Laia Duch Soler

Report SP3

L'objectiu d'aquest seminari és realitzar més exercicis de ffmpeg i realitzar un contenidor de Docker. El primer exercici consisteix a canviar la resolució d'un vídeo d'entrada creant un nou vídeo de sortida. Aquest nou vídeo, el volem codificar amb diferents codecs. El següent exercici consisteix a crear un vídeo de sortida de comparació, és a dir, que en una mateixa pantalla aparegui el mateix vídeo amb diferent codificació per veure la diferència. El tercer exercici consisteix a crear una interfície, per dur a terme-ho he utilitzat la llibreria tkinter. L'últim exercici consisteix a crear un contenidor Docker que contingui ffmpeg i que dugui a terme algun tipus de processament de vídeo.

Exercici 1

Aquest exercici el trobem en l'script anomenat *codecs.py*. En aquest script he creat una classe anomenada *VideoConverter* en la qual té una funció per canviar la resolució del vídeo i a més a més té funcions per codificar un vídeo d'entrada en diferents codecs.

En primer lloc, volem convertir el vídeo d'entrada BBB amb aquestes resolucions: 720p 480p 360x240 160x120. Per cada resolució es crea un vídeo de sortida anomenat "output_*resolution*.mp4". A continuació, volem codificar el vídeo d'entrada amb diferents codecs. Per cada tipus de codificació, es crea un video nou anomenat "output_*codec*", depenent del codec el vídeo l'extensió serà de tipus .mp4 o .webm.

Exercici 2

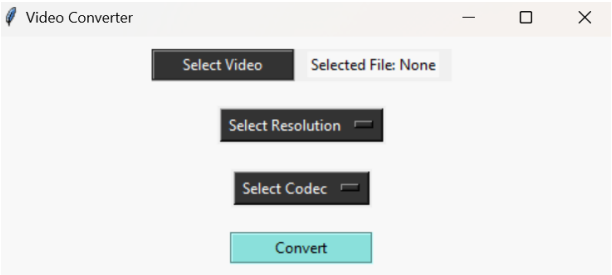
Aquest exercici he creat un nou script anomenat *comparison.py*. Per a aquest exercici s'utilitza els vídeos codificats anteriorment. Per realitzar la comparació fem servir els codecs VP8 i VP9. Es crea un nou vídeo de sortida anomenat "comparison_video.mp4".

El codificador VP9 és una versió més eficient i avançada que VP8, ja que ofereix una millor eficiència de compressió de manera que ens proporciona una qualitat de vídeo similar amb una taxa de bits més baixa. Tot i que la qualitat de vídeo dels dos codificadors és similar podem trobar alguna diferència i, per tant, podem veure com la qualitat del vídeo codificat en VP9 és millor que la del VP8. Veiem que en el vídeo codificat en VP9 té millor resolució i podem veure les imatges correctament, mentre que en el vídeo codificat en VP8 quan hi ha molt de moviment veiem la imatge una mica pixelada.

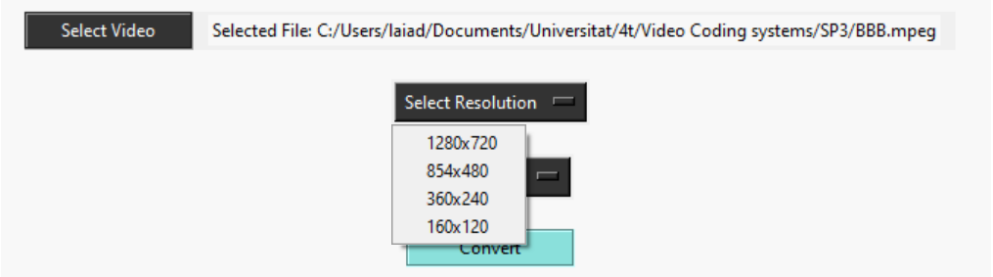
Exercici 3

Aquest exercici l'he realitzat en l'script *VideoConverter_GUI.py*. Aquest codi utilitza la biblioteca tkinter per crear una interfície gràfica d'usuari (GUI) que permet seleccionar un fitxer de vídeo, escollir opcions de resolució i codec, i després convertir el vídeo a un format especificat utilitzant ffmpeg. La interfície inclou botons per seleccionar el vídeo, especificar la resolució i el codec, i realitzar la conversió.

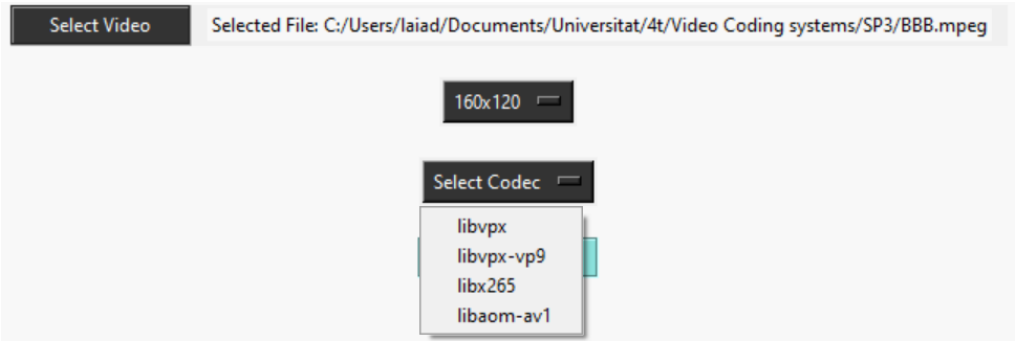
Al executar el script ens crea la següent interfície gràfica:



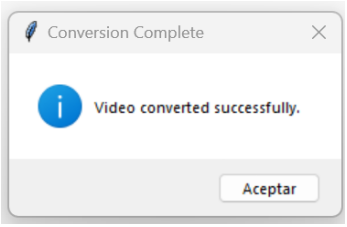
En la següent imatge veiem com l'usuari ha seleccionat un vídeo del seu dispositiu i en el botó *Select Resolution* observem les diferents opcions:



A continuació veiem els diferents tipus de codificacions que l'usuari pot escollir:



Un cop seleccionat tot, cliquem al botó *Convert* ens deixarà escollir on volem guardar el nou vídeo de sortida i amb quin nom el volem guardar. Un cop el vídeo ha estat convertir ens sortirà la següent finestra:



Exercici 4

Aquest exercici consisteix en explorar i descobrir els Dockers. Docker és una plataforma de virtualització de contenidors que permet als desenvolupadors empaquetar, distribuir i executar aplicacions amb totes les seves dependències en un entorn aïllat, conegut com a contenidor.

L'objectiu és crear un contenidor de Docker que contingui ffmpeg, en el meu cas tinc un script de python anomenat *convert_to_grayscale.py* en el qual converteix un vídeo d'entrada en escala de grisos. He volgut crear el contenidor ffmpeg amb aquest script. Aleshores, el fitxer Dockerfile s'utilitza per construir una imatge de Docker. En primer lloc, utilitza la imatge oficial de Python versió 3.8 com a base i instal·la l'eina ffmpeg. El directori de treball que s'estableix és en "/app", i copia l'script de python *convert_to_grayscale.py*. En aquest directori s'instal·len totes les biblioteques necessàries de Python i defineix la comanda que s'executarà quan s'inici el contenidor, en aquest cas és executar l'script utilitzant Python. Aquests dos arxius els trobem dins de la carpeta /SP3/docker_project.

Per veure el contenidor i la imatge cal utilitzar la següent comanda:

docker pull laiaduch/scav