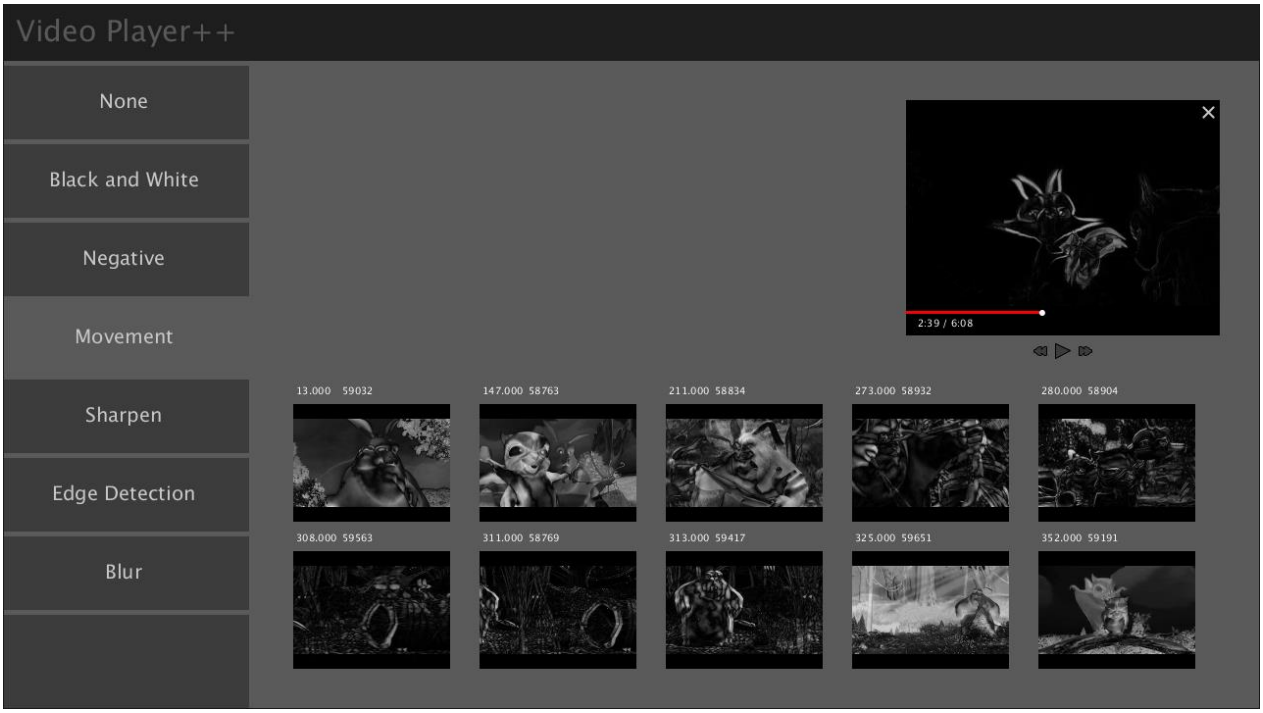
UNIVERSIDADE da MADEIRA

Masters in Informatics Engineering

Multimedia Systems

Teacher: Diogo Cabral

Video Player++



Project by:

- Diogo Cruz, nº 2030115

- Diogo Nóbrega, nº 2027415

Introduction

We were asked to develop an application using Processing or openFrameworks that detects “relevant” moments in a video. The definition of “relevant moment” should be defined by us. It could be the quantity of movement, edges, a color or any other characteristic that would be interesting.

These moments should be represented by thumbnails and these should allow to navigate in the video. In our implementation of this project we tried to be as creative as possible, giving the user a reliable and intuitive video player interface.

User Interface

In our Video Player, the user can either select a file from directories (Default file explorer) or capture a video from the computer’s camera (if it exists / is enabled / is supported by Processing).

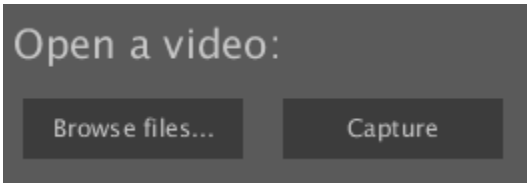


Figure 1 - Ways of selecting a video

Our interface consists of one “screen/menu” giving the user the possibility to change the desired effect at any time. Changing the effect also causes the thumbnail selection to update and offer relevant ones.

