A Static Video Summarization Approach With Automatic Shot Detection Using Color Histograms

E. J. Y. Cayllahua-Cahuina, G. Cámara-Chávez, D. Menotti
UFOP - Federal University of Ouro Preto
Computing Department
Ouro Preto, MG, Brazil
Email: {ecayllahua1, gcamarac, menottid}@gmail.com

***Abstract*—Shot detection has been widely used in video summarization for video analysis, indexing, and browsing. In this paper, we present an approach for static video summarization using histograms information for an automatic shot detection. The principal component analysis (PCA) is used in order to reduce the dimensionality of the feature vector. We propose the use of Fuzzy-ART and Fuzzy C-Means algorithms to automatically detect the number of clusters in the video and consequently extract the shots from the original video. The process is entirely automatic and no *a priori* human interaction is needed. The storyboards produced by our model are compared with the ones presented by the Open Video Project.**

***Index Terms*—video summarization, shot detection, keyframe extraction, fuzzy clustering, histograms.**

I. Introduction

The volume of multimedia information such as text, audio, still images, animation, and video is growing every day. The accumulated volume of this information can become a large collection of data. It would be an arduous work if a human tries to process such a large volume of data and even, at a certain scale, it would be impossible. Video is a perfect example of multimedia information. Video information is growing exponentially, and each day an enormous quantity of video is uploaded to the internet. TV video information is generated every day and security cameras generate hours of video. It is necessary to develop a model in order to manage all this information. Video summarization aims to give to a user a synthetic and useful visual summary of a video sequence.

Thus, a video summary is a short version of an entire video sequence. The video summary can be represented into two fashions: a static video storyboard and a dynamic video skimming. Dynamic video skimming consists in selecting the most relevant small dynamic portions of audio and video in order to generate the video summary. On the other hand, static video storyboard is interested in selecting the most relevant frames (*keyframes*) of a video sequence and generate the correspondent video summary. Obviously, in this case, the key part is to recognize these relevant frames or portions of video. The models in the literature have different points of view of what is relevant and what is not and the way to extract these relevant frames.

A raw video consists of a sequence of video shots. A shot is defined as an image sequence that presents continuous action which is captured from a single operation of a single camera and its visual content can be represented by keyframes. In order to extract the important *keyframes* from a video, we need to segment it first, usually into shots, and then analyze which will be the most representative frame in the set of frames that compose the detected shot.

In this paper, we propose the use of the Fuzzy-ART [1] algorithm to automatically find the possible number of shots and we later use the Fuzzy C-Means [2] algorihtm to discover and extract the *keyframes* from the detected shots. Our approach is based on [3] but our modification give us the main benefit that no previous human interaction is needed as it can operate in an unsupervised way and still provide satisfactory summaries. Moreover, the proposed model is not computationally expensive, compared to the original and other models from the literature.

The remainder of this paper is organized as follows. Section II provides a literature overview. Section III presents the proposed model and the details about it. In Section IV, we describe the tests and discuss the results. Finally, in Section V, we present our conclusions and the future works.

II. Related Work

In order to generate a correct and complete summarization of a given video, the employed model would have to perform an optimal understanding of the video semantic content. However, automatic understanding of the semantic content of videos is a very complex task and is still far beyond the intelligence of today's computing systems, despite the significant advances in computer vision, image processing, pattern recognition, and machine learning algorithms.

In order to capture the semantic of the video, some approaches [4], [5], [3] try to process a single feature of a video content, such as color histogram, motion, *etc*. A video is a very complex collection of data. Then, it is quite difficult to effectively discriminate the most meaningful parts in a video using a single feature. This is the reason that the most recent approaches try to combine all possible features. In order to overcome this problem, recent works combine different features. For example, in [6], [7], [8], besides visual information, audio data is used. Textual information which is

