Fast Repetition Detection in TV Streams using Repetition Patterns

Cérès Carton, Patrick Gros

Fast Repetition Detection in TV streams using duration patterns

Cérès Carton
Inria
Rennes, France
Email: Ceres.Carton@irisa.fr

Patrick Gros
Inria
Rennes, France
Email: Patrick.Gros@inria.fr

***Abstract*—This paper presents a new method to detect video repetitions in a TV stream. This method aims at reducing the number of image descriptors that have to be computed. First, shot durations are used as hash keys to propose a first set of potential repetitions. Second, a statistical test and visual descriptors are used to verify and complete these first guesses in order to improve the results by reducing the number of false detections and by retrieving the maximal length of the repeated segments. The method is tested in the context of a TV structuring application using one month of continuous TV stream.**

I. Introduction

Television daily produces massive amounts of videos. Diversification of broadcast possibilities and storage devices has recently given rise to the emergence of many new services and novel TV programs consumption schemes. These new services are basically aimed at making audio-visual content available to users without any constraints on location and/or time. This can be possible if and only if the TV stream structure is available with a very good precision.

Unfortunately, the representation of a digital video does not convey explicitly its structure. The Event Information Table included in many TV streams is not more accurate than a usual program guide found in a magazine. Retrieving any information from a stream remains very difficult. TV streams have a strong and stable but hidden structure made of programs and breaks that we want to discover. The recovery of this structure from raw data is known as the TV structuring problem. The key technique needed to solve this problem is the detection of repeated segments like commercials, followed by a classification of these segments. This allows recovering the structure of the video and recognizing the nature of all pieces of the video stream.

A lot of research has been done to identify similar video clips in a video dataset. Most of this research was conducted with two main purposes in mind: video copy detection and video similarity search for content-based retrieval. All these techniques rely on the computation of images descriptors, and on the comparison of these descriptors in order to find which images are similar or nearly identical. The development of new indexing algorithms allowed to reduce drastically the computing time necessary to compare a descriptors to a huge set of descriptors, up to one hundred billions of descriptors. Comparing all the descriptors of a large set two by two remains more complex, but new algorithms will emerge. On the other hand, such methods cannot reduce the fixed cost of the computation of the image descriptors themselves. Reducing the number of image descriptors needed, not just by subsampling the video stream by a method or another, but by computing only the useful ones, is the objective of the method we propose.

In this paper, we focus on efficiently discovering unknown and repeated sequences in very long TV streams (up to several months). The target repetitions are commercials, jingles, trailers or even longer programs. Since we are looking for long repeated segments, that usually contain several shots, we propose to use a technique inspired from DiscId[1], the technique used to recognize musical CDs that is based on the length of the various tracks of each CD. In a similar way, we propose to use shot durations in order to retrieve repeated sequences. These durations are used as hash keys and allow detecting very efficiently the stream parts of interest. Since the method is error prone due to the extreme simplicity of the descriptor used, we use then traditional visual descriptors in order to verify initial guesses and to complete them in order to get maximal length repetitions.

The rest of the paper is organized as follows. Section 2 provides a brief state of the art and section 3 presents the method and the results we obtained.

II. State of the Art

Several works are related to ours: those focusing on similar video recognition and those related to TV stream structuring. As a matter of fact, the former ones provides the basic tools necessary to the latter ones.

A. Similar Video Recognition

A lot of work has been done on video retrieval for video copy detection or video similarity search. These works consider a video extract as a query and try to retrieve all the similar clips in a large database. This is a typical search-by-example problem that benefits directly from what has been done in the domain of image retrieval, since most techniques consider the video as a simple set of ordered images and first retrieve images before eventually doing a post-processing based on image ordering. It is clearly out of the scope of this paper to do a complete survey of this extremely active field of research these last years, but here are the basics.

These techniques first compute image descriptors. These can be local descriptors like SIFT [8] or the many variants that

[1] See http://en.wikipedia.org/wiki/CDDB

have been proposed afterwards. They can also be global ones like GIST [11] for example. Another way to get robust and discriminant global descriptors is to pack the local descriptors in a global one. This was first proposed by Czurka [3] (bag-of-keypoints) for image categorization and by Sivic and Zisserman [15] in the context of video recognition (bag of visual words - BOVW).

