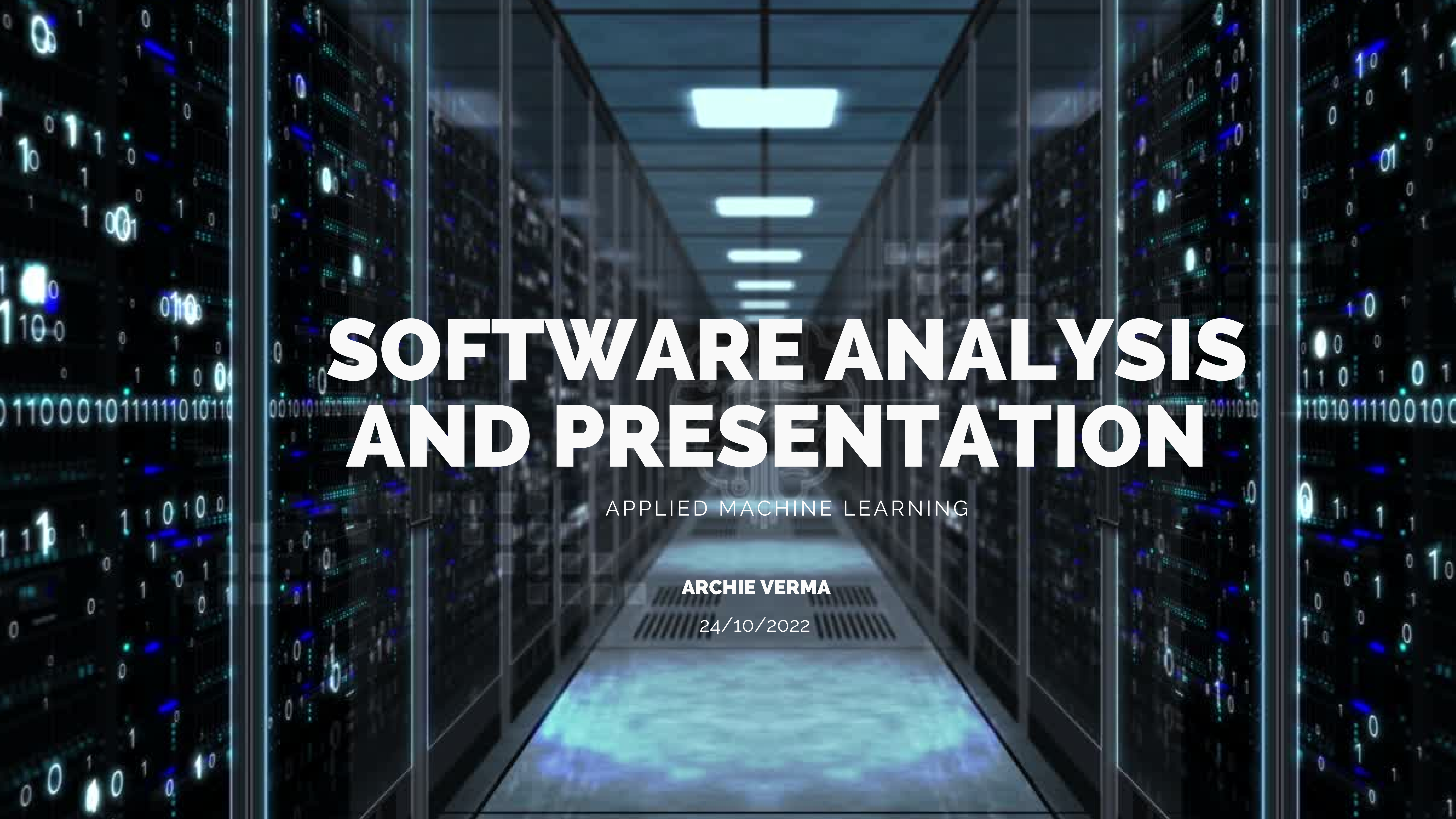


SOFTWARE ANALYSIS AND PRESENTATION

APPLIED MACHINE LEARNING

ARCHIE VERMA

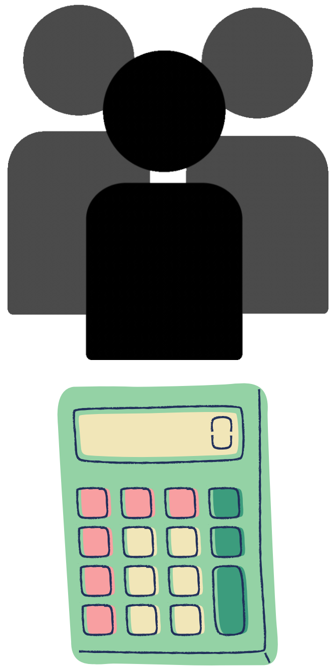
24/10/2022

Table of Contents

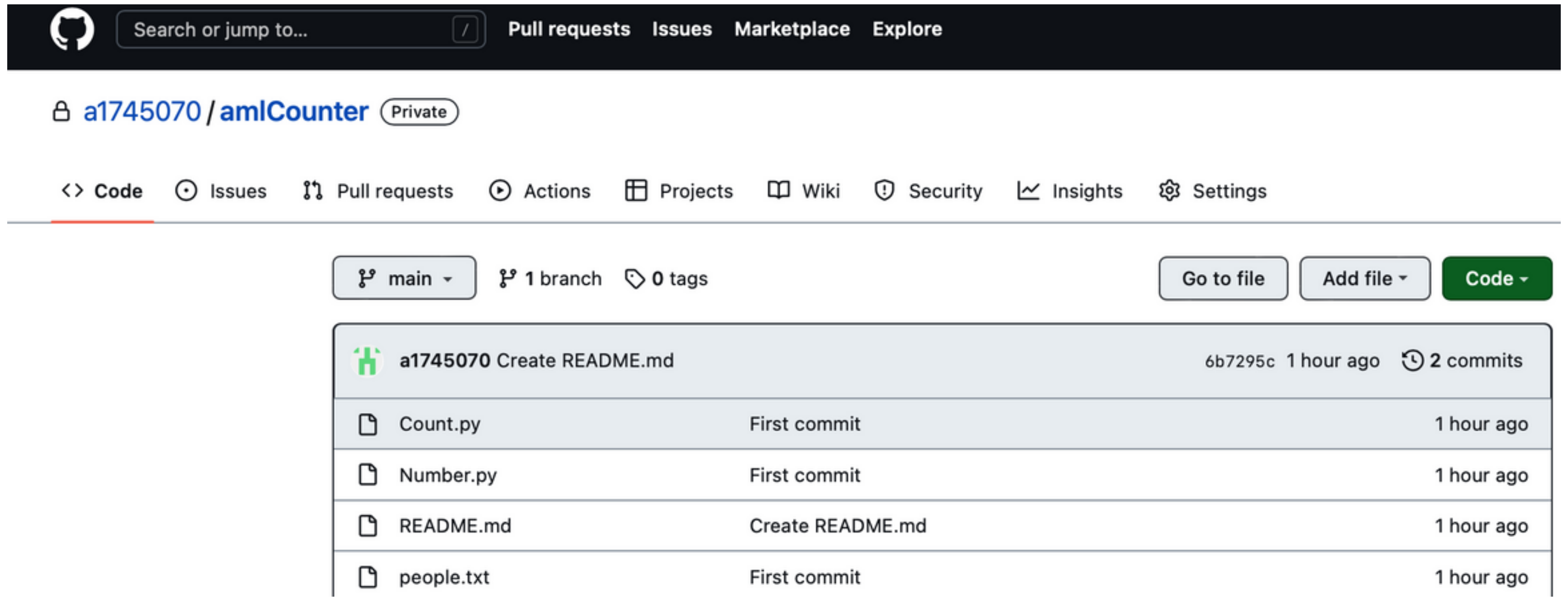
INTRODUCTION

IDEA RECAP:

An application to count the number of people entering and exiting an area over a given period from a video using machine learning and computer vision, which can solve the problem of doing architectural audits manually. As well as be