REPORT

Submitted by:
Anish Taneja
Ashutosh Bajpai

Registration No and Roll No:
11706358, 30
11704819, 29

ASSIGNMENT

TOPIC: HEAD COUNTING APPLICATIONS (PEOPLE COUNTER)

Submitted To:

Sanjay Kumar Sir

School of Computer Science & Engineering Lovely Professional University, Phagwara

INTRODUCTION

In recent times, security has become one of the utmost to our everyday life. To properly manage security it is of great importance to introduce video surveillance. Video surveillance helps in reducing and preventing theft, real time monitoring to observe the behaviour or action of people, increase productivity, and provide evidence during investigation. Video surveillance can be used to count the number of people entering or leaving a building. The tracking and counting of people is a field that has gained a lot of attention in the last few years due to the advancement of image processing and computer technology. By counting the information can be used to identify traffic patterns hourly, monitor events and to optimize labour in shopping malls and markets. The camera makes the entire process automated and helps to identify threats in different areas, makes it easier to evacuate people as well as to know areas that require more attention during an emergency. According to people, counting and conversion rate should be used to measure the performance of stores on an on-going basis to frequently identify areas for improvement. Automatic counting of people, entering or exiting a region of interest, is very important for both business and security applications. To properly manage security in various places, it is of great importance to introduce video surveillance. This paper introduces an automatic people counting system which can count multiple people who interact in the region of interest, by using only one camera.

PYTHON LIBRARIES USED IN PROJECT

- **NUMPY** : NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

❖ A powerful N-dimensional array object
❖ Sophisticated (broadcasting) functions
❖ Tools for integrating C/C++ and Fortran code
❖ Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

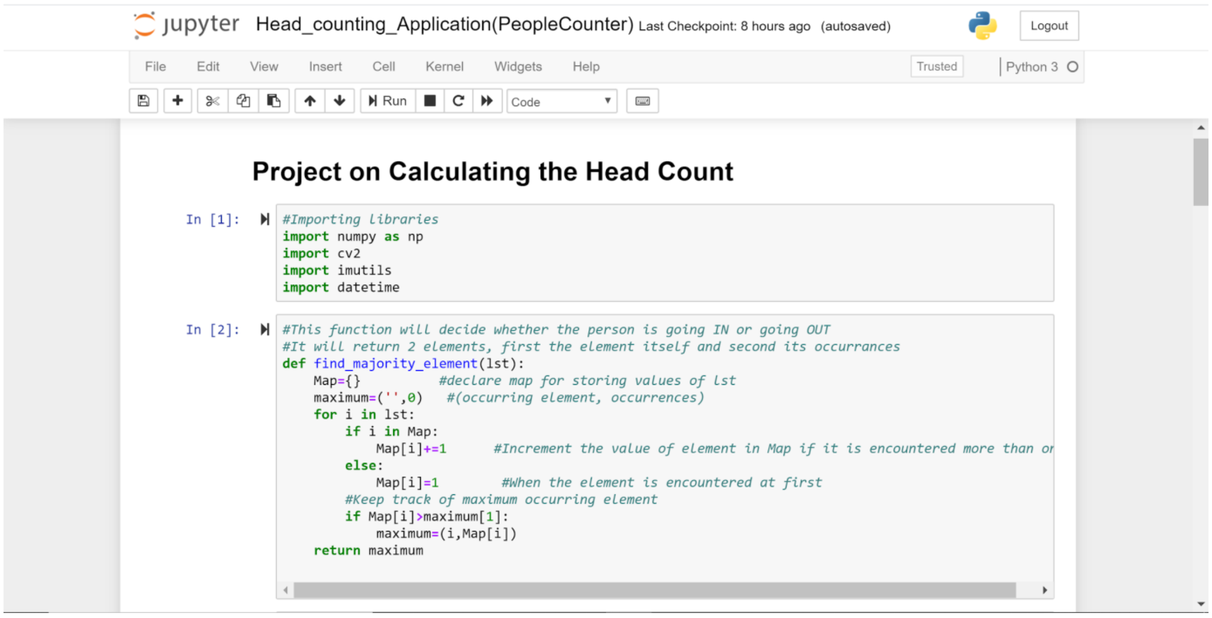
- **OPEN CV :** Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases the number of weapons in your arsenal. Besides that, several other libraries like SciPy, Matplotlib which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

- **IMUTILS** : A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much more easier with OpenCV and both Python 2.7 and Python 3.

