# Practice Project - OpenCV based : People Counting -Object Detection

## Table of Contents

## Problem Statement:

```
People Counter Based on OpenCV with concept of program counting number of people incomming and outgoing a particular door.
```

### 1.OpenCV People Counter

This program counts number of people incomming and outgoing a particular door. The example video is shot with Raspberry Pi Camera. I use OpenCV and Python 2.7. Make sure you install the numpy, cv2, imutils before you run the program.

```
Steps to execute the program with video source
```

Make sure you install the above mentioned dependencies. Place your video file and replace people-capture.mp4 in the line video = cv2.VideoCapture("people-capture.mp4") with your video filename Open your Terminal in OpenCV Environment Run python counter.py

### 2. Importing Libraries

In [ ]:

```python
import numpy as np
import time
import imutils
import cv2
```

### 3.Python Implementation

In [ ]:

```python
import numpy as np
import time
import imutils
import cv2

avg = None
video = cv2.VideoCapture("data/people-capture.mp4")
xvalues = list()
motion = list()
count1 = 0
count2 = 0
def find_majority(k):
    myMap = {}
    maximum = ( '', 0 ) # (occurring element, occurrences)
    for n in k:
        if n in myMap: myMap[n] += 1
        else: myMap[n] = 1

        # Keep track of maximum on the go
        if myMap[n] > maximum[1]: maximum = (n,myMap[n])

    return maximum

while 1:
    ret, frame = video.read()
    flag = True
    text=""

    frame = imutils.resize(frame, width=500)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (21, 21), 0)

    if avg is None:
        print ("[INFO] starting background model...")
        avg = gray.copy().astype("float")
        continue

    cv2.accumulateWeighted(gray, avg, 0.5)
    frameDelta = cv2.absdiff(gray, cv2.convertScaleAbs(avg))
    thresh = cv2.threshold(frameDelta, 5, 255, cv2.THRESH_BINARY)[1]
    thresh = cv2.dilate(thresh, None, iterations=2)
    (_, cnts, _) = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    for c in cnts:
        if cv2.contourArea(c) < 5000:
            continue
        (x, y, w, h) = cv2.boundingRect(c)
        xvalues.append(x)
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)
        flag = False

    no_x = len(xvalues)

    if (no_x > 2):
        difference = xvalues[no_x - 1] - xvalues[no_x - 2]
        if(difference > 0):
            motion.append(1)
        else:
            motion.append(0)

    if flag is True:
        if (no_x > 5):
            val, times = find_majority(motion)
            if val == 1 and times >= 15:
                count1 += 1
            else:
                count2 += 1

        xvalues = list()
        motion = list()

    cv2.line(frame, (260, 0), (260,480), (0,255,0), 2)
    cv2.line(frame, (420, 0), (420,480), (0,255,0), 2)
    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, 0, 255), 2)
    cv2.imshow("Frame",frame)
    cv2.imshow("Gray",gray)
    cv2.imshow("FrameDelta",frameDelta)

    key = cv2.waitKey(1) & 0xFF
    if key == ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```