Figure 2 - Available video effects

When selecting an effect, the analysis and selection of the most relevant thumbnails occurs asynchronously, so that it doesn't stop video playback. This way the user doesn't have to wait for this process to finish before starting/continuing to watch the video. We also implemented a progress bar in the analysis and selection of thumbnails to alert the user that this process takes a bit of time and to inform on its status (depends on the length of the video).

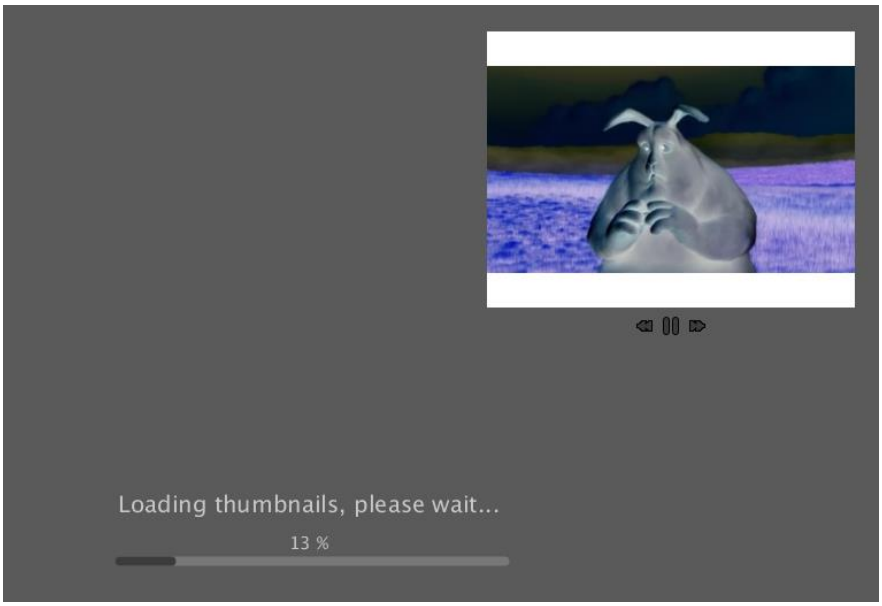


Figure 3 - Indication of progress in the analysis of frames, while video continues to play

The video itself has some basic controls, common to other video players, like jumping forward / backward (buttons and video edges), pausing / playing, dragging of timeline bar to seek, close the video, displaying its length, amongst others.

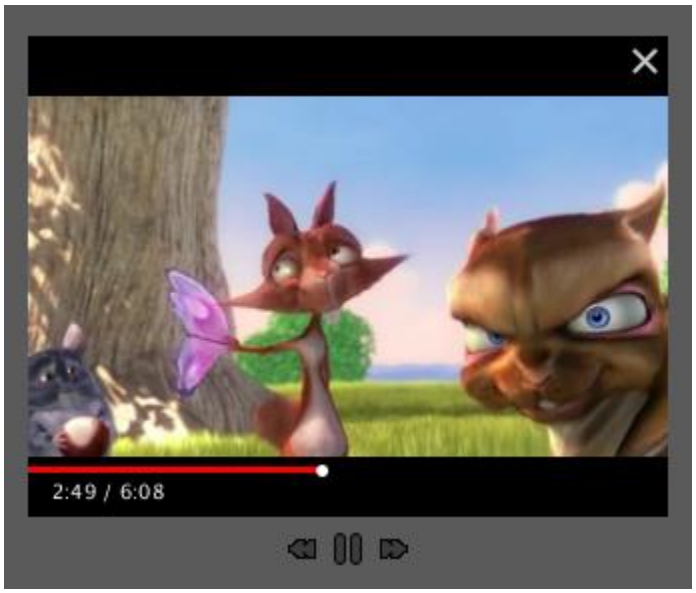


Figure 4 - Basic video controls

After the generation of relevant moments in the video according to the select effect, it's also possible to jump to a specific frame in the video by clicking the desired thumbnail.

