Video Summarization

Edward Jorge Yuri Cayllahua Cahuina, Guillermo Camara
PPGCC - Programa de Pós-Graduação em Ciência da Computação
UFOP - Universidade Federal de Ouro Preto
Ouro Preto, Minas Gerais, Brasil
email: ecayllahua1@gmail.com, gcamarac@gmail.com

***Resumo*—In this paper we study several approaches for video summarization. We focus our efforts in static video summarization. Different methods are reviewed and we implement one the methods based on spatiotemporal features. This method is proved to be effective in video summarization.**

***Keywords*-Video Summarization, static video storyboard, algorithms, spatiotemporal features.**

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. A perfect example of this task is video.

Video information is growing exponentially: Each day an enormous quantity of video is uploaded to the internet; TV video information is generated every day; 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 in 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 of a video sequence and generate the correspondent video summary. Obviously, the key part is to recognize these relevant frames or portions of video, and this adds a certain subjectivism to the methods in the literature because different methods have different points of view of what is relevant and what is not.

Many methods have been proposed for video summarization, dynamic video skimming usually have very complex models which demands a long time for its implementation, so for this first part we have chosen to develop a static video storyboard which is more suitable for the time frame we have been given.

We have implemented the method proposed by [1]. This method first extracts the frames from the video and computes their spatiotemporal features. For this extraction the method uses the spatiotemporal Hessian matrix which proves to be a good feature extractor and also provides a measure of the activity that happens within each frame. Later, this information is processed to extract the most important frames (*keyframes*) based in the frames with higher activity and finally, it constructs a clip with the *keyframes* that are considered relevant.

This article is structured as follows. Section II gives a brief explanation of some of the methods that we have reviewed and we provide a wider presentation of the method we have implemented. In Section III we will review and analyze the complexity of the algorithms that we have developed. After that, in Section IV we present the different results that we have obtained in our experiments and we will discuss about them. Finally, in Section V we present some conclusions and we also propose some of future work to be done.

II. Methods

First we need to define some of the concepts we will use in this paper:

- A video: is a sequence of consecutive frames.
- A frame: A single synchronized picture on a roll of movie film.
- *keyframe*: A single frame that can represent other frames in the same video.

In the following, we review some of the methods that have been proposed by researchers in this area.

A. A Video Summarization Approach based on Machine Learning

In [2], the authors propose a method that relies on machine learning. The method first detects what they consider the principal features; these features are based on pixel values, edges and histograms. Then these features are used into their machine learning system to predict video transitions. For each video transition if the value computed by the neural network exceeds a threshold then a *keyframe* is detected and is marked for the construction of the video summarization. According to their experiments this method is robust when dealing with movies where a lot of video effects such as *Cut*, *Fade-in*, *Fade-out* or *Dissolve* are presented.

B. An Improved Sub-optimal Video Summarization Algorithm

In [3], a greedy algorithm is proposed which has more simplicity and a good performance but is not as robust as [2]. The method proposed takes as input the desired temporal rate, *i.e.* the total number of frames T_f that the final summary will have. Then the method adds the first frame by default to the summary. Afterwards, the method computes the distortions of the current frame. If the distortion is large enough and we have not reached T_f the frame is added to the video summary. This greedy algorithm performs fast but according to their results the final summary is not robust enough.

C. MINMAX Optimal Video Summarization

In [4], they present an algorithm based on dynamic programming where a MINMAX optimization is used although we do not know how robust their model is. Their method is also based on the distortions of the frames, but they use the dynamic programming approach in order to minimize the maximum distortion of the video summary under the premise that this will result into a better video summary for the final user.

D. Video Summarization from Spatio-Temporal Features

Most of the methods proposed in the literature follow a general scheme. We can subdivide this scheme into two principal steps. The first step is to somehow extract from the original video the frames that are considered relevant. In the second step all the frames that where considered relevant are used to construct the final video summary. We have chosen to implement the method proposed by [1]. This method follows the general scheme that we have just described.