effective to show if people are being cautious and taking necessary covid precautions.



This application was coded in python and the code was stored on on GitHub.

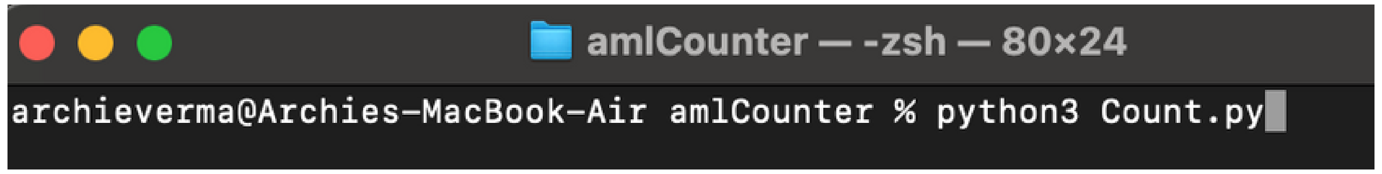


a1745070 / amlCounter Private

1 branch 0 tags

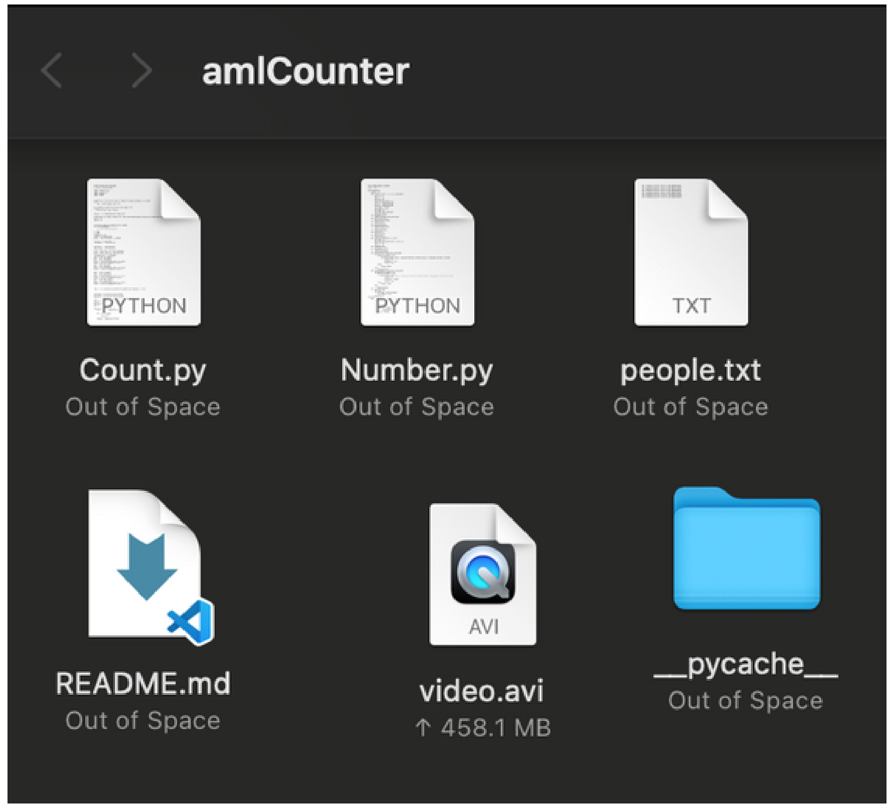
a1745070 Create README.md 6b7295c 1 hour ago 2 commits

Count.py	First commit	1 hour ago
Number.py	First commit	1 hour ago
README.md	Create README.md	1 hour ago
people.txt	First commit	1 hour ago

