Real-time Contact Tracing During a Pandemic using Multi-camera Video Object Tracking

Maha Yaghi‡, Tasnim Basmaji‡, Reem Salim, Jawad Yousaf, Huma Zia and Mohammed Ghazal*
Electrical and Computer Engineering Department
Abu Dhabi University
Abu Dhabi, UAE
*Correspondence: mohammed.ghazal@adu.ac.ae
‡These authors contributed equally to this work.

2020 International Conference on Decision Aid Sciences and Application (DASA) | 978-1-7281-9677-0/20/$31.00 ©2020 IEEE | DOI: 10.1109/DASA51403.2020.9317132

***Abstract*—Due to the COVID19 pandemic, contact tracing and moving object tracking are gaining more popularity in automated video surveillance systems in computer vision and video processing. The application of contact tracing and moving object tracking is critical in applying pandemic control measures and is getting more important day by day. This work proposes a computer vision-based algorithm for contact tracing using stationary surveillance cameras. The input videos are converted into a bird's eye view where all moving objects are detected, and the distances between them are calculated. The algorithm performs background subtraction to isolate foreground objects, morphological operations to remove the noise, and blob analysis to identify the connected regions in the resulting foreground video. Kalman filters to estimate objects' motion in the video calculates Euclidean distance between the objects to trace object contacts. This algorithm can be utilized in almost all public places such as shopping malls, airport terminals, and educational institutions. It allows identifying, assessing, and managing people who might have been exposed to the disease. The testing data was collected in a home environment, and the stationary camera was replaced with a mobile phone camera fixed on a tripod. The work was implemented and tested, and the results verified the feasibility and effectiveness of the proposed method. The system was able to detect the objects in the input video frame and estimate the distance between them across multiple cameras.**

***Index Terms*—Computer Vision, Video Processing, Contact Tracing, Object Tracking**

I. Introduction

Infectious diseases are diseases caused by microorganisms, including bacteria, viruses, fungi, and parasites that can be passed through direct or indirect contact with people or animals or by consuming contaminated food or water [1]. The recent COVID-19 is an infectious disease caused by a virus that spreads from person-to-person through droplet and contact transmission [2]. Various measures have been taken worldwide to reduce the spread of the virus between people, including lockdowns, quarantines, distant learning, and social distancing, leading to severe consequences [3]. Contact tracing is the process of identification of people who came in contact with an infected person [4]. It is one of the control measures employed by health authorities' personnel for decades to prevent the spread of infectious diseases through contact and stop transmission. Multiple technological solutions have been introduced to facilitate contact tracing, including several mobile applications and microcontroller-based systems; however, these solutions raised many security concerns and are not published yet. The recent progress in object detection and tracking has also encouraged more research due to todayś pandemic. A computer vision-based algorithm that utilizes the available infrastructure for contact tracing is one of the highly anticipated solutions at the current time [5].

Although computer vision-based contact tracing solutions have not been published, object segmentation, tracking, and recognition across the frames of a video is a classical computer vision problem that has been introduced in multiple publications throughout the years. Solutions proposed are based on either static cameras to track moving objects or cameras attached to the moving objects [6]. Video surveillance is one of the applications of indoor object tracking that gained much popularity and represents the first part of the proposed system. Authors in [7] proposed a robust tracking algorithm achieved using feature fusion and multiple cameras. The proposed method integrates spatial position, shape, and color information to track object blobs. The trajectories obtained from individual cameras are incorporated by an extended Kalman filter to resolve object occlusion. The authors proposed another tracking solution in [8]. The proposed system is based on panoramic cameras and a map of the indoor environment. The system applies a background subtraction method to detect human beings; then, it searches the foreground pixels to find the pixel whose location represents the object's location and matches their location, providing precise location information and allowing object tracking. Authors in [9] proposed an algorithm to track pedestrians and detect situations where people may be in danger using an automated thermal video surveillance system. The system involves detection, tracking, and recognition. Detection and tracking are performed on a thermal input image to improve the recognition stage. However, the system is limited to detect and track one person at a time. Additionally, the study in [10] showcases a blob recognition and tracking method. Although this approach is not targeting humans directly, it uses a method to detect blobs, which can be people. The system begins by inputting images, binarizing, and smoothing them for noise reduction. Background subtraction is then performed to segment the

foreground from the background. The next stages include rectification, blob detection, and blob tracking performed based on the Euclidean minimum distance. Similarly, a Computer Vision-based tracking system was presented in [11]. The system involves several stages, including data acquisition, background detection, and tracking.

