**VIDEO SUMMARIZATION**

**Abhay Mathur, Saksham Saraswat**

Arizona State University

[mathurabhay22@yahoo.in](mailto:mathurabhay22@yahoo.in), ssarasw3@asu.edu

1. **Problem Statement**

Media is always evolving and the evolution of digital video has led to the production of many new multimedia applications. Many professional and education applications involve utilization of large volumes of video. Making a summary of these videos can improve the efficiency and the usefulness of these multimedia applications, and of course save time. YouTube.com is a wide used commercial application with millions of users and several billions of videos. Summarization of videos can also be convenient for the consumers of these commercial applications. So how do we come up with an algorithm which allows us to extract the video, make a summary and evaluate the summary?

1. **Related Work**

Video summarization is basically of two types, key frame extraction and video skims.

* 1. **Key Frame Extraction**

Key Frames in a video are the ones which best represents the summary of the video. Works done in the past have used different techniques to extract key frames.[3] Previous works have presented the frames in different color spaces (viz RGB, YUV). Clustering based methods have used different low-level or high-level features to differentiate between the frames. Previous works also differ from each other on the clustering algorithm they used to create clusters of frames, including hierarchical clustering. Furthermore, there are some algorithms which, instead of using clustering, used web image priors [4].

* 1. **Video Skims**

The other type of video summarization produces short clips of videos to represent the summary. Some methods take a few frames around the key frames to make a video skim. Other more complex methods tries to understand the semantic meaning of the videos by using techniques like camera-motion detection to produce video skims[5].

Few of the other summarization algorithms are mentioned below:

**Vgraph** – Is a method based on partitioning the video frames into different shots using color features.

**LSTM** – Long short term memory are a special kind of recurring neural networks. Features are fed into the lstm chains which then gives the importance of each feature.

**SIFT** – Scale-invariant feature transform is an algorithm used in computer vision to detect and describe local features in images. This method suggests to detect video shots by detecting a key objects in each frame and then tracking motion of these objects to detect shot boundaries.

Our approach takes advantages from the aforementioned works. Moreover, we describe an evaluation method, which many of the previous works failed to do.

1. **Our Approach**
2. **Uniform Sampling**

Instead of analyzing each frame, we sample the video uniformly to get 100 frames. The sampling rate is adjusted according to the total number of frames in the video. Since we only want 100 frames, the longer the video, lesser is the sampling frequency. Of course, if we skip this step, we would get better results, but at the cost of time. If we decide to work on every frame, the time complexity of the tool will increase manifolds, and by the time the tool will output a summary, the user would have easily watched the whole video, which would eliminate the purpose of this tool. The response time of this tool must be much less that the duration of the video.

1. **Creation of feature vectors (HSV Histograms)**

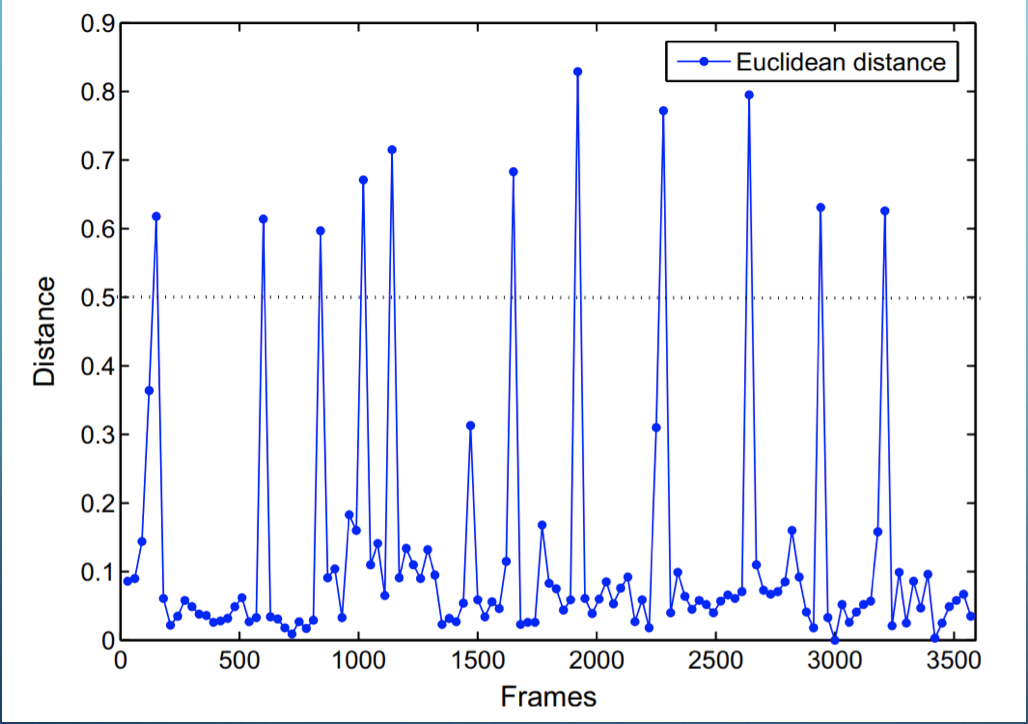
HSV space is a better descriptor of images as compared to RGB space. Hence, we convert all the sampled frames received after step one to HSV space. Now, we create 3 histograms per frame, corresponding to hue, saturation and intensity values. Now we concatenate these 3 histograms (of size 256 each), to create a feature vector (of size 256+256+256=768) for each frame.

