(Semi) Automatic Infinite Entertainment System

Ana Sofia Afonso
48303
asl.afonso@campus.fct.unl.pt

Sara Fernandes
47872
sj.fernandes@campus.fct.unl.pt

1. INTRODUCTION

This project's main goal is to continuously play video based on information retrieved from a camera.

In order to achieve this, the report will carry out a study of the state of the art regarding video interfaces, dynamic video searching, content-based similarity search, among others. Furthermore, we will also specify the interface that will be created, comprising it's features as well as the type of metadata that will be stored for each video.

Regarding the project, there are two main parts: there's a camera that retrieves video and audio from the environment, and the other part will extract characteristics from the videos and use these to determine which video is playing at the moment.

The final user will only watch a continuous stream of videos (they won't be able to pause, play, or choose the videos that will be played). The operator will be able to visualize the videos' metadata, make playlists, browse through the content and watch the live feed of what is happening on the end user's side.

2. STATE OF THE ART

Video exploration and analysis is a long and strenuous process because it involves searching through hours of videos. There have been several suggestions in order to make it easier for a user to search through video content, as well as propositions on how to make a video browsing interface more user friendly.

Since videos can either be several hours long, or only a few minutes, there have been several suggestions of scrollbar design in order to enable dynamic video browsing, that make it possible to change the slider's scale as well as the scrolling speed. These sliders, although useful for specific situations, have yet to be widely used and replace the usual scrollbar. [1]

According to Cees G.M. Snoek, a video search method based on text results in a disappointing retrieval performance. [2] Using annotations linked to videos, it is possible for the user to quickly see the content of a video and locate the interesting parts. That is what happens in TAV (Temporal Annotation Viewing), a video browser interface. In this interface, it is possible to add annotations to videos, and therefore that makes it possible to search videos by annotations, instead of just by their titles. This allows for a better video search as well as facilitate the skimming through the video content. [3]

There are also techniques used in order to improve navigation of educational videos that are similar to the one described above. These include having a highlight storyboard below the video, showing images of parts of a video. After conducting user testing, users found that this type of interface helped them to scan the video in an efficient and easy way, since they could easily identify the most important parts of a video.[4]

In order to summarize videos, it is possible to obtain a representative number of shots of a video using Machine Learning, which allows a quick assessment of the video content. [5] This is useful for creating video browsing interfaces that are easier to use.

There is a system that provides high quality content-based similarity search for unlabeled continuous archived video data by extracting visual and audio sample frames in time intervals, converts it from RGB into HSV colors and uses the system proposed by Sticker [6]. This tool has a small metadata overhead, which makes it useful and allows for a practical implementation for video search. [7]

A paper about 3D Face recognition using Kinect vises to solve a problem of 2D face recognition systems with a depth system that aids the process. This system prevents problems such as pose invariance, illumination variation and face images at arbitrary angles. [8]

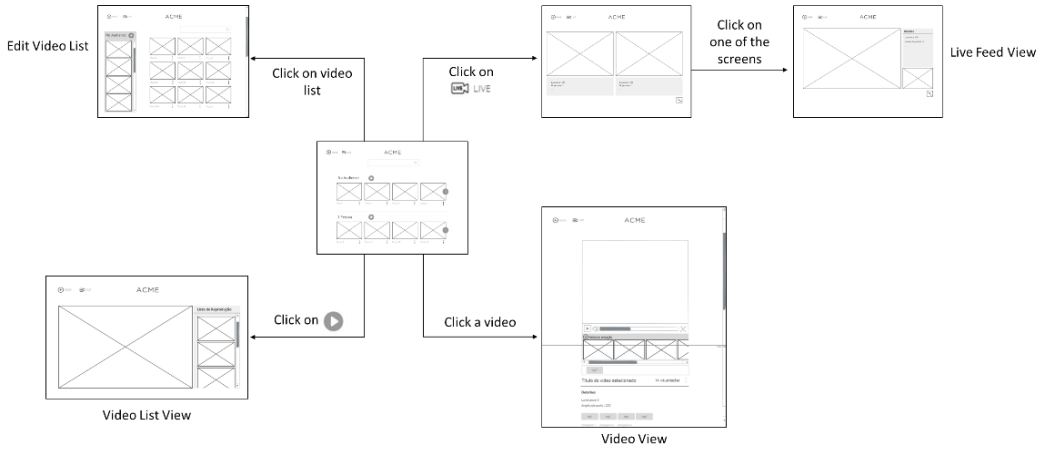

Figure 1 - Storyboard of the Operator Interface.

