Laboratory Exercise
OpenCV Videos

Submitted by:

Group 1

SO	PI	Category	Exceptional 4	Acceptable 3	Marginal 2	Unacceptable 1	Score
b	1	**Compliance** 30%	All procedures were followed, the output is as expected, additional related functionalities were augmented.	All procedures were followed and the output is as expected.	All procedures were followed but the output is not as expected.	Did not follow the set procedures	
b	1	**Analysis** 20%	Data interpretation is professionally written with appropriate and clear illustrations.	Data is clearly and correctly explained with proper illustrations.	Data is not clearly explained, has minor flaws, or no illustrations	There is no data explanation about the data or results.	
b	1	**Validity** 20%	The implementation uses the concepts and principles of the experiment as well as advanced topics.	The implementation uses the concepts and principles of the theory discussed for the experiment.	Implementation did not clearly express the use of theory discussed for the experiment.	There is no implementation.	
b	1	**Interpretation** 20%	The conclusion is professionally written and points the theories in the experiment and its implications in engineering.	The conclusion points to the main ideas and applications of the theory in the experiment.	The conclusion does not point out the main ideas and applications of the theory in the experiment.	There is no conclusion.	
		Format and Clarity 10%	Follows the prescribed format, observes proper and technical grammar, and observes proper citation and referencing according to IEEE journal standards.	Follows the prescribed format, observes proper and technical grammar, and observes proper IEEE citation and referencing.	Did not follow the prescribed format, has poor grammar, or incorrect citations and references scheme.	Did not follow the prescribed format, has poor grammar, and has no citations and references.	
		TOTAL SCORE					

Group Members			
STUDENT NUMBER	**NAME**	**CONTRIBUTION**	**SCORE**
201113488	CRUZ, SUZETTE L.		
201912207	LABAO, GIO VINCENT P.		
202020002	SALVADOR, DANIEL D.		
202020001	SALVADOR, GABRIEL D.		

Submitted to:
Engr. Dexter James L. Cuaresma

Date:
02/12/2024

OBJECTIVES

- To be create a simple code to read video, display video and save video.
- To create a simple python program to video using camera and display it in pychram using OpenCV.

DISCUSSION

Capture Video from Camera

Often, we have to capture live stream with a camera. OpenCV provides a very simple interface to do this. Let's capture a video from the camera (I am using the built-in webcam on my laptop), convert it into grayscale video and display it. Just a simple task to get started.

To capture a video, you need to create a VideoCapture object. Its argument can be either the device index or the name of a video file. A device index is just the number to specify which camera. Normally one camera will be connected (as in my case). So I simply pass 0 (or -1). You can select the second camera by passing 1 and so on. After that, you can capture frame-by-frame. But at the end, don't forget to release the capture. (OpenCV, 2020)

Playing Video form File

Playing video from file is the same as capturing it from camera, just change the camera index to a video file name. Also while displaying the frame, use appropriate time for cv.waitKey(). If it is too less, video will be very fast and if it is too high, video will be slow (Well, that is how you can display videos in slow motion). 25 milliseconds will be OK in normal cases. (OpenCV, 2020)

Saving a Video

So we capture a video and process it frame-by-frame, and we want to save that video. For images, it is very simple: just use cv.imwrite(). Here, a little more work is required.

This time we create a VideoWriter object. We should specify the output file name (eg: output.avi). Then we should specify the FourCC code (details in next paragraph). Then number of frames per second (fps) and frame size should be passed. And the last one is the isColor flag. If it is True, the encoder expect color frame, otherwise it works with grayscale frame. (OpenCV, 2020)

FourCC is a 4-byte code used to specify the video codec. The list of available codes can be found in fourcc.org. It is platform dependent. The following codecs work fine for me.

