CS 1002 – PROGRAMMING IN PYTHON

MINI PROJECT – Write up, Code & Output

TOPIC :

Motion detection and its uses in home automation

SUMMARY / ABSTRACT:

This is a Python code that uses the OpenCV library to capture live video from a webcam and detect motion in it. It then records the start and end times of the motion and stores them in a Pandas DataFrame. Finally, it saves the recorded times in a CSV file and closes all open windows.

The code works by initializing a static background frame and then continuously capturing new frames fromthe video feed. It compares each new frame with the static background frame to identify any changes (i.e., motion). If motion is detected, the code captures the current time and records it as the start or end time of the motion.

The code then displays several images on the screen to visualize the captured frames, including the gray scale image, the difference between the current frame and the static background frame, a black and white image with the moving object highlighted, and a color image with a green rectangle drawn around the moving object. The code uses a while loop to keep capturing frames until the user presses the ‘m’ key.

At the end, it stores the recorded times in a Pandas DataFrame and saves it to aCSV file. Finally, it releases the video feed and closes all open windows.

TEAM MEMBERS

- A . Evans William
- Achyuth Mukund
- Haneef Ahmad
- Medhini Bhat
- Harshini. S
- Divya. M

IMPORTED LIBRARIES

Import cv2 : imports the OpenCV (cv2) library for computer vision tasks.

Import pandas as pnd: imports the Pandas library and gives it an alias “pnd”. Pandas is a popular library for data manipulation and analysis.

Import time: imports the time module which provides various time-related functions.

From datetime import datetime : Imports the datetime class from the datetime module. The datetime class provides methods to work with dates and times.

IMPORTED MODULES

Cv2 – This is the OpenCV library for computer vision tasks.

Pandas – This is the Pandas library for data manipulation and analysis.

Time – This is the Python time module which provides various time-related functions.

Datetime – This is the Python datetime module which provides classes for working with dates and times.

CODE :

```
# Importing the OpenCV libraries
import cv2
# Importing the Pandas libraries
import pandas as pnd
# Importing the time module
import time
# Importing the datetime function of the datetime module
from datetime import datetime
# Assigning our static back position as None for initial frames
staticBack = None
```

```
# List of the tracks when any motion is detected in
the frame

motionList = [ None, None ]

# A new list for capturing the time when movement
detected

time = []

# Initializing DataFrame using pandas with Initial and
Final column

dFrame = pnd.DataFrame(columns = ["Initial",
"Final"])

# Capturing Video from our system's Webcam

mainVideo = cv2.VideoCapture(0)

# Using an infinite while loop to capture images as a
video

while True:

    # Reading each frame or image from the video

    check, frame = mainVideo.read()

    # Initializing motion as Statics frame

    motion = 0

    # Creating a gray frame from colour images

    grayImage = cv2.cvtColor(frame,
cv2.COLOR_BGR2GRAY)

    # Creating a GaussianBlur from the gray scale
image to find changes
```

```
    grayFrame = cv2.GaussianBlur(grayImage, (21, 21),
0)
    # In first iteration, we make a gray frame from
initial static frame
    if staticBack is None:
        staticBack = grayFrame
        continue
    # Calculation of difference between static and
gray frame we created
    differFrame = cv2.absdiff(staticBack, grayFrame)
    # Highlighting the change between static
background and current gray frame
    threshFrame = cv2.threshold(differFrame, 30, 255,
cv2.THRESH_BINARY)[1]
    threshFrame = cv2.dilate(threshFrame, None,
iterations = 2)
    # Finding contour from the moving object in the
frame
    contis,_ = cv2.findContours(threshFrame.copy(),
                    cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    for contr in contis:
        if cv2.contourArea(contr) < 10000:
            continue
        motion = 1
```

```
        (x, y, w, h) = cv2.boundingRect(contr)
        # Creating a green rectangle around the moving
object
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255,
0), 3)
    # Adding the motion status from frame
     motionList.append(motion)
     motionList = motionList[-2:]
     # Adding the motion Start time
     if motionList[-1] == 1 and motionList[-2] == 0:
        time.append(datetime.now())
    # Adding the motion End time
     if motionList[-1] == 0 and motionList[-2] == 1:
        time.append(datetime.now())
    # Displaying captured image in the gray scale
     cv2.imshow("This is the image captured in the
Gray Frame", grayFrame)
     # Displaying the difference between current frame
and the initial static frame
     cv2.imshow("Difference between the two
frames", differFrame)
     # Displaying the black and white images from the
video on the frame screen
     cv2.imshow("This is a Threshold Frame created
```

```
from the system's Webcam", threshFrame)
   # Displaying contour of the object through the
color frame
   cv2.imshow("This is one example of the Color
Frame from the system's webcam", frame)
   # Creating a key to wait
   key = cv2.waitKey(1)
   # Ending the whole process with the 'm' key of our
system
   if key == ord('m'):
      # Appending time when something is moving on
the screen
      if motion == 1:
         time.append(datetime.now())
      break
# Adding the time of motion inside the data frame
for a in range(0, len(time), 2):
   dFrame = dFrame.append({"Initial" : time[a],
"Final" : time[a + 1]}, ignore_index = True)
# Creating a CSV file where all movements will be
recorded
dFrame.to_csv("MovementsTimeFile.csv")


# Releasing the video
```

