Social Distance Monitoring System using MATLAB

Name: Yazeed Eldigair

Date: 15-Jun-2020

Abstract

This research paper discusses the implementation of a social distancing monitoring system developed in MATLAB. Pedestrians from a surveillance camera video footage are detected, tracked, and checked whether they are maintaining a sufficient social distance between others. Using an adaptive background modeling technique, the people in the footage are marked as foreground. Their locations are then mapped to a bird's eye view of the scene and distances between them get calculated.

I. Introduction

The COVID-19 pandemic has taken the world by storm and resulted in various aspects of our daily lives being affected by it. Ever since the outbreak, all science disciplines have taken the initiative to play a role in the fight against the virus. One field in particular that can have a huge impact on reducing the spread of the virus is the Computer Vision field. Using surveillance cameras and the implementation of social distancing monitoring algorithms, contact tracing can become significantly easier and more reliable.

With the help of machine learning, article [1] provides a method a of estimating measurements of distances between walking pedestrians in a video while displaying a bird's eye view of the scene. In order to project the images taken from a monocular camera, the algorithm asks the user to input four points in the video that are parallel to the view from above. The shape drawn then helps in projecting the scene to a bird's eye view. For pedestrian detection, the algorithm uses deep learning to localize the pedestrians in the scene and display them in the bird's eye view.

[2] also uses quite a similar approach to tackle the problem. When calibrating the scene, a scale factor was estimated to determine the number of pixels that represent 6 feet in real life. For detection, a pedestrian detection network was used that is based on the Faster R-CNN architecture.

II. Proposed Algorithm

Figure 1 illustrates the flow diagram of the system proposed. Initially, a frame is read from the camera or the video of pedestrian surveillance footage. The frame read is used to update the background model using a mixture of gaussians. The result will give us a binary image with pedestrians marked as foreground objects. A series of morphological operations are performed in order to pre-process the binary image obtained. Using a blob analyzer, the statistics of each object that differentiates it from other objects in the frame are obtained. We then update our assigned and unassigned tracks. After acquiring the measurements required to obtain a bird's eye view of the scene, each object in the frame is projected to the bird's eye view and is represented by a dot. The distance between each person and the other is calculated using the scale factor to convert from pixels to meters. In case the distance between any two people is less than 1.8 meters, they are marked as red dots.

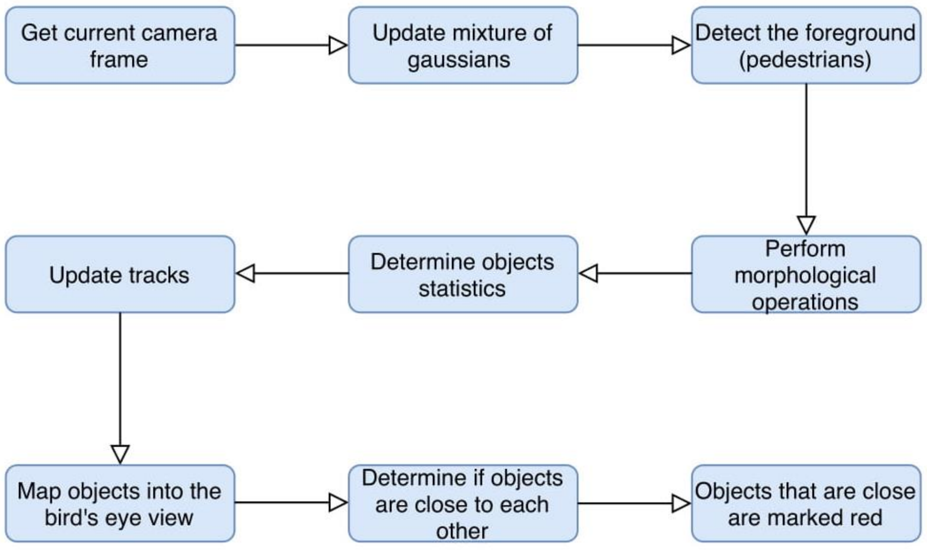
Get current camera frame
Update mixture of gaussians
Detect the foreground (pedestrians)
Perform morphological operations
Determine objects statistics
Update tracks
Map objects into the bird's eye view
Determine if objects are close to each other
Objects that are close are marked red

Figure 1. Flow diagram of the proposed algorithm

Calibration:

The video used for testing the algorithm is a monocular camera view of a pedestrian surveillance footage. Since the camera angle is fixed and there is no motion involved; in order to acquire a bird's eye view representation of the scene, it was necessary to determine real-life measurements in the scene. Various methods can be implemented to obtain a bird's eye view but for the purpose of this project, the height at which the camera is surveilling was determined and its pitch was estimated. Using these to measurements and the aid of MATLAB's built-in functions, the top view representation was generated. Figure 2 displays the result after applying the transformation on one of the frames of the video

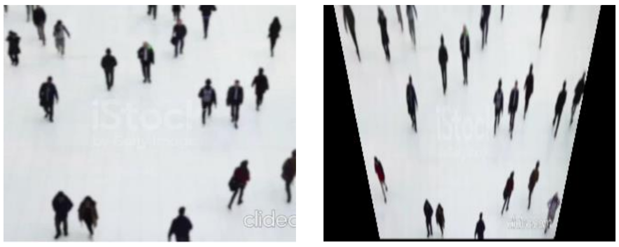

Figure 2. Frame read from the video on the left and the output after mapping on the right

Detection:

Stauffer's mixture of Gaussian models was adapted in this method for the purpose of detecting the foreground in each frame received. Initially, 10 frames are used as initialization frames in order to estimate the means and standard deviations of the Gaussian models. In this implementation, 5 Gaussian models were used to model the system. The models were then updated according to the pixel values they encounter. This results in the formation of a binary image that segregates the background from the foreground which are pedestrians in this case.

