Person Tracking and Distance Traveled Estimation Using Tracktor++

Benjamin Dion[1], Tyler Glenn[2], Saurabh Sinha[3]

***Abstract*— The goal of this project was to leverage a combination of computer vision topics to create a tool that estimates the distance traveled by persons in the field of view of a single camera. It uses a combination of tracking-by-detection, homography, and frame-frame distance calculator to estimate the distance traveled by each individual. For tracking-by-detection, the team used an open source Computer Vision and Pattern Recognition software available for public use known as Tracktor++. Tracktor++ leverages on a bounding box without any available training or optimization data, proving to be the ideal choice for this novel implementation where data isn't readily available to train the model. Using manual point-to-point correspondence for the first frame, a homography matrix was generated. By calculating the distance traveled by the bottom center of bounding boxes between each frame, the tool estimates the total pixel distance traveled. Then, using scale calculated manually by estimating a known object size in the video, the model scales the distance to a physical distance (feet, meter). Overall, this model successfully estimates the distance traveled based on visual assessment of the video. Our github repository is located here: https://github.com/tglenn28/EECS504Proj**

I. BACKGROUND AND IMPACT

Given amateur and local sports are not nearly as well funded, the idea was to introduce a low entry cost technology that would allow individuals to improve their performance and compete on a deeper level, while studying opportunities to improve safety through the data. The main motivation for this project was to enhance amateur sports and safety in various applications around the world. The inspiration came from major league sports, such as football, soccer, and basketball that offer next generation statistics play by play. The purpose of this project is to create a minimum viable product (MVP) to demo this using footage taken with a regular camera.

The long-term vision is to implement an app that allows any user with a smartphone or single camera setup to leverage computer vision and track their distance traveled as part of their health and fitness vision. The coach of a sports team may find this tool useful to track player performances, strategy, and other fitness metrics to improve training regiments and team strategy with a simple, one camera setup. This could be done in a positive fashion without any impact to privacy as no personal data needs to be collected.

An additional potential use of this system would be to use it on camera footage from webcams or security cameras in public areas. This could be useful for local governments to estimate foot traffic as a function of location or time of day to help make informed decisions regarding infrastructure updates. Also, during the COVID-19 pandemic, a slightly modified version of the algorithm could be used to score the public on social-distancing by measuring how far people stay away from each other.

[1]Benjamin Dion is a Master of Automotive Engineering student at University of Michigan, Ann Arbor, MI, USA bedion@umich.edu

[2]Tyler Glenn is a Master of Automotive Engineering student at University of Michigan, Ann Arbor, MI, USA tyglenn@umich.edu

[3]Saurabh Sinha is a Master of Automotive Engineering student at University of Michigan, Ann Arbor, MI, USA sinhasau@umich.edu

II. METHOD

A. PERSON TRACKING

In order to detect the people moving in a scene, the team ran the Tracktor++ algorithm developed by Bergmann, Meinhardt, and Leal-Taixe [1]. Tracktor++ is a tracking-by-detection algorithm that uses a two-step process to track multiple objects from frame-to-frame in a video. It uses a Faster-RCNN regression model to identify a person while simultaneously classifying them, and a separate detection model to determine if any new people have entered the frame. Tracktor++ outputs the data as a text file containing frame number, object tracking ID, the pixel location of the top-left corner of the object bounding box, and the bounding box dimensions.

B. TARGET POSITION ESTIMATION

To determine the distance traveled by each person during the scene, we simplified the problem from a 3D estimation problem into a 2D estimation problem. We do this by assuming that the bottom of the bounding box corresponds to where the person's feet touch the ground. We also assume that the ground is flat. With these assumptions, the next thing we need to solve the estimation problem is a projective transformation that maps pixel coordinates in the original image to a 2D coordinate system on the ground plane. By referencing visual content in the scene, we selected four points that corresponds to a rectangle on the ground plane and we estimated the aspect ratio of that rectangle. With these points, we were able to determine the homography matrix using the perspective transform function in OpenCV, which leverages the least squares error minimization shown below in eq. (1). Using this homography matrix, we were able to "flatten" the image, which gives us a clearly defined ground thereby making it easier to determine the distance traveled.

