Laia Duch Soler

Report Seminar 2

For this seminar, I have created three different Python scripts. The main script is called 'main.py,' which includes a class named *VideoProcessing*. This class contains various methods I've developed to perform tasks 1, 2, and 3 of the seminar. The script also features an interactive section where the user must choose which task to execute.
The 'subtitles.py' script is designed for task 4, and it inherits from the previous script to test its functions. Finally, the 'histogram.py' script is crafted to accomplish task 6. This script also inherits from the first script to test the created function.

Task 1
This task involves taking the original video (BBB.mpeg), trimming it to the first 9 seconds, and calculating macroblocks and motion vectors using ffmpeg commands. The result is saved in the video 'BBB_motionVectors.mpeg'. To perform this task, we need to select option 1.

Task 2
In task 2, it takes the first 50 seconds of the original video and creates an .mp4 container. This container includes three different audio tracks (mp3 mono, mp3 stereo, AAC). To accomplish this, we create the three audio files and use ffmpeg to export the container to .mp4. The result of the new container is named 'BBB_container.mp4'.

Task 3
In task 3, we create another method in the class that, given an input .mp4 container, tells us how many video and audio tracks it contains. The results are displayed in the terminal.

Task 4 i 5
For these tasks, we create a new script called 'subtitle.py.' The objective is to create an output with the original video and integrated subtitles. To achieve this, we need to have yt-dlp installed. The function *download_video* allows us to download a YouTube video by entering its URL and saves it as .webm. The function *download_subtitles* downloads the subtitles of the chosen video and saves this file. The *integrate_subtitles* function takes the original video and, using ffmpeg, integrates the subtitles. This script inherits from the main script, and to test it, we need to choose option 4. To verify that it works, I have chosen a YouTube video of a song, but it can be tested for any YouTube video that has subtitles.

Task 6
In task 6, we extract the YUV histogram from the original video and create a new container in which the video track contains this histogram. To extract the histogram, we use ffmpeg, which takes the input video, calculates the pixel level histogram, resizes the histogram, overlays it on the original video, and generates a new output video with this overlay. We create a new script and inherit it from the main script.