Measuring and Visualizing Social Distancing Using Deep Learning and 3D Computer Vision

Bilal AbdulRahman[1], **Zhigang Zhu**[1,2]
[1] The CUNY Graduate Center, New York, NY, USA
[2] The CUNY City College, New York, NY, USA
babdulrahman@gradcenter.cuny.edu, zzhu@ccny.cuny.edu

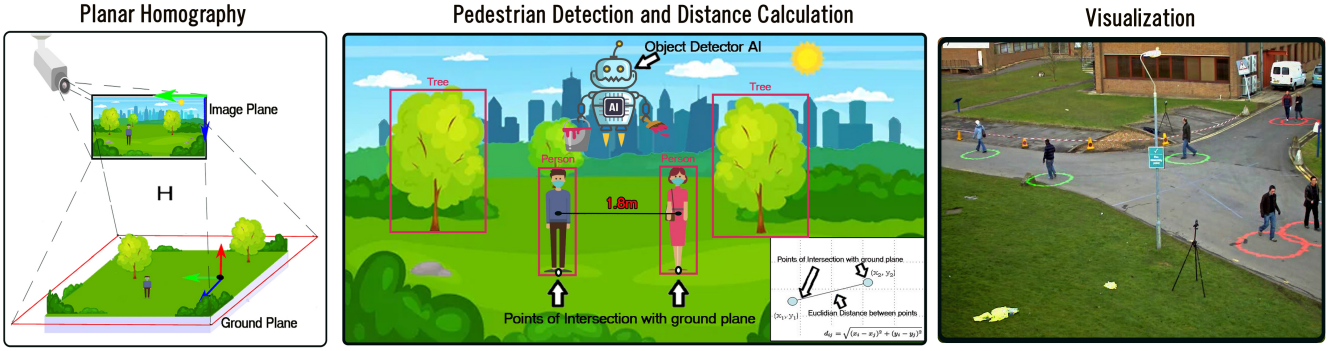


Figure 1: Components in measuring and visualizing social distance: planar homography, pedestrian detection, distance calculation, and visualization.

Abstract

Social Distancing has proved a necessary measure in controlling the spread of Coronavirus. The CDC (Center for disease control and prevention) in the United States recommends 6 feet as a safe distance between individuals. Therefore, a surveillance system capable of measuring distances between individuals can prove beneficial in limiting the spread. Video surveillance systems have already been introduced in various fields of our daily life to enhance security and protect individuals and sensitive infrastructure. In this work we present a way to use the existing video surveillance infrastructure to measure and monitor social distancing. We present a practical approach to monitor this distance using artificial intelligence and projective geometry techniques. Our approach, after initial setup, works in real-time requiring only a monocular surveillance camera feed. The proposed approach utilizes YOLO v4 neural network object detector for detecting pedestrians in the camera's view. Projective transformation is used to localize the pedestrians on the ground. Finally, the real world distances between pedestrians is calculated and visualized with the right perspective and occlusion relations as if the distance marks are actually on the ground. All the implementation is in real-time, and is performed on python using the OPENCV libraries and the YOLO v4 neural net with pretrained weights. Experimental results are provided to validate our approach. The code of this work will be made publicly available at GitHub upon acceptance.

AAAI Fall 2020 Symposium on AI for Social Good.

Introduction

The world is amidst a global pandemic. Coronavirus is continuing its spread across the world, with more than four million confirmed cases in 188 countries. At the time of starting this work, more than 300,000 people had lost their lives to the virus(CDC July,2020). Governments around the world are scrambling to come up with ways to contain the virus and limit the spread. Visual surveillance systems, which are common in urban environments, aim at providing safety in everyday life. High quality surveillance cameras are already present in most busy streets and departmental stores. Although they were initially commissioned for security surveillance, they can be re-purposed to help limit the spread of COVID-19. The CDC (Centers for disease control and prevention) recommends 6 feet as a safe distance between individuals to limit the spread (CDC July,2020). In this paper we present a practical way to monitor this dis-

tance. Our approach goes through four steps (Figure 1): (1) Planar homograph estimation; (2) pedestrian detection; (3) distance calculation; and (4) social distancing visualization.