In Fedora: DIVX, XVID, MJPG, X264, WMV1, WMV2. (XVID is more preferable. MJPG results in high size video. X264 gives very small size video)

In Windows: DIVX (More to be tested and added)
In OSX: MJPG (.mp4), DIVX (.avi), X264 (.mkv).
FourCC code is passed as `cv.VideoWriter_fourcc('M','J','P','G')or cv.VideoWriter_fourcc(*'MJPG')` for MJPG. (OpenCV, 2020)

MATERIALS

Software:

- PyCharm
- OpenCv
- Anaconda
- Python

PROCEDURES

1. Run the given code below and write your observation

a. Capture Video From Camera

main.py

```python
import numpy as np
import cv2 as cv
cap = cv.VideoCapture(0)
if not cap.isOpened():
    print("Cannot open camera")
    exit()
while True:
    # Capture frame-by-frame
    ret, frame = cap.read()
    # if frame is read correctly ret is True
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break

    # Display the resulting frame
    cv.imshow('frame', frame)
    if cv.waitKey(1) == ord('q'):
        break
# When everything done, release the capture
cap.release()
cv.destroyAllWindows()
```

b. Playing Video from File

main.py

```python
import numpy as np
import cv2 as cv
cap = cv.VideoCapture('video_file_name.mp4/.avi/....etc')
while cap.isOpened():
    ret, frame = cap.read()
    # if frame is read correctly ret is True
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break
    cv.imshow('frame', frame)
    if cv.waitKey(1) == ord('q'):
        break
cap.release()
cv.destroyAllWindows()
```

Note: upload your video file in the directory folder

c. Saving a Video

Note: You can run your created video file by using the code in letter b to check your output.

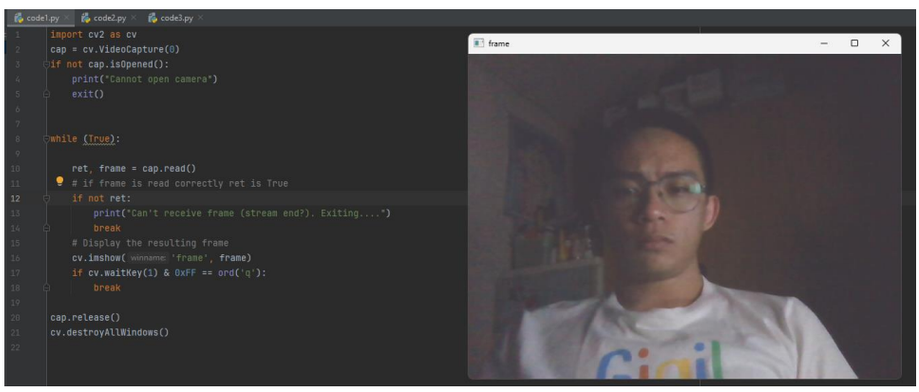
```python
import numpy as np
import cv2 as cv
cap = cv.VideoCapture(0)
# Define the codec and create VideoWriter object
fourcc = cv.VideoWriter_fourcc(*'MJPG')
out = cv.VideoWriter('output.mp4', fourcc, 20.0, (640,  480))
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        print("Can't receive frame (stream end?). Exiting ...")
        break

    cv.imshow('frame', frame)
    if cv.waitKey(1) == ord('q'):
        break
# Release everything if job is finished
cap.release()
out.release()
cv.destroyAllWindows()
```

2. Take a screen shot of your design and **describe each image**.
3. Save your works as Lab2_GroupNo. as a PDF file. And upload it to the submission file.