The second component is an algorithm to compare efficiently these descriptors. In the case of BOVW, this is achieved by inverted files, a simple but extremely efficient technique. When the descriptors are represented by non-sparse high dimensional vectors, specific algorithms were designed to circumvent the curse of dimensionality problem. Their goal is to find which are the closest points of a given query point in a high dimensional space. Finding close points is not so difficult, but proving they are the closest is the hardest part since it requires considering almost all the data. That's why approximate search techniques were developed, with various ways to control the approximation done. Such algorithms were for example developed by Lejsek et al. [7] or Joly and Poullot [6], [13].

B. TV Structuring

Two different kinds of TV structuring techniques have been proposed. The first one relies on a huge amount of manual annotation and infers the structure of the stream from these annotations. This structure an be use to predict the start and end time of every program, and the stream is analyzed just around these predicted times to refine these predictions. Such a method was developed by Poli [12]. Other methods take a bottom-up approach. Repetitions are detected since they are rather frequent and allow to identify many structuring elements like the TV breaks. This detection is followed by a classification step where each repeated segment is given a label according to its type: commercial, short program, trailer, sponsor credits... The Event Information Table (an electronic program guide embedded in the TV stream itself) is finally used to attribute titles to the programs. The difficulty comes from the necessity to compare the whole stream with itself to detect the repeated segments. Given the induced complexity, this requires new approaches with simpler descriptors [10], or to adapt a technique which considers the stream progressively [9]. Poullot [14] also built such a system, but dedicated to a much finer comparison than what is needed for TV structuring.

Even if the descriptors used are simpler, Naturel's one has only 64 bits per image [10], they remain based on the computation of the descriptors of all images or of a subsample of the images. Here, we propose to use something much simpler to find initial guesses and to reduce the use of the descriptors to verify or complete these first assumptions.

III. Fast Repetition Detection

Context and Problem

As written before, our goal is to develop a new method to detect efficiently the repeated segments of TV stream up to several months long, in order to further compute its structure in terms of programs and breaks. The underlying idea is to limit the number of image descriptors to be computed: this is possible because we are only interested in rather long repeated segments. This allows using a more basic descriptor.

Vocabulary

In this paper, the basic unit of video we consider is a shot. A shot is mainly characterized by its *duration*, i.e. its number of frames. We call *segment* any contiguous set of shots. A segment is characterized by the number of shots it contains, its *length*, but it can also be described by the sequence of the durations of the shots it contains. As an example, a segment of 3 shots whose durations are 10, 19 and 24 can be described by the vector (10, 19, 24). Such a vector will be called a *duration pattern* or a *shot duration pattern* in the rest of the paper.

A *sequence* is a set of segments that are repeated, i.e. that represent the same content up to some noise. Each of these repeated segments is called an *occurrence*. Chronological order is used to sort the occurrences of a sequence. In this way, we call first occurrence the first one to appear in the stream.

Method Overview

The general working scheme of our solution is as follows. First, the stream is segmented automatically into shots. Then, the shot durations are used as hash keys to retrieve shots of same duration, or series of shots with the same duration pattern. A selection on these shot duration patterns is made through a statistical test and a content-based test. Then, image descriptors are used to extend these sequences to their maximum duration.

Our solution does not make any hypothesis on the length or the frequency of repeated sequences. All the results presented in this article were performed on a one-month TV stream. This TV stream was recorded from a digital broadcast of a French TV channel, in 2008.

Evaluation Method

The method is evaluated using only *precision*. No recall measure is computed here because no ground truth is available on our corpus. Annotating manually the whole corpus in term of repetitions would be too expensive. The precision is computed automatically: An accurate visual descriptor is used in order to verify the the repetition detected. Thus, on Fig. 1, 100 % means that all detected repetitions are correct.

We proceed in few steps. For each sequence, the first occurrence is used as a reference. Then:

1) We compute the L_2 distance between the GIST of each pair of corresponding frames of the reference and of the i-th occurrences. The frames are declared identical is their distance is lower than 0.9;
2) We use a decision criterion:
   - if more than 80 % of frames of each occurrence are identical to those of the reference, the sequence is considered as *correct*.
   - if at least one occurrence has less than 80 % of frames identical, the sequence is considered as *incorrect*.