- **DATETIME** : In Python, date and time are not a data type of its own, but a module named datetime can be imported to work with the date as well as time. The Datetime module comes built into Python, so there is no need to install it externally.

  Datetime module supplies classes to work with date and time. These classes provide a number of functions to deal with dates, times and time intervals. Date and datetime are an object in Python, so when you manipulate them, you are actually manipulating objects and not string or timestamps.

CODE SNIPPETS

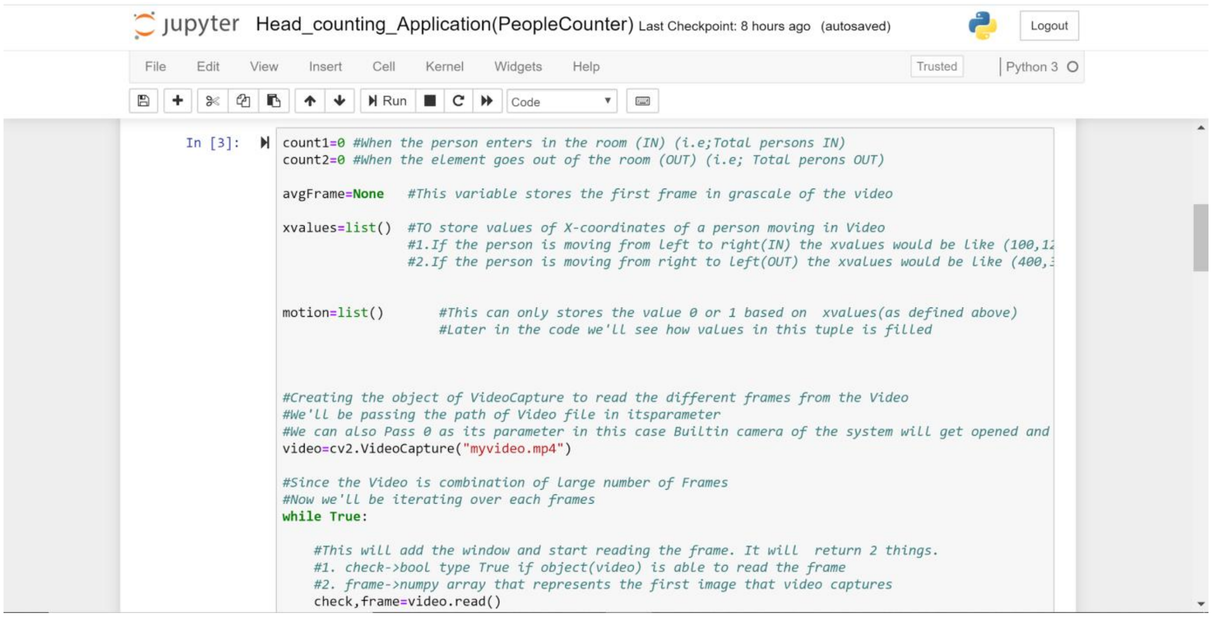

jupyter
Head_counting_Application(PeopleCounter) Last Checkpoint: 8 hours ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

Project on Calculating the Head Count

In [1]:
```python
#Importing libraries
import numpy as np
import cv2
import imutils
import datetime
```