This paper presents a vision-based real-time contact tracing and moving object tracking system to ensure social distancing, especially in the current pandemic. The remainder of the paper is organized as follows: Section II describes the workflow of the proposed system, Section III shows and reviews experimental results, and Section IV concludes the paper with a summary of key points discussed.

II. Proposed System

The proposed system is a computer vision-based contact tracing system that consists of multiple pipelines combined to perform automatic detection and tracking of moving objects in a video from stationary cameras converted to a bird's eye view. As shown in Fig. 1, the algorithm's pre-processing stage begins by applying a background subtraction algorithm based on Gaussian mixture models to detect the moving object, morphological operators to eliminate noise, and blob analysis to detect the connected pixels that will mostly form objects. Once objects are detected, the motion track of each object is estimated by a Kalman filter. The association of detections to the same object is based on the estimated motion. The filter estimates the track's location and determines the likelihood of the detection-track association. Due to the possibility of having some unassigned tracks, track maintenance is the next step in this algorithm. This step guarantees that the assigned tracks are continuously updated, and the unassigned tracks are marked invisible. The algorithm then calculates the euclidean distance between detected objects associated with tracks for contact tracing while applying color codes that reflect other moving objects' distances. Object tracking across multiple cameras is implemented using centroids, and the area of detected blobs is identified during the blob analysis step. Once an object leaves one camera's scope and enters the other, the last detected centroid and the blob's area are used to identify the object across multiple cameras. Each step of the proposed algorithm is explained in detail below.

A. Bird Eye View Scene Transformation

A birds-eye view is an overall-top view of a scene used as a pre-processing stage in certain applications. In the proposed system, every video frame is pre-processed and transformed into a birds-eye view, using inverse perspective mapping, for accurate distance measurement, presented at later stages. Camera intrinsics, including focal length, principal point, and image size, are collected using the CameraCalibrator toolbox in Matlab. The calibration accuracy is evaluated by examining the projection errors, camera extrinsic, or undistorted images. Once the camera intrinsics are exported, the camera's height and pitch, the area in front of the camera, and the sides must be

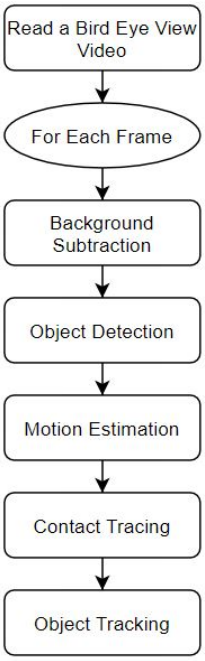


Fig. 1. Flowchart of proposed algorithm

set based on trials. The transformation algorithm is described as follows:

1) Create an object to perform the transforms
2) Shift the pixels (x_1, y_1) of the frame I, of size (MxN), to place the center at point (0,0).

$$x_2 = x_1 - \frac{N}{2} \tag{1}$$

$$y_2 = y_1 - \frac{M}{2} \tag{2}$$

3) Rotate the frame pixels by the specified mounting angle.
4) Scale the frame height to overcome the reduction in size due to rotation to get frame J.
5) Project frame J on a two-dimensional plane using the pinhole model technique and the predefined focal length.