A threshold of 80 % was used not to be penalized by a too strict threshold on distance between two frames.

A. *Shot-Based Detection*

The first step of the method is shot detection. Any technique can be used, ours is based on GIST features. The key property required for the shot detector is its repeatability, i.e. its ability to segment exactly the same way a content that is broadcasted twice. On the other hand, the impact of missing a few transitions or of over segmenting is weak as long as the detector is repeatable. As a matter of fact, all shot detection algorithm requires to compute some sort of image description, and to compute it on almost all images, something we would like to avoid.

It should be noticed that the descriptors used for shot detection are usually very basic (like color histograms), fast to compute and not discriminant enough for video similarity detection. Many shot segmentation algorithms are several times faster than real-time. So the computation of these descriptors is not the bottleneck of TV structuring algorithms that require the computation of another more discriminative descriptor on the images in order to detect the repetitions.

We propose to use a really simple descriptor for each shot: its duration measured in number of frames. With this simple descriptor, we avoid the frame sub-sampling and memory management issues. With an average of 19170 shots per day in our corpus, it implies that 19170 integers are enough to describe a day of video. 16-bit integers are large enough to code durations for shots up to 45mn long. So 38340 bytes are enough for a day, and 1MB is enough to store the description of a full month. The main idea is to characterize a segment containing one or several shots by the series of the durations of these shots, what we call its duration pattern. If two sequences have the duration pattern, and especially if that pattern is *sufficiently* rare or unexpected, they may represent the same TV stream segment.

The algorithm works as follows: The first loop builds iteratively a series of associative or hash tables. $T[L]$ contains the segments of L shots sorted by their duration patterns. For example, $T[1][23]$ contains all the shots whose duration is 23, and $T[2][23, 42]$ contains the segments of two shots having respectively 23 and 42 shots. Since we look for repetetions, bins with only one element are withdrawn.

Data: A segmented video
Result: All sequences with a same duration pattern
- length $L = 1$
- create a table $T[1]$ of all the shots indexed by their duration
- remove the shots that are alone in their bin
while $T[L] \neq \emptyset$ **do**
  - add 1 to L
  - create a table $T[L]$
  for *every segment* $s \in T[L-1]$ **do**
    - add to s its following shot
    - index s in $T[L]$ by its duration pattern
  end
  - withdraw the bins of $T[L]$ that contain 1 segment
end
while $L \geqslant 3$ **do**
  remove 1 to L
  for $s \in T[L]$ **do**
    - remove the subsegments of s from $T[L-1]...T[1]$
  end
end

Algorithm 1: Shot based detection

The second loop of the algorithm cleans the tables of redundant information: when two segments of 5 shots have the same duration pattern, their corresponding subsegments of 4, 3... shots have also similar patterns, but are of no interest, and thus removed from their respective tables. The result of the algorithm is contained in the tables $T[L]$. Each of these tables contains in a given bin all the maximal segments sharing a same shot duration pattern, i.e. the repetitions we are looking for. Constructing these tables iteratively avoids the enumeration of all possible segments of a given length L. A segment is considered only if it has a subsegment of length $L-1$ that is repeated.

The major advantage of this shot based detection is its efficiency. One month of video is treated in only 12 seconds for the first loop, followed by 44 seconds of post-processing for the second loop. The drawback is that precision is poor for short length sequences, as it can be observed on the blue curve "base" in figure 1.

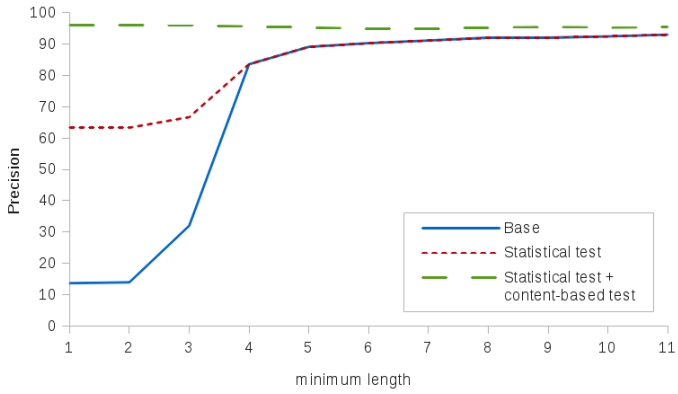

Fig. 1. Precision of the repetition detected. X-axis: nb of shots in the sequences, Y-axis: precision obtained for these sequences.