For a better understanding, Figure 1 shows how the model works. First, we load the video. Then, we extract its corresponding frames. For each frame contained in the video, we detect its spatiotemporal features using the Hessian matrix. Afterwards, we use these features to compute the level of activity of the frame. If this level is too high then we can flag this frame as a *keyframe*. After processing all the frames in the video, we have a set of *keyframes*, we will filter these *keyframes* in order to extract the most representatives. Finally, once we have our set of most representative *keyframes* we use them to construct a video sequence. The final result will is summarized video.

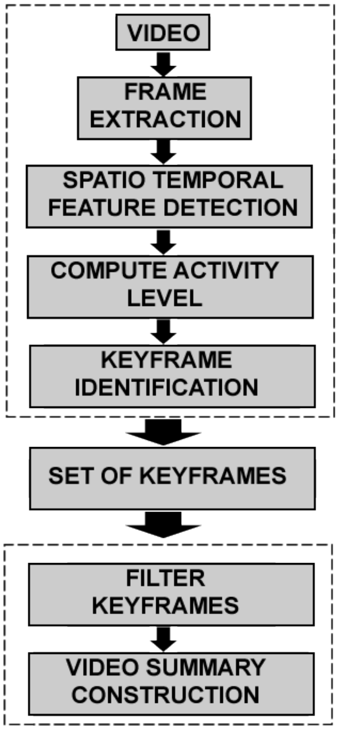

Figura 1. Model Scheme for Video Summarization from Spatio-Temporal Features

1) Spatio-Temporal Feature Detection: We have used the Hessian matrix, which is known to produce stable features and is also used in object recognition and feature matching applications. We will extend its use to a 3D scenario, since a video is a sequence of frames and a frame can be considered as an image I synchronized in the video. Our frame will have information about space and time, therefore we will represent the information contained in a frame as: $I(x, y, t)$, where x, y is the spatial information and t is the temporal information of I frame. Our Hessian matrix is defined as:

$$H(I) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial x \partial t} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} & \frac{\partial^2 I}{\partial y \partial t} \\ \frac{\partial^2 I}{\partial x \partial t} & \frac{\partial^2 I}{\partial y \partial t} & \frac{\partial^2 I}{\partial t^2} \end{bmatrix} \tag{1}$$

We work with frame I as an image, so for each pixel in I we use Equation 1 to compute its corresponding Hessian Matrix H. We are using the following masks to compute the second derivative:

$$mx = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix} \tag{2}$$

$$my = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \tag{3}$$

$$mxy = \frac{1}{4} \times \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{4}$$

2) Frame Activity Level: We have defined a threshold th. Since we have calculated H we now compute the determinant of the matrix H.If $det(H) > th$ then the pixel is marked as a feature in the image. We store all the features detected in another matrix I' of the same size of I. Once all the pixels have been processed, we count the number of features in matrix I' and we will store this value in a structure n, so for each frame I we will have its correspondent $n(I)$. We will consider as *keyframes* the ones with the most salient activity levels. In order to extract the *keyframes* we use the concept of local maxima.

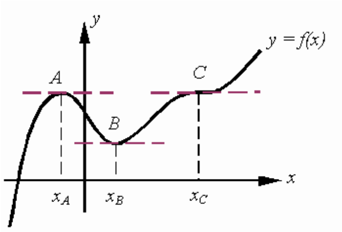

Figura 2. Local Maxima of a Function

As seen in Figure 2 a local maxima occurs at A. We can detect the local maxima of a function using:

$$\frac{\partial^2 f}{\partial x^2} < 0 \tag{5}$$

For example, in Figure 3, we show the level of activity of a certain video. As we can see, in Figure 3, if we were to detect the local maxima in that signal, a lot of *keyframes* would be selected making our summary useless. The idea of filtering the signal so that we can detect only the ones with the most salient levels is useful here.

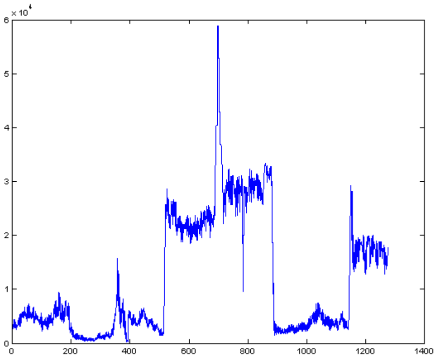

Figura 3. Activity levels of a movie

3) Keyframes Filtering: We now have in n the activity levels of all the frames. The ones with the most salient activity levels are our *keyframes*. As we have seen in Figure 3, filtering the signal gives us a better summary. We have used the median filter in order to filter this signal.

The median filter is known to attenuate the high frequencies while smoothing the signal. Applying the median filter allows us to attenuate these "fake" high levels of activity. So after the filtering only the real and most salient frames are selected. In Figure 4 we can see the final result after filtering the signal of Figure 3. As we can see, filtering the signal of levels of activity produces a much better summary.

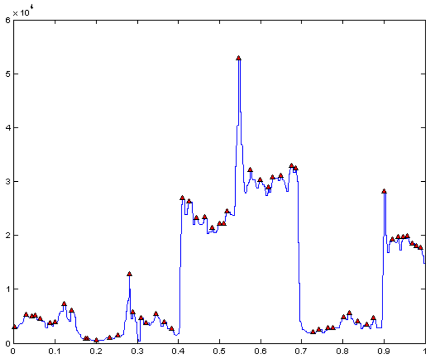

Figura 4. Filtered Signal of the Activity Levels

III. Complexity Analysis

We now analyze the complexity of our model. The main component that we analyze is the spatiotemporal feature detection. Given a video V containing a total number of frames L. We have to process all the L frames in order to extract the summary from V, then for each frame we compute the Hessian matrix for each pixel. So for a frame I of size $M \times N$. Given that for each pixel we perform 9 local operations in order to compute the second derivative in x and y, there is also 3 operations for x and another 3 for y. Computing the determinant takes another 9 local operations. Making it a total of 24 local operations for each pixel, this causes the process to take up to $L \times 24 \times M \times N$ operations. The complexity would be: $\mathcal{O}(24LMN)$. We could ignore the constant 24 , and the complexity would be $\mathcal{O}(LMN)$.

Computing the activity level can be executed in a single operation and can be performed during the spatiotemporal feature detection stage. After that, once we have the potential *keyframes*, they are stored in a linear structure (vector). Filtering the *keyframes* is performed in a linear operation with complexity $\mathcal{O}(n)$, where n is the vector containing the activity level values. Constructing the video summarization is a linear operation with $\mathcal{O}(K)$ where k is the length of the vector of the most salient *keyframes*.

In order to process the video we have to process all the frames, but we do not have to load all at the same time. For each operation we only need three frames in order to compute the Hessian matrix, therefore we could consider an approach where we would only need to load into memory $3 \times M \times N$ values. The vector of levels of activity is of size n and it can not be larger than the total length of the movie, *i.e.* $n < L$.

IV. Experiments

The model is coded in Matlab and the experiments have been executed on a Core 2 Quad Intel processor with 4 GBytes of RAM with Windows 7 as O.S. . A movie is usually composed by several frames. For example a 1 minute movie can contain about 1100 frames. It is difficult to show a whole sequence of a video in this document, therefore the test shown here is a extraction of a video.

In Figure 5, we show the results from the model. We have taken a small sequence of frames to show how the model works. In this small sequence of frames, the model has detected *keyframe* 60 as the one with the most salient level of activity. The remaining of the frames are disposed and only *keyframe* 60 is considered for the final video summary.

So far, there is no standard data base for video summarization tests. Moreover, the real test in how good or how bad the video summary is can only be judged by a final user and his/her judgement can also be very subjective making it difficult to set a metric. That is why in the literature most of the tests have been executed with local videos and most of the times no surveys on final users were executed.

V. Conclusions and Future Work

The following conclusions have been formulated:

- The most important part in video summarization is to give a solid model to extract the *keyframes*.
- *Keyframes* can be very subjective, what one person can consider important information for another person it is not.
- This area of research is still very young, there is still no standard database for researchers to analyze their methods.

Some future works:

- Change the computation of the second derivative, because using masks can be very time consuming.
- Extend the model for dynamic video summarization.

Referências

[1] R. Laganière, P. Lambert, and B. E. Ionescu, "Video summarization from spatio-temporal features," 2008.

[2] W. Ren and Y. Zhu, "A video summarization approach based on machine learning," *International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IEEE*, 2008.

[3] L. Coelho, L. A. D. S. Cruz, L. Ferreira, and P. A. Assunção, "An improved sub-optimal video summarization algorithm," *International Symposium ELMAR (Electronics in Marine) - 2010, IEEE*, 2010.

[4] Z. Li, G. M. Schuster, and A. K. Katsaggelos, "Minmax optimal video summarization," *IEEE Transactions On Circuits And Systems For Video Technology*, 2005.

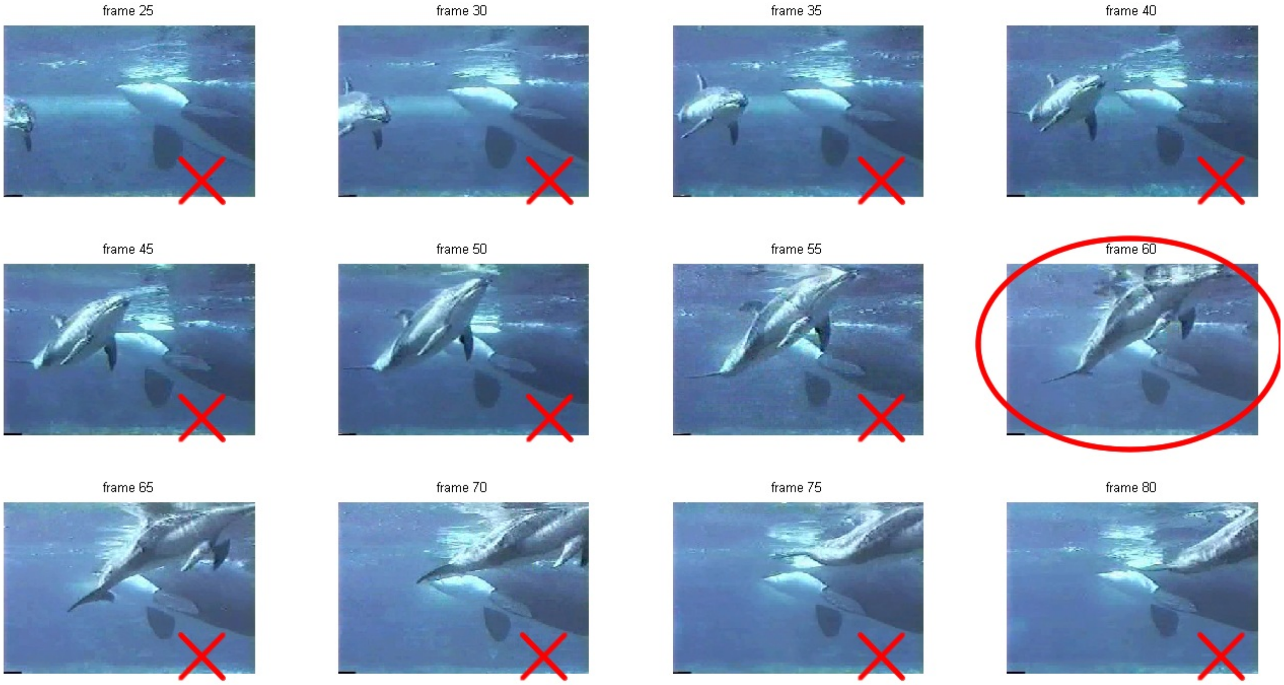

Figura 5. Frame 60 identified as the *keyframe* of its neighbors