Software Requirements and Specifications for People Counter

Prepared By: Group 13

Names and ID:
Nabiul Hoque Khandakar - 1631164642
Talha Yamin - 1712731642
Golam Kibria Anik - 1712345642

Course: CSE 323
Section: 11
Date of Submission : 14/12/2019

Table of Contents

1.INTRODUCTION

1.1 Purpose - The purpose of our project is to build the titled “People Counter” with OpenCV and Python. Using OpenCV, we’ll count the number of people who are heading “in” or “out” of a designated space in real-time. This space can be any bounded location from homes and office spaces to malls and stadiums. This can help to reduce electricity and utility costs as there is no scope for human error. Due to a lack of funds and resources we could not practically build and construct the entire apparatus of the project and as such this report serves to mainly show the theoretical and programming aspects.

1.2 Intended Audience - The intended audience of our project ranges from any business or organization to individuals such as mainstream consumers.

1. Household and Office Spaces- Used for tasks such as video surveillance and security.
2. Commercial Spaces - Used for the aforementioned tasks as well as statistical analysis of people accessing an area, and other added value products.

1.3 Product Scope - This program offers a complete and easy way to identify and count the different numbers of people entering and exiting a space by locating them with green circles. The benefits of this is that an automated system will keep track of people that are entering and exiting a place. Its main objective is to identify only human beings of all types regardless of age, gender, height etc. using a pre-trained machine learning algorithm. The algorithm can be made to run on any central system attached to surveillancing equipment at entrances and exits of the desired space.The goal is to make possible for home and business owners to track people traffic in and out of their spaces and possibly provide more value added services such as automation (i.e Turning On or Off electronic components based on people traffic in a specific space) provided that they have already installed necessary additional components for their desired action.

1.4 Reading Suggestions and References - This document contains the necessary requirements and some aspects of the analysis of the requirements and is organized based on the paper at the link ieeexplore.ieee.org/document/622685 and the blog post at the link www.pyimagesearch.com/2018/08/13/opencv-people-counter/ . All images have been taken courtesy of the links above as well.

2 Overall Description

2.1 Product Perspective - This document presents an application for counting people through a single fixed camera. This system performs the count distinction between input and output of people moving through the supervised area. The counter requires two steps: detection and tracking. The detection is based on finding people's heads through preprocessed image correlation with several circular patterns. Tracking is made through the application of a Single Shot Detector(SSD) to determine the trajectory of the candidates. Finally, the system updates the counters based on the direction of the trajectories. Problematic situations, such as occlusions, people grouped in different ways, scene luminance changes, etc. can be dealt with if the need arises. Figure below shows the possible camera views.

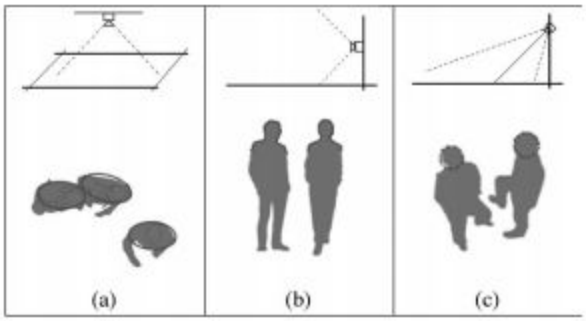

Fig. 1. Possible camera views. (a) Overhead view. (b) Front view. (c) Inclined view.

2.2 Understanding object detection vs. object tracking - To understand the scope of our project we must first understand the difference between object detection and object tracking and how both have been used to serve our aim. When we apply object detection we are determining where in an image or frame an object is. An object detector is typically more

computationally expensive, and therefore slower, than an object tracking algorithm. Examples of object detection algorithms include Haar cascades, HOG + Linear SVM, and deep learning-based object detectors such as Faster R-CNNs, YOLO, and Single Shot Detectors (SSDs).

An object tracker, on the other hand, will accept the input *(x, y)*-coordinates of where an object is in an image and will:

1. Assign a unique ID to that particular object.
2. Track the object as it moves around a video stream, predicting the new object location in the next frame based on various attributes of the frame (gradient, optical flow, etc.).

Examples of object tracking algorithms include MedianFlow, MOSSE, GOTURN, kernelized correlation filters, and discriminative correlation filters, to name a few. With the differences cleared between both let us focus on how both are used simultaneously for our tracking algorithm.