1. **Finding the Value of K**

The feature vectors obtained in the previous step will serve as one of the input to the K-means algorithm. The other input is the value of K. In this step, we try to find the appropriate value of K for the video.

We plot a graph representing pairwise Euclidean distance between each pair of frames, as shown in Fig 1. Then we calculate a threshold value based on the minimum and maximum values received in this graph. The number of peaks received in this graph, which are above the threshold value), represent the number of times there was a significant change in the scene or shot. In Fig 1, we can see that there are 11 peaks above the threshold value (0.5), indicating that the appropriate value of K (and hence number of clusters) should be 12 (11+1). The formula we used for calculating threshold is

Where *min* and *max* represents the minimum and maximum distance value received in the graph

******

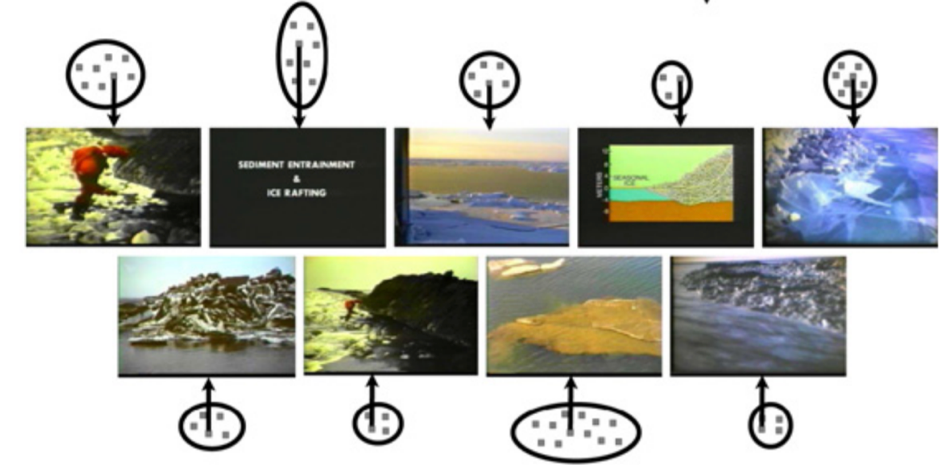
*Fig. 1: Pairwise Euclidean distance between the frames*

1. **K – Means Clustering**

All feature vectors of the frames and K are input to the K-means clustering algorithm. The value of the ‘Distance’ parameter was given as ‘cityblock’. Call to this function, creates K number of clusters containing one or more frames.

1. **Extraction of Key frames**

One of the outputs of the K-means clustering algorithm is a matrix containing distances of every point to each centroid. We analyze this matrix to see which frame is closest to the centroid of its cluster. The frame closest to the centroid of the cluster is chosen as the key frame and included in the summary. We also have the option to give initial seeds for clusters but that doesn’t guarantee better efficiency so we just let the program decide the centroid of each cluster. Only one key frame per cluster is extracted. This step is showed in Fig 2.