B. Background Subtraction

The proposed algorithm applies a Matlab implemented background subtraction algorithm to detect the moving objects in each frame. The foreground detector applied compares each frame to a background model to determine whether each pixel belongs to the background or the foreground. Then, it computes a foreground binary mask, where the pixel value of 1 corresponds to the foreground, and a pixel value of 0 corresponds to a background, hence allowing the detection of non-stationary objects in a video taken using a stationary camera. The first frame in the video is set to be the background model (B). The difference image (D) is found by subtracting the background model from each frame (I). Then the calculated values are compared against a threshold to generate the binary mask. The diagram in Fig. 2 shows how the algorithm works. The number of training frames, number of Gaussians,

and minimum background ratio are all tunable parameters that can be changed based on the system requirements.

$$D = | I - B |$$

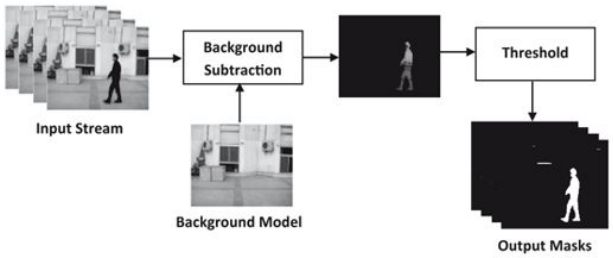


Fig. 2. Background subtraction workflow diagram

C. Object Detection

The next stage of the system is to detect objects and report their specifications. This stage involves two stages: (1)applying morphological operations and (2)blob analysis, as depicted in Fig. 3.

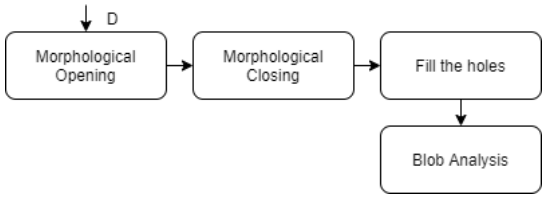


Fig. 3. Pipeline of the Object Detection algorithm

1) Morphological Operations: After performing motion segmentation using the foreground detector and generating the binary mask, the algorithm performs several morphological operations to remove noisy pixels and fill in the blobs' holes. Morphological opening is first applied to the image, which erodes the image then dilates it to remove the small objects from the binary mask while preserving the shape and size of the large objects. Morphological closing is next applied to fill the small holes within the mask's objects by dilating the mask then eroding it. Finally, a flood-fill operation is performed to fill the background pixels that cannot be reached within the object by filling in the background from the image's edge. This operation will help reduce false positives and eliminate noisy pixels.

2) Blob Analysis: Since the connected regions of foreground pixels showing in the mask mostly correspond to moving objects, the algorithm performs blob analysis to detect and identify all connected components and compute their characteristics, including area, centroid, and bounding box. Then, rectangles will be drawn on the objects using the bounding boxes to highlight the frame's moving objects. Both centroids and areas of the objects will be used to identify and track objects across cameras and calculate their distance.

D. Motion Estimation

The motion estimation stage uses the Kalman filter to estimate each object's current state and future state in the video. In this system, Kalman filter is used to set predictions about the track locations and the probability of assigning a detection to a track. The process begins by creating an array of tracks to maintain the state of the tracked objects. Each track is a structure representing a moving object in the video. The track structure is designed to show the number of frames since the track was first detected, the total number of frames in which the track was detected, and the number of consecutive frames in which the track was not detected. The number of frames in which the track existed in or disappeared is necessary to distinguish noise from moving objects. Once the number of visible track frames increases above a threshold, the object will be displayed and assigned to a track using the Hungarian algorithm. On the contrary, the object will be excluded once the number of consecutive invisible track frames exceeds a threshold. In this case, the algorithm assumes that the object has left the field of view and deletes the track.