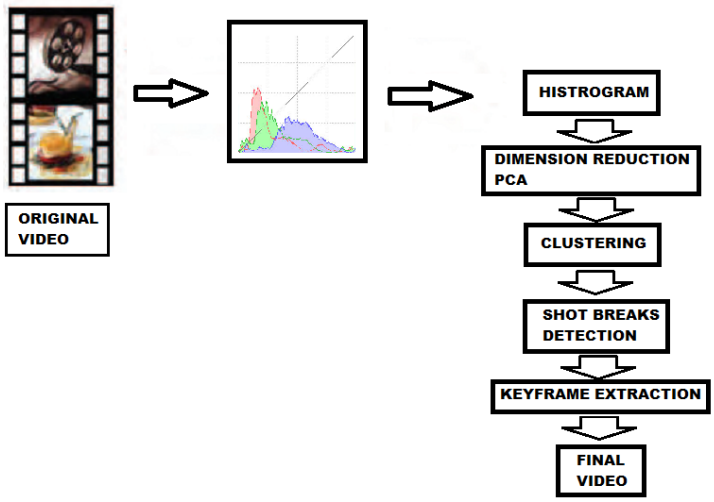

Fig. 1. General View of the approach

usually present in most films and sitcoms are also used in other approaches such as [9], [10].

As these models use more information they get specialized in certain type of videos. For example, a video recorded by a mobile phone will not contain any textual information but its file name. Complex models based on textual or even audio information can not operate on generic videos, but the models based only on visual information will have more success in generic videos. One technique that is usually used on computer vision is color histogram.

Histograms have been widely used by many video summarization techniques, several models such as [11], [12], [13], [14] have used histograms as a visual descriptor. The main reason for using histograms in video summarization is that it provides significant information about a frame and it is not computationally expensive. That is reason we use histograms in our model. Applying histograms to our problem, will provide us the visual information describing the color distribution in the frames. We can use this information to group similar frames and extract possible *keyframes*.

In [3], the concept of histogram evolution is used for summarizing the video. Although the model generates more accurate summaries, it requires some human interaction making it not automatic. The model extracts the color histogram for each frame and work as follows. Once computed, a dimensionality reduction is performed by PCA (Principal Component Analysis). This process reduces the dimensions and form a 2-D feature space, where each frame is represented as a point in this new space. Then a Fuzzy C-Means algorithm is executed in order to cluster and define the most important segments of the video. Once these segments are detected, the *keyframes* are selected and used for generating the final video, *i.e.* the summary.

The main drawback of [3] is that this model is not entirely automatic since we need humans to provide some previous knowledge as for example the possible number of scenes on the video. This kind of knowledge is usually not known *a priori*. Some previous works have tried to cope with this problem using histogram difference in order to detect the possible shots in the video [15], [16]. Our proposed model automatizes the shot detection using histograms and clustering algorithms.

III. Summarizing Video Sequences Using Color Histograms

The main contribution of this paper is to automatically identify the number of clusters (*shots*) of the video. An overview of our proposed model is shown in Figure 1. In the followings subsection, we present in detail each step of our model.

A. Histogram Computation

Histograms are widely used in computer vision. They can describe the color features when applied to a video frame. In this approach, we use a RGB (Red, Green and Blue) histogram of a frame, taking into consideration that the videos used in our experiments are colored. The original model uses grayscale histograms when RGB values are not present. The RGB color histogram provides distribution information of colors for a given video frame. Let us consider that the values of each color channel goes from 0 to 255, making a total of 256 values and that for a given color frame each pixel would contain a combination of these three color channels. Therefore a RGB histogram for a frame should be represented by a structure of size $256 \times 256 \times 256$, *i.e.*, the RGB cube. A minute of video usually has more than 400 frames, then it would be computationally expensive to operate such a structure for each frame. In order to reduce this complexity, we only use 16 bins for each channel. This value was obtained empirically from our tests. Once a RGB histogram is computed, the matrix is

transformed and stored in a vector of size $16 \times 16 \times 16$, such that each frame is represented by a vector of size 4096.

B. Dimension Reduction

The result from our previous step can be seen as a one dimensional vector representing the frequency of the color histogram. Then the PCA is executed on this vector in order to reduce the dimension of the histogram features. In order to choose the number of principal components, we check the variances from the original vector and we set a fixed threshold to choose the most significant components. We dynamically choose only the principal components which variance is greater than 50%. The value of this threshold is obtained from our experiments. Using this scheme, the process usually indicates us to use the first 2 to 8 principal components. The reason why we can not use a fixed number of principal components is that, the frames from different movies usually have different expressions of color. Fixing the number of principal components to use in this reduction step as done in the original model is not a good idea, since we can probably lost discriminative information. In Figure 2, we can appreciate how the frames of an entire video are represented in a 2-D space generated by PCA.