B. *Verification of the Shot-Based Detected Sequences*

In the previous section, we have presented our method to detect the segments with similar shot duration patterns. The method is really efficient, works even for segments reduced to 1 or 2 shots, but the price to pay is a low precision for the short sequences. This is due to the fact that the similarity between duration patterns is not equivalent to the one between video contents. In order to improve this precision, we use two different tests: the first one is based on the probability of appearance of each duration pattern, the second one is based on the visual information of the sequence.

1) Statistical test of detected segments: This test is based on the probability of appearance of a segment of a given duration in the TV stream, and use results introduced by Castro et al. [2] on significant motif detection in time series. It relies on shots durations only, no additional information is necessary.

Statistical hypothesis tests are widely used to help in decision-making. A null hypothesis (H_0) is defined. The test aims at finding whether there is enough evidence in the data to reject this hypothesis. Here, we want to verify whether the repetition of a shot duration pattern in the stream is meaningful, i.e. unexpected and thus carries some interesting information, or just due to chance and is thus uninteresting. So the null hypothesis (H_0) corresponds to "the repetition of this shot duration pattern is not meaningful, not exceptional", while the alternative hypothesis (H_1) is "this repetition is exceptional".

We first computed the histogram of the shot durations on figure 2. If we assume that shots length are independent and identically distributed in a TV stream, this histogram tells how often a given duration pattern should appear in the stream, i.e. a probability of appearance of a duration pattern given our assumption. The p-value measures whether this probability of appearance is higher than the actual number of appearance or not.

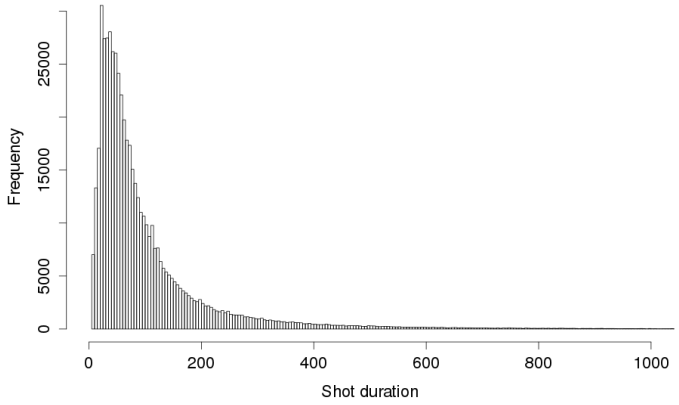

Fig. 2. Histogram of the shot durations.

For reference model, we used Bernoulli trials. For a given duration pattern δ, its number of appearance is $N(\delta) = \sum_{i=1}^{n} Y_i$ where Y_i is the Bernoulli random variable:

$$Y_i = \begin{cases} 1 & \text{if } \delta \text{ occurs in position i in the stream} \\ 0 & \text{otherwise} \end{cases}$$

with probability $p(Y_i) = \mu$. The pattern count $N(\delta)$ is a sum of Bernoulli random variables. Therefore it follows a binomial distribution: $N(\delta) \sim \mathcal{B}(n, \mu)$. Then, the p-value is calculated by the complement of the binomial cumulative density function:

$$\mathbb{P}(\mathcal{B}(n,\mu) \geqslant N^{obs}(\delta)) = 1 - \sum_{k=0}^{N(\delta)-1} \binom{k}{n} \mu^k (1-\mu)^{n-k}$$

If this p-value is lower then α, it means that the pattern δ appears sufficiently more often than expected to let us consider it as meaningful. The segments sharing this pattern have a good chance to represent a same content. α is the risk of saying "the pattern is not exceptional" by mistake. To determine its value, we used the corrected risk of Bonferroni [4], [5].

In classical hypothesis-testing, the p-value is compared to a fixed α significance level, such as 0.05 or 0.01. Here, we apply a test for each discovered sequence, i.e. the number of tests applied is the number of distinct sequences N_d. If α is set to 0.05 and we apply 100 000 simultaneous tests to sequences, one would expect to find 5000 significant sequences by chance alone. This issue is known as the multiple hypothesis-testing problem. To control the number of false discoveries, the significance level is set to values strictly smaller than 0.05 or 0.01. Bonferroni adjustment is a classical and simple approach. We adjust α to $\alpha' = \alpha/d$ where d is the number of hypothesis tests performed.