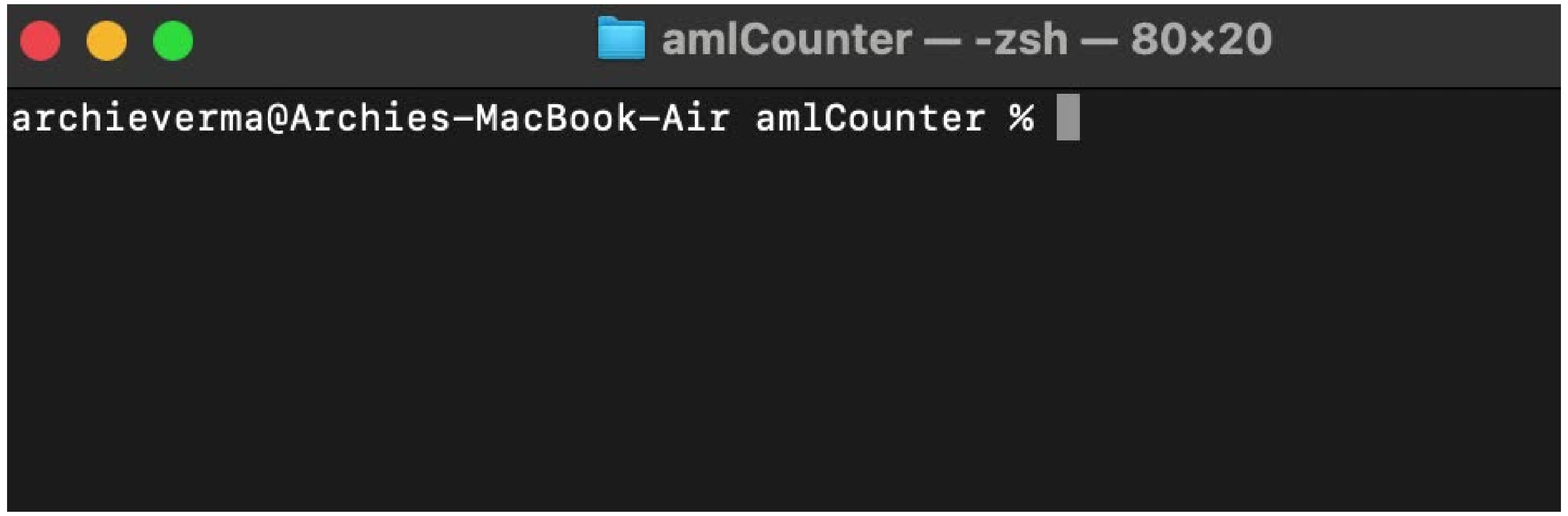


amlCounter — -zsh — 80×24

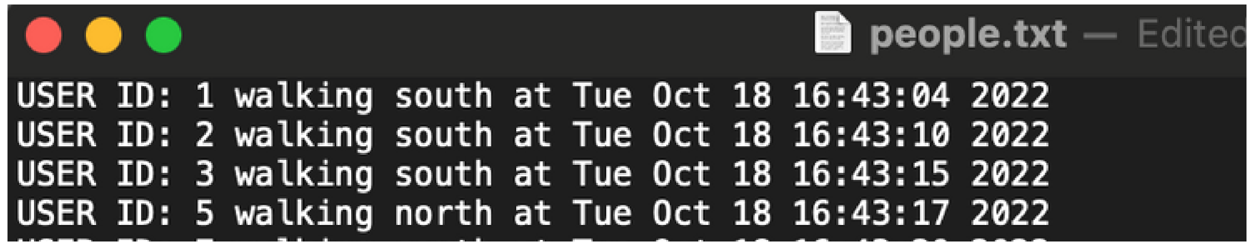
```
archieverma@Archies-MacBook-Air amlCounter % python3 Count.py
```



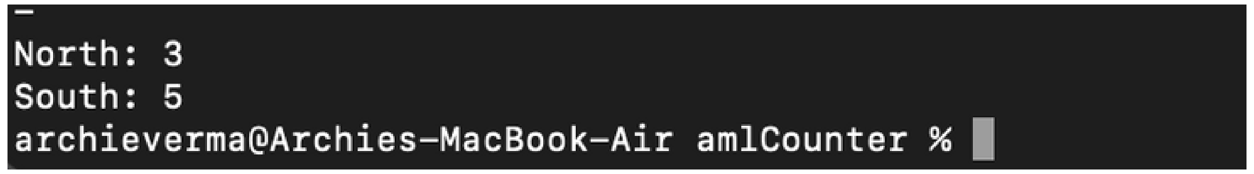
APPLICATION RESULTS



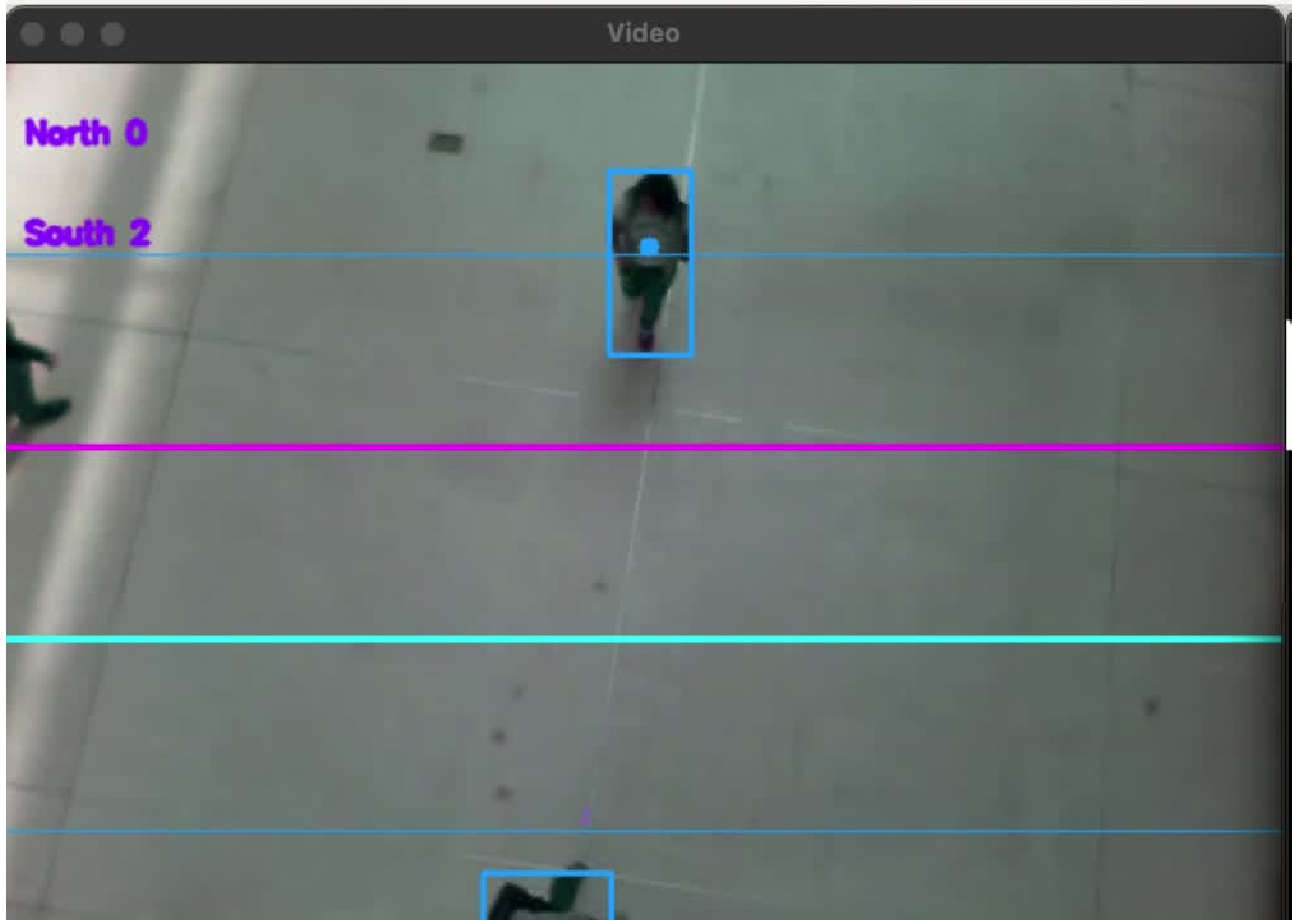
```
archieverma@Archies-MacBook-Air amlCounter % 
```

