Crowd Tracking Techniques for Evaluation of Adherence to Social Distancing Measures

Jacob Desman, Emily Chang, Andrew Cornelio, Sandy Shi

EN.601.461/661 Computer Vision - Final Report

1 Introduction

The discovery and subsequent rapid spread of COVID-19 has brought with it a variety of regulations aimed to prevent further transmission of the disease. One such guideline is social distancing: putting approximately 6 feet of distance between oneself and other people. This guideline is based on how far the virus could be transferred by airborne droplets and has been shown to be the most effective means of reducing COVID-19 transmission [1,2].

Unfortunately, in-person enforcement of social distancing guidelines on a broad scale is difficult and poses risk of infection to the monitors. As a result, regulatory bodies have instead turned to methods of gauging social distancing remotely, such as through bluetooth, radio-frequency identification (RFID) sensors, and, most notably, analyzing video feeds from network cameras. Many recent publications have combined methods in computer vision, machine learning, and artificial intelligence to calculate interpersonal distance [3-6]. In this project, we aim to explore and analyze different crowd tracking techniques as well as developing our own.

2 Project Goals

As stated in the proposal, our initial project goals were to first **establish a baseline functional system,** by replicating the results from recent preprint papers. As a first step, the method described in Ghodgaonkar, et al. (2020) was recreated, as it employs a relatively straightforward means of estimating interpersonal distance via bounding box geometries [3]. However, this method relies heavily on making approximations regarding size and positioning of persons within the scene, which in turn requires many detections per frame to work accurately.

In order to be more robust to varied scenes and to better encapsulate image geometry, we extended this simple baseline system. Here, we propose a multi-step pipeline (Fig. 1):

1. The user selects points corresponding to a square region for which an image to bird's eye view homography can be calculated
2. Deep learning person detection system automatically detects people's locations in image and transfers them onto the ground plane
3. From user-defined height of a person, system infers camera geometry and determines the camera angles to assist in approximating distances between people
4. Draw circles representing proper social distance on the bird's eye view reprojection of the scene

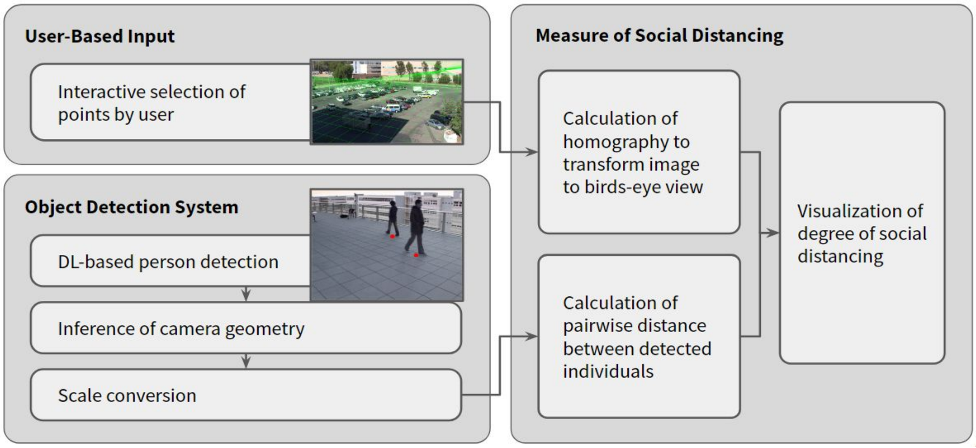
User-Based Input
Interactive selection of points by user
Object Detection System
DL-based person detection
Inference of camera geometry
Scale conversion
Measure of Social Distancing
Calculation of homography to transform image to birds-eye view
Calculation of pairwise distance between detected individuals
Visualization of degree of social distancing

Figure 1. Proposed pipeline for the social distancing tool.

3 Methods

For the purposes of evaluating and testing the efficacy of these methods, video data from the Video and Image Retrieval and Analysis Tool (VIRAT) dataset was used [7]. Besides being realistic and diverse, the VIRAT dataset also provides well-estimated "ground truth" homographies for each camera view. This allowed us to develop a system to calculate the homography in parallel with the rest of our pipeline and check the similarity of results.