This test has a major advantage: it does not impact efficiency of our method since it takes only 4s on a one-month corpus and does not require to compute any visual descriptor.

On our one-month long corpus, 87129 statistical tests are performed. 78 % of sequences are rejected by the test (67980 sequences). Further investigations on the accepted sequences showed that they were correct for 20.1 % of them. The precision increases from 13.7 % to 63.3 %, but it can be seen on figure 1, it was improved only for the shorter sequences.

2) Content-based verification of detected segments: After the statistical test, we use another test based on the comparison of the GIST descriptors of chosen images. Given a number N defined by the user and a sequence, i.e. a set of k possibly repeated segments provided by Algorithm 1, we choose randomly N images in the first occurrence of the sequence, compute their GIST, and compute also the GIST of the corresponding images in the other occurrences. If the L_2 distance between the GIST of any pair of corresponding images remains lower than a threshold (0.9), the sequence is declared as correct.

Let's assume the sequence has n occurrences. In the worst case, i.e. when the sequence is correct, N $\times$ n visual descriptors are to be computed. When the sequence is incorrect, the computation can be stopped prematurely.

On figure 1, we can see that the combination of the statistical test and the content-based test provides excellent results in term of precision. The content-based test alone would have provided equivalent results, but that would have been less efficient in computation time. As a matter of fact, the statistical test rejects 78 % of the sequences, as seen in the previous subsection, and thus reduces the number of image descriptors to be computed. Finally, this test requires 41mn of computation to process a one-month long stream.

		Content-based test	
		Accepted	Rejected
Statistical test	Accepted	1080	4300
	Rejected	973	67007

TABLE I. COMPARISON OF RESULTS OF STATISTICAL AND CONTENT-BASED TEST ON SHORT SEQUENCES

3) The case of short sequences: Table I provides a quantitative comparison of both tests. As the content-based test is used as the reference method to evaluate the precision in our experiments, it is assumed to be errorless. So the table shows that the statistical test is a good tool to improve precision, since it rejects most of the incorrect sequences. The price to

pay is that is also reject 47,4 % of the correct sequences. This would degrade the recall a lot if we were able to compute it. However, it should be noticed, that the main advantage of the statistical test is its speed: without it, the content-based test would require 2 more hours to complete its work.

4) The case of long sequences: For longer sequences, only 4.96 % of the sequences are incorrect (519 incorrect sequences over a total of 10462). This really good result is logical since the longer a sequence is, the more exceptional its appearance is. We decided to study in more details these long incorrect sequences.

After both tests, we realized that some incorrect sequences still remained. Among these, some were long sequences with more than 10 shots per occurrence. These sequences are in fact composed of two subsequences. The first one is identical in all occurrences, when the other one is different in some of the occurrences. This phenomenon is represented in figure 3. $sequence_1$ is the sequence shared by all occurrences. $sequence_2$ is the sequence not shared by all occurrences: it appears only in the second and the third occurrences.

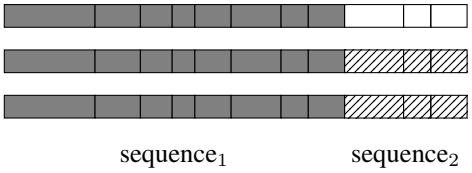

Fig. 3. The problem of long incorrect sequences

We added a specific test to detect this pattern, based on visual descriptors. With our one month long corpus, 70 sequences were concerned. This detection has a major advantage, we can be really confident in results obtained for long sequences. The major drawback is that it is time-consuming since all long sequences have to be checked. However, as only 70 sequences are concerned by this phenomenon on our one-month corpus, time computation only increased by 110 seconds.

C. Completion of maximum length sequences

After using both tests (statistical and content-based), we obtain a set of sequences with a good precision. However, these sequences are not maximal in length. This is due to some defaults of the shot segmentation algorithm. Figure 4 shows two typical examples of these defaults.