C. Automatic Cluster Detection

We consider scene boundary detection as a classification problem. After applying the algorithms, analogous frames will be grouped together in one cluster. There will also exist a degree (value) of membership assigned to each frame. Once we detect these clusters, we extract the closest frame to each centroid of a class, this frame will be tagged as a *keyframe*.

So, in order to detect the different shots, we perform two clustering algorithms. We automatically detect the number of clusters for a given video based on the feature vector previously extracted as shown in Section III-B. We carry out this action by using the Fuzzy-ART clustering algorithm. This algorithm performs an unsupervised clustering of our data and automatically detects the number of possible clusters.

Later with the number of clusters detected we execute the Fuzzy C-Means algorithm. The Fuzzy C-Means is a popular technique for classification, and is commonly used in pattern recognition and image processing problems.

We show an example of this process in Figure 2, where the number of clusters are indicated by the colors and its cluster centers detected are marked with "O".

D. Final Video Generation

Once we have detected all the *keyframes* from our video, we extract a 10 frame neighborhood surrounding for each *keyframe*. With this set of frames, we produce the final video, *i.e.*, the summary.

IV. Experiments and Results

Performing an objective evaluation procedure for a video summarization method is a difficult problem. Taking into account that a user evaluation of a video summary can be very subjective and that so far no standardized metric has been adopted by the researchers. Evaluating a video summary is still an open problem. The test videos used by the original model [3] are not available for downloading, that is why we have decided to use the videos from the "Open Video Project" [17]. This video database is an open dataset and the videos that are available there are usually used by researchers that are involved in computer vision. Another benefit from using this database is that every video is accompanied by its corresponding storyboard, making it possible to perform an evaluation comparing their storyboard and ours. In order to generate a video summarization, we have to identify the *keyframes*. The more *keyframes* we identify the more informative our summarization becomes.

We will show our results as a storyboard of each video and we will compare our results to the storyboard of Open Video. The videos that are presented here are:

- NASA 25th Anniversary Show, Segment 07. File: "anni007.mpg"
- Giant on the Bighorn, segment 05 of 9. File: "BOR11_005.mpg"
- NASA 25th Anniversary Show, Segment 01. File: "anni001.mpg"
- Old mail coach at Ford, U.S.P.O.. File: "match0868.mpg"
- Drift Ice as a Geologic Agent, segment 07 of 11. File: "UGS12_007.mpg"
- Drift Ice as a Geologic Agent, segment 10 of 11. File: "UGS12_010.mpg"

We present all the results in figures, every figure will have two parts. Part (A) shows the storyboard presented by Open Video and Part (B) shows the storyboard we got from our model. The frames marked with a dashed line are the matched frames between the two storyboards. Additionally, the frame marked with dots is a *keyframe* that our model successfully detected but is not present in the storyboard from Open Video.

In Figure 3, we show the result for video "anni007.mpg". As we can see, the model is successful identifying most of the *keyframes* and even identifying one additional *keyframe* not present in the Open Video storyboard. In Figure 4, we show the result for video "BOR11_005.mpg". In this case, our model identifies four additional *keyframes* not detected by Open Video. In Figure 5, we show the result for video "anni001.mpg". In this case, the model failed to identify some *keyframes* present in the Open Video storyboard. As we are only using a color histogram as a descriptor, the model is inheriting its disadvantages. The main disadvantage is that we are only using color distribution and no spatial information is taken into account, therefore it is possible that the model can cluster two totally different frames into the same group, this causes that some *keyframes* will not be detected successfully. In Figure 6, we show the result for video "match0868.mpg". In this case, the model identifies repetitive *keyframes*. This happens as a consequence of color histogram being susceptible

