracy. Therefore, the amount of storage needed will also increase. Likewise, a larger threshold, with higher values, will save a large amount of space in a database, at the expense of turning the stored information useless for video fingerprinting purposes.

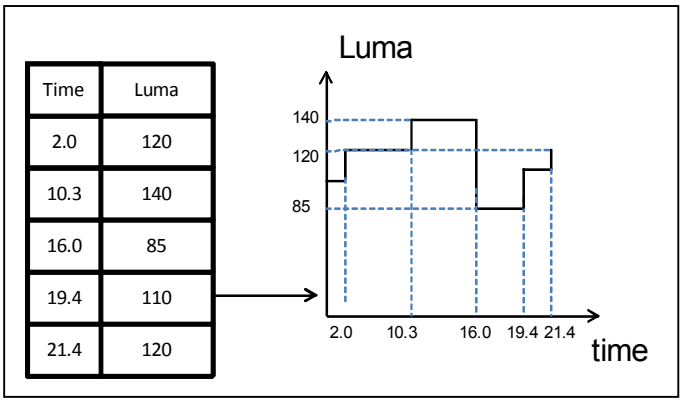

Figure 2. Information clustering algorithm

Figure 2 shows how that information, taken from the previous index detection algorithm, builds the graph on the right hand. When having the information stored in a database,

depicted in the left hand, it is easy to reconstruct the information, while saving a large amount of information in the database. Instead of storing Luma information about every second or frame, only the indexes are stored.

Additionally to the information pertaining to each movie, a look-up table for the values of Luma is used. This table allows to quickly search movies that have a specific Luma value in any of the movie indexes.

B. Search algorithm

The search algorithm consists of several levels or layers of filtering. By providing several layers of identification, we can achieve faster searches and use fewer resources. Using this approach, it is possible to discard movies that are clearly not related to our source video, or clip, and only increase the effort of the search with videos that have a higher probability of being a match.

The initial filtering consists on calculating the clustered information of a given clip to be found, or source movie, calculate its time indexes and each index value of Luma. Then the algorithm will proceed to make two analysis. Firstly, search the look-up tables for movies with indexes of similar values of Luma. Secondly, analyze the time indexes of the selected movies and consider for further analysis only the movies that have a similar number and time distribution of indexes. That information is extremely effective in an initial phase, since the indexes and their values of Luma provide a simple, and computationally effective, way to discard unrelated movies, while retaining a high level of uniqueness. It is, therefore, possible to spend less time in calculations on movies that are easily identified as not being a match.

Next step consists on analyzing all the videos that were previously deemed to be possible matches of our source video. In this step, different analysis approaches are used. We make use of the Tanimoto correlation factor[7], defined in (1).

$$T(A,B) = \frac{A \cdot B}{\|A\|^2 + \|B\|^2 - A \cdot B} \quad (1)$$

The Tanimoto correlation compares vectors of values and calculates the probability of those vectors to be similar. In more detail, the vectors that are used in to this similarity algorithm are the values, already reconstructed from the database, of each index of the source video and the video in the database. Then, each group of *n* seconds, is analyzed. However, not every index from the source video will be compared to every index of the video in the database. Instead, if a threshold is not met from that comparison, the algorithm will jump to the next index of that movie and a new comparison will be initiated. If a threshold is attained, then a thorough examination is done, further comparing every second within the current index and the next one. Allowing the algorithm to jump to the next relevant section makes the algorithm faster, depending on the threshold used. High gains are achievable at the expense of an increased number of False Negatives; that is, videos that are not successfully found.

The subsequent and final step is an analysis of the distance of the vectors, defined by (2).

$$D = \sqrt{\sum_{i=0}^{n} (A_i - B_i)^2} \quad (2)$$

Those vectors are the same that were previously used in the Tanimoto correlation algorithm. However, this turn, the distance between those vectors is calculated and the lower the distance, the higher the probability that those vectors are the same and, therefore, that the videos are a match.