*Fig. 2: Showing extraction of key frames from key clusters.*

1. **Evaluation**

**4.1 Database**

The database we used for evaluation of our project can be downloaded from with link: <https://sites.google.com/site/vsummsite/download>. This website contains 50 videos from Open Video. All videos are in MPEG-1 format (30 fps, 352 x 240 pixels), in color and with sound. These videos are distributed among several genres (documentary, educational, ephemeral, historical, lecture) and their duration varies from 1 to 4 minutes and approximately 75 minutes of video in total.

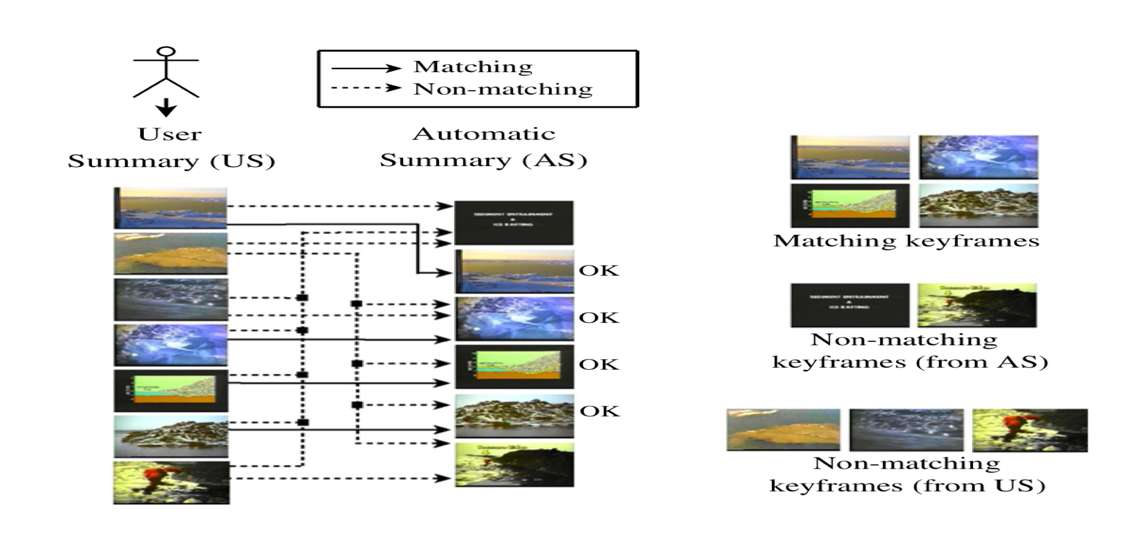
It also contains 250 user summaries. These summaries were created manually by 50 users, each one dealing with 5 videos, meaning that each video has 5 video summaries created by 5 different users.

**4.2 Calculating Efficiency**

Evaluation is necessary to assess the efficiency of our tool. We compare the summary generated by our tool to five different summaries manually created by five different users for the same video. The names of the output frames are of the form *‘Framexxxx’* where *xxxx* represents the frame number. The summaries are compared using the frame numbers. Here we assume that one shot/scene is made of at least 100 frames. We match the frame numbers in manually created summaries with our summary. If difference in two frame numbers is less than or equal to 100, we consider these frames similar.

Efficiency is calculated as the percentage of manually created frames found in automatically created frame set. This process is shown in Fig 3.