$$E_{LS} = \sum_i \|r_i\|^2 = \sum_i \left\| f\left(x_i; p\right) - x_i^{'} \right\|^2 \qquad (1)$$

C. DISTANCE CALCULATION

Once the homography matrix was determined, we found the pixel location of the bottom-center of the bounding box. By moving the pixel location in question from the top-left to the bottom-center, it gives us a more accurate idea of how far the person traveled with respect to the ground. Once the pixel location of the bottom-center was found, we applied the homography matrix to that pixel location to translate it from the original image to the ground plane coordinate system. These pixel locations could then be used to determine the total distance traveled in transformed image pixels (*pd*) using Eq. (2) to find the distance traveled from one frame to the next.

$$pd = \sum_{1}^{n} \sqrt{(x_n - x_{n-1})^2 + (y_n - y_{n-1})^2} \qquad (2)$$

In order to account for noise in the data and reduce error, we then applied a moving average filter to the transformed points. After this filter was applied, we could again use Eq. (2) to achieve much more accurate results. This gave us the total distance traveled by a person in pixels. To convert the pixel distance to real-world distance traveled, we needed to use an object with a known size (in meters), and generate a scale factor with respect to number of pixels that equated to that distance. Applying this scale factor, we were able to determine real world distance traveled in both meters and feet, respectively.

III. PROTOTYPE

During our initial implementation of Tracktor++, the team attempted to use multiple videos of amateur basketball players playing pickup basketball [2][3]. After conversion to images, these videos were only 720p and had frame rates lower than 30 Hz. Tracktor++ was able to track the players on the court, but had a very difficult time maintaining a consistent trace on each player. This resulted in Tracktor++ losing players in multiple frames, starting new track ID's on the same player, or switching a track ID from following one player to another when they crossed paths. These inconsistencies made distance estimation for each player nearly impossible, and due to Covid-19, creating our own ground truth became virtually impossible. So, the team decided to pivot and focus our prototype on a dataset that we knew worked with the Tracktor++ algorithm: the MOT17 Dataset Sequence 03 (MOT17-03) [4].

In order to determine the homography matrix needed to do the projective transformation for the MOT17-03 scene, we looked for four points that we knew created a known shape. On the sidewalk, there are multiple darker bricks that are rectangular. This gave us a good reference to help us determine if the homography matrix was transforming the scene in the correct way. If the dark bricks looked rectangular in the flattened image, we knew the transformation was successful. Since the scene in question used a single, static camera, we were able to generate the homography matrix once, and apply it across the entire scene. If the camera was moving, we would need to regenerate the homography matrix for every frame in order to get an accurate distance measurement.

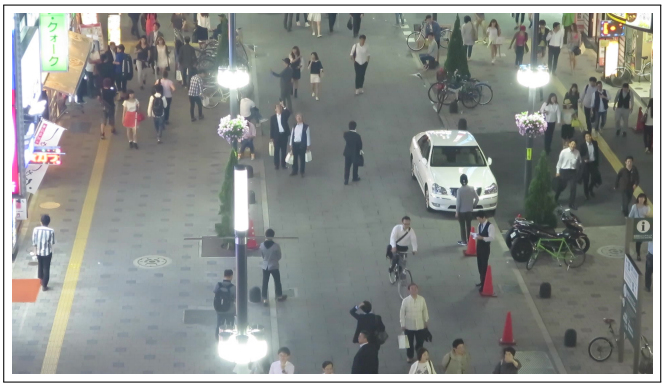

Fig. 1. MOT17-03 Frame 1

Another thing to note about our prototype is that we ran the scene offline. It is possible to run the prototype online.

The video we analyzed used a frame rate of 30 Hz, and was a high-quality 1920x1080 resolution image. The frame rate may not need to be that high, but we did find that the Tracktor++ algorithm had a hard time maintaining a continuous track on an individual if the image resolution was too low. For this reason, we highly recommend using an image quality of at least 1080p for any video analysis.