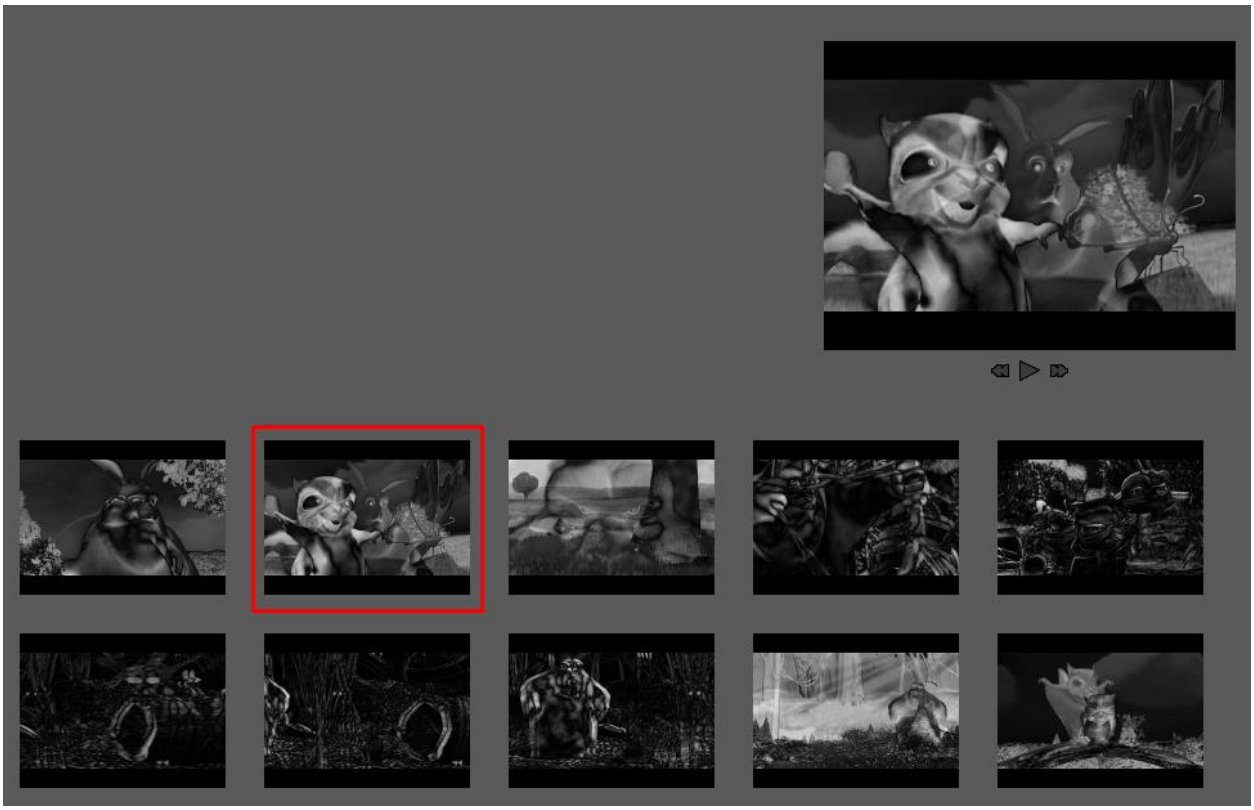


Figure 5 - Jump to the frame in the selected thumbnail (Movement Effect)

Architecture

As our user interface is quite complex and has a lot of elements (tabs, buttons, thumbnails, …) we opted for an approach based on the separation of concerns concept. Each element has its own state and behavior, can draw itself, can receive inputs and trigger actions.

The UI can be thought of as a tree, as every element can have other child elements. For instance, in our case, the video player itself has child elements (traditional control buttons, …).

Every UI element implements a UIElement interface and, as such, can define its own method to handle any input (mouse presses, mouse dragged, …). When the user triggers any input event, that input is checked by each appropriate element to determine whether it was directed towards it and, if it was, that element can then handle it by carrying out its defined behavior. For example, a mouse press is checked for by the video player itself. If it is handled by it or by one of its child elements, then that input event is can be dismissed. If it's not, then it is checked by the effect tabs. If it's also not handled by the tabs, then it keeps getting delegated to the next appropriate element until it is either handled by one or there are no further elements to handle it.

This separation of concerns concept is key in modern user interfaces. We chose to follow an approach based on it because it fits very well with our implementation, allowing us to maintain a very organized code base that is very open to change.