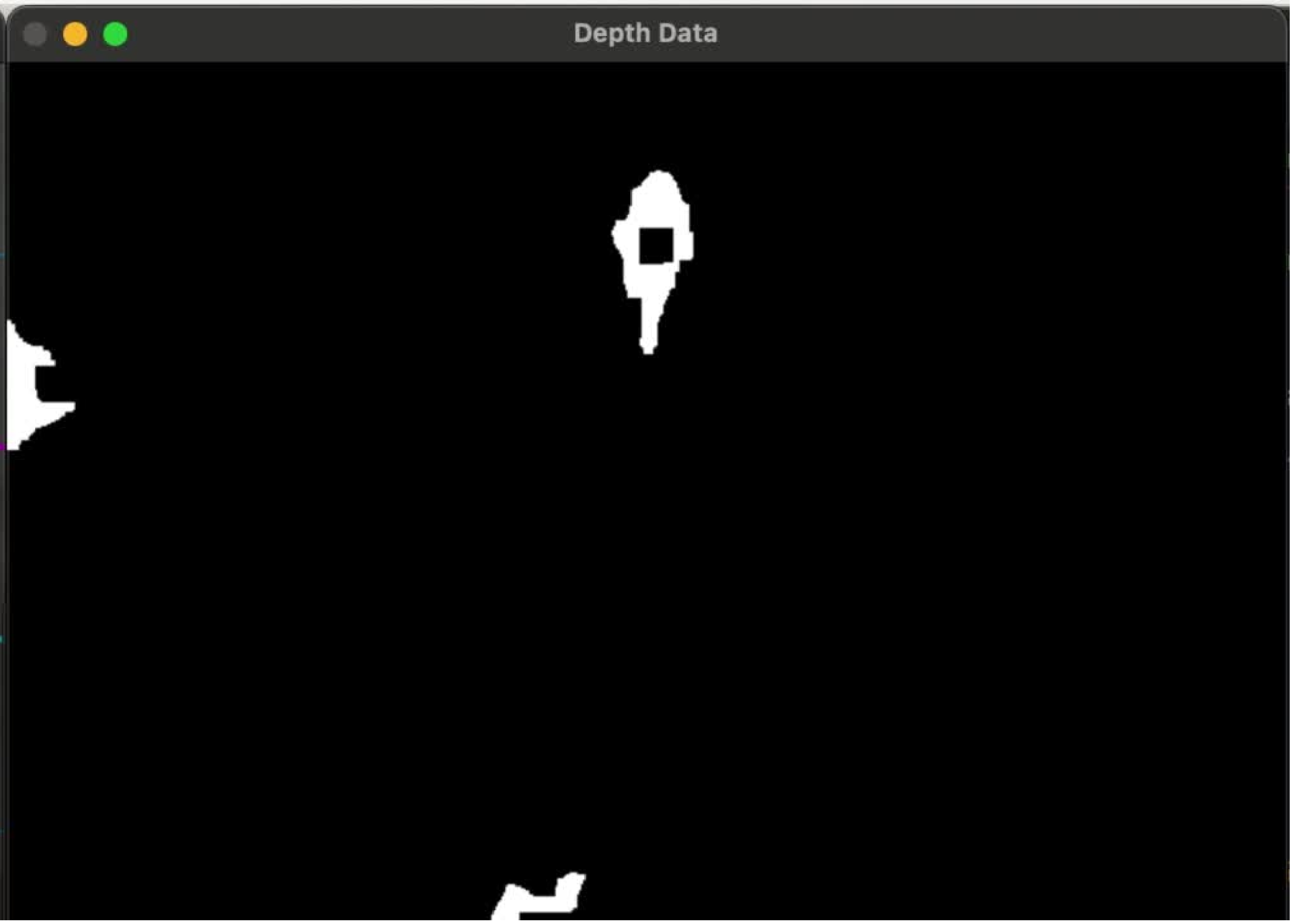


```
USER ID: 1 walking south at Tue Oct 18 16:43:04 2022
USER ID: 2 walking south at Tue Oct 18 16:43:10 2022
USER ID: 3 walking south at Tue Oct 18 16:43:15 2022
USER ID: 5 walking north at Tue Oct 18 16:43:17 2022
```