RESULTS AND DISCUSSION

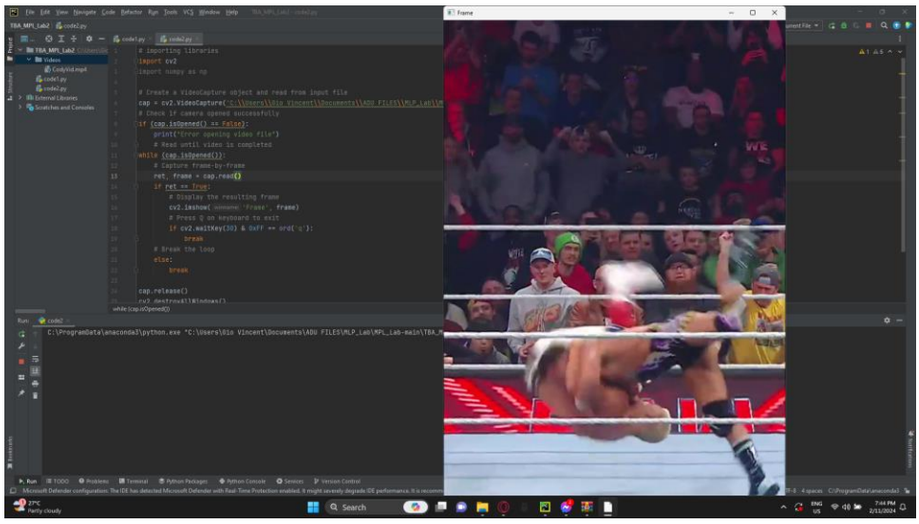
- OpenCV:

a. Example 1



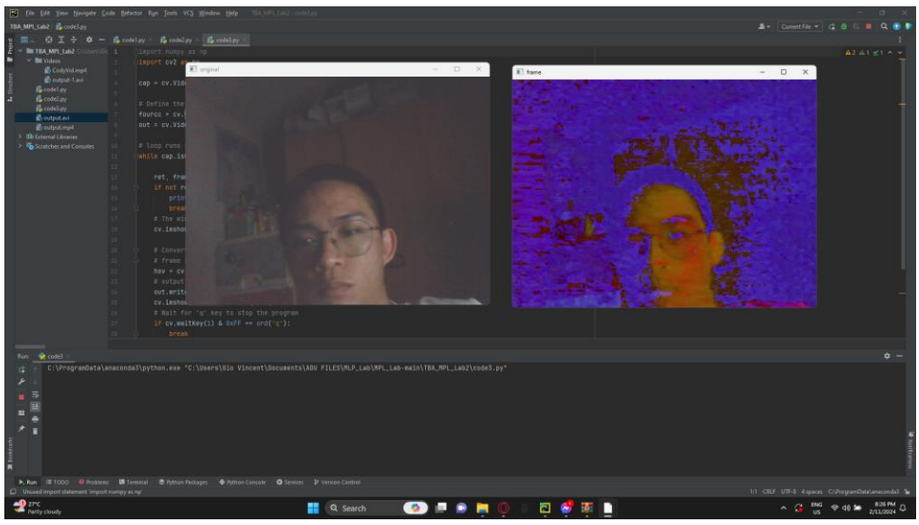
Analysis : For this example 1, we notice that the camera opens even if we did not trigger the camera button on the computer. But once the camera is covered it will detect and prompt an error “Cannot open camera.” We also found a code error when trying to exit the video. We found out on line 17 it needs to add “& 0xFF” before the “== ord(’q’)”.

b. Example 2



Analysis : For this example 2, we have experienced code error on the file path. As we tried to open a video from a file, it needed to have a double backslash to find the real path of the video file. Once we add the backslash, it recognizes and the video runs. The video displayed does not have audio and changing the frames will speed up the video if it is 25 or lower and slow down if 30 above.

c. Example 3



Analysis : For this example 3, the goal is to save a video file. Compare from the other examples it shows two different prompt screens, one is original and the other is frame. There is also difference from the output the original has clearer output while the frame has a blurry output. The group added lines of code to show two different output frames since the given code does not record any video. However, adding the HSV frame enables the recording albeit not as clear to the original video.

The link below will redirect to the group's Github repository for the laboratory experiment:

TBA_MLP_Lab2_Group1

CONCLUSION

In conclusion, this laboratory experiment to use OpenCV library for video and image has been a hands-on introduction to python libraries. We tried with videos, looked at frames and tried camera functionality without clicking any functions from the computer. Recording videos in HSV depict colors as perceived by the human eye. This can be used in AI recognition depending on how it can be applied. Using different color spaces can also be used. This practical experience equips us with valuable knowledge applicable in fields like image recognition and video surveillance. This laboratory will guide one step ahead to head start future projects and advancement in video processing through machine learning.