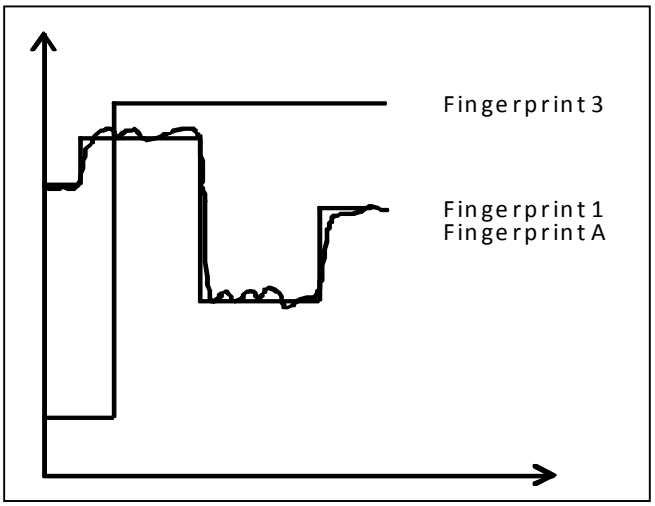

Figure 3. Example of fingerprint comparison

Figure 3 illustrates how the initial stages of the fingerprint matching process work. Fingerprints A, 1 and 3 are compared. Fingerprint A would be the original movie before clustering its information and fingerprint 1 would be the fingerprint of movie A after clustering its information.

Clearly, fingerprints 1 and A are a match. However, fingerprint 3 is related to another movie and will be discarded from further comparison.

V. RESULTS

To test our proposed algorithm, we compare it with the CGO algorithm. The targeted movies were downloaded from popular UGC websites, amounting to over 700 minutes of video.

The movies have varying resolutions (from VGA (640x480) to FullHD (1920x1080)), lengths (from 1 minute to 60 minutes) and frame-rates (15fps to 30fps). The variety of the videos allows us to make the tests in a large diversity of genres like documentaries, music clips, movie trailers, animations and static image clips generated by users.

The parameters used for the CGO are the same defined by the authors, each frame was partitioned into 8 parts (4 columns and 2 rows), threshold of 0.4 and the search sequence is made

of 100 consecutive frames (10 seconds). The parameters for the SUJ algorithm are set to provide a clearer comparison with CGO. Therefore, the searched sequence length is also 10 seconds. In order to provide a better insight of the influence the clustering can have, a varying database threshold, between 1 and 30, is analyzed.

The proposed algorithm was tested by searching a sequence of each movie against itself and the remaining 209 videos in the database. Each test will, in the end, have 44100 comparisons.

A. *Analysis of database results*

Table I presents the comparison between SUJ, the proposed solution, and the CGO-based solution. In the proposed solution the needed storage for the fingerprints of all 210 movies never exceeds 5Mbytes, while the CGO solution needs over 100Mbytes.

TABLE I. DATABASE RESULTS

SUJ		**CGO**
DB threshold	***Size (Kbytes)***	***Size (Kbytes)***
1	4944	107900
2	3390	
3	2611	
5	1783	
10	1020	
15	678	
20	512	
25	412	
30	348	

Moreover, several thresholds for the database are presented. Those thresholds are used in the information clustering algorithm, when deciding what degree of Luma variance can be allowed until a new index is created. Lower values of this threshold mean that lower variances of Luma will create a richer set of indexes for a movie, causing the size of the database to grow. On the other hand, using high thresholds will permit a lower size of the database. The implications of such thresholds will be further analyzed in the following sub-sections.

B. *Analysis of timing results*

This sub-section will present the results obtained when the time performance of the presented solutions is analyzed.

TABLE II. INITIAL IMPORT MEASUREMENTS

	SUJ	**CGO**
Time (minutes)	181	4538

As explained in Section III, the two compared solutions have large differences in the initial extraction of fingerprints of movies to the databases. More specifically, the CGO solution needs to have several calculations done, such as frame partitioning, conversion and the CGO calculations. Those calculations, as presented in Table II are very expensive, and completely shatter any possibilities of using CGO in a real-time environment, with an initial fingerprint extraction time of over 75 hours. On the other hand, the SUJ platform took 3 hours to extract the fingerprint of 773 minutes of video, faster than real-time.Table III presents the obtained times for the search of videos. Here, a section of each one of the 210 videos was looked up in the database and averaged.

TABLE III. LOOK UP TIME RESULTS

SUJ		**CGO**
DB threshold	***Avg. Time (s)***	***Avg. Time (s)***
1	74.16	92.41
2	28.38	
3	20.48	
5	13.27	
10	8.84	
15	8.66	
20	10.1	
25	12.42	
30	15.11	

The results from SUJ is faster than CGO. It is also relevant to look into the DB threshold column. A low threshold will make the number of indexes grow and the lookup time increase. As the threshold grows the lookups become faster. However, a too large threshold will make the lookup time grow. This is due to the fact that valuable information is discarded when using large thresholds, making the algorithm analyze sections of videos that are not a match, in later stages of the algorithm. An example of this is depicted in the following figure.