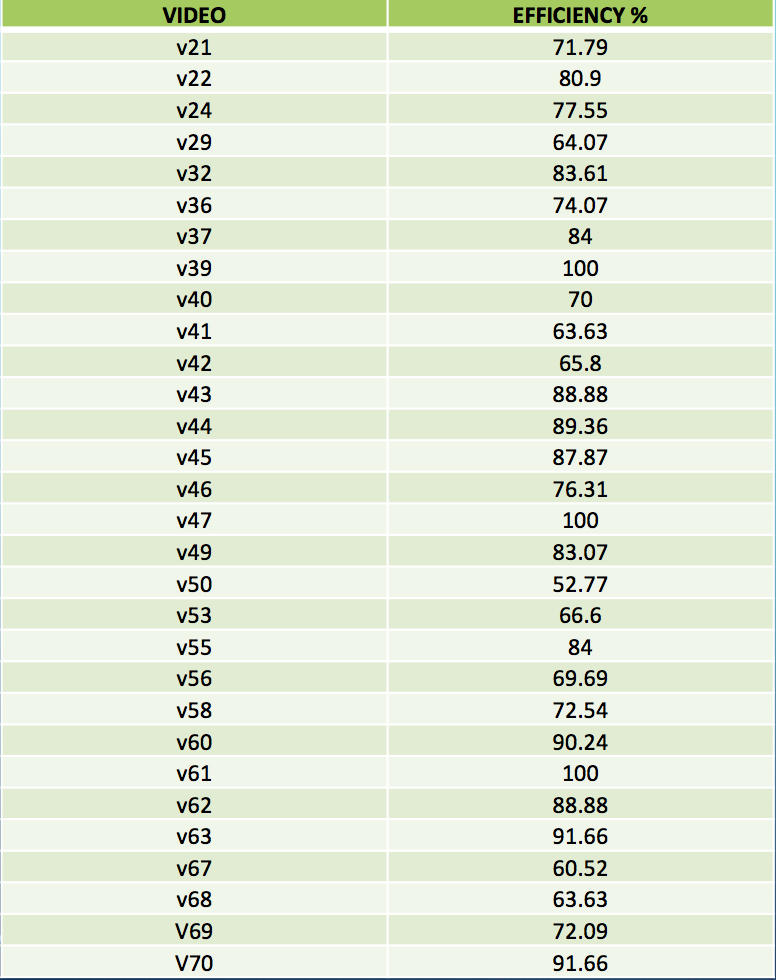
Another more complex way to compare the two summaries is by comparing histograms of the two frames. This method would require setting a threshold value for the difference in the two histograms; and taking decision based on if the difference between the histograms is greater than the threshold value.



*Fig. 3: Comparison of Automatic Summary and User Summary*

1. **Results**

We calculated the efficiency percentage of 50 videos of different types. The efficiency of videos ranged from 60% to 100%, as shown in Fig 4. The efficiency of the summary depends on how visually dissimilar the neighboring shots are. Efficiency is less for videos in which different scenes look much similar. Fig. 4 shows some of the efficiency percentages.



*Fig 4.*

1. **Conclusion**

Summarization of a video is an effective process since it allows the users to preview a summarized version of long videos which would improve the efficiency of users and the application used for commercial and educational purposes. Our program was implemented in matlab since it has useful inbuilt functions, for example, K-means clustering, calculating histogram etc. Our approach proved to be much efficient because it is based on how humans perceive the video as sequence of different scenes. We represent our video frames as HSV histograms. K-means algorithms divide these frames in different clusters based on their histogram values. And then we extract one frame per cluster which can be the best representation of the whole cluster. We had database of manually created summaries by 5 people, which we used to evaluate our tool.

1. **References**

*[1] VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method -* [*http://laplace.dcc.ufmg.br/npdi/uploads/96a40be5-db2d-f171.pdf*](http://laplace.dcc.ufmg.br/npdi/uploads/96a40be5-db2d-f171.pdf)

*[2]:Video Summarization: an overview of techniques -* [*http://home.iitk.ac.in/~kanishkg/Video\_Summarization\_Final\_Report.pdf*](http://home.iitk.ac.in/~kanishkg/Video_Summarization_Final_Report.pdf)

*[3] N. Ejaz, I. Mehmood, and S. W. Baik, “Efficient visual attention based framework for extracting key frames from videos,” Signal Processing: Image Communication, vol. 28, no. 1, pp. 34–44, 2013.*

*[4] G. Kim, L. Sigal, and E. P. Xing, “Joint summarization of large-scale collections of web images and videos for storyline reconstruction,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 4225–4232.*

*[5] K. Chorianopoulos, “Collective intelligence within web video,” Human-centric Computing and Information Sciences, vol. 3, no. 1, p. 1, 2013.*