3.1 User-Based Homography Selection

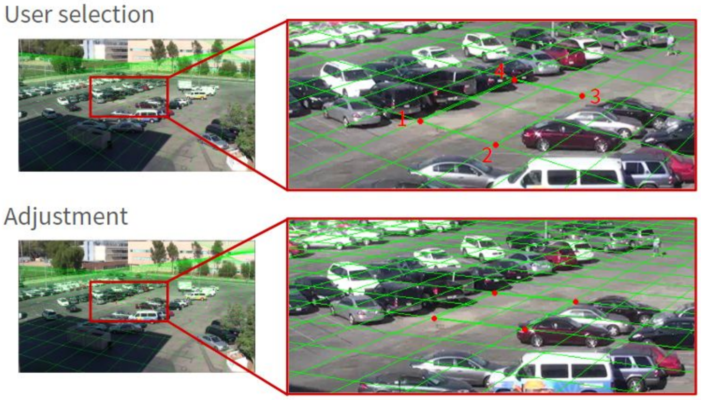

Figure 2. Example of user-based homography selection process. If the initial point selection does not produce a fitting homography, the user can adjust the points as need be.

Our pipeline begins with user-based fitting of the ground plane. The user selects four points corresponding to a square in world space (Fig. 2). OpenCV interfaces are used to allow user selection, drawing of the "ground plane," and adjusting of point positioning following initial selection. A homography is then calculated to transform the image space into the bird's eye view (see Fig. 4, left).

3.2 Person Detection

Deep learning is used to find bounding boxes corresponding to individuals in the scene. Two deep learning-based detection systems, YOLO and DETR, were evaluated for these purposes. YOLO, which stands for "You Only Look Once", is a widely used object detection model [8]. DETR, which stands for "DEtection TRansformer", is a more recent framework released by Facebook [9]. These models take in images and produce the coordinates of a bounding box around the people in the image. However, calculation of the ground plane requires a person's position on that plane, not a bounding box. To get this point, we used the center point of the bottom edge of the bounding box to approximate the people's feet positions. This assumption worked most of the time, although it failed if the person's lower body was occluded or when there were significant shadows, causing the bounding box to incorrectly represent the individual. The YOLO and DETR models were tested on a variety of scenes to see which was the most accurate and consistent, and DETR showed better performance on videos taken at close, medium, and far distances overall.

3.3 Inference of Camera Geometry

Our system uses a homography, which is defined up to scale. Therefore, we need to determine a scale factor to convert the units of the bird's eye view into feet. We do this by assuming the average height of the person and using simple projection to make assumptions about the dimensions of the scene.

Here, we make the simple assumption that the camera looks at the ground at a specified angle, and that angle does not vary between people despite the field of view. While this results in the best calibration for people in the center of the image, it serves as a principal approximation in our pipeline to simplify the problem. We then use OpenCV's decomposeHomographyMatrix function to determine the rotation matrix for the camera pose. However, the function requires an intrinsic camera matrix, which we do not require in advance to make the system generalizable to any video. Since we determine our own scale factor independent of camera properties, we set arbitrary focal length parameters and set the camera offsets to be the middle of the image. Once we have candidate rotations from the aforementioned function, we can further decompose those into the Euler angle about each axis using OpenCV's RQDecomp3x3 function. The decomposed rotation matrix is then used to determine the angle at which the camera is looking at the ground.

Using this rotation to infer scale, we first assume that the average height of a person is 5.4 feet tall [3]. This allows for simple projection of the head onto the ground space. By calculating the expected length in feet between the feet and the projection of the head onto the ground, and that same distance in the homography space, we can make a conversion factor that converts units in the bird's eye view into feet (Fig. 3). Users can theoretically choose their own scale factor, if they so wish. While these assumptions break down in certain cases which will be discussed later, our goal was to create a system which has more realistic assumptions than prior work, which merely estimates distance metrics.

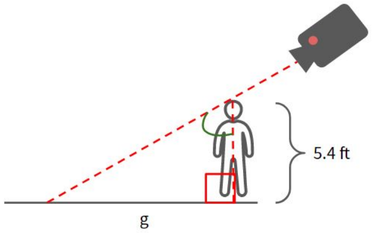

Figure 3. Diagram illustrating the method. Note that with two angles and a side known, the ground plane can easily be determined geometrically.

