P2: FFMPEG

In this document it is described the FFMPeg Editor program to manipulate video and audio files using the ffmpeg library. This program contains a principle script call it window.py that is the responsible to call the functions that will edit our files, and show it in a small window that functions as options, where the user can interact without modify any script. Then the program looks like:

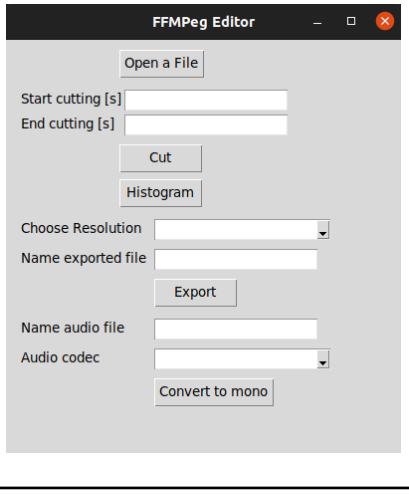

Figure 1: FFMPeg Editor interface

As can be observe, there are several options, let's focus on the first, where the user need to select an input mp4 file and then from a given time interval in seconds, where it can be specified in the entry of start cutting and end cutting. Pressing the *cut* button it will generated a mp4 file with the fragment chosen, but if the end cutting is a lower time than the start cutting it will show a window error:

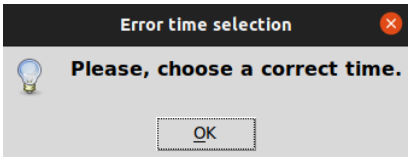

Figure 2: Window error time selection

This method to cut videos it could be found in clip_cutter.py where there is a function named cut fragment which needs as parameters the input file the star cutting and end cutting, to execute the following command:

```
ffmpeg_extract_subclip(file, start_time, end_time, targetname="test.mp4")
```

The next option it shows the yuv histogram of the cutted video and the video at the same time. This method it is in yuv_histogram.py file where there is defined the function histogram that gets as parameter the video cutted and execute in a terminal the following command:

```
ffplay file -vf split=2[a][b][b]histogram format=yuva444p[hh][a][hh]overlay
```

And it will show the output below:

Figure 3: YUV histogram

Below the histogram option there is another to export the cutted video in a given name and resolution as the entry's says, if not one of those are chose the program it will show a window error advertising the user:

Figure 4: Window advertising selection error

The resolutions that can be chose are, 1280x720p, 854x480p, 360x240p and 160x120p all of this with a 16:9 aspect ratio. This method it is located in resize_clipy.py where there is a function named *change_resolution(name, w, h)* where the enter parameters are the name of the output file, and w, h the width and height of the specified resolution. Then this functions executes from the terminal the next command:

```
ffmpeg -i input -vf scale=:h -preset slow -crf 18 name
```

The last option extracts from the previous video the audio and converts from stereo to mono and then it is exported with the codec specified by the user, that can be mp3, flac, aac or wav. As the previous option if it is not given a name and codec it will show another window error:

Figure 5: Window advertising of wrong given name or codec

There is a script named audio_to_mono.py with *mono_codec(name, codec)* function that gets as parameter the name of the output audio and the wished codec, then executes the following command in the terminal:

```
ffmpeg -i input -ac 1 name.codec
```