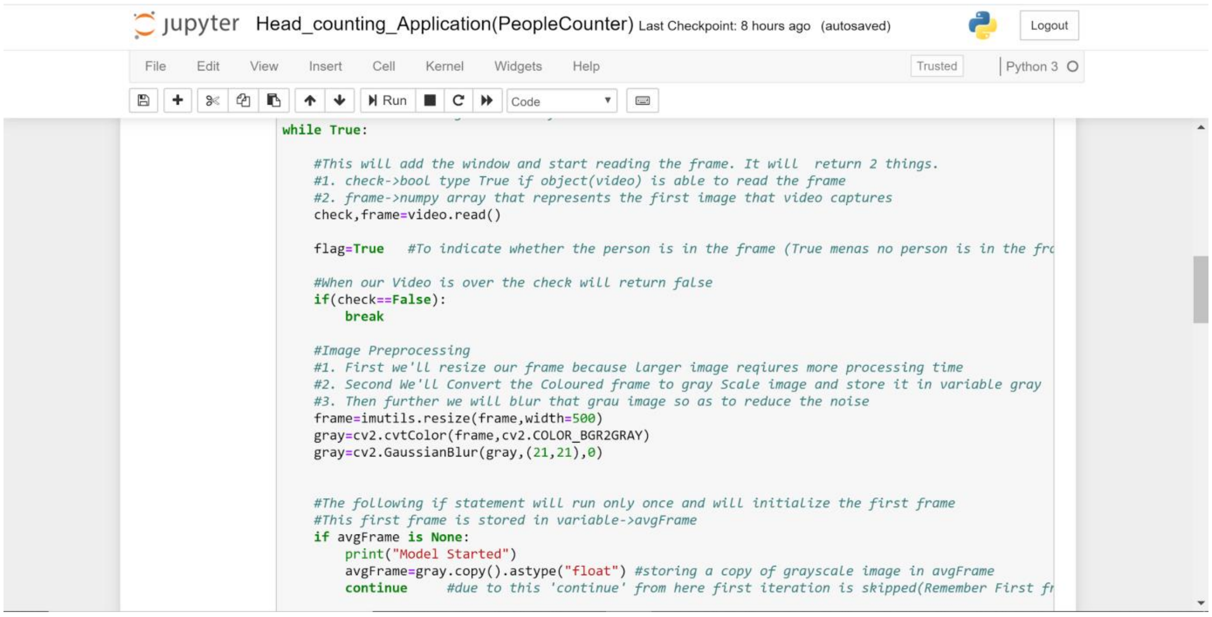
In [2]:
```python
#This function will decide whether the person is going IN or going OUT
#It will return 2 elements, first the element itself and second its occurrances
def find_majority_element(lst):
    Map={}          #declare map for storing values of lst
    maximum=('',0)   #(occurring element, occurrences)
    for i in lst:
        if i in Map:
            Map[i]+=1      #Increment the value of element in Map if it is encountered more than or
        else:
            Map[i]=1        #When the element is encountered at first
        #Keep track of maximum occurring element
        if Map[i]>maximum[1]:
            maximum=(i,Map[i])
    return maximum
```

Firstly we import libraries required for implementation of the code i.e Numpy, OpenCv, Imutils and Datetime. The function (find_majority_element) decides whether the person is going in or out. This function will return 2 elements, first the element itself and second its occurrences.


jupyter Head_counting_Application(PeopleCounter) Last Checkpoint: 8 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
In [3]:

```python
count1=0 #When the person enters in the room (IN) (i.e;Total persons IN)
count2=0 #When the element goes out of the room (OUT) (i.e; Total perons OUT)

avgFrame=None   #This variable stores the first frame in grascale of the video

xvalues=list()  #TO store values of X-coordinates of a person moving in Video
                #1.If the person is moving from left to right(IN) the xvalues would be like (100,12
                #2.If the person is moving from right to left(OUT) the xvalues would be like (400,


motion=list()       #This can only stores the value 0 or 1 based on  xvalues(as defined above)
                    #Later in the code we'll see how values in this tuple is filled



#Creating the object of VideoCapture to read the different frames from the Video
#We'll be passing the path of Video file in itsparameter
#We can also Pass 0 as its parameter in this case Builtin camera of the system will get opened and
video=cv2.VideoCapture("myvideo.mp4")

#Since the Video is combination of large number of Frames
#Now we'll be iterating over each frames
while True:

    #This will add the window and start reading the frame. It will  return 2 things.
    #1. check->bool type True if object(video) is able to read the frame
    #2. frame->numpy array that represents the first image that video captures
    check,frame=video.read()
```

In the above code snippet we are creating the object of VideoCapture to read the different frames from the Video. We'll be passing the path of the Video file in its parameter. We can also Pass 0 as its parameter in this case. BuiltIn camera of the system will get opened and will capture the first frame of the video.


jupyter Head_counting_Application(PeopleCounter) Last Checkpoint: 8 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

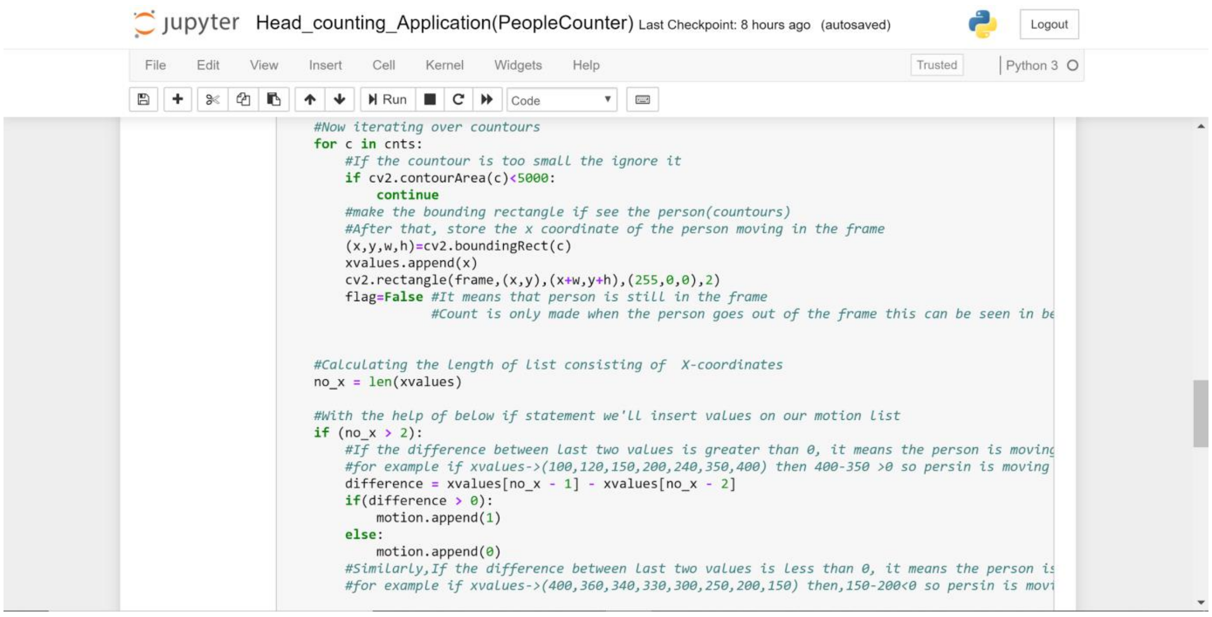
```python
while True:

    #This will add the window and start reading the frame. It will  return 2 things.
    #1. check->bool type True if object(video) is able to read the frame
    #2. frame->numpy array that represents the first image that video captures
    check,frame=video.read()

    flag=True   #To indicate whether the person is in the frame (True menas no person is in the fr

    #When our Video is over the check will return false
    if(check==False):
        break

    #Image Preprocessing
    #1. First we'll resize our frame because larger image reqiures more processing time
    #2. Second We'll Convert the Coloured frame to gray Scale image and store it in variable gray
    #3. Then further we will blur that grau image so as to reduce the noise
    frame=imutils.resize(frame,width=500)
    gray=cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    gray=cv2.GaussianBlur(gray,(21,21),0)


    #The following if statement will run only once and will initialize the first frame
    #This first frame is stored in variable->avgFrame
    if avgFrame is None:
        print("Model Started")
        avgFrame=gray.copy().astype("float") #storing a copy of grayscale image in avgFrame
        continue     #due to this 'continue' from here first iteration is skipped(Remember First fr
```

Since the Video is a combination of a large number of Frames. Now we'll be iterating over each frame. This will add the window and start reading the frame. It will return 2 things:

- Check->bool type True if object(video) is able to read the frame
- Frame->numpy array that represents the first image that video captures

Image Preprocessing

- First we'll resize our frame because larger images require more processing time.
- Second We'll Convert the Coloured frame to grayScale image and store it in variable gray.
- Then further we will blur that gray image so as to reduce the noise.

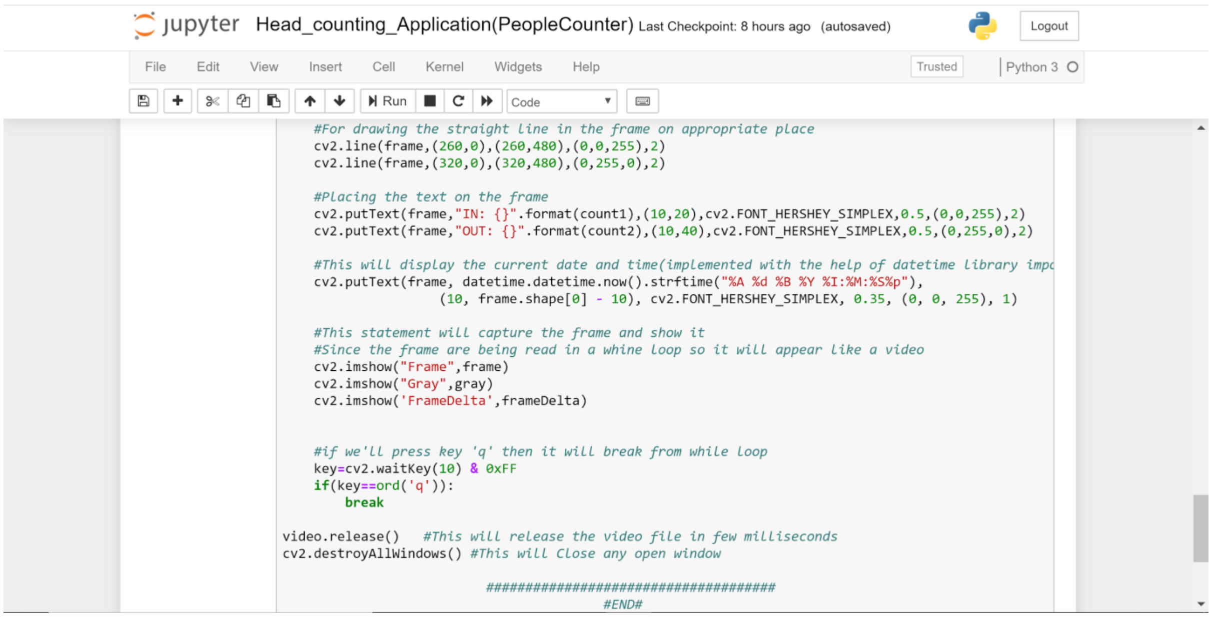

jupyter
Head_counting_Application(PeopleCounter) Last Checkpoint: 8 hours ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

```
    #Now iterating over countours
    for c in cnts:
        #If the countour is too small the ignore it
        if cv2.contourArea(c)<5000:
            continue
        #make the bounding rectangle if see the person(countours)
        #After that, store the x coordinate of the person moving in the frame
        (x,y,w,h)=cv2.boundingRect(c)
        xvalues.append(x)
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        flag=False #It means that person is still in the frame
                   #Count is only made when the person goes out of the frame this can be seen in be


    #Calculating the length of list consisting of  X-coordinates
    no_x = len(xvalues)

    #With the help of below if statement we'll insert values on our motion list
    if (no_x > 2):
        #If the difference between last two values is greater than 0, it means the person is moving
        #for example if xvalues->(100,120,150,200,240,350,400) then 400-350 >0 so persin is moving
        difference = xvalues[no_x - 1] - xvalues[no_x - 2]
        if(difference > 0):
            motion.append(1)
        else:
            motion.append(0)
        #Similarly,If the difference between last two values is less than 0, it means the person is
        #for example if xvalues->(400,360,340,330,300,250,200,150) then,150-200<0 so persin is movi
```

Now we'll find contours in this threshold image(A contour refers to the outline of an object).To find contours in an image, we need the OpenCV "cv2.findContours" function. It Accepts 3 parameters:

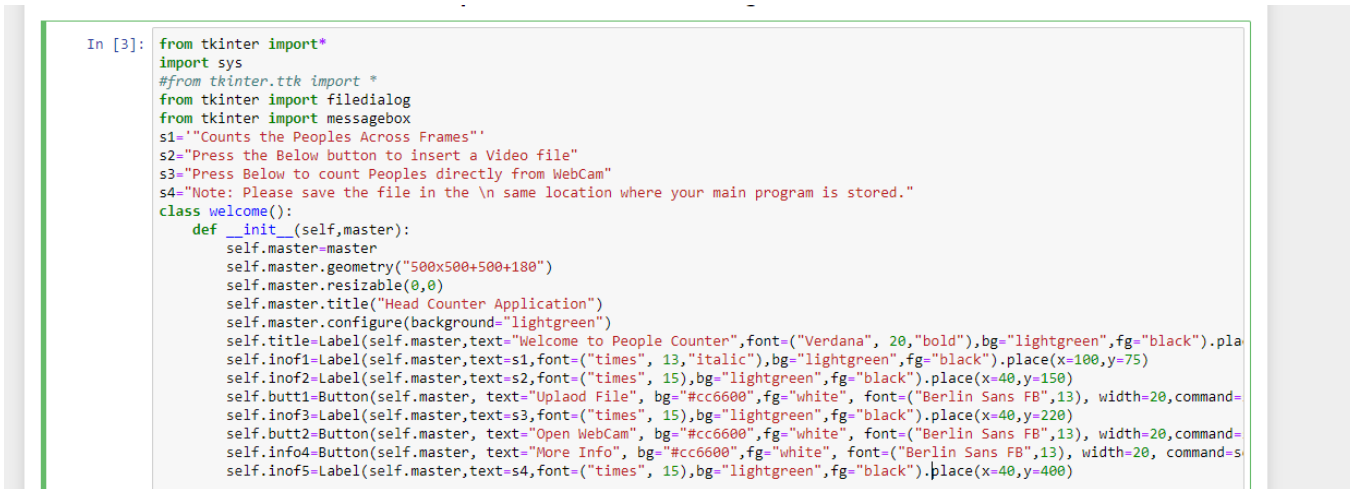
- Copy of threshold image(Because this function is destructive in nature so we pass the copy)
- cv2.RETR_EXTERNAL tells OpenCV to compute the hierarchy (relationship) between contours
- We tell OpenCV to compress the contours to save space using cv2.CV_CHAIN_APPROX_SIMPLE.

(cnts,_)=cv2.findContours(thresh.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

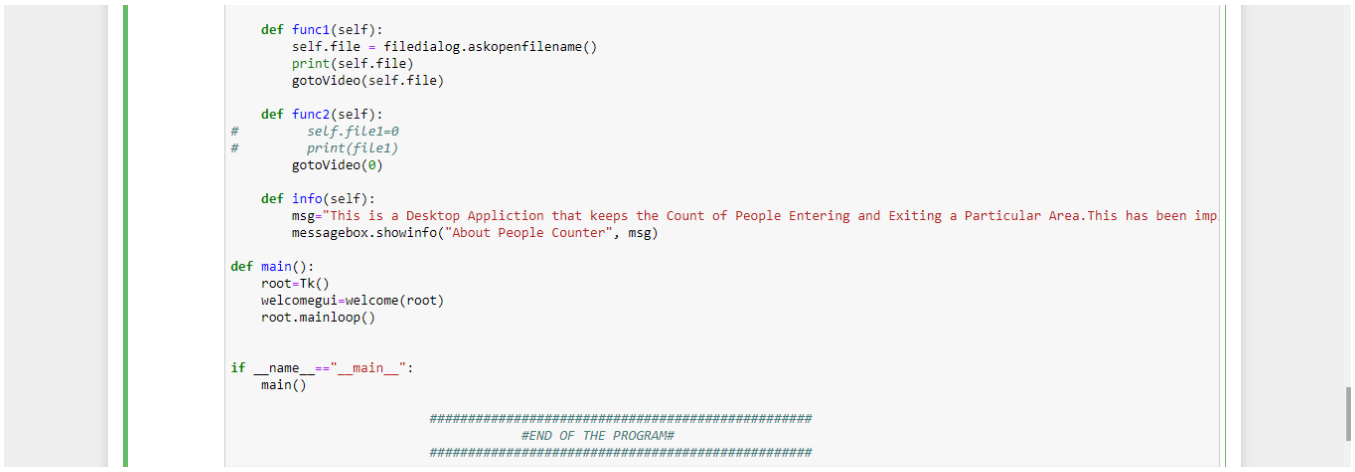

jupyter Head_counting_Application(PeopleCounter) Last Checkpoint: 8 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
        #For drawing the straight line in the frame on appropriate place
        cv2.line(frame,(260,0),(260,480),(0,0,255),2)
        cv2.line(frame,(320,0),(320,480),(0,255,0),2)

        #Placing the text on the frame
        cv2.putText(frame,"IN: {}".format(count1),(10,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,(0,0,255),2)
        cv2.putText(frame,"OUT: {}".format(count2),(10,40),cv2.FONT_HERSHEY_SIMPLEX,0.5,(0,255,0),2)

        #This will display the current date and time(implemented with the help of datetime library impo
        cv2.putText(frame, datetime.datetime.now().strftime("%A %d %B %Y %I:%M:%S%p"),
                (10, frame.shape[0] - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.35, (0, 0, 255), 1)

        #This statement will capture the frame and show it
        #Since the frame are being read in a whine loop so it will appear like a video
        cv2.imshow("Frame",frame)
        cv2.imshow("Gray",gray)
        cv2.imshow('FrameDelta',frameDelta)


        #if we'll press key 'q' then it will break from while loop
        key=cv2.waitKey(10) & 0xFF
        if(key==ord('q')):
            break

    video.release()   #This will release the video file in few milliseconds
    cv2.destroyAllWindows() #This will Close any open window

                      ######################################
                                     #END#
```

The Below Code Contains the implementation of GUI through tkinter-

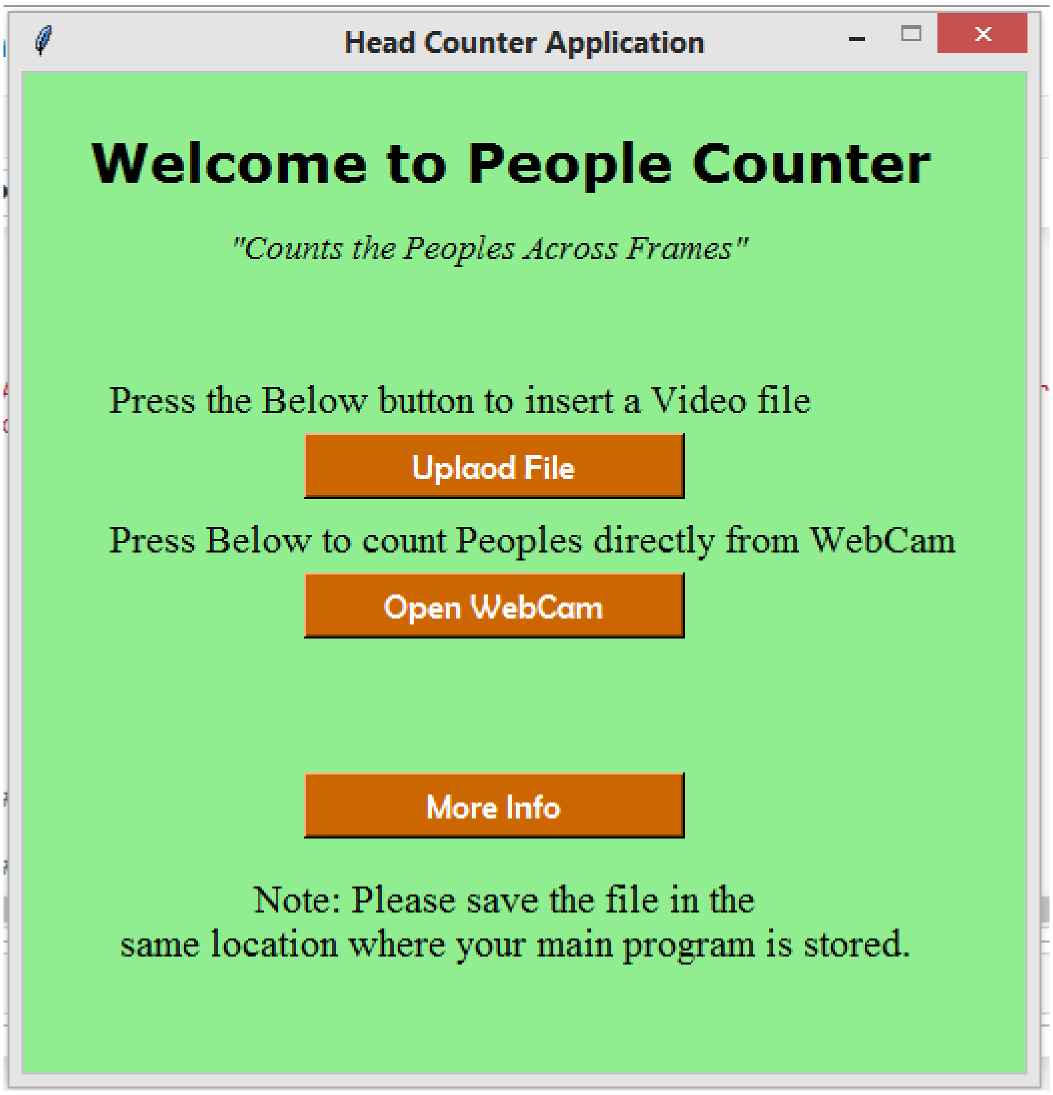


```
In [3]: from tkinter import*
        import sys
        #from tkinter.ttk import *
        from tkinter import filedialog
        from tkinter import messagebox
        s1='"Counts the Peoples Across Frames"'
        s2="Press the Below button to insert a Video file"
        s3="Press Below to count Peoples directly from WebCam"
        s4="Note: Please save the file in the \n same location where your main program is stored."
        class welcome():
            def __init__(self,master):
                self.master=master
                self.master.geometry("500x500+500+180")
                self.master.resizable(0,0)
                self.master.title("Head Counter Application")
                self.master.configure(background="lightgreen")
                self.title=Label(self.master,text="Welcome to People Counter",font=("Verdana", 20,"bold"),bg="lightgreen",fg="black").pla
                self.inof1=Label(self.master,text=s1,font=("times", 13,"italic"),bg="lightgreen",fg="black").place(x=100,y=75)
                self.inof2=Label(self.master,text=s2,font=("times", 15),bg="lightgreen",fg="black").place(x=40,y=150)
                self.butt1=Button(self.master, text="Uplaod File", bg="#cc6600",fg="white", font=("Berlin Sans FB",13), width=20,command=
                self.inof3=Label(self.master,text=s3,font=("times", 15),bg="lightgreen",fg="black").place(x=40,y=220)
                self.butt2=Button(self.master, text="Open WebCam", bg="#cc6600",fg="white", font=("Berlin Sans FB",13), width=20,command=
                self.info4=Button(self.master, text="More Info", bg="#cc6600",fg="white", font=("Berlin Sans FB",13), width=20, command=s
                self.inof5=Label(self.master,text=s4,font=("times", 15),bg="lightgreen",fg="black").place(x=40,y=400)
```



```
    def func1(self):
        self.file = filedialog.askopenfilename()
        print(self.file)
        gotoVideo(self.file)

    def func2(self):
#         self.file1=0
#         print(file1)
        gotoVideo(0)

    def info(self):
        msg="This is a Desktop Appliction that keeps the Count of People Entering and Exiting a Particular Area.This has been imp
        messagebox.showinfo("About People Counter", msg)

def main():
    root=Tk()
    welcomegui=welcome(root)
    root.mainloop()


if __name__=="__main__":
    main()

                          ##################################################
                                      #END OF THE PROGRAM#
                          ##################################################
```

Result of GUI Code Snippets-



Head Counter Application

Welcome to People Counter

"Counts the Peoples Across Frames"

Press the Below button to insert a Video file

Press Below to count Peoples directly from WebCam

Note: Please save the file in the
same location where your main program is stored.

GITHUB REPOSITORY LINK

https://github.com/as-hu/People-Counter