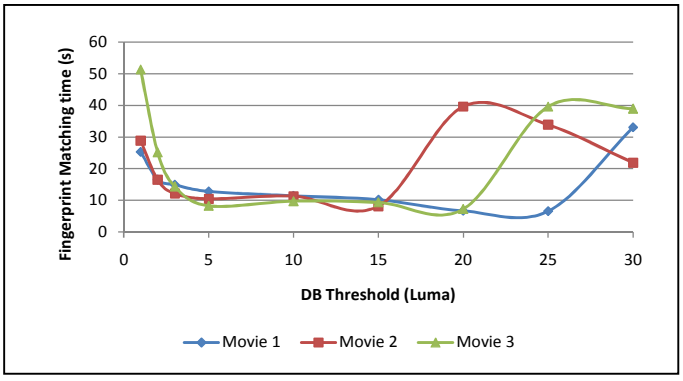

Figure 4. Example of fingerprint timings for 3 movies

In Figure 4, the timings converge to lower values up to a database threshold of 15. However, after that threshold the values become higher and unreliable.

C. *Analysis of robustness results*

A video fingerprinting platform needs to give accurate results. That accuracy can be expressed by the amount of videos reported as found, but being mismatches (False Positives) or by the number of times a video is known to be in a database, but the algorithm not being able to find the section of the video (False Negatives)[8].

Our presented solution considers more relevant to properly find every video, even if the result will increase the reported number of False Positives.

TABLE IV. SUJ ROBUSTNESS RESULTS

DB threshold	False Positives	False Negatives
1	8	0
2	12	0
3	9	0
5	11	0
10	11	3
15	8	8
20	9	6
25	11	14
30	9	18

From the results of Table IV, our solution can effectively find all the matches of the searched videos in 3 of the analyzed database thresholds (thresholds 1 to 5).

As the database threshold increases, the number of hits also increases, due to loss of information. The clustering algorithm will have less indexes and group larger areas of Luma, leading to zones with no unique variations of Luma that can identify smaller segments of video.

TABLE V. CGO ROBUSTNESS RESULTS

False Positives	False Negatives
211	1

From the obtained results, both solutions propose a low number of False Negatives. In contrast, CGO presents a larger amount of False Positives. While having some False Positives would be expected, this amount of False Positives would negatively impact the use of this algorithm in our platform.

VI. SUMMARY AND FUTURE WORK

In this paper, we propose a novel video fingerprinting method based on Tanimoto correlation. We also propose a method to greatly reduce the amount of database information needed to effectively store fingerprints of videos.

The results show that our solution can achieve such an improvement. Moreover, our method can reduce the amount of False Positives and False Negatives, reduce the time spend on fingerprint matching and reduce the amount of initial processing so that it can effectively be incorporated in a real-time, practical, situation.

The future work will comprise further improvements and comparisons, testing the algorithm against resizing, compression algorithms, brightness and gamma alterations, subtitling, frame cropping and rotation modifications.

REFERENCES

[1] Sunil Lee and Chang D. Yoo, “Robust Video Fingerprinting for Content-Based Video Identification”, in IEEE Trans. Circuits and Systems for Video Technology, vol. 18, no. 7, pp983-988, July, 2008

[2] Seungjae Lee , Young Ho Suh, Video fingerprinting based on orientation of luminance centroid, Proceedings of the 2009 IEEE international conference on Multimedia and Expo, p.1386-1389, June 28-July 03, 2009, New York, NY, USA

[3] David G. Lowe, Object Recognition from Local Scale-Invariant Features, Proceedings of the International Conference on Computer Vision-Volume 2, p.1150, September 20-25, 1999

[4] S.C. Cheung and Avideh Zakhor, "Efficient video similarity measurement with video signature," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 13, no. 1, pp. 59-74, Jan. 2003.

[5] Changick Kim and B. Vasudev, "Spatiotemporal sequence matching for efficient video copy detection," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 15, no. 1, pp. 127-132, Jan. 2005.

[6] Job Oostveen , Ton Kalker , Jaap Haitsma, Feature Extraction and a Database Strategy for Video Fingerprinting, Proceedings of the 5th International Conference on Recent Advances in Visual Information Systems, p.117-128, March 11-13, 2002

[7] T.T. Tanimoto, 1957, IBM Internal Report 17th Nov.

[8] L.M. James and L.C. David, *Decision and Estimation Theory*. New York; McGraw-Hill, 1978, pp27-38