For achieving the goal of a practical measuring and visualization for social distancing, well-studied computer vision techniques such as planar homography and object detection using neural networks are employed. As the first step (the initialization), the homography or projective transformation is computed between the ground plane in the real world and ground plane as it is represented in the image plane of a camera. Homography is required to be computed only once and can be done before hand. We will use the estimated homograph of the ground plane along with the point of intersection of a pedestrian with the ground plane to localize the pedestrian. Then in the second step, the YOLO v4 neural network object detector(Bochkovskiy, Wang, and Liao 2020) is used for detection pedestrians on surveillance video feed. In the third step, the point of intersection of each pedestrian with the ground plane is then estimated and localized using the pre-calculated homography, and the distance between any two of them are calculated. Finally, in the fourth step, the distances between pedestrians are visualized with the right perspective and occlusion relations using a simple Augmented Reality composition of the real scene background, the social distance marks on the ground, and the images of the detected pedestrians, as if the marks are actually on the ground.

Since the only computationally expensive task is the object detection, which is a real-time implementation of the YOLO object detector, this approach can be run in real time. Furthermore, this approach only requires 4 points of correspondence on the ground visible in the video feed for estimating the planar homography, which can easily be sourced from online maps (if the view is outdoors) or floor plans (if the view is indoors). Therefore, after the initial setup step, this approach will work requiring only the video feed (steps two to four).

The results of the applied social distancing approach will be presented using the camera database from the Performance Evaluation of Tracking and Surveillance Challenges (PETS) in the year 2009 (Ferryman and Shahrokni 2009). All the implementation is performed on python using the OPENCV libraries and the YOLO v4 pretrained neural net. The results show reasonable amount of success, and the code of this work will be made publicly available at GitHub upon acceptance.

The paper is organized as the following. Section discusses related work on social distancing monitoring using AI and computer vision. After this, the four steps of our approach are detailed in four sections: Section discusses the calculation of the ground plan homography; Section describes the real-time pedestrain detection step using the YoLo v4 neural net; Section details the procedures of pedestrian localization using the piercing point of each detected person on the planar ground, and distance calculation between any two of them; and Section presents a number of ways to visualize the social distancing measurement results and our final choice. Then Section provides our experimental results on the PETs dataset, with a video supplement, and finally in Section we conclude our work with some discussions of future work.

Related work

Due to the importance of social distancing measures, a number of works have appeared in the last few months. Similar to our work, a social distancing monitoring approach (Punn, Sonbhadra, and Agarwal 2020) that utilizes YOLOv3 and Deepsort on CCTV footage was proposed to detect and track pedestrians followed by calculating a violation index for non-social distancing behaviors. The way this approach estimates the position of pedestrians is unclear as there is no implementation discussion on this part and the work mostly focuses on the deep sort algorithm. Another work(Cristani et al. 2020) defines social distancing monitoring as a visual social distancing problem. This work introduces a skeleton for a detection based approach to inter-personal distance measuring. It also discusses the effect of social context on people's social distancing and raises privacy concerns.

Several prototypes utilizing machine learning and AI along with cameras and sensors have been developed as commercial solutions for social distancing monitoring and enforcing. Landing AI (13 June,2020) has proposed a social distancing detector using a surveillance camera to highlight social distancing violations. The work in (Prateek Khandelwal 2020) details a systems that was deployed in a manufacturing plant to monitor worker activity. This type of implementation requires knowledge of a lot more of camera parameters and complex transformations compared to our work.

In addition to surveillance cameras, LiDAR based (14 July,2020) and stereo camera based (16 June,2020) systems are also commercially deployed. These approaches require special hardware made for the specific application. A very recent work(Yang et al. 2020) proposes a non-intrusive warning system with softer omni-directional audio-visual cues. It evaluates critical social density and modulates inflow into a region-of-interest. The approach in its implementation is very similar to ours. Although ours further focuses on a better visual representation using projective geometry.

Planar Homography Estimation

2D projective geometry is the study of properties of the projective plane P_2 that are invariant under a group of transformations known as projectivities (Hartley and Zisserman 2003). A projectivity is also called a homography. Hence, homography is a special case of projective geometry. It enables the mapping of points in spaces with different dimensionality. A point x' observed in a view (such as an image) can be mapped into its corresponding point x in another perspective or coordinate system (such as a ground plane) (Figure 2). Planar homography is a linear transformation on 3 homogeneous vectors represented by a non-singular 3x3 matrix H (up to a scale factor):

$$s \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = H \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} \quad (1)$$

where (x_1,x_2,x_3) are the homogeneous coordinates on the ground plane and (x'_1,x'_2,x'_3) are the homogeneous coordinates on the image, s is the scale factor and H is the homography matrix with 8 DoF (degrees of freedom) as it is estimated up to a scale, and H is a homogeneous matrix. Due to the scale factor, only the ratio of the matrix elements is significant. Lending to its homogeneous nature, It is generally normalized (Aghajan and Cavallaro 2009) with

h_{33}=1

or

h_{11}+h_{12}+h_{13}+h_{21}+h_{22}+h_{23}+h_{31}+h_{32}+h_{33}=1.

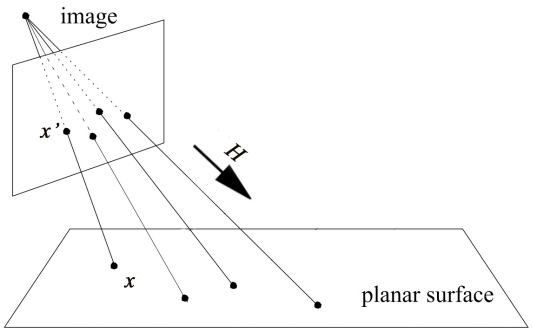


Figure 2: A visual representation of homography and the corresponding points

We can use this property to localize the pedestrians in view and calculate their locations on the ground with respect to each other. For this purpose, we need to find the homography matrix between ground plane in real world coordinates and ground plane as it is represented in the image plane. This can be achieved by finding four corresponding points between the two planes and using them to find the matrix elements. Let the inhomogeneous coordinates of a pair of matching points x and x' in the world and image plane be $(x, y) = (x_1/x_3, x_2/x_3)$ and $(x' , y') = (x'_1/x'_3, x'_2/x'_3)$, respectively. The projective transformation of these points can be written in inhomogeneous form as:

$$x = \frac{x_1}{x_3} = \frac{h_{11}x' + h_{12}y' + h_{13}}{h_{31}x' + h_{32}y' + h_{33}} \quad (2)$$

$$y = \frac{x_2}{x_3} = \frac{h_{21}x' + h_{22}y' + h_{23}}{h_{31}x' + h_{32}y' + h_{33}} \quad (3)$$

Each point correspondence generates two equations for the elements of H, which after multiplying out are:

$$x(h_{31}x' + h_{32}y' + h_{33}) = h_{11}x' + h_{12}y' + h_{13} \quad (4)$$

$$y(h_{31}x' + h_{32}y' + h_{33}) = h_{21}x' + h_{22}y' + h_{23} \quad (5)$$

These equations are linear in the elements of H. Four point correspondences lead to eight such linear equations in the entries of H, which are sufficient to solve for H up to an insignificant multiplicative factor. In other words, we only need to identify four points on an image with corresponding four points on the ground plane with known locations (i.e., relative locations/distances in feet) (Figure 2). The only restriction is that the four points must be in "general position", which means that no three points are col-linear. One thing to note here is the computation of the Homography matrix H in this way does not require knowledge of any of the camera's parameters or the pose of the plane (Hartley and Zisserman 2003).

Now that it is possible to compute point correspondences from the 2D space to the 3D world and vice versa, it is also possible to determine the number of pedestrians and their exact (relative) location in a scene. This further requires finding the point of intersection of the pedestrians with the ground plane which is achieved in the following steps.

Real-time Pedestrian Detection Using YOLO

Object detection is the process of locating where certain objects are in an image and then correctly classifying those objects. Yolo is a fast, accurate neural network for object detection (Redmon et al. 2016). Yolo network architecture is like a FCNN (fully convolutional neural network) . The architecture splits the input image into grids and for each grid generates bounding boxes and class probabilities for those bounding boxes. This type of model has several advantages over traditional classifier-based systems. It looks at the whole image at test time, so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike systems like R-CNN which require thousands of passes for a single image. This makes it extremely fast, more than 1000x faster than R-CNN and 100x faster than Fast R-CNN. (Redmon et al. 2016).

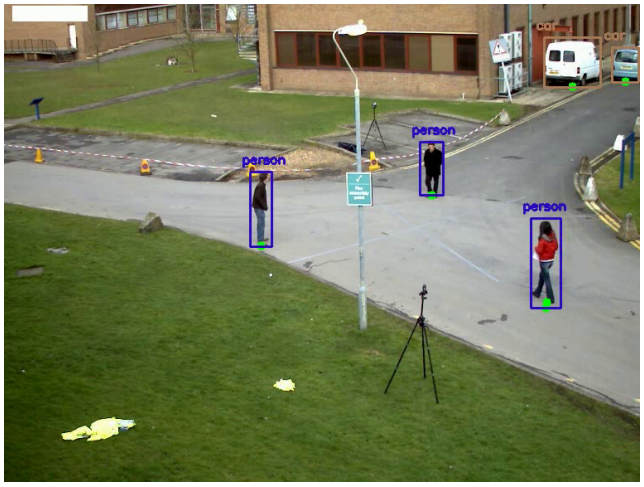


Figure 3: Bounding boxes drawn on the original frame from the result of the YOLO neural net

In this work, The fourth revision of Yolo (Yolo v4) is

used to detect pedestrians in the video feed as it is faster and more accurate than previous generations(Bochkovskiy, Wang, and Liao 2020). Each frame is forwarded to the Yolo neural net as input and it outputs predictions with confidence levels as to what objects are present in the frame. The locations of these predictions are returned as a center point ($c_{x'}$,$c_{y'}$) and height (h_b) and width (w_b) of an enclosing bounding box . All predictions below a 50 percent confidence level threshold are rejected. Finally, using the coordinates provided by the neural net, bounded boxes are drawn on the image frame, after non maximal suppression, using software. (Figure 3).

Pedestrian Localization and Distance Calculation

As previously discussed in Section , we need to find the piercing point of the pedestrians with the ground plane in order to localize them on the ground using homography. We can know the location of each pedestrian in the video feed by filtering out the "person" class from the predictions of the YOLO nueral net.

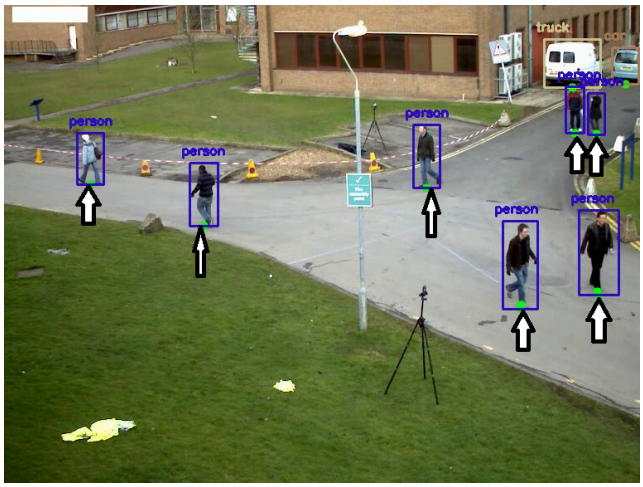


Figure 4: Arrows indicating the estimated point of intersection

To estimate the piercing point with the ground plane we apply some heuristics: the pedestrian is standing vertically and whole body of the pedestrian is visible in the video feed and is detected by the object detector. We then use the center point of the bottom side of the bounding rectangle returned by the YOLO object detector as a estimation of the piercing point for the pedestrian (As shown in Figure 4).This point is either between or on the pedestrian's feet and parallel to the pedestrians head, so is a good estimation of the pedestrian's location with respect to the ground plane. The piercing point ($p_{x'}$,$p_{y'}$) will have the same x' coordinate as the center point of the bounding rectangle returned by the Yolo nueral net. The y' coordinate can be calculated by subtracting the distance from the midpoint to the bottom side of the bounding rectangle to the y' coordinate of the center point ($c_{y'}$). Therefore:

$$(p_{x'}, p_{y'}) = (c_{x'}, c_{y'} - \frac{h_b}{2}) \tag{6}$$

Where h_b is the height of the bounding box. Finally, these coordinates are transformed into real world coordinates (p_x,p_y) using the homography matrix H calculated in Section .

The knowledge of the coordinates of all pedestrians in the image with respect to the ground plane in the real world reduces the problem to calculating distance between two points in 2D plane. Regardless the coordinate system selected on the ground plane, we can find the distance between any two pedestrians by calculating the Euclidean distance(Szabo 2015) between their points of intersections on the ground plane as

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{7}$$

where d_{ij} is the distance between any two pedestrians having piercing points (x_i, y_i) and (x_j,y_j) on the ground.

Social Distancing Visualization

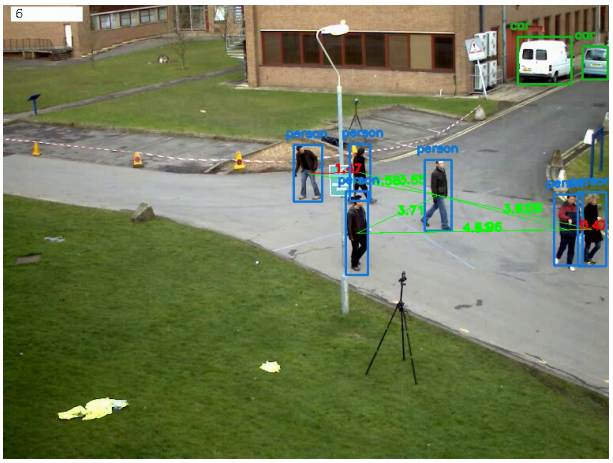
The distances need to be represented in an intuitive way that can be easily interpreted and is not overwhelming to the viewer. Initially, the raw representation was adding green distance lines between pedestrians if the distance was less than 5 meters (16.4 feet) and displaying the distance on top of the lines. These lines would turn red if the distance went below 2 meters or roughly 6 feet (Figure 5a).This representation becomes overwhelming and confusing if there are many pedestrians in view (Figure 5b).

For an image-based visualization, the best representation we could come up with was green circles of a 3-foot radius around each pedestrian. When any two circles intersect it can be interpreted as the two pedestrians are closer than 6 feet and the circles turn red alerting the viewer. Now the challenge faced implementing this was drawing circles in real time that would correctly reflect the distance in the real world on the image. As drawing circles of fixed size would not show the distance accurately. Also, to make it work in real time transforming each point of a circle before drawing it on the image was not computationally viable.

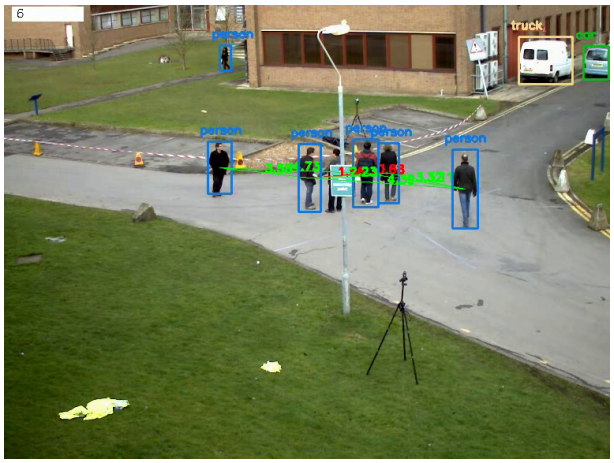
Therefore, what we ended up creating was a separate blank image which represented a scaled top down view of the real-world ground plane. This blank image would act as a buffer between the ground plane and image. The reasons for scaling the blank image were twofold: so that all of the ground plane coordinates visible in the camera feed were represented as positive pixel values on the blank image and the blank image size would not be excessive. The circles were then drawn on this blank image (Figure 6) and transformed using homography and overlayed on to the original image using the following relations.

A point P_i on the image of the ground plane could be represented as:

$$P_i = H_{ip}.P_p \tag{8}$$



(a)



(b)

Figure 5: Social distancing initial representation with bounding boxes and lines: (a) Bounding boxes for pedestrians and lines between them in green (greater than 6 feet) and red (less than 6 feet). (b) Dense crowds may overwhelm the viewer in this kind of view

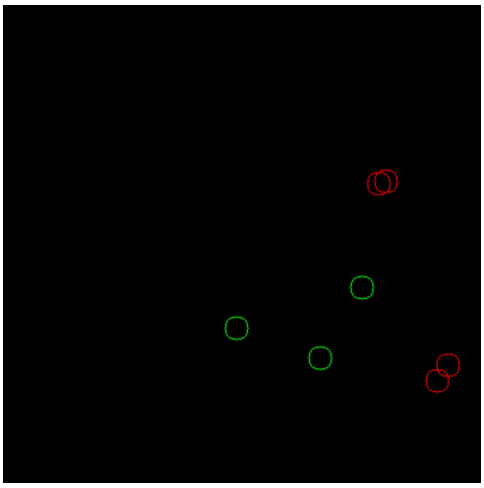


Figure 6: Scaled blank image top down view

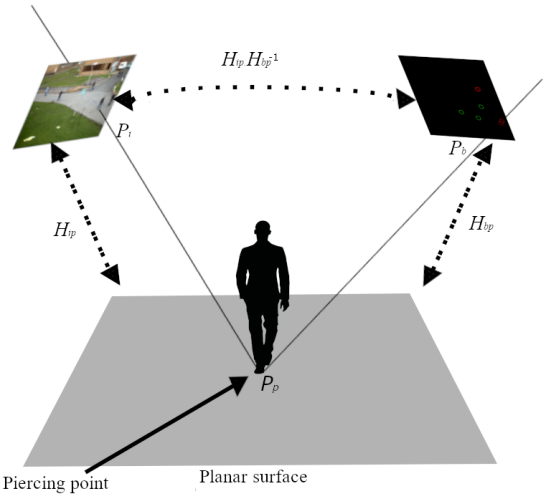


Figure 7: The homography between two images induced by a world plane (the concatenation of two homography matrices).

where P_p is the point on the ground plane in real world coordinates and H_{ip} is the projective transformation or homography between the two planes. The same point on the blank image P_b can be represented as

$$P_b = H_{bp}.P_p \tag{9}$$

where P_b is the point on the blank image and H_{bp} is the projective tranformation between real world ground plane and blank image (Figure 7).

Now Equation (9) can be written as:

$$P_p = H_{bp}^{-1}.P_b \tag{10}$$

Putting value of P_p in Equation (8) from Equation (10) we get:

$$P_i = H_{ip}.H_{bp}^{-1}.P_b \tag{11}$$

Thus, by using this relation each point in the blank image is mapped onto the camera frame. and since since the blank image is just a scaled representation of the real world coordinates, this resulted in circles which were correctly scaled (Figure 8b). Background subtraction (Figure 8a), which is the processing filtering out moving objects in a video, was used to overlay human contours on top of the circles in the original feed to make the circles look as if they are actually on the ground. The blank image in Figure 6 is also used in the result to represent the top down view of the ground plane.

(a) (b) (c)

Figure 8: Social distancing final representation. (a)Masks of the pedestrian's contours after background subtraction. (b)Social distancing enhanced visualization with circles around pedestrians. The image is a real-time composition of background, circles and pedestrian images (using the masks in (a)), showing the right perspectives, sizes and occlusions. (c)Top down view with the social distancing circles added to the corresponding satellite image.

Results

This work is implemented on the PETS 2009 s2, L1 View 001. We use a calibrated camera with calibration parameters already provided in the data set to find corresponding points for the purpose of computing homography. That being said, it is not necessary for the camera to be calibrated for this work to be implemented (Hartley and Zisserman 2003). To compute planar homography only 4 corresponding points between 2 planes are required as already explained in Section .

The application is run on a Intel core i7-8750H CPU and a Nvidia GTX 1050tiM GPU and 16GB RAM. The CCTV video feed has 7 fps while the YOLO v4 can process up to 11fps on this hardware. This results in smooth real-time operation.

The result of our work can be seen in the YouTube video here: https://youtu.be/4wrkB8QDJ_g. The video also has the initial raw representation of the bounding boxes with the distance lines followed by our refined presentation with circles. The blank image of the top down view has the satellite image of the scene added to it courtesy of google maps for visual appeal (Figure 8c).

Conclusion and Discussion

We have presented a way of monitoring social distancing using surveillance cameras. The pedestrian detecting part was performed using YOLO neural net and the localization, relative distance and realistic circular overlays are achieved using planar homography and image composition (background, circles, pedestrians). This work in its current form can be implemented for real world use. Nevertheless, the performance and tracking can be further enhanced if the following features are also implemented.

As of now, background subtraction is only used for visual appeal in this work. Ground plane piercing point detection can be made more accurate by using background subtraction to detect human contours inside the bounding boxes and then detect the feet from these contours. If object detection and tracking algorithm is used such as (Wang et al. 2019), it can be used for contact tracing and can also be used to implement a timer on the violation of social distancing. In the case that the feet of a person are not visible due to occlusion, we can use the average height of a person to roughly estimate the feet location from the location of the head. Furthermore, multiple camera angles can help deal with occlusions. Using a less computationally expensive object detection algorithm such as (Wojke, Bewley, and Paulus 2017) can enable the system to use lesser resources and thus can be run on lesser hardware.

Acknowledgments

This work is supported by a CUNY Graduate Center Scholarship. The second author is also supported by NSF via the Partnerships for Innovation Program (Award #1827505) and ODNI via the Intelligence Community Center for Academic Excellence (IC CAE) at Rutgers University (Awards #HHM402-19-1-0003 and #HHM402-18-1-0007).

References

July,2020. *Social Distance Monitoring*. Https://levelfivesupplies.com/social-distance-monitoring/.

June,2020. *Landing AI Creates an AI Tool to Help Customers Monitor Social Distancing in the Workplace*. Https://landing.ai/landing-ai-creates-an-ai-tool-to-help-customers-monitor-social-distancing-in-the-workplace.

June,2020. *Using 3D Cameras to Monitor Social Distancing*. Https://www.stereolabs.com/blog/using-3d-cameras-to-monitor-social-distancing/.

Aghajan, H.; and Cavallaro, A. 2009. *Multi-Camera Networks: Principles and Applications*. USA: Academic Press, Inc. ISBN 0123746337.

Bochkovskiy, A.; Wang, C.-Y.; and Liao, H.-Y. M. 2020. "YOLOv4: Optimal Speed and Accuracy of Object Detection". *ArXiv* 2004.10934.

CDC. July,2020. *Social Distancing, Quarantine, and Isolation*. Https://www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/social-distancing.html.

Cristani, M.; Del Bue, A.; Murino, V.; Setti, F.; and Vinciarelli, A. 2020. "The Visual Social Distancing Problem". *IEEE Access* PP. doi:10.1109/ACCESS.2020.3008370.

Ferryman, J.; and Shahrokni, A. 2009. "PETS2009: Dataset and challenge". In *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 1–6. doi:10.1109/PETS-WINTER.2009.5399556.

Hartley, R.; and Zisserman, A. 2003. *Multiple View Geometry in Computer Vision*. USA: Cambridge University Press, 2 edition. ISBN 0521540518.

Prateek Khandelwal, Anuj Khandelwal, S. A. D. T. N. X. A. R. 2020. "Using Computer Vision to enhance Safety of Workforce in Manufacturing in a Post COVID World". *ArXiv* 2005.05287.

Punn, N. S.; Sonbhadra, S. K.; and Agarwal, S. 2020. "Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques". *ArXiv* 2005.01385.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. "You Only Look Once: Unified, Real-Time Object Detection". In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788.

Szabo, F. 2015. *The Linear Algebra Survival Guide: Illustrated with Mathematica*. USA: Academic Press, Inc. ISBN 0124095208.

Wang, Z.; Zheng, L.; Liu, Y.; and Wang, S. 2019. "Towards Real-Time Multi-Object Tracking". *ArXiv* 1909.12605.

Wojke, N.; Bewley, A.; and Paulus, D. 2017. "Simple online and realtime tracking with a deep association metric". In *2017 IEEE International Conference on Image Processing (ICIP)*, 3645–3649.

Yang, D.; Yurtsever, E.; Renganathan, V.; Redmill, K. A.; and Ozguner, U. 2020. "A Vision-based Social Distancing and Critical Density Detection System for COVID-19". *ArXiv* 2007.03578.