Besides recognizing faces, there have been studies carried out about emotion recognition. One of the techniques used involves exploring the content clues such as event, objects and scenes in a video. This method aims to calculate the likelihood of the objects in context of an event in a particular scene representing a particular emotion. [9]

3. SPECIFICATIONS

The goal of this assignment is to continuously play video based on information recovered from a camera. Based on the information retrieved from the camera, such as how many people are present, or what they're wearing, the system should be able to play adequate videos based on the analysis of the videos' metadata.

The metadata extracted from the videos consists of: tags or keywords, luminance, color, edge distribution, number of times a specific object appears in the video frame, number of traces appearing in the video, texture characteristics, rhythm and audio amplitude. The data will be stored in an XML file, so that it is only necessary to process each video once.

3.1 Interface

The interface for this assignment will consist in two parts: one for the final user, that only shows the videos in a continuous stream, and the operator view, that enables to browse the videos, create playlists of videos, visualize the metadata extracted from the videos, view playlists of videos grouped by events (such as videos to play when there's no audience), among others.

When the operator hovers the mouse over a video, a series of representative shots will be shown. This aims to make it possible for the operator to quickly understand the content of that specific video.

Each video will have a group of tags which will make it easier to know what that video is about and simplify the search process. The lists of videos (similar to playlists), will make it easier to group similar videos and minimize the need of searching. This way, a playlist about a certain theme can be selected instead of selecting only one video at a time.

When the operator is watching a video, there will be a list of the most important parts of that video underneath it, which we will call annotations. When an annotation is clicked, the video jumps to the annotation's timestamp, and it's possible to check the keywords associated to that part of the video. The goal of this feature is to make it possible for the operator to quickly understand what the most important parts of a video are, which becomes even more useful in long videos.

The storyboard in Figure 1 presents the main features of the interface of the operator. It is possible to view a video, a list of videos, view the live feed and edit a list of videos. As side functionalities it's also possible to add videos to the interface, as well as tags and new lists of videos.

3.2 Shortcuts and key binds

Figure 2 shows the keyboard shortcuts that we will implement in order to improve the interface usability for the operator. S will enable to Stop the video, P will make it possible to pause and unpause the video. L will show the live feed window, showing both the camera and video playing at the moment. In order to put the video in full screen, the operator can use the key bind Alt + Enter, and ESC or Alt + Enter again to get out of the full screen mode.

The numbers from 0 to 9 will make it possible to navigate the video in segments, 0 corresponding to the beginning of the video and 9 to the last segment. In order to enable easier navigation through a video, it is possible to use the horizontal arrows to push the video back or forward 10 seconds at a time. The vertical arrows can be used to control de volume of the video.

Finally, the key binds Ctrl+N (Next) and Ctrl+B (Before) allow the operator to play the next or previous video when watching a playlist of videos.

Figure 2 - Keyboard shortcuts and keybinds to be created in our interface.

4. ALGORITHMS AND TECHNIQUES

4.1 Object

The ORB algorithm receives two grayscale images as input, one with the object that we intend to detect, and the other one with a frame of the current video being analysed.

Using openCv's BFMatcher, both images are analysed and we retrieve an output keypoint for each one, as well as a descriptor Matrix.

With the knn Neighbours algorithm, both images are compared, giving a new matrix as an output. Finally, the values in the matrix are compared and, if the results in line zero are below or the same as the results in line one, that is a match, meaning that the object is present in that specific frame.

4.2 Texture

24 Gabor kernels were calculated,with 4 different frequencies and 6 different orientations, with 3x3 matrixes, with sigma of value 5, orientation ranges from 30º to 210º with intervals of 30º, the frequency ranges from 1 to 8 with intervals of 2, and finally the offset is always 0.5. With this Kernels we calculate the mean and the standard deviation of each grayscale frame.

4.3 Edge Distribution

For the edge distribution, we used openCv's cv::Mat in order to create the edge filters (vertical edge filter, horizontal edge filter, 45 degree edge filter, 135 degree edge filter and non-directional edge filter).