2.3 Combining both object detection and object tracking - Highly accurate object trackers will combine the concept of object detection and object tracking into a single algorithm, typically divided into two phases:

- Phase 1 — Detecting: During the detection phase we are running our computationally more expensive object tracker to

  (1) Detect if new objects have entered our view.

  (2) See if we can find objects that were “lost” during the tracking phase. That is to say any object that has left our designated premises.

For each detected object we create or update an object tracker with the new bounding box coordinates. Since our object detector is more computationally expensive, we only run this phase once every *N* frames.

- Phase 2 — Tracking: When we are not in the "detecting" phase we are in the "tracking" phase. For each of our detected objects, we create an object tracker to track the object as it moves around the frame. Our object tracker should be faster and more efficient than the object detector so that we can track an object as soon as it comes onto our boundary and subsequently track it. Tracking continues until the *N*-th frame has been reached and then the object detector is re-run. The entire process then repeats. The benefit of this hybrid approach is that we can apply highly accurate object detection methods without as much of the computational burden. A similar tracking approach is used in building the People Counter.

2.4 Combining object tracking algorithms - A step by step rundown of the centroid tracking algorithm. Figures are named after corresponding steps.

Step #1- Accept a set of bounding boxes and compute their corresponding centroids (i.e., the center of the bounding boxes):
The bounding boxes themselves can be provided by either:

1. An object detector (such as HOG + Linear SVM, Faster R- CNN, SSDs, etc.)
2. Or an object tracker (such as correlation filters).

In the image below there are two objects to track in this initial iteration of the algorithm.

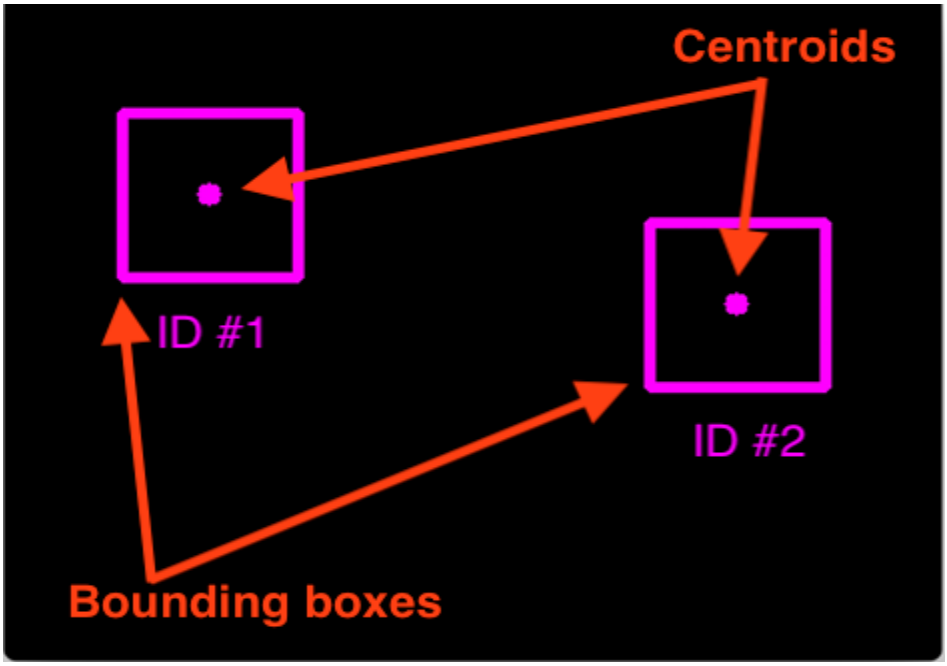

Figure 1

Step #2 - Compute the Euclidean distance between any new centroids (yellow) and existing centroids (purple). The centroid tracking algorithm makes the assumption that pairs of centroids with minimum Euclidean distance between them must be the same object ID.

In the image there are two existing centroids (*purple*) and three new centroids (*yellow*), implying that a new object has been detected (since there is one more new centroid vs. the already old centroid). The arrows then represent computing the Euclidean distances between all purple centroids and all yellow centroids.

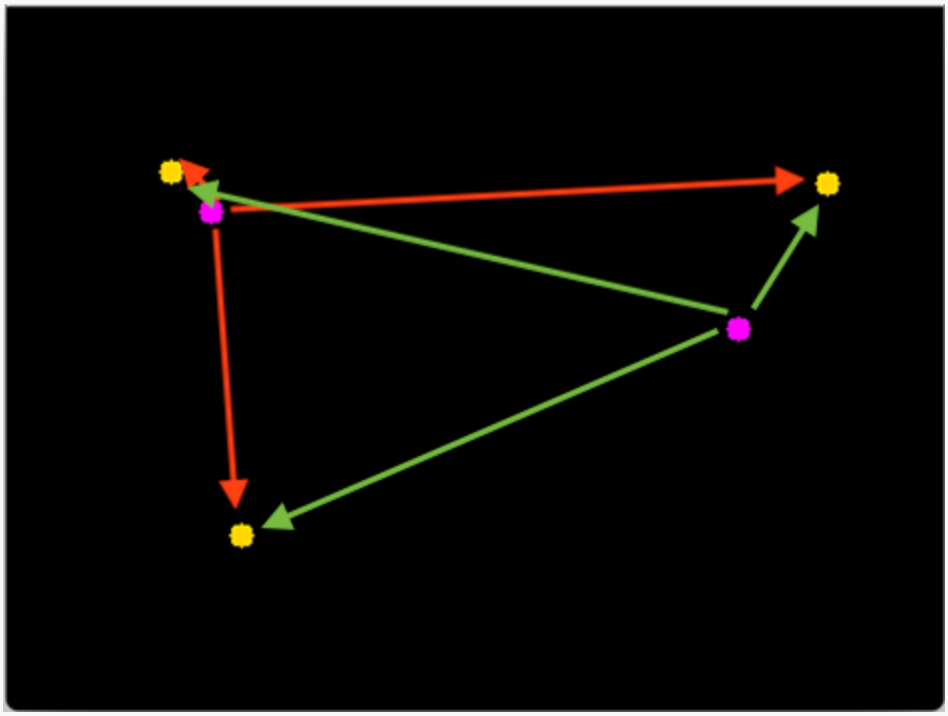

Figure 2

Step #3 - The calculated Euclidean distances are associated with object IDs. The centroid tracker choses to associate centroids that minimize their respective Euclidean distances. As for the point in the bottom-left that didn't get associated with anything the next step solves this dilemma.

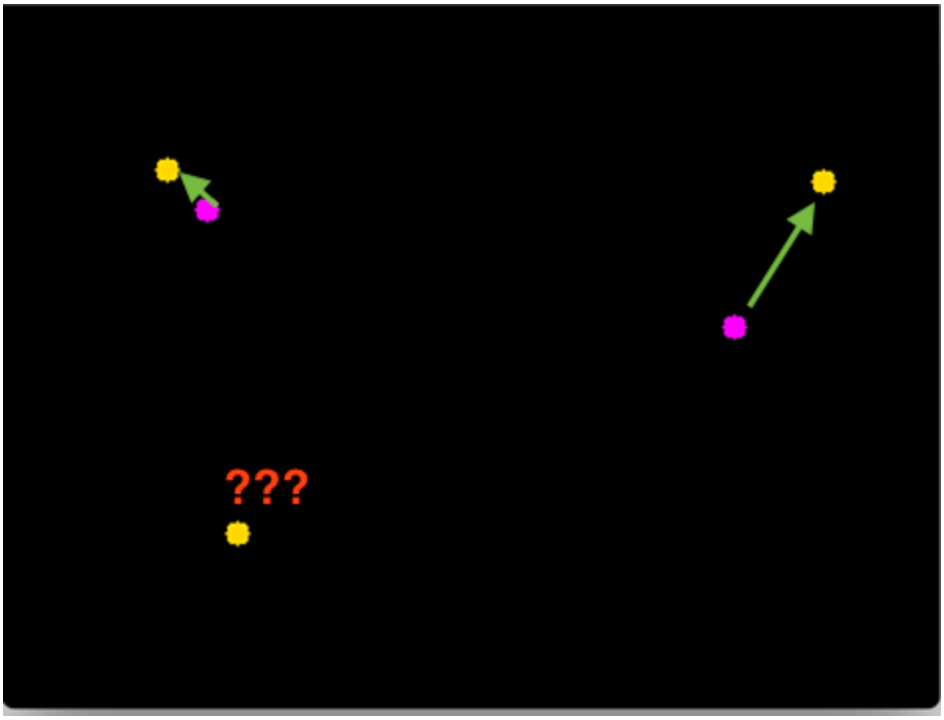

Figure 3

Step #4 - Registering new objects simply means adding the new object to the list of tracked objects by:

1. Assigning it a new object ID
2. Storing the centroid of the bounding box coordinates for the new object

As per the image in Step#3 there was a new object that wasn't matched with an existing object. To fix this it was now registered as an object with ID#3 shown in the image on the next page.

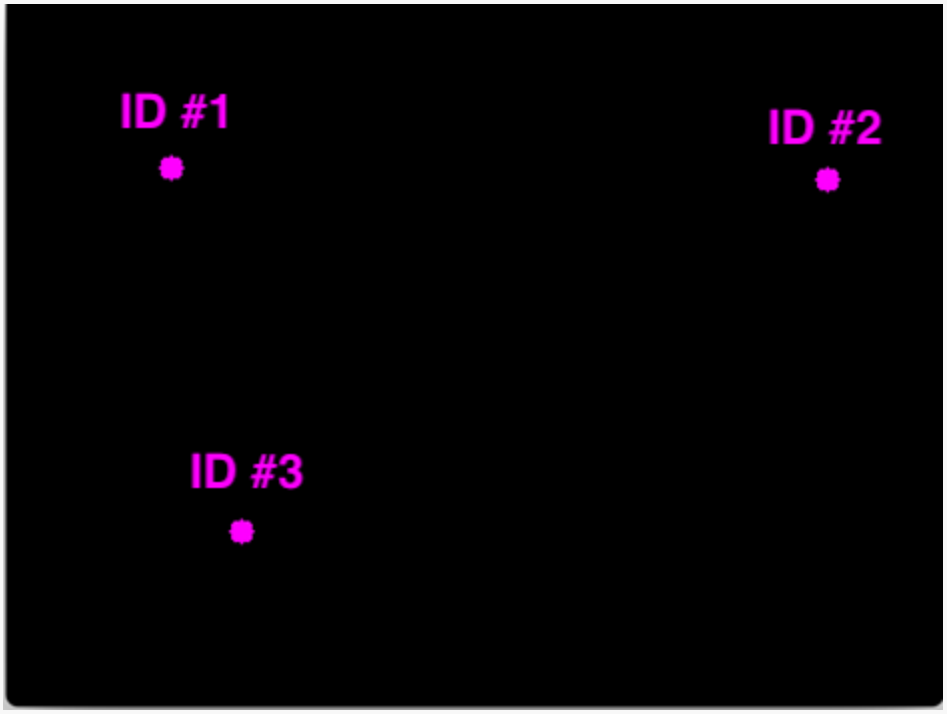

Figure 4

Step#5 - In the event that an object has been lost or has left the field of view, it can simply be deregistered. The definition of “lost” or is “no longer visible” really depends on the exact application and so for the People Counter, people IDs are deregistered when they cannot be matched to any existing person objects for 40 consecutive frames.

2.5 Operating Environment and Dependencies - The different Python libraries used to build our people counting application are:

- NumPy
- OpenCV - for standard computer vision/image processing functions, along with the deep learning object detector for people counting.
- dlib - implementation of correlation filters as it is easier to work with for the project
- imutils

Additionally, the `pyimagesearch` module, the Python driver script used to start the people counter and example videos have been used. Any machine running Windows or Mac OS having python and the above mentioned libraries and modules will be able to use the people counter.

3 System Features

Since the program does not exist as a complete software such as an application or central software it can be designed as such based on the type of client looking to use it.

3.1 Home and Office Automation - This piece of software can be used as a controller for automating electrical components in any household or workplace. That is, it can be used to turn said electrical components, be they appliances or simple components, on or off based on the presence of people in a room or larger space.

3.2 Statistical Analysis of Customer Traffic - Business owners seeking for customer analytics can install this on their storefronts to track customer interaction and any numerical data associated with the comings and goings of their customer and/or staff.

4 Other Nonfunctional Requirements

4.1 Performance Requirements - The system should be able to track any given amount of people in a single frame and should be able to operate on a 24 hour basis given that the lighting conditions are optimal**.**

4.2 Security Requirements - The system should operate under strict supervision as footage captured by the camera should be confidential to everyone else outside the owner's/user's designated space.