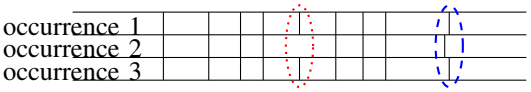

Fig. 4. Segmented sequences defaults

The dotted ellipse shows an example of shot transition that is not detected, when the dashed ellipse shows an example of a shot transition that is slightly temporally shifted. Such defaults shorten the common part of the various occurrences. So we developed a method in order to recover the full length of the occurrences.

Visual descriptors are used to compensate these defaults. We compare frame by frame the distances between corresponding visual descriptors. To define precise boundaries and in order not to define a threshold on these distances, we use the Page-Hinkley Test (PHT) [1]. PHT is a sequential analysis technique typically used for monitoring change detection. It allows efficient detection of changes in the normal behavior of a process that is established by a model.

Let define a discrete signal $\{y_0, y_1, \ldots, y_N\}$. PHT is applied iteratively for each y_n. Its aim is to detect when the signal increases (it can be adapted to detect a decrease of the signal). If we consider current value y_n, then:

$$\begin{aligned} U_0 &= 0 \\ U_n &= \sum_{k=1}^{n}(y_k - \mu_0 - \frac{v_m}{2}), n \geqslant 1 \\ m_n &= \min_{0 \leqslant k \leqslant n} U_k \end{aligned}$$

where:
- μ_0 is the signal mean computed on past values $y_0, \ldots, y_n$;
- η is a threshold depending on the admissible false alarm rate (= 0.5);
- v_m is the gap authorized for the mean (= 1).

The test is positive when $U_n - m_n > \eta$.

The use of Page-Hinkley test has several advantages. First, as its computation is incremental, we minimize calculations. Second, it is minimal in terms of delay time until detection. Another major advantage is that we don't need to fix a threshold on the distance between visual descriptors, and the test allows little changes on descriptor distances. A threshold technique on descriptor distance could not face a local increase in distance, without major rupture. Page-Hinkley test can deal with such a situation.

For a given sequence, all occurrences are compared to the first one, and the test is performed on the images before and after the segments (resp. $area_1$ and $area_2$ on figure 5) to check if we have found correct beginning and end of this particular repetition. When all occurrences have been compared, only the minimal extensions before and after the occurrences are kept. Furthermore, we add an additional threshold to invalidate such extensions when they are smaller than 10 images.

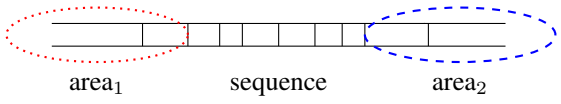

Fig. 5. Application areas of PHT

Precision does not change with completion (we do not introduce errors) but as it can be seen in figure 6, sequences are longer after completion than before. T-test on sequences length in these two groups (after completion and before) proves a significant difference (p-value = 1.629e-13). This completion requires 1 hour and 20 minutes of computation.

D. Scalability of the method

As it can be seen in figure 7, experimental results show that our method is quadratic ($\mathcal{O}(n^2)$) for small corpora and tends to be linear for larger corpora.