4.4 Rhythm

Using the value obtained through the edge distribution calculation for each video, the difference between the grayscale histograms is assessed. If this value is higher than a certain threshold (in this case, we used ⅔ of the total image pixels), that frame is included.

In the end, we are able to get a sum of all the frames that are higher than the threshold, having the total number of frames that exceed it.

5. APPLICATION IMPLEMENTATION

When the application first loads, it checks the videos directory for video files. In case it isn't empty, it proceeds to iterate through this directory and fill an array of videos and calculate the metadata that will be saved to the XML file.

The values that are calculated are the number of faces appearing in every video frame,

6. CONCLUSIONS

This paper allowed us to learn the language C++, and provided an opportunity to learn about openFrameworks and OpenCV, since the group had no prior knowledge of it.

Furthermore, it was possible to put in practice some of the algorithms, techniques and subjects learned in the theoretical classes, which we consider to be a very positive aspect of the project.

Regarding the application, there are some functionalities that we couldn't implement, such as changing the video according to the number of faces.

Besides that, it also wasn't possible to implement the user interface. However, the Live Feed View acts as a debugger for the admin, and can be used to mimic the behavior of the user interface.

We would have liked to have more time in order to tweak these aspects, as well as to improve overall code and class organization.

7. REFERENCES

[1] Wolfgang Hürst, Georg Götz, and Philipp Jarvers. 2004. Advanced user interfaces for dynamic video browsing. In *Proceedings of the 12th annual ACM international conference on Multimedia* (MULTIMEDIA '04). ACM, New York, NY, USA, 742-743. DOI: https://doi.org/10.1145/1027527.1027694

[2] Cees G.M. Snoek. 2010. The mediamill search engine video. In *Proceedings of the 18th ACM international conference on Multimedia* (MM '10). ACM, New York, NY, USA, 1323-1324. DOI: https://doi.org/10.1145/1873951.1874212

[3] Stefanie Müller, Gregor Miller, and Sidney Fels. 2010. Using temporal video annotation as a navigational aid for video browsing. In *Adjunct proceedings of the 23nd annual ACM symposium on User interface software and technology* (UIST '10). ACM, New York, NY, USA, 445-446. DOI: https://doi.org/10.1145/1866218.1866263

[4] Juho Kim, Philip J. Guo, Carrie J. Cai, Shang-Wen (Daniel) Li, Krzysztof Z. Gajos, and Robert C. Miller. 2014. Data-driven interaction techniques for improving navigation of educational videos. In *Proceedings of the 27th annual ACM symposium on User interface software and technology* (UIST '14). ACM, New York, NY, USA, 563-572. DOI: https://doi.org/10.1145/2642918.2647389 Ding, W. and Marchionini, G. 1997. *A Study on Video Browsing Strategies*. Technical Report. University of Maryland at College Park.

[5] Marie-luce Viaud, Olivier Buisson, Agnes Saulnier, and Clement Guenais. 2010. Video exploration: from multimedia content analysis to interactive visualization. In *Proceedings of the 18th ACM international conference on Multimedia* (MM '10). ACM, New York, NY, USA, 1311-1314. DOI: https://doi.org/10.1145/1873951.1874209

[6] . Zhe Wang, Matthew D. Hoffman, Perry R. Cook, and Kai Li. 2006. VFerret: content-based similarity search tool for continuous archived video. In Proceedings of the 3rd ACM workshop on Continuous archival and retrieval of personal experiences (CARPE '06). ACM, New York, NY, USA, 19-26. DOI=http://dx.doi.org/10.1145/1178657.1178663)

[7] Sticker (M. Stricker and M. Orengo. Similarity of color images. In SPIE Conference on Storage and Retrieval for Image and Video Databases III, volume 2420, pages 381-392, Feb. 1995)

[8] . Rahul Ajmera, Aditya Nigam, and Phalguni Gupta. 2014. 3D Face Recognition using Kinect. In Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing (ICVGIP '14). ACM, New York, NY, USA, Article 76, 8 pages. DOI: https://doi.org/10.1145/2683483.2683559

[9] Chen Chen, Zuxuan Wu, and Yu-Gang Jiang. 2016. Emotion in Context: Deep Semantic Feature Fusion for Video Emotion Recognition. In Proceedings of the 24th ACM international conference on Multimedia (MM '16). ACM, New York, NY, USA, 127-131. DOI: https://doi.org/10.1145/2964284.2967196