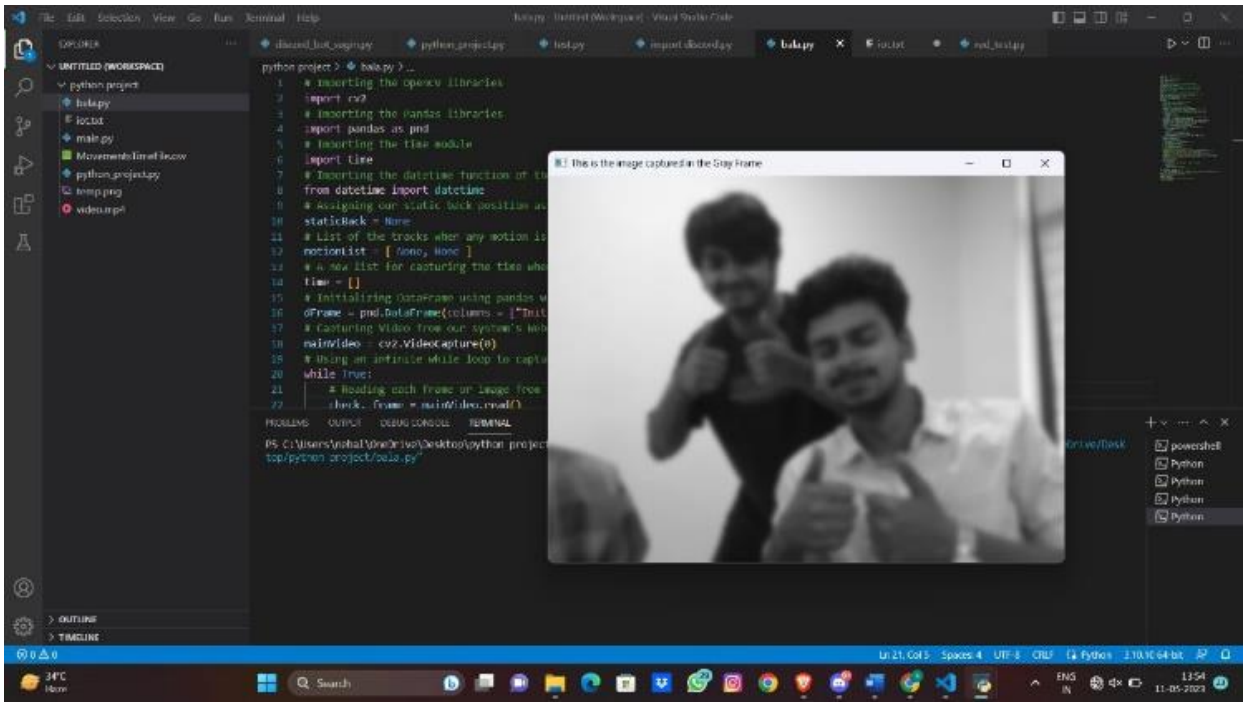
mainVideo.release()