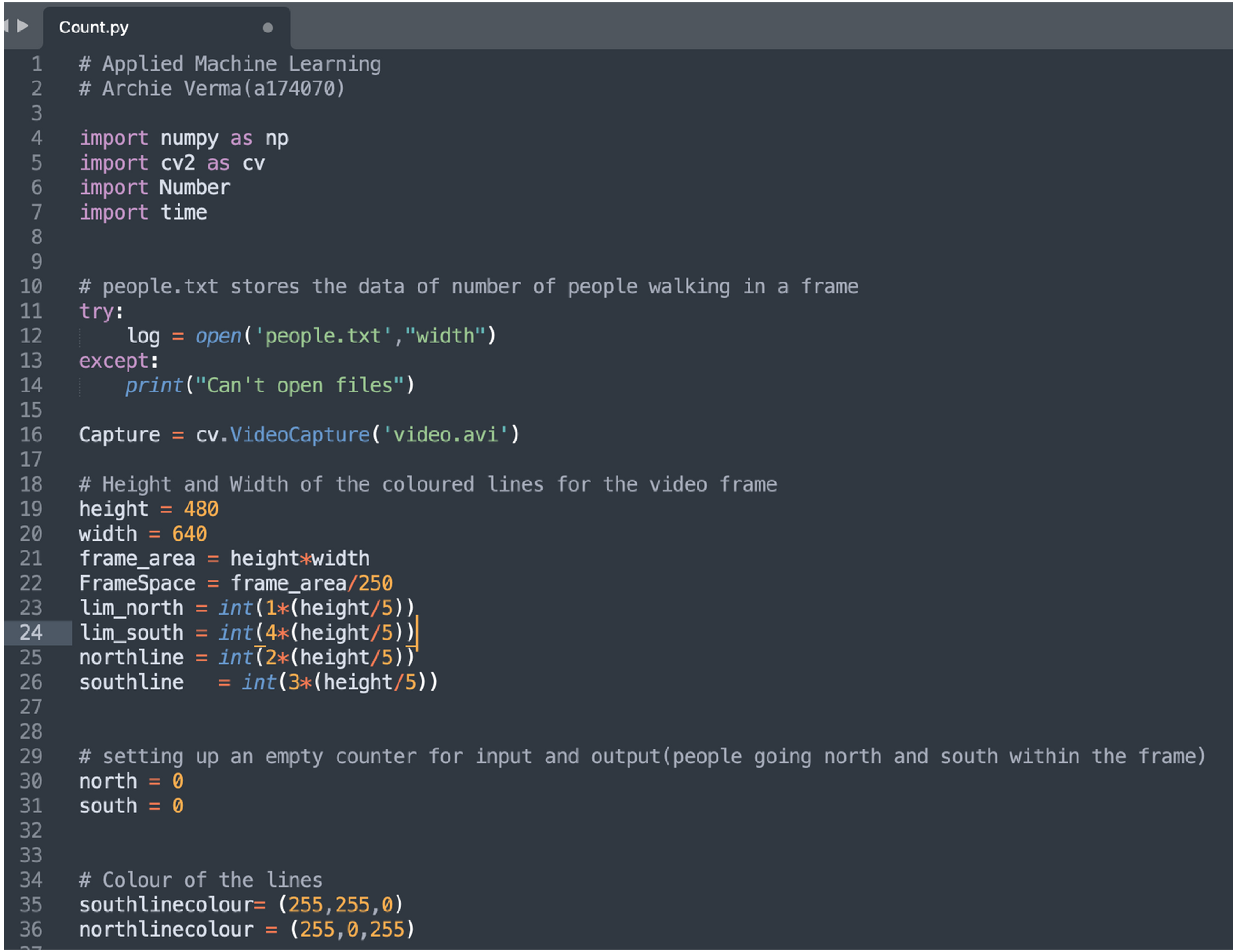


```
North: 3
South: 5
archieverma@Archies-MacBook-Air amlCounter % 
```

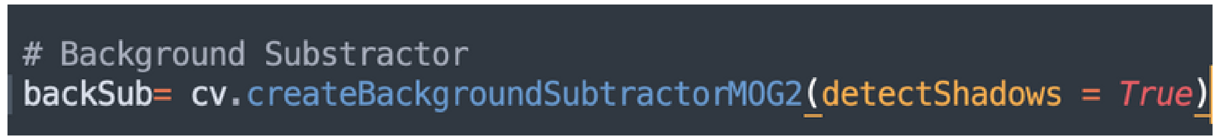




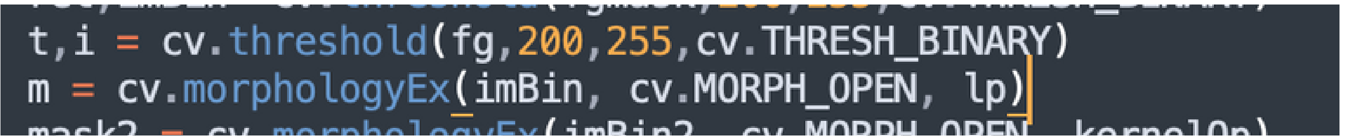
CODE HIGHLIGHTS



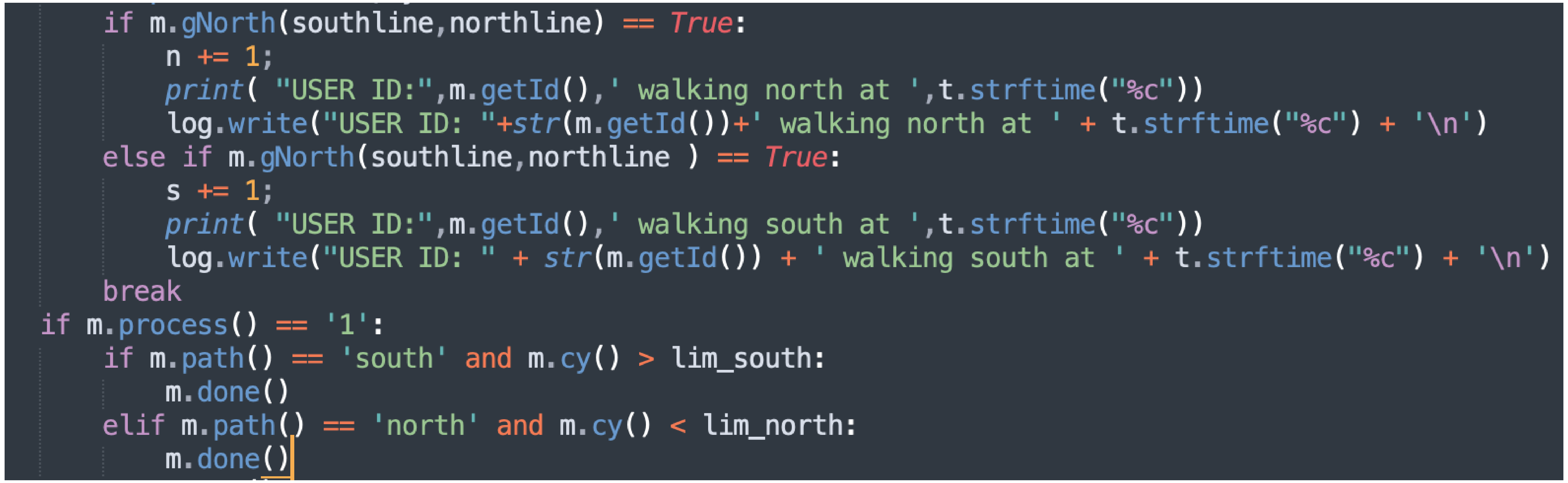
```python
# Applied Machine Learning
# Archie Verma(a174070)

import numpy as np
import cv2 as cv
import Number
import time


# people.txt stores the data of number of people walking in a frame
try:
    log = open('people.txt',"width")
except:
    print("Can't open files")

Capture = cv.VideoCapture('video.avi')

# Height and Width of the coloured lines for the video frame
height = 480
width = 640
frame_area = height*width
FrameSpace = frame_area/250
lim_north = int(1*(height/5))
lim_south = int(4*(height/5))
northline = int(2*(height/5))
southline    = int(3*(height/5))


# setting up an empty counter for input and output(people going north and south within the frame)
north = 0
south = 0


# Colour of the lines
southlinecolour= (255,255,0)
northlinecolour = (255,0,255)
```



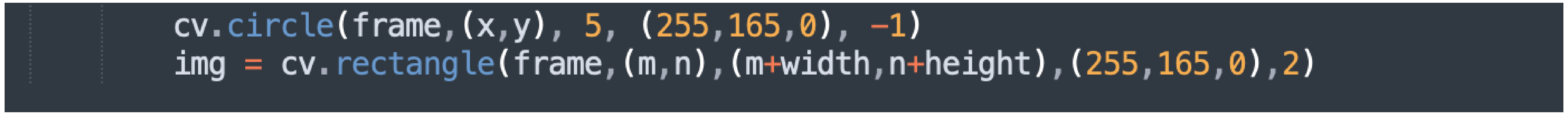
```
# Background Substractor
backSub= cv.createBackgroundSubtractorMOG2(detectShadows = True)
```



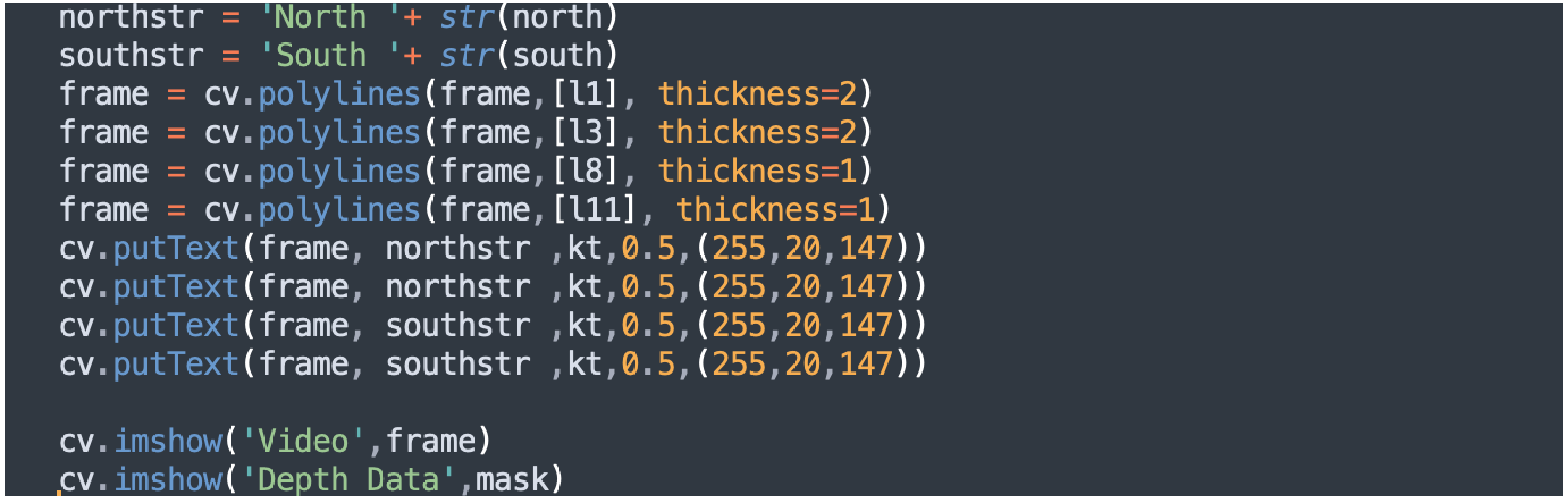
```
t,i = cv.threshold(fg,200,255,cv.THRESH_BINARY)
m = cv.morphologyEx(imBin, cv.MORPH_OPEN, lp)
mask2 = cv.morphologyEx(imBin2, cv.MORPH_OPEN, kernelOp)
```



```
        if m.gNorth(southline,northline) == True:
            n += 1;
            print( "USER ID:",m.getId(),' walking north at ',t.strftime("%c"))
            log.write("USER ID: "+str(m.getId())+' walking north at ' + t.strftime("%c") + '\n')
        else if m.gNorth(southline,northline ) == True:
            s += 1;
            print( "USER ID:",m.getId(),' walking south at ',t.strftime("%c"))
            log.write("USER ID: " + str(m.getId()) + ' walking south at ' + t.strftime("%c") + '\n')
        break
    if m.process() == '1':
        if m.path() == 'south' and m.cy() > lim_south:
            m.done()
        elif m.path() == 'north' and m.cy() < lim_north:
            m.done()
```

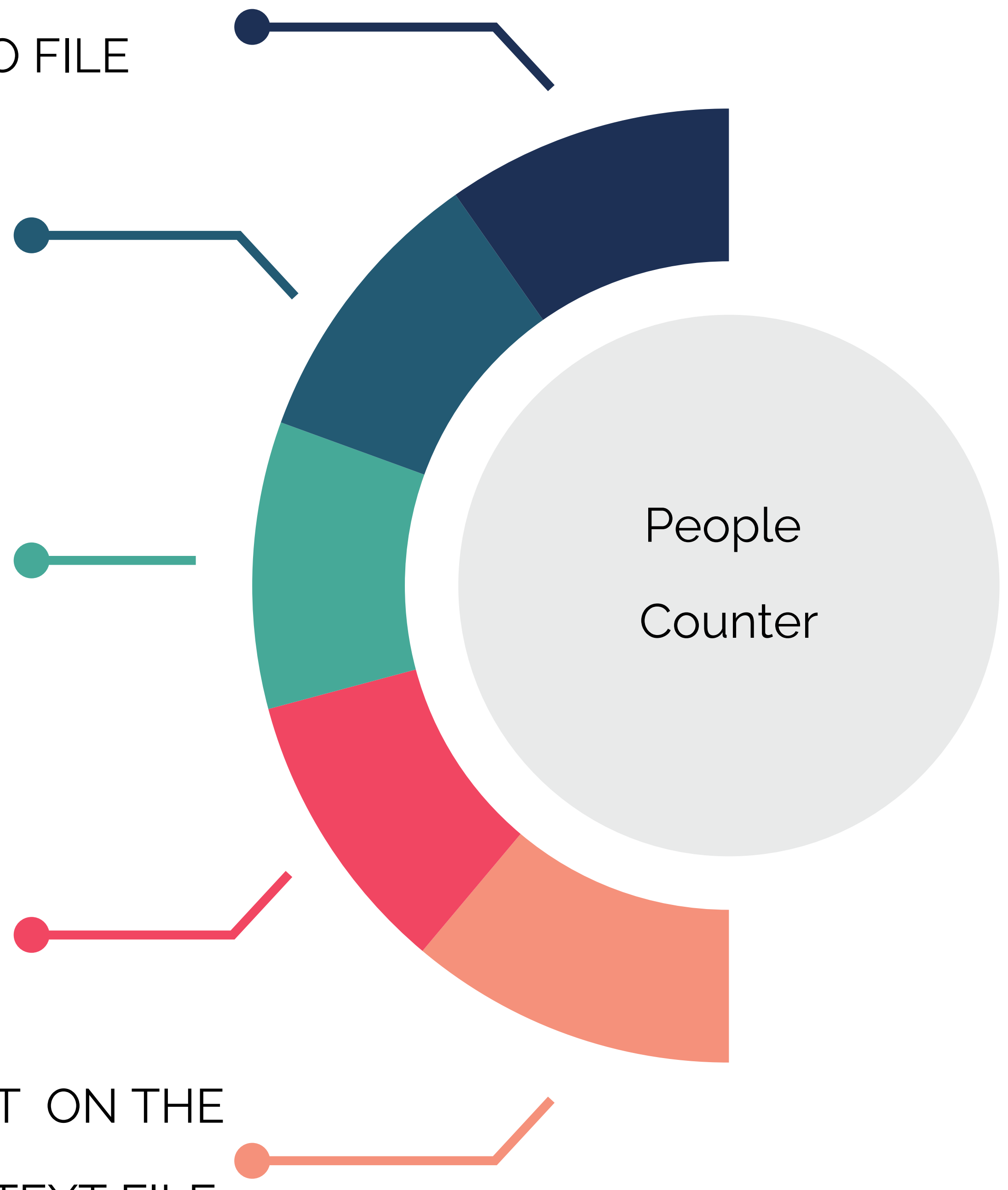


```
            cv.circle(frame,(x,y), 5, (255,165,0), -1)
            img = cv.rectangle(frame,(m,n),(m+width,n+height),(255,165,0),2)
```



```
    northstr = 'North '+ str(north)
    southstr = 'South '+ str(south)
    frame = cv.polylines(frame,[l1], thickness=2)
    frame = cv.polylines(frame,[l3], thickness=2)
    frame = cv.polylines(frame,[l8], thickness=1)
    frame = cv.polylines(frame,[l11], thickness=1)
    cv.putText(frame, northstr ,kt,0.5,(255,20,147))
    cv.putText(frame, northstr ,kt,0.5,(255,20,147))
    cv.putText(frame, southstr ,kt,0.5,(255,20,147))
    cv.putText(frame, southstr ,kt,0.5,(255,20,147))

    cv.imshow('Video',frame)
    cv.imshow('Depth Data',mask)
```

CODING PROCESS



CAPTURING AND READING THE VIDEO FILE

MEASURING THE VIDEO FRAME AND CREATING DEPTH DATA

MAKING BLUE BOX FRAMES ON MOVING INDIVIDUALS

COUNTING PEOPLE FROM DEPTH DATA

PRINTING RESULT ON THE VIDEO AND IN A TEXT FILE

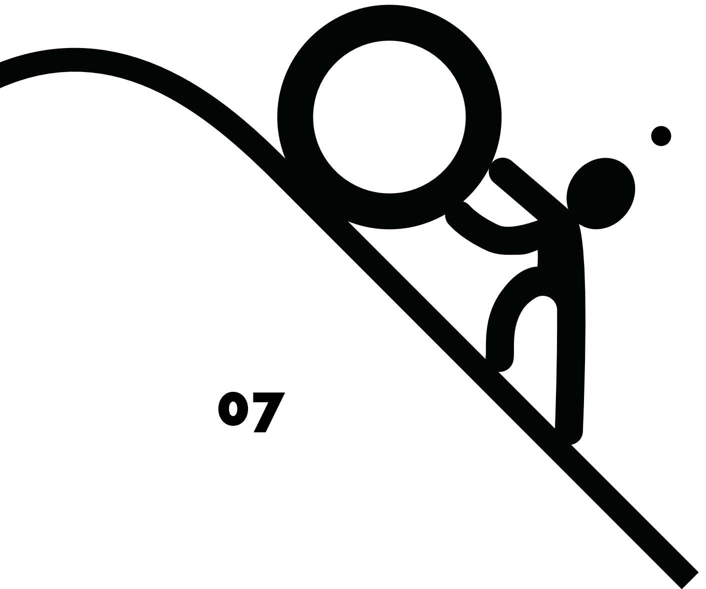
People Counter

CHALLENGES AND PROBLEM SOLVING APPROACHES

- Installation of cv2 and numPy on my laptop took a long time as I got a wrong version installed and it gave me numerous errors while using capture to read the test video but I was able to figure it out through stack overflow

- Studying different open cv functions and their use for detecting individuals in the video- this was done through research online and on GitHub about over cv and similar application ,reading some research papers which have used similar methods to build a counting application

- Implementing the frame detection process using a test avi video in python, this was done through research and code analysis through GitHub.

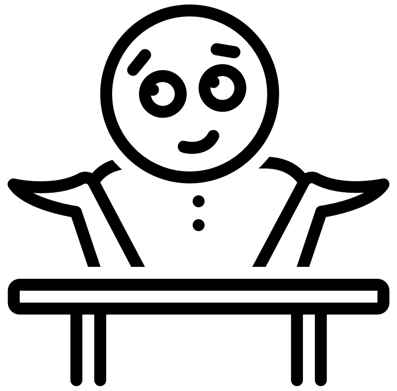


MACHINE LEARNING ELEMENTS

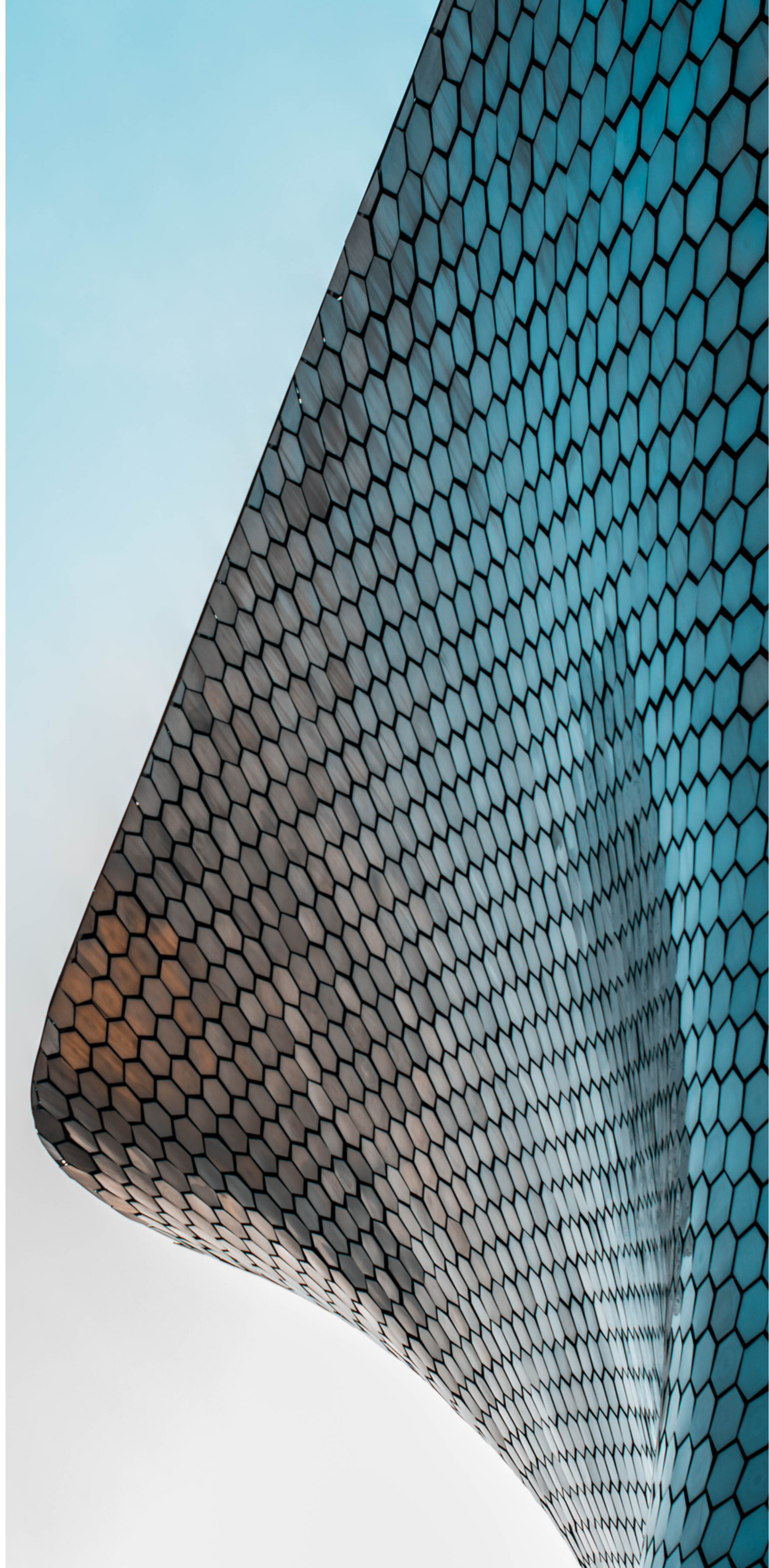
- Use of OpenCV and NumPy
- The data is generated by processing and converting the test video into depth data as was seen in the results.
- That data is getting evaluated through various open CV functions as were shown in the code like createBackgroungSubstractor for generating depth data, polylines used to draw a polygon, thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze data, putText to draw a text string for number on the video and imshow to display an image in a window, circle and rectangle to point people and draw rectangles on top of them.

PROJECT GOAL

- The project goal was achieved as the basic application components were implemented which was counting number of people in any video frame and saving the results.
- This goals wan't achieved completely as the application doesn't give that accurate results for videos with too many people present as the depth data generated isn't very clear and the group becomes like a blob.
- The data does not show any complex graphs for clear results as currently it is being performed for a video and is not in terms of a big class room setting.
- It is not detecting people wearing masks and distance between them to show covid safety yet.



FUTURE PLAN



1

Implementing and testing the application for a video with a crowd aiming to get accurate results.

2

Implementing the application for classrooms for architectural audits with personalised graphs and clear reading of number of people present per hour

3

Implementing the application to have additional covid features to show if people are social distancing, wearing masks etc .

REFERENCES

People counting systems with opencv, python, and ubidots (no date) Ubidots Help Center. Available at: https://help.ubidots.com/en/articles/1674356-people-counting-systems-with-opencv-python-and-ubidots [Accessed: October 21, 2022].

Meel, V., 2022. YOLOv3: Real-Time Object Detection Algorithm (Guide) - viso.ai. [online] viso.ai. Available at: <https://viso.ai/deep-learning/yolov3-overview/> [Accessed 21 October 2022].

GitHub. Available at: https://github.com/[Accessed 21 October 2022].

E. Bondi, L. Seidenari, A. D. Bagdanov and A. Del Bimbo, "Real- time people counting from depth imagery of crowded environments," 2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2014, pp. 337-342, doi: 10.1109/AVSS.2014.6918691.

Shakhadri, S., 2022. People Counting and Tracking Project using Deep Learning. [online] Analytics Vidhya. Available at: <https://www.analyticsvidhya.com/blog/2021/11/complete-guide-to-people- counting-and-tracking-end-to- end-deep-learning-project/> [Accessed 11 September 2022].

Thank You