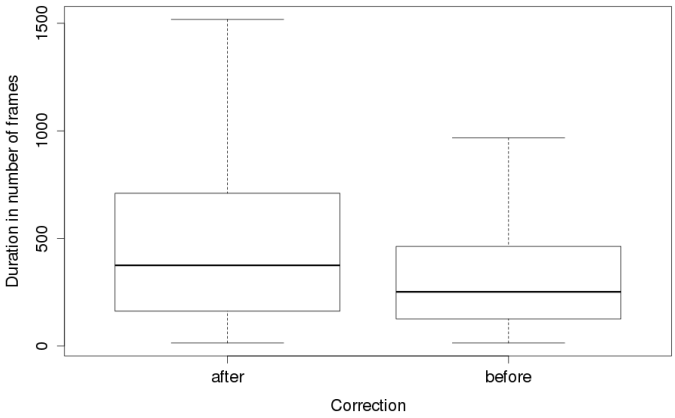

Fig. 6. Distribution of duration depending on correction

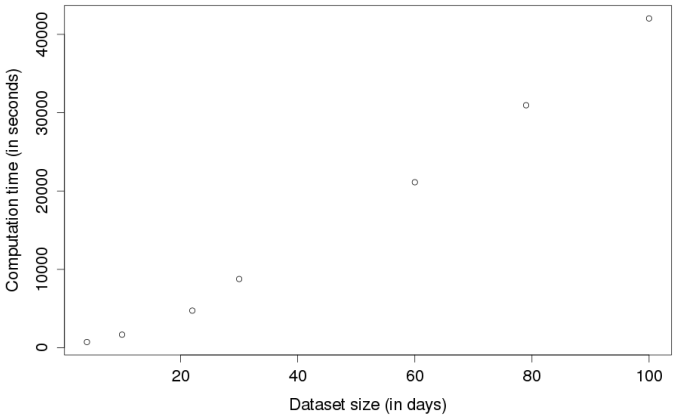

Fig. 7. Execution time depending on corpus size

IV. Conclusions

In this paper, we present a new method for detecting repetitions in TV streams. Experimental results prove a good precision for long TV streams and a good scalability. For example, one month of TV is processed in 2h05 (not including the shot segmentation) with our method, with a precision of 92.15 %. One of the goals was to reduce the number of image descriptors to compute. Table II shows how many descriptors are needed.

Corpus size	Nb of descriptors needed (percentage of the total nb of frames)
1 day	53054 (2.5 %)
10 days	3291082 (15.2 %)
30 days	9989222 (15.4 %)

TABLE II. Number of image descriptors computed with respect to the size of the corpus

Two things come out of this table. When a single day is used, there are few repetitions in the stream (jingles, sponsor credits... are not repeated), and the number of descriptors computed remains very low. When several days are considered, the number of repetition increases and so is the number of descriptors computed. But this number does not seem to vary much with the length of the corpus. The one-day-long corpus is a very special case in fact.

A perspective of this work would be to still limit the computation time by using simpler image descriptors, especially to sort out the short sequences. Since we do not compare everything with everything, less discriminative descriptors should be sufficient, for example to solve the issue of the incorrect long sequences of section III-B4, but also to extend the segments. When two repeated segments are found, what follows is either identical or completely unrelated. So very simple cues should be enough to check what is the real situation.

Detection of repetitions is a first step for valuation of TV streams: valuation of archives or on-demand TV. Further works will be done to classify sequences obtained by our method for TV streams structuring.

Acknowledgment

This work was partly achieved as part of the Quaero Programme, funded by OSEO, French State agency for innovation. The authors would like to thank Sébastien Campion for his experimental support.

References

[1] M. Basseville. Detecting changes in signals and systems a survey. *Automatica*, 24(3):309 – 326, 1988.

[2] N. C. Castro and P. J. Azevedo. Significant motifs in time series. *Stat. Anal. Data Min.*, 5(1):35–53, Feb. 2012.

[3] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *European Conference on Computer Vision*, Prague Czech Republic, May 2003.

[4] S. Hanhijärvi. Multiple hypothesis testing in pattern discovery. In *Proceedings of the 14th international conference on Discovery science*, DS'11, pages 122–134, 2011.

[5] B. J.M. and A. D.G. Multiple significance tests: The bonferroni method. *Bristish Medical Journal*, 310(6973):170, 1995.

[6] A. Joly, O. Buisson, and C. Frlicot. Content-based copy detection using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2):293–306, 2007.

[7] H. Lejsek, F. H. smundsson, B. . Jnsson, and L. Amsaleg. Nv-tree: An efficient disk-based index for approximate search in very large high-dimensional collections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):869–883, May 2009.

[8] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[9] G. Manson and S.-A. Berrani. Automatic tv broadcast structuring. *International Journal of Digital Multimedia Broadcasting*, 2010.

[10] X. Naturel and P. Gros. Detecting repeats for video structuring. *Multimedia Tools and Applications*, 38(2):233–252, May 2008.

[11] A. Olivia and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145175, 2001.

[12] J.-P. Poli. An automatic television stream structuring system for television archives holders. *Multimedia Systems*, 14(5):255–275, 2008.

[13] S. Poullot, O. Buisson, and M. Crucianu. Z-grid based probabilistic retrieval for scaling up content-based copy detection. In *ACM International Conference on Image and Video Retrieval*, Amsterdam, The Netherlands, July 2007.

[14] S. Poullot, M. Crucianu, and O. Buisson. Scalable mining of large video databases using copy detection. In *ACM Multimedia*, Vancouver, Canada, October 2008.

[15] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003.