 # Lastly, Destroying all the open windows with openCV

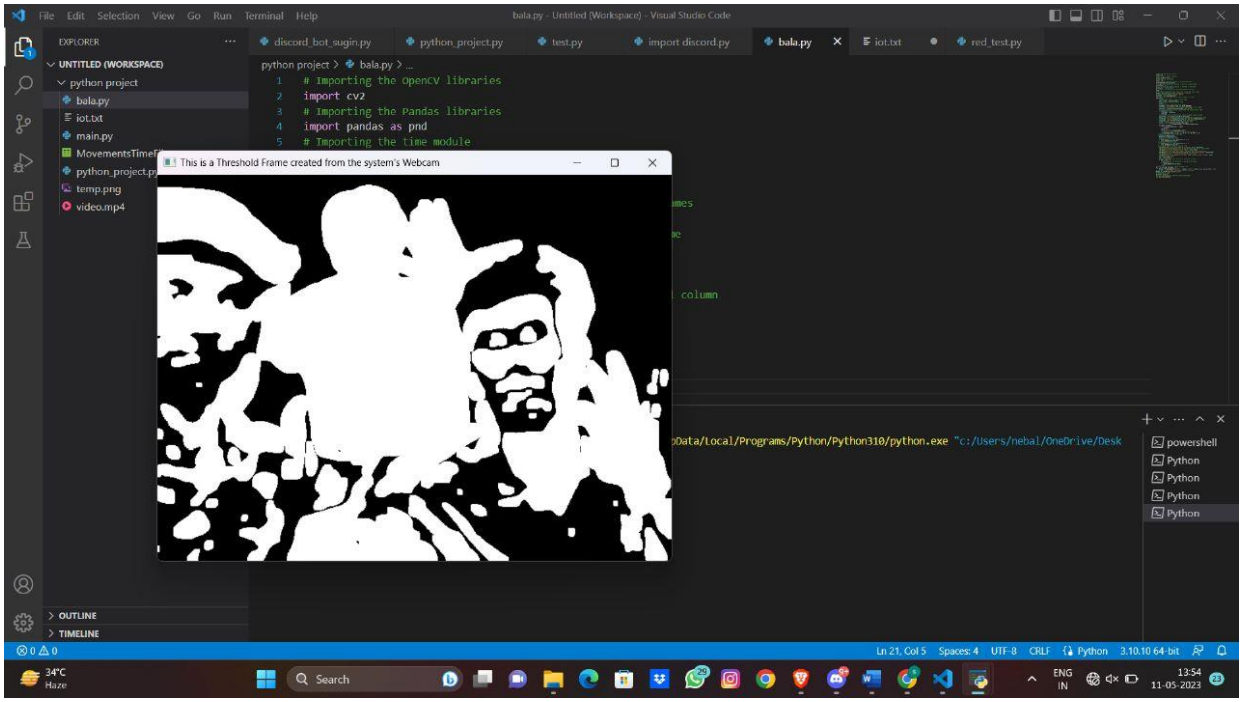
cv2.destroyAllWindows()

OUTPUT :

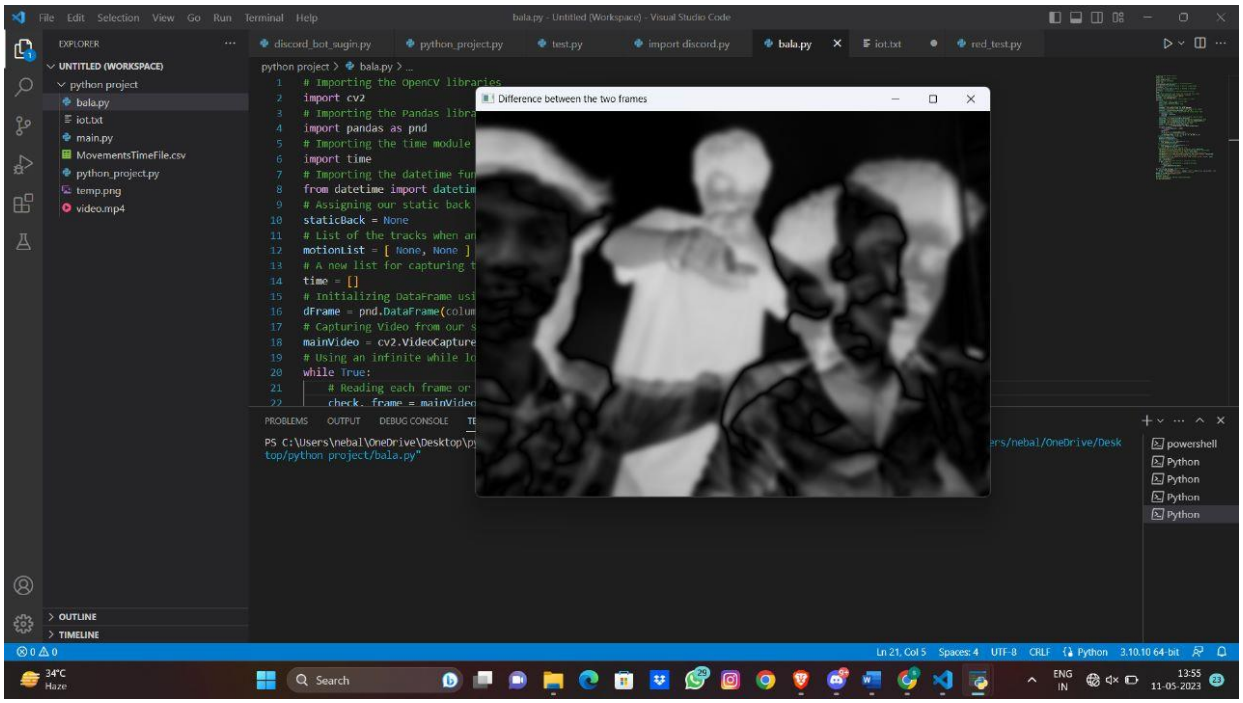
1. **Image captured in gray frame :**



2.Example of a threshold frame created from a system's webcam :



3. Difference between the two frames :



4. One example of a colour frame from the system's webcam :