IV. RESULTS

Our prototype was run on the MOT17-03 dataset, which contains a scene of people walking in a street in the middle of a city block [4]. This scene was used in the development of the Tracktor++ algorithm, and was known to track people cleanly. The scene also had multiple objects with known sizes that could be used to determine scaling factor, and objects with known shapes that could be used to determine how well the homography transform performed. An example frame from the MOT17-03 dataset that has been run through the Tracktor++ object tracking algorithm can be seen in Fig. 2.

Upon running the relatively state-of-the-art algorithm on a video with the ground truth of one hundred and forty eight individuals, Tracktor++ detected 342 unique individuals. These extra detections range from multiple identifications for the same person to a improper detection of a construction cone as a moving object.

Upon performing the homography transform to the scene, we confirmed the homography looked right based on the rectangular bricks on the sidewalk (see Fig. 6). We then applied the homography to the pixel locations of each box and determined the distance traveled in pixels (Eq. 2). Using the known size of the manhole cover in the street (22-inches or 0.5588 meters), we were able to determine the scaling factor for the scene. Applying this scaling factor, we found the real-world distance traveled for every person detected by the Tracktor++ algorithm in the scene. An example of the path people took can be seen in Fig. 4. The distance traveled

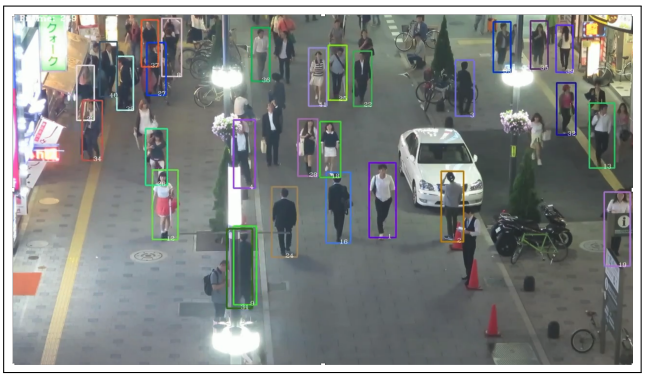

Fig. 2. Example of Tracktor++ output with bounding boxes

for 9 of the 342 observed people in the scene can be seen in Fig. 5.

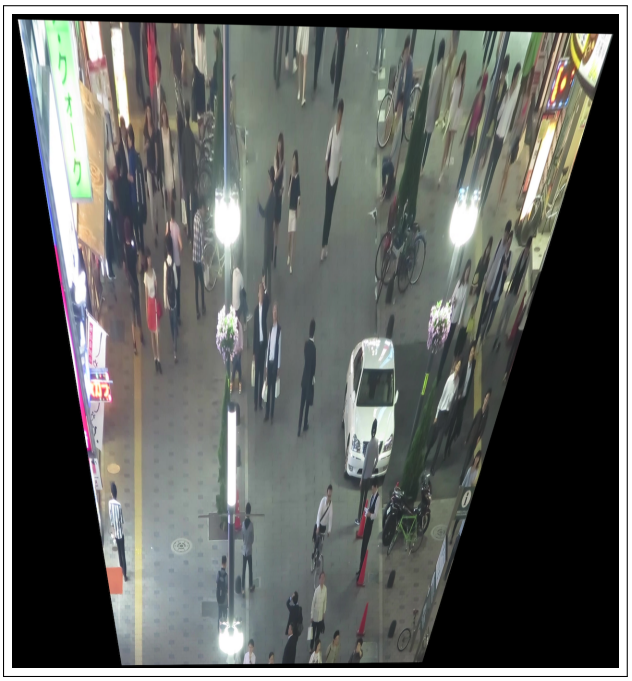

Fig. 3. Homography Result

V. CONCLUSION

In conclusion, our minimum viable product has proven to be successful in estimating the distance traveled by multiple individuals in a given frame if the tracking software used provides a stable, accurate trace.