After an incoming frame has been binarized, morphological operations are applied to it to attenuate noise and fill in holes. The operations performed are shown in Figure 3. A binary image obtained after the operations is shown in Figure 4.

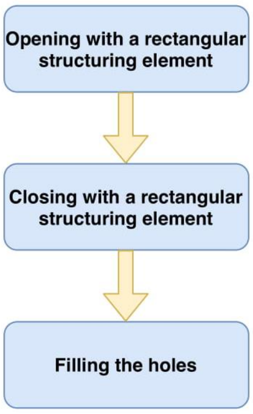
Opening with a rectangular structuring element
Closing with a rectangular structuring element
Filling the holes

Figure 3.Morphological operations on binary images

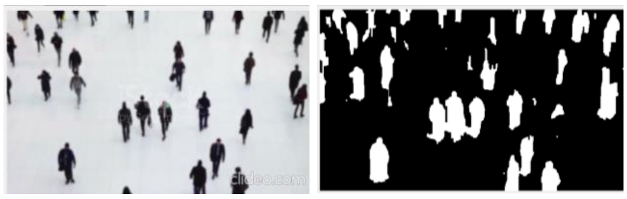

Figure 4. Output after detection and morphological operations

Tracking:

The minimum bounding boxes and the centroids of all the detected objects are then determined which would help us in tracking each object. Firstly, using a Kalman filter, the future location of the centroid and the minimum bounding box are predicted. Depending on the detections, assigned and unassigned tracks are updated. Tracks that have become old and outdated since they stopped appearing after a period of time get destroyed. Meanwhile, from unassigned tracks, we create new tracks. These are the same tracking steps followed by [3]. Figure 5 shows the result of tracking people throughout the video while keeping their identity.

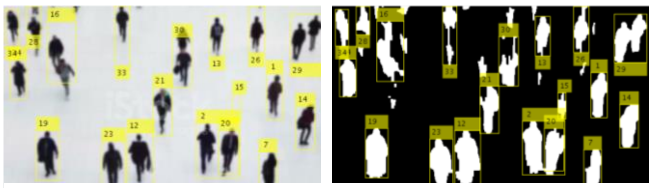

Figure 5. Object tracking throughout the frames

Monitoring Distance:

From the video, it was estimated that 1.8m (6ft) was equivalent to about 45 pixels in the bird's eye view. Using this measurement, we check whether there are any other dots that are within a 45 pixel radius of the object. In case there is, we flag the object; otherwise, the object is green in the bird's eye view.

III. Results and Discussion

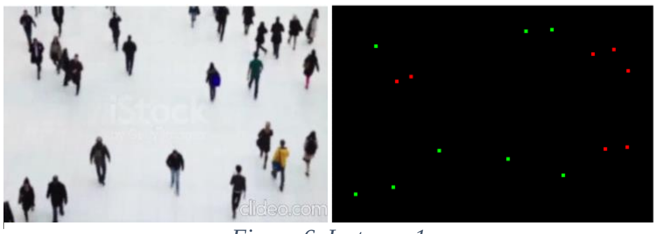

Figure 6. Instance 1

Figure 6 shows an instance during the period of surveillance. The image on the left is the frame of the video that is currently being processed. The image on the right is the bird's eye view adapted after transforming the pixel locations of the centroids of the detected objects to the projected plane.

The green dots represent individuals who are at least 1.8m (6 ft) away from anyone else; whereas the red dots represent people who are not maintaining a sufficient social distance between others.

As can be seen from the Figure 6, not all pedestrians were successfully detected by the algorithm. Whenever there are multiple people clumped up, the algorithm is more likely to miss on a few of those people. This is mainly because when people are very close, they might be detected as only object rather than multiple

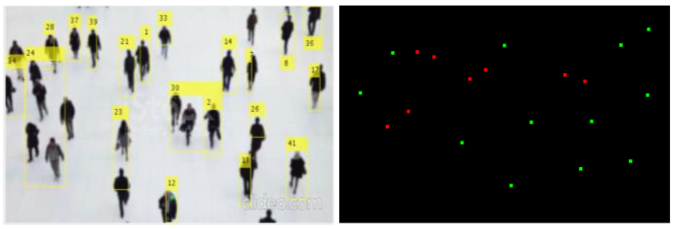
Figure 7. Instance 2

Figure 7 depicts another instance from the surveillance video. Once again it is observable that the algorithm does a good job at detecting individuals who are maintaining a good social distance; however, when people are really close to each other, they would sometimes get detected as one person.

IV. Conculsion

There is vast room for improvement. Primarily, it is essential to be able to differentiate between objects that are close to each other which seems to be a problem in this implementation. Moreover, detection is not perfect as from time to time objects which appear in the video do not get detected as foreground; thus, different GMM parameters or a different approach can be tested to improve performance. The appearance of the same object throughout multiple scenes has not been tested either. Finally, distance between one person and the other is being calculated but not displayed in the bird's eye view of the scene.

V. References

[1] A. Anwar, "Towards Data Science," April 2020. [Online]. Available: https://towardsdatascience.com/monitoring-social-distancing-using-ai-c5b81da44c9f. [Accessed June 2020].

[2] "Landing AI," Landing AI, April 2020. [Online]. Available: https://landing.ai/landing-ai-creates-an-ai-tool-to-help-customers-monitor-social-distancing-in-the-workplace/. [Accessed June 2020].

[3] "MathWorks," MathWorks, [Online]. Available: https://www.mathworks.com/help/vision/examples/motion-based-multiple-object-tracking.html. [Accessed May 2020].