Relevance Methods

We implemented all the video processing effects that we looked at during the course. We start by collecting a sample of frames from the video, specifically one per second, and apply the selected effect. If the video is shorter than 10 seconds, the interval we use is smaller so that we can get a minimum number of thumbnail samples (default – 10). After collecting the samples, we analyze each one according to the respective relevance method and assign each one a score. We keep the 10 best ones (default number of thumbnails) and order them chronologically.

We settled in the following relevance criteria:

Effect	Relevance Method
None	Thumbnails divide the video in equal intervals.
Black and White	Thumbnails are chosen based on how dark they are. To obtain these thumbnails, we analyze each one, summing up each RGB component of each pixel. The thumbnails with the smaller total sum are darker and are chosen.
Negative	Thumbnails are chosen based on how much difference in color there is between the original image and their negative counterpart. It might be difficult to perceive as it's not particularly intuitive. For each pixel we sum up the difference in color between the original frame and the negative one. The thumbnails with the larger total sum are chosen.
Movement	Thumbnails are chosen based on how much movement there is relative to the previous frame. Usually these moments represent cuts in a scene and we opted not to filter them out as we think it's an interesting outcome. We apply the movement difference effect, subtracting one frame from the other, and convert it to black and white. Then we count the number of pixels that aren't totally black since these are the difference between frames and represent change/movement. The thumbnails with the higher number of counted pixels portray more movement and are chosen.
Sharpen	Thumbnails are chosen based on how sharp they are. To get a measure for how sharp an image is, we first apply edge detection to it (also converts it to black and white) and then count how many edges were detected. Edge detection is a good way to measure how sharp an image is, since a blurry image is characterized by blurred/non-sharp edges. It can be difficult to see in a normal non-blurred video since all thumbnails are sharpened (after the effect is applied) and it's hard to distinguish between a sharp thumbnail and a slightly sharper one. However, in a video with some portions pre-blurred, it's clear that thumbnails from those portions don't get chosen. After applying edge detection, we count the number of pixels that have one of the three RGB components' values higher than 100. Essentially, all the pixels with some brightness. The thumbnails with the higher number of counted pixels have more edges and are sharper, and so are chosen.

Edge Detection	Thumbnails are chosen based on how many edges are detected. We first convert it to black and white. Then we count the number of pixels that have one of the three RGB components' values higher than 100. Essentially, all the pixels with some brightness. The more pixels we count the more edges there are. The thumbnails with the higher number of counted pixels are chosen.
Blur	Thumbnails are chosen based on how blurred they are. To get a measure for how blurred an image is, we first apply edge detection to it (also converts it to black and white) and then count how many edges were detected, for the same reason as in the sharpness method. It can be difficult to see as all thumbnails are blurred (after the effect is applied) and typically it's difficult to distinguish between them. However, in a video with some portions pre-blurred it's clear that thumbnails from those portions are blurrier and get chosen. To choose the best ones we follow an identical procedure to the sharpness method. After applying edge detection, we count the number of pixels that have one of the three RGB components' values higher than 50 (we look for pixels less bright than in the sharpness method to compensate for the fact that there are way less edges). The less pixels we count the less edges there are and the blurrier the thumbnail is. The thumbnails with the lower number of counted pixels are chosen.

Discussion and Conclusions

We developed a video player using Processing. This video player allows its users to select a video from the file system or capture with a video camera and apply effects to it including movement detection, edge detection, among others, in real time.

The user can navigate through the video (capture not supported) using a number of thumbnails chosen based on the currently selected effect. The user can select another effect at any time and the thumbnails provided are updated asynchronously, so that the user can keep watching/interacting with the video in the meantime.

It's also possible to close the current video and open another one or even start capturing live video via camera.

To sum up, we were able to explore the concepts shown during the semester and put them to use in a creative way. Sometimes we paired a couple of concepts to analyze a frame, other times we played around with different thresholds to get a specific effect's thumbnail selection just right.

All in all, we believe that with the implementation of this project we were able to apply the knowledge obtained in the lectures/labs and gain a better understanding in the area of video processing, but most of all, we believe that we developed a great application that provides interesting functionality to its user and does so through a well suited user interface.