Throughout the prototype development, we found a few system limitations that should be discussed. The Tracktor++ algorithm's accuracy is reduced significantly if the video is less than 30 frames per second and has an image quality lower than 1080p. Tracktor++ also struggles when multiple people are moving in the same direction and overlap for multiple frames. Sudden changes in direction can also confuse the algorithm. Additionally, Tracktor++ changes the size of the bounding box, depending on how the person moves their body. This size variation from frame-to-frame introduced

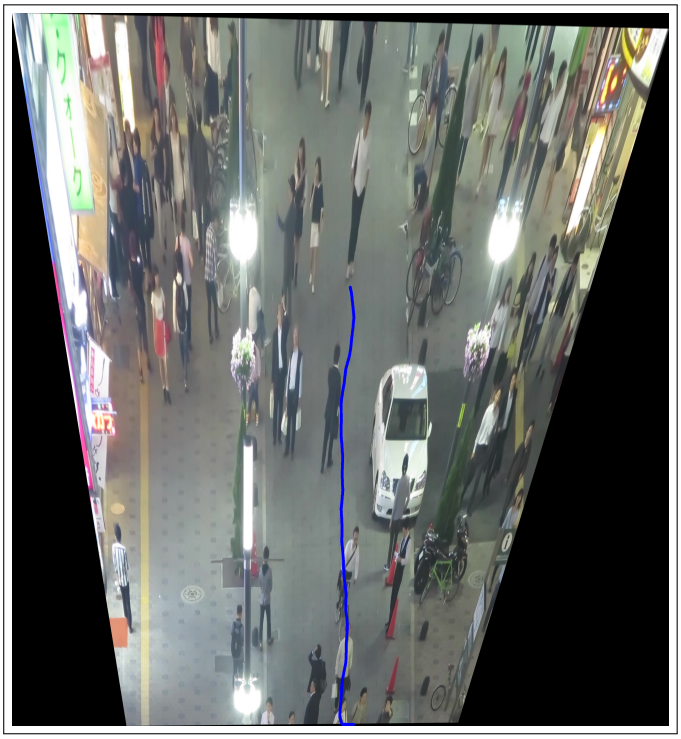

Fig. 4. Path Map for Person Track ID 1

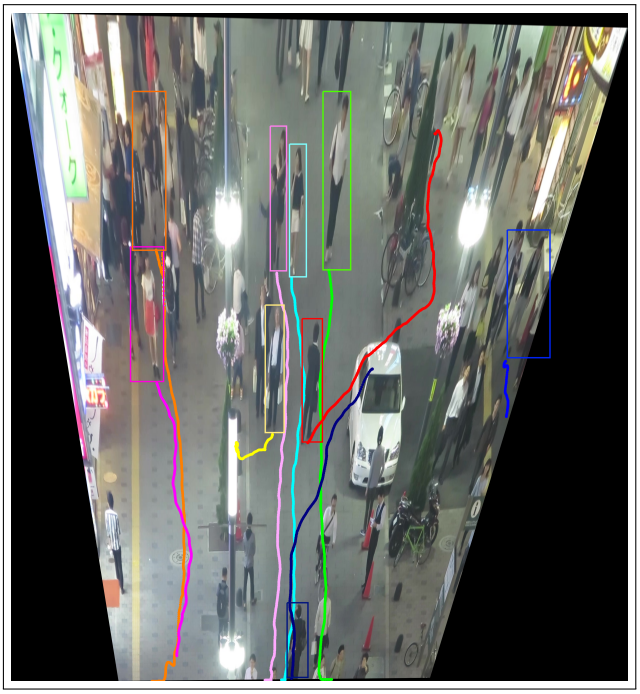

Fig. 5. Visual path representation nine select individuals

Object Tracking Number	Distance Traveled in Meters	Distance Traveled in Feet
1	2.1529	7.0638
2	14.7469	48.3846
3	12.9905	42.6290
4	17.6092	57.7758
6	3.2218	10.5707
8	9.8447	32.3005
10	14.2817	46.8583
21	14.5741	47.8178
31	11.6285	38.1531

Fig. 6. Distance Traveled by the nine select individuals in Fig. 5

additional noise into the distance calculation by moving the bottom-center of the bounding box drastically if someone raised their arm. To reduce the impact of this error, we implemented the moving average filter. While not completely eliminating the noise, it helped reduce it enough to provide feasible results. Our method also requires a reference frame of an object with known dimensions to properly convert distance traveled from pixel distance to physical distance in meters or feet.

VI. FUTURE WORK

Future work to improve the performance of the system includes continuous homography generation to allow for the system to work with a moving camera. In addition, the team would also try using different multi-object tracking algorