What's the utility?

Crowd_Count++ is aimed at counting the amount of people in a room, in a crowd, at a bus station, or in almost any other situation. There are many examples of uses for our system. One such example is the MTA buses in New York City. Being able to feed that footage into a software system that counts the amount of passengers on a bus would enable the MTA to make better decisions when dispatching buses from their depots, and thus creating a more efficient transportation system for the citizens of NYC. This method would also be cheaper than installing hardware sensors on each bus. Also, it can be used by business for counting the number of people that travel through their location and probably do better advertisement strategies.

What’s the challenge?

The main challenges in counting people indoors or outdoors is that it is not 100% accurate. Computer vision is an active area of research and must use predefined algorithms such as Haars Algorithm and Histogram of Oriented Gradients. They have a high accuracy only if the input image is not too clustered with objects (people in our case) and a fixed position in an image. Our images are constantly changing since people can be closer or further apart from the camera which causes when we run our algorithms with a fixed size window to miss some objects. Even if we are able to change the scanning window size for each image, we cannot still say without looking at the image what would be the optimal window size. Also, we have considered using background subtraction for improving our algorithm since some objects never change position. Having those areas shown as black can improve our algorithm since we can make those areas black in our original image that otherwise might have been considered false positives. However, the challenge with this approach is that we need to find an optimal threshold for each image to make it black.

What’s your algorithm/creativity?

Our methodology for counting people is as follows:

Given a video input in any format and location we convert the video footage into a .mov format since we are using OpenCv libraries for extracting frames from the video. Once the frame is extracted, we standardize the frame to 1024 x 1024 and use it as input for our face detection algorithm. Our indoors algorithm uses the Haar Cascade as means to detect people. We run the algorithm using front and profile view since people are not always looking at the camera. After this there might be some overlap between the front and profile view algorithms, therefore we run the image through a Non-Maximum Suppression which takes each overlap as 1. After this the count is saved into a Mongo Database, and the results can be displayed in the front end by querying for location or date.