3.4 Determining Degree of Social Distancing

After a scale is applied to the bird's eye view, pairwise distances are calculated for each person to indicate social distancing violations between people. Circles are drawn around each individual's feet in the bird's eye view to visualize proximity, making it clear when people are nearby or violating distancing.

4 Results & Evaluation

Qualitatively, we are able to yield the best results for cameras that look at a scene from far away, or at angles closer to the bird's eye view. Since we assume a constant scaling factor, which is based on a single person, the approximation will be most accurate for people close to the center of the image. The bird's eye representation of the scene appears fairly accurate in such optimal regions (Fig. 4), but this breaks down in closer quarters. Code for Figure 4, provided as an example, can be found within our GitHub repository.

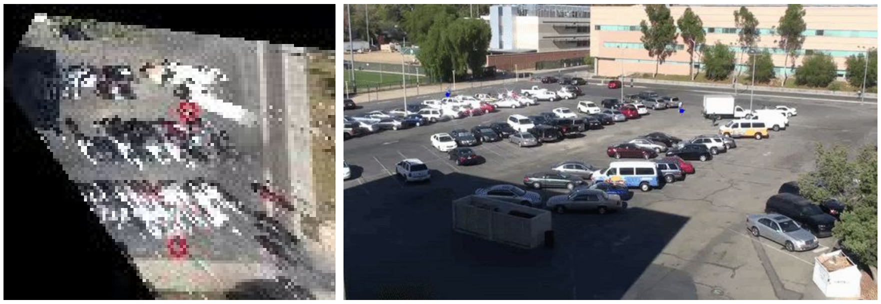

Figure 4. Results from our pipeline over a sample video from the VIRAT data set. Note the reasonable person detection and scaling of the image.

We also sought to determine how well the scaling factor was being determined. However, because it is difficult to find data sets with real, pairwise distance estimation or the depth information we need, our group instead turned to analysis of car lengths within frames of the VIRAT data set as an approximation (Fig. 5). Depending on the source, the average car length is between 14-16 feet; therefore, we expect distances in this range.

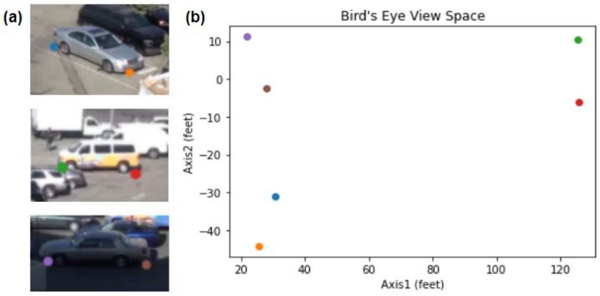

Figure 5. (a) Car lengths in feet, from top to bottom: 14.2 feet, 16.5 feet, 15.1 feet. **(b)** Representation of these cars in the bird's eye view space. Note that the axes are not even.

5 Discussion

5.1 Limitations

There are few limitations to this work. The first is detection of the ground plane. The pipeline is highly dependent on this, as inaccurate selections yield poor results. The next is the assumption of scaling. We assume a simple camera model where one "ray" is projected from the camera view. In reality, this assumption breaks down when considering that people further from the camera will have a ray at a slightly different angle, but we cannot compensate for this because we do not estimate the field of view. This provides an explanation for why our system works best with images that are far away: objects of interest are generally in the middle of the frame and the difference in the angles becomes less extreme. Additionally, it can be difficult to infer a realistic scale factor for people directly in front of the camera (Appendix Fig. 1). Therefore, inferring the field of view of the camera and computing a scale factor for a local region around a person could improve the robustness of our method.

5.2 Future Work

Deep learning-based approaches for constructing the bird's eye view homography have been explored in the past, involving automatic detection of vanishing points and lines [10]. Work like this would prevent the need for user homography estimation. Furthermore, the aforementioned method accounted for the field of view of the camera, which we do not. Other work has explored depth information based on a similar image, thereby removing the need for our camera reconstruction process [11]. Other work, which we have touched upon before, has also used the information gathered from detection and tracking to create density maps indicating where the highest concentration of social distancing infractions occur [6].

5.3 Main Takeaways

We have three main takeaways. First, especially considering how complex our project was, we realized the importance of setting up a clear guide/timeline for the project and trying to break down your goal into a few steps that you know how to tackle. Second, we learned that simple principles like homography transformations can produce useful real world applications even though they are relatively simple. For example, we used a homography transformation to turn a camera image to a bird's eye view image, which we used to calculate real world distances. Finally, we realized real world projects involve the integration of many separate components to create a powerful end product.

5.4 Course Feedback

When researching methods for use in our project, we found that deep learning played an integral role in many approaches to people detection methods and other relevant computer vision research. Though we ended up obtaining a good balance between deep learning and computer vision methods, perhaps learning about an applied approach to deep learning earlier in the course would be beneficial to have as background knowledge before going into the project.

6 Source Code

The source code for this project can be found at https://github.com/endernac/CV-Final-Project.

References

[1] Rubin, D., Huang, J., and Fisher, B.T., "Association of Social Distancing, Population Density, and Temperature With the Instantaneous Reproduction Number of SARS-CoV-2 in Counties Across the United States," *JAMA Network Open* 2020; 3(7):e2016099. doi:10.1001/jamanetworkopen.2020.16099

[2] Qureshi, Z., Jones, N., Temple, R., Larwood, J.P.J., Greenhalgh, T., and Bourouiba, L., "What is the evidence to support the 2-metre social distancing rule to reduce COVID-19 transmission?," *The Centre for Evidence-Based Medicine*, 2020.

[3] Ghodgaonkar I., Chakraborty, S., Banna, V., Allcroft, S., Metwaly, M., Bordwell, F., Kimura, K., Zhao, X., Goel, A., Tung, ., Chinnakotla, A., Xue, N., Lu, Y.H., Ward, M.D., Zakharov, W., Ebert, D.S., Barbarash, D.M., and Thiruvathukal, G.K.,, "Analyzing Worldwide Social Distancing through Large-Scale Computer Vision," arXiv:2008.12363v1, 2020.

[4] Aghaei, M., Bustreo, M., Wang, Y., Bailo, G., Morerio, P., and Del Bue, A., "Single Image Human Proxemics Estimation for Visual Social Distancing," arXiv:2011.02018v2, 2020.

[5] Cristani, M., Del Bue, A., Murino, V., Setti, F., and Vinciarelli, A., "The Visual Social Distancing Problem," arXiv:2005.04813v1, 2020.

[6] Rezaei, M., and Azarmi, M., "DeepSOCIAL: Social Distancing Monitoring and Infection Risk Assessment in COVID-19 Pandemic," arXiv:2008.11672v3, 2020.

[7] "A Large-scale Benchmark Dataset for Event Recognition in Surveillance Video" by Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J.K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xiaoyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai, in Proceedings of IEEE Computer Vision and Pattern Recognition (CVPR), 2011.

[8] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A., "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* doi: 10.1109/CVPR.2016.91.

[9] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S., "End-to-End Object Detection with Transformers," arXiv:2005.12872v3, 2020.

[10] Abbas, A., and Zisserman, A., "A Geometric Approach to Obtain a Bird's Eye View From an Image," arXiv:1905.02231v2, 2020.

[11] Zhao, C., Sun, Q., Zhang, C., Tang, Y., and Qian, F., "Monocular Depth Estimation Based On Deep Learning: An Overview," arXiv:2003.06620v2, 2020.

Appendix

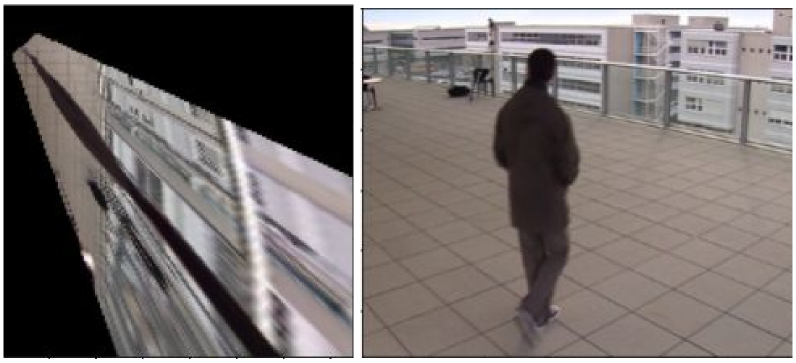

Figure 1. Results from a scene on a terrace. Note how close the camera is to the person of interest and the shallow angle of the camera. This results in a highly warped homography, making it difficult to calibrate based off of the head and foot position. This illustrates some of the shortcomings of the system.