E. Contact Tracing

The contact tracing step is implemented when the blob analyzer detects more than one centroid, i.e., multiple objects are moving in the video. The x and y coordinates of the centroids are used to calculate the euclidean distance between the objects. Once the distance is calculated, it is compared to a threshold distance value to ensure social distancing between objects (people). Although the system is developed to detect the moving objects, the contact tracing is designed to store the last known centroid location of a once moving object to consider the scenario of when a person walks into the frame and sits somewhere. The distance is first calculated in pixels using the euclidean distance, then converted into centimeters by multiplying the calculated distance ($dist_{px}$) with a spatial calibration factor (C.Factor) using the formulas below. The calibration factor is obtained by measuring an object of known length in the real world (L_{cm}) then dividing it by the measured number of pixels in the bird's eye view image (L_{px}). A color-coded label is then added to each object bounding box in the frame along with a line identifying the object's ID and connecting it to other objects in the frame. The color of the label is based on the distance between objects. If the distance exceeds a predefined safety threshold, object labels become red, indicating a possible contact between them; otherwise, the labels will be yellow.

$$C.Factor = \frac{L_{cm}}{L_{px}}$$

$$dist_{cm(i,j)} = dist_{px} * C.Factor$$

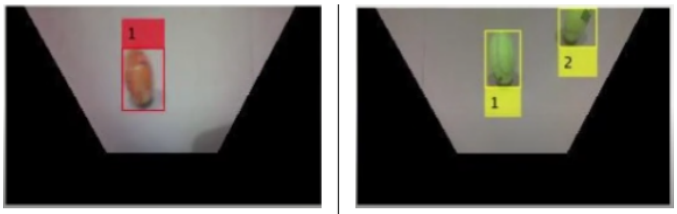
F. Object Tracking

Object tracking in a single camera's scope relies on creating tracks to follow moving objects' status within a frame. Typically, once an object leaves the first camera's scope and enters the second camera's scope, a new label is generated

and given to the object, making tracking across the cameras difficult. The proposed algorithm is designed to allow object tracking across multiple cameras by detecting the entering and exiting objects in a frame. A predefined frame margin *m* is set to represent the black left and right regions in the image frame where the image does not exist in the cameras. The algorithm continuously records the first and the last *x* and *y* coordinates (x_i, y_i) and (x_{i+1}, y_{i+1}) of the object's centroid along with the estimated area of the object's bounding box and the label of the object. If the *x* position of the centroid in the first camera *c1* is less than *m*, the object is said to be exiting the scope of *c1*. Similarly, if the object's centroid *x* position subtracted from the width of the image is less than *m*, the object is said to be entering the scope of *c2*. Once a new object enters the scope of *c2*, the area of the object's bounding box is compared against the area of the exited object's bounding box. In case the area was the same, the exited object's data is saved into the data of the new object in the frame of *c2*. Accordingly, the object will maintain the same label across the two scenes. On the other hand, if the object entered from a different location, the algorithm will generate a new label for the new object that does not match any of the cameras' previously given labels.

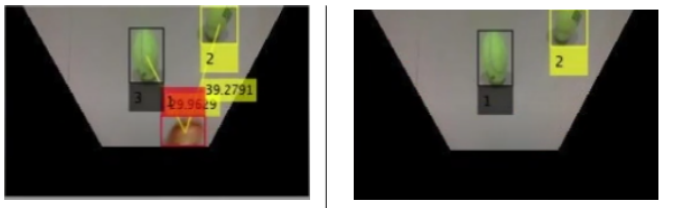
III. Results and Discussion

The system was tested with different hyper-parameters settings on a scenario where one moving object exists in *c1* scope, and two objects exist in *c2* scope to verify the feasibility and effectiveness of the proposed method. Bird's eye view transformation was applied, and the newly generated videos were used to test the proposed algorithm. For testing purposes, the three balls represent people and have different labels, a red label represents an infected person, a yellow label represents a healthy person, and a black label represents the testing needed class due to social distancing violation. Distance measurements are also visible on the screen to ascertain the violation. Initially, one infected object exists in *c1* scope, and two healthy objects maintaining social distancing exist in *c2* scope, as shown in Fig. 4. The scenario starts with the infected object of ID 1 exiting the scope of *c1* to the scope of *c2*. As the object leaves the first camera's scope and enters the second one's scope, their labels are also copied with them. When the object with ID 1 approaches the object with ID 3 and violates the social distance, object three is marked for testing, and the bounding box changes color to black for visualization purposes, as shown in Fig. 5. The object with ID 2 remains unaffected due to maintaining social distancing with objects 1 and 3. The system was also tested on a third scenario using a surveillance camera video available in Matlab. As shown in Fig. 6: Left, two individuals maintained social distancing, and accordingly, the system gave them a yellow label. Later in the video, the distance between them decreases to become less than the threshold defined, thus leading into marking both individuals for testing, as illustrated in Fig. 6: Right.



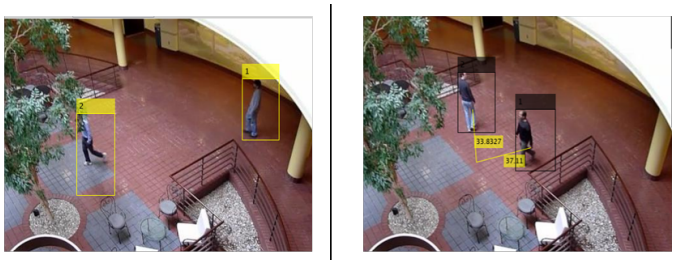
Left: Camera 1 view Right: Camera 2 view

Fig. 4. System output for scenario 1



Left: Social distancing violated Right: Infected object leaves the view

Fig. 5. System output for scenario 2



Left: 2 individuals maintaining social distancing

Right: 2 individuals violating social distancing

Fig. 6. System output for scenario 3

IV. Conclusion

The work proposed in this paper is a computer vision-based algorithm for contact tracing using stationary surveillance cameras. The algorithm begins by converting the input video into a bird's eye view where all moving objects are detected, and the distances between them are calculated. The algorithm performs background subtraction to isolate foreground objects, then morphological operations to remove the noise. Next, it conducts blob analysis to identify the connected regions in the resulting foreground video and applies Kalman filters to estimate objects' motion in the video. Eventually, it calculates Euclidean distance between the objects to trace object contacts. This algorithm allows identifying, assessing, and managing people who might have been exposed to the disease through contact transmission. The system was able to detect the objects in the input video frame and estimate the distance between them across multiple cameras. The results obtained verify the feasibility and effectiveness of the proposed algorithm.

REFERENCES

[1] (2020) Infectious diseases. [Online]. Available: https://www.mayoclinic.org/diseases-conditions/infectious-diseases/symptoms-causes/syc-20351173

[2] K. Song, S. Jiao, Q. Zhu, and H. Wu, "A proactive and practical covid-19 testing strategy," *IEEE Engineering Management Review*, pp. 1–1, 2020.

[3] E. Hernández-Orallo, P. Manzoni, C. T. Calafate, and J. Cano, "Evaluating how smartphone contact tracing technology can reduce the spread of infectious diseases: The case of covid-19," *IEEE Access*, vol. 8, pp. 99 083–99 097, 2020.

[4] J. Berglund, "Tracking covid-19: There's an app for that," *IEEE Pulse*, vol. 11, no. 4, pp. 14–17, 2020.

[5] W. H. Organization, "Contact tracing in the context of covid-19," May 2020.

[6] A. Morar *et al.*, "A comprehensive survey of indoor localization methods based on computer vision," *Sensors*, 2020.

[7] Q. Zhoua and J. Aggarwalb, "Object tracking in an outdoor environment using fusion of features and cameras," *Elsevier*, vol. 24, p. 1244–1255, 2006.

[8] Y. Sun *et al.*, "Device-free human localization using panoramic camera and indoor map," 2016.

[9] C. S. Sanoj, N. Vijayaraj, and D. Rajalakshmi, "Vision approach of human detection and tracking using focus tracing analysis," in *2013 International Conference on Information Communication and Embedded Systems (ICICES)*, 2013, pp. 64–68.

[10] F. Wang, X. Ren, and Z.Liu, "A robust blob recognition and tracking method in vision-based multi-touch technique," in *2008 IEEE International Symposium on Parallel and Distributed Processing with Applications*, 2008, pp. 971–974.

[11] E. Monier, P. Wilhelm, and U. Rückert, "A computer vision based tracking system for indoor team sports," in *The fourth International Conference on Intelligent Computing and Information Systems*, Cairo, Egypt, 19 - 22 Mar. 2009.