Workshop-1

Group-31
SENECA POLYTECHNIC

Workshop-1

Declaration

We, Steven David Pillay, Olha Hodovaniuk, Ruyuan Sun, and Syed Moonis Iqbal, declare that the attached assignment is our own work in accordance with the Seneca Academic Policy. We have not copied any part of this assignment, manually or electronically, from any other source, including websites, unless specified as references. We have not distributed our work to other students.

Task Distribution

Name	Task(s)
Steven David Pillay	Setup Local Environment / Walkthrough for Anaconda Setup
Olha Hodovaniuk	Setup Local Environment / Setup Team Meeting
Ruyuan Sun	Setup Local Environment / Draft the Team Contract
Syed Moonis Iqbal	Setup Local Environment

Questions

Briefly explain what you do and don't understand in this code?

Steven David Pillay

This code simply does the following:

- Imports the necessary modules.
- Set up the video input file. (Video1.mpg)
- Sets up the video output file by retrieving the default frame width, height, frame rate and codec.
- Loop through all the frames and convert the frame from BGR to HSV color space and display it.
- Releases the resources and closes all the windows.

Olha Hodovaniuk

From this code I infer that it reads a video file, processes each frame, and then writes the processed frames to a new video file in order to test OpenCV. It starts by importing the libraries, playing the input video, and checking if I may open the file successfully. Then it retrieves the frame dimensions, defines the codec and creates a VideoWriter object for the output video. In a loop, it reads frame-by-frame, converts it to HSV colour space, writes the HSV frame to the output file, and displays both the original and HSV frames. I may exit from the loop by pressing ‘Q’. After that, the code releases resources and closes all OpenCV windows. I have a good understanding of what this code does; however, I had a few questions that comments in the code and my group helped me understand.

Syed Moonis Iqbal

The purpose of this code is to reads frames from a video, creates an output file, and writes to that file at the set framerate.

- The code imports the OpenCV library, opens the video file 'inputvideofilepath', and checks if the video opened successfully.
- It then gets the video's height and width, to get the video's resolution.
- A variable named 'frame_rate' is defined, which sets the framerate for the output video.
- A VideoWriter object is initialised using the 'DIVX' codec, output file path, resolution and frame_rate.
- The code then reads the input video, converts it from BGR to HSV colorspace, and then writes to 'TestVideo1.avi' at the set framerate.
- Afterwards, the code waits on the user to press q to to exit. The waitKey(25) waits for 25 milliseconds for the key to be pressed and quits the program once the key is pressed.
- Finally, it frees up the memory used by the video input and output objects.

Ruyuan Sun

What I Understand:

- **frame_width = int(cap.get(3))** and **frame_height = int(cap.get(4))** are used to get the video's width and height.
- **codec = cv2.VideoWriter_fourcc(*'DIVX')** specifies the codec used for encoding the video.
- **out = cv2.VideoWriter(outputvideofilepath, codec, frame_rate, resolution, True)** initializes the VideoWriter object with the output file path, codec, frame rate, resolution, and color flag.
- **ret, frame = cap.read()** reads a frame from the video, where **ret** indicates success and **frame** contains the image data.
- **hsvframe = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)** converts a frame from BGR to HSV color space.
- **if cv2.waitKey(25) & 0xFF == ord('q'):** checks if the 'q' key is pressed to break the loop.
- **cap.release()** and **out.release()** release the video capture and writer objects, respectively, to free up resources.

What I Don’t Understand:

- I am not entirely sure how the system determines the default video resolution and why it varies.
- The syntax ***'DIVX'** for unpacking the codec characters into the **VideoWriter_fourcc** function is unclear.
- The reason for using **& 0xFF** in the **cv2.waitKey** function is not clear.

Change the frame-rate parameter {5, 10, 20, 25, 30} and check both size and duration of the saved video. Can you briefly explain what is going on?

Steven David Pillay

Observation

The lower the frame rate, the larger the video file and the longer the duration of the video, and vice versa.

Explanation

When we lower the frame rate, the output appears as a slow-motion version of the original video, showing each frame for a longer duration. Conversely, increasing the frame rate results in a time-lapsed video.

Olha Hodovaniuk

After I changed the frame_rate parameter in the script to 5, 10, 20, 25, and 30 frames per second, it affected the size and duration of the saved video. When I changed frame_rate to 5, resulting in less frames per second, I made the video file smaller and longer in time for the same amount of footage. This is because fewer frames are stored and presented more slowly. On other hand, when I changed frame_rate to higher frame rates (to 30) produced larger video files and shorter durations, as more frames are recorded and displayed more quickly, I have increased the file size because of the higher number of frames per second. As a result, increasing the frame rate increases file size but decreases the playback duration, and vice versa.

Syed Moonis Iqbal

Changing the 'frame_rate' only significantly affects the duration of the video, and not the size. Since changing the 'frame_rate' parameter only alters how often the frame is written to a video (or how many frames the video has in 1 second), the duration becomes shorter or longer, depending on a higher or lower framerate. As the same number of frames are written to the file, the size remains the same.

Ruyuan Sun

Frame Rate: 5 fps Size: 9.2 MB Duration: 1:16	Frame Rate: 10 fps Size: 9.2 MB Duration: 0:38
Frame Rate: 20 fps Size: 9.2 MB Duration: 0:19	Frame Rate: 20 fps Size: 9.2 MB Duration: 0:19
Frame Rate: 25 fps Size: 9.2 MB Duration: 0:15	Frame Rate: 30 fps Size: 9.2 MB Duration: 0:12

Changing the frame rate affects the duration of the saved video. Higher frame rates result in shorter video durations because the same number of frames is displayed in a shorter amount of time, making the video play faster. The file size remains constant at 9.2 MB because the total number of frames and their resolution does not change, only the display rate of the frames changes.