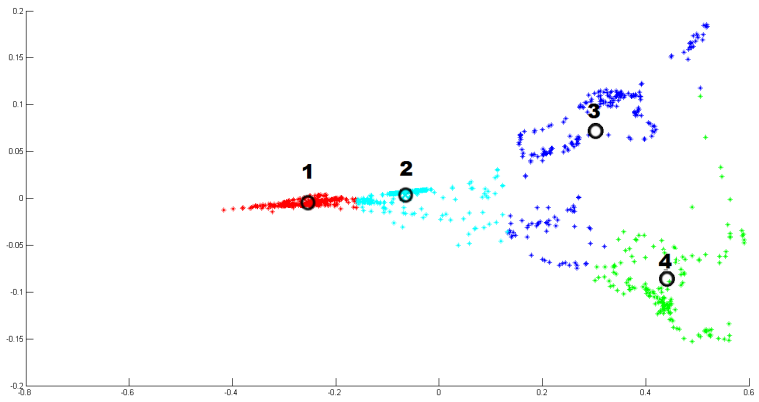

Fig. 2. 2-D representation of a video and Clusters detected

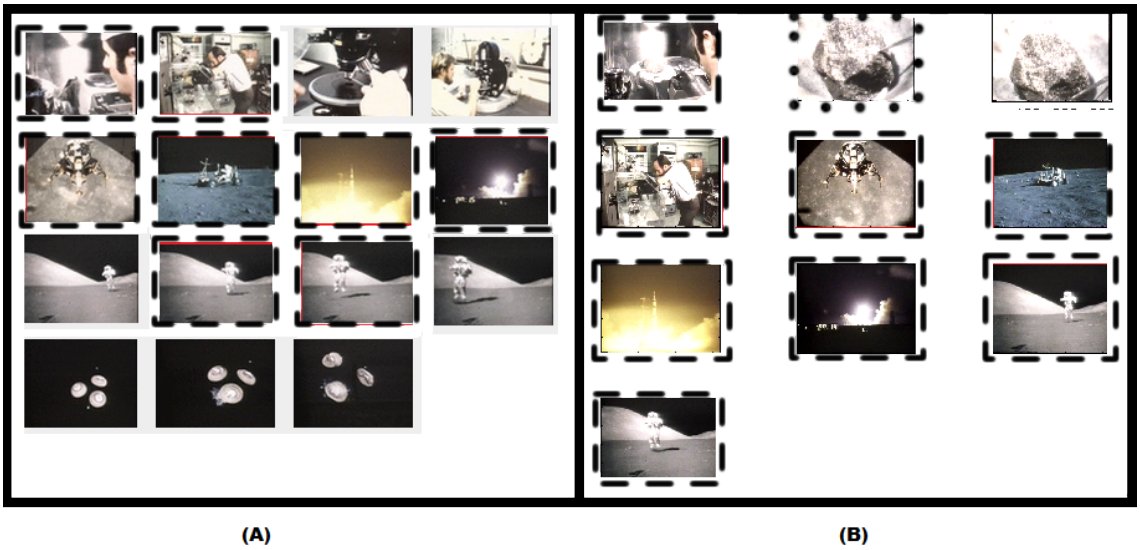

Fig. 3. Video "anni007.mpg". Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

to lighting intensity changes, two or more similar frames can be detected as different due to this disadvantage. The same issue is present in Figure 7, where the model identifies 3 similar *keyframes* but two additional *keyframes* not present in the Open Video storyboard are detected. In Figure 8, only 1 *keyframe* from the Open Video Project is not detected but we successfully identify two more *keyframes* present on the original video.

In Table I, we show the list of videos we have used for our tests, they are all available in the Open Video Database. The table contains the information about the tested video, its file name in the database and the next three columns provide us the following information:

- Matched *keyframes*, this column indicates the number of *keyframes* that were similar between our storyboard and the Open Video storyboard.
- Missed *keyframes*, this column indicates the number of *keyframes* that were not detected by our model and are present in the Open Video storyboard.
- Found *keyframes*, this column indicates the number of *keyframes* that our model successfully detected and that are not present in the Open Video storyboard.

As we can see, using just the color feature is not discriminative in all the cases, but it still generates good summaries and is in affinity to our goal of proposing an automatic video summarization algorithm that is not computationally expensive.

V. Conclusions and Future Work

In this paper, we introduced a static video summarization approach using histograms information for an automatic shot detection. We used the principal component analysis to reduce the dimensionality of our feature vector and we propose the use of Fuzzy-ART and Fuzzy C-means algorithms to

Fig. 4. Video "BOR11_005.mpg". Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

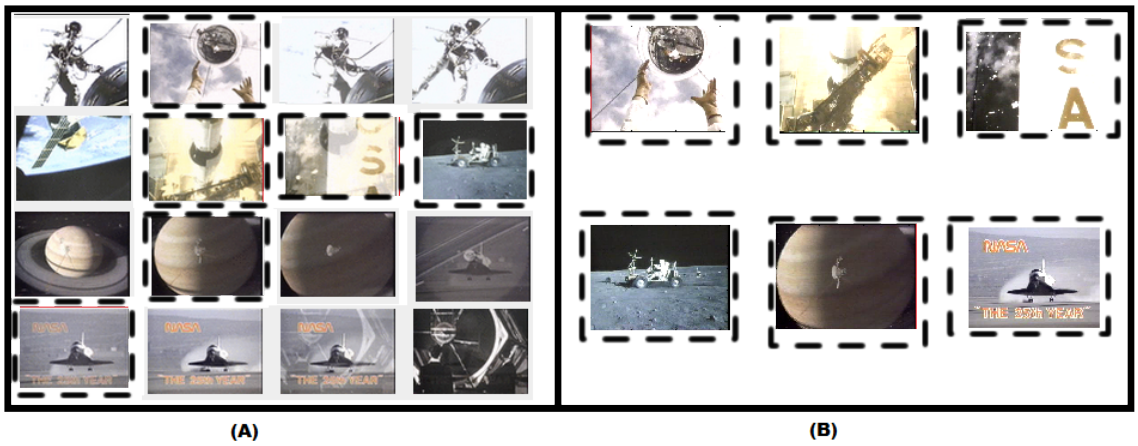

Fig. 5. Video "anni001.mpg". Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

Video Name	File Name	Matched *keyframes*	Missed *keyframes*	Found *keyframes*
NASA 25th Anniversary Show, Segment 07	anni007.mpg	12	3	2
Drift Ice as a Geologic Agent, segment 07 of 11	UGS12_007	3	1	2
Giant on the Bighorn, segment 05 of 9	BOR11_005	3	1	4
Drift Ice as a Geologic Agent, segment 10 of 11	UGS12_010	4	1	0
NASA 25th Anniversary Show, Segment 01	anni001.mpg	9	8	1
New Indians, Segment 09	indi009.mpg	4	6	0
Old mail coach at Ford, U.S.P.O.	0868.mpg	3	1	0
The Colorado, segment 10 of 10	BOR02_010.mpg	1	1	0
A New Horizon, segment 13 of 13	BOR10_013	2	0	4
NASAKSN - Go	NASAKSN-Go.mpg	2	5	4
NASAKSNN - Why Can't Teddy Travel On The Space Shuttle	NASAKSN- WhyCantTeddy TravelOnTheSpaceShuttle.mpg	4	0	2

TABLE I
THIS TABLE SHOWS THE VIDEOS WE HAVE USED FOR OUR TESTS, THEY ARE AVAILABLE IN THE OPEN VIDEO DATABASE.

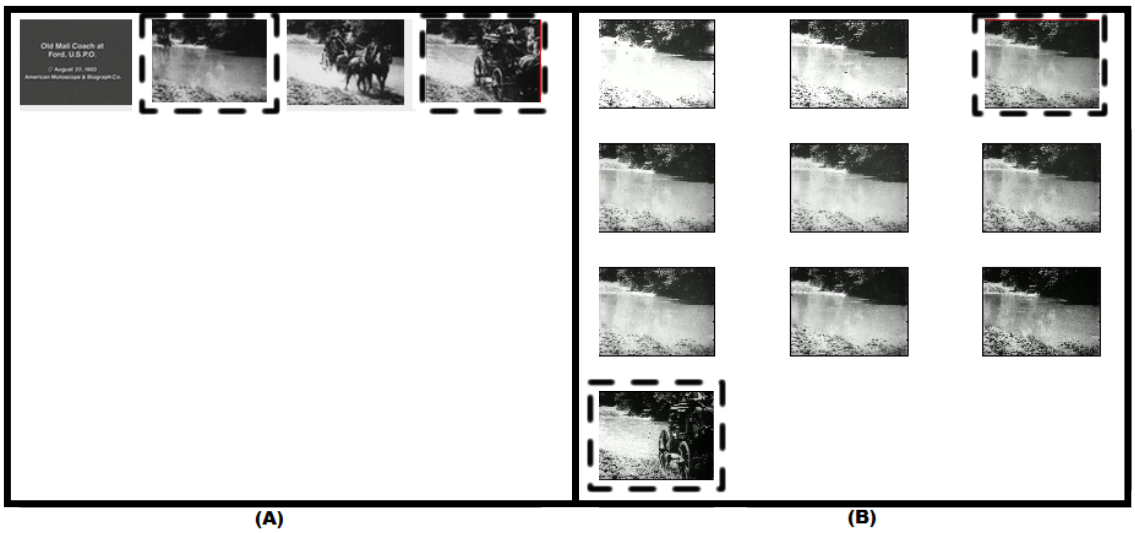

Fig. 6. Video “match0868.mpg”. Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

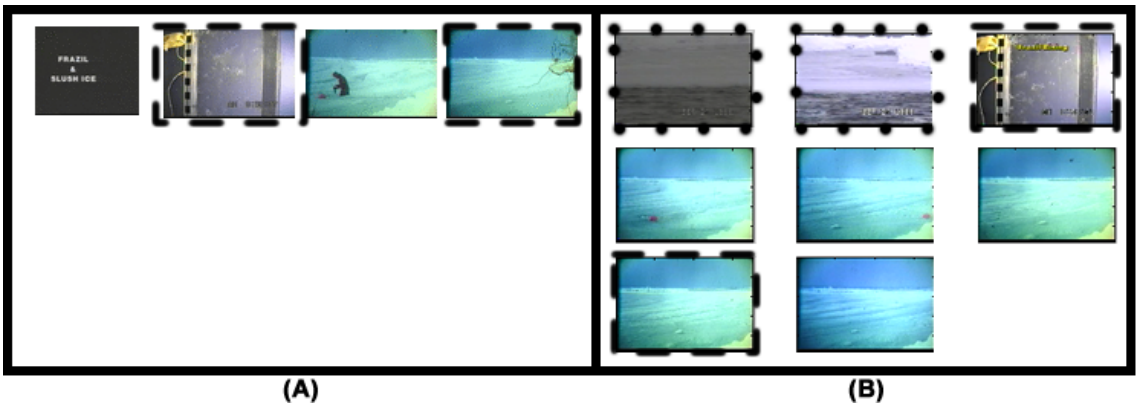

Fig. 7. Video “UGS12_007.mpg”. Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

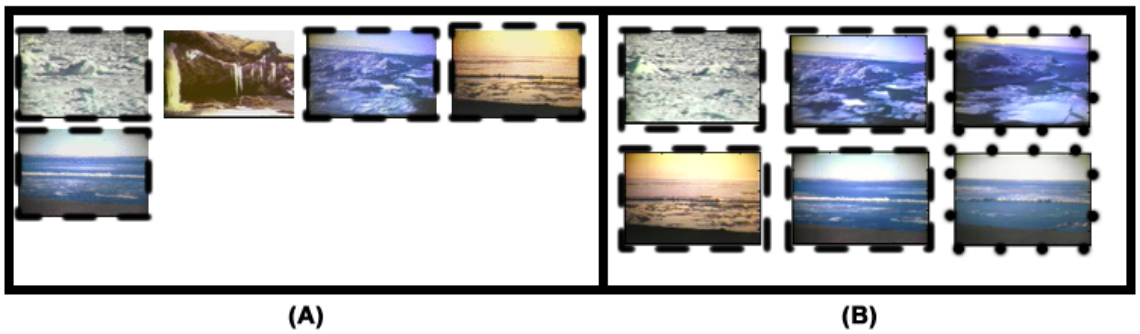

Fig. 8. Video “UGS12_010.mpg”. Figure (A) shows the storyboard from Open Video, Figure (B) shows the storyboard from our results. The frames marked with the dashed line are the matched frames between the two storyboards

automatically detect the number of clusters in the video and consequently extract the shots from the original video, respectively. The main advantage of this approach is that the entire process is automatic and no *a priori* human interaction is needed.

The results obtained from our test show that the model is effective for finding *keyframes* and is not computationally expensive. However as shown in the experiments, the use a single source of information, *i.e.*, color information, is not enough to provide us discriminative information. Therefore, as future work we plan to study the use of spatial or texture information that can be extracted using non computationally expensive algorithms. We impose this requirement on the algorithms load due to the fact that a large number of frames (images) is processed when dealing with videos. And performing complex computations over this quantity of images would require an huge amount of time making the whole process unbearable.

In this research area, drawbacks are found as follows. In order to produce a perfect video summary a model has to completely capture the semantic of the video. Semantic comprehension is one the most open problems in our area. No model has been created yet that can perfectly solve semantic understanding. Therefore, video summarization is subject to this problem. There is also a problem of subjectivism. That is, extracting something that we would consider semantically relevant might not be considered that relevant for other persons.

VI. Acknowledgements

The authors would like to thank FAPEMIG, CAPES and CNPq for the financial support.

References

[1] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy art: an adaptive resonance algorithm for rapid, stable classification of analog patterns," in *International Joint Conference on Neural Networks (IJCNN)*, vol. 2, 1991, pp. 411–416.

[2] M.-C. Hung and D.-L. Yang, "An efficient fuzzy c-means clustering algorithm," in *IEEE International Conference on Data Mining (ICDM)*, 2001, pp. 225–232.

[3] T. Wan and Z. Qin, "A new technique for summarizing video sequences through histogram evolution," in *International Conference on Signal Processing and Communications (SPCOM)*, 2010, pp. 1–5.

[4] R. Laganière, P. Lambert, and B. E. Ionescu, "Video summarization from spatio-temporal features," in *ACM TRECVid Video Summarization Workshop (TVS)*, 2008, pp. 144–148.

[5] W. Ren and Y. Zhu, "A video summarization approach based on machine learning," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP)*, 2008, pp. 450–453.

[6] J. Nam and A. H. Tewfik, "Dynamic video summarization and visualization," in *ACM International Conference on Multimedia (MULTIMEDIA)*, 1999, pp. 53–56.

[7] F. Chen, M. Cooper, and J. Adcock, "Video summarization preserving dynamic content," in *ACM TRECVID Video Summarization Workshop (TVS)*, 2007, pp. 40–44.

[8] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang, "Automatic video summarization by graph modeling," in *IEEE International Conference on Computer Vision (ICCV)*, 2003, pp. 104–109.

[9] B.-W. Chen, J.-C. Wang, and J. Wang, "A novel video summarization based on mining the story-structure and semantic relations among concept entities," *IEEE Transactions on Multimedia*, vol. 11, no. 2, pp. 295–312, 2009.

[10] L. Li, K. Zhou, G.-R. Xue, H. Zha, and Y. Yu, "Video summarization via transferrable structured learning," in *International conference on World Wide Web (WWW)*, 2011, pp. 287–296.

[11] X. Yang and Z. Wei, "A novel video summarization algorithm," in *International Conference on Multimedia Technology (ICMT)*, 2011, pp. 98–101.

[12] J. Jiang and X.-P. Zhang, "Gaussian mixture vector quantization-based video summarization using independent component analysis," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2010, pp. 443–448.

[13] W. Bailer, E. Dumont, S. Essid, and B. Merialdo, "A collaborative approach to automatic rushes video summarization," in *IEEE International Conference on Image Processing (ICIP)*, 2008, pp. 29–32.

[14] S. de Avila, A. da Luz, and A. A. de Araújo, "Vsumm: A simple and efficient approach for automatic video summarization," in *IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2008, pp. 449–452.

[15] F. Hong-cai, Y. Xiao-juan, M. Wei, and Y. Cao, "A shot boundary detection method based on color space," in *E-Business and E-Government (ICEE), 2010 International Conference on*, may 2010, pp. 1647 –1650.

[16] G. G. L. Priya and S. Domnic, "Video cut detection using dominant color features," in *Proceedings of the First International Conference on Intelligent Interactive Technologies and Multimedia*, ser. IITM '10. New York, NY, USA: ACM, 2010, pp. 130–134. [Online]. Available: http://doi.acm.org/10.1145/1963564.1963586

[17] Open Video Project, "Open video database," http://www.open-video.org, 2011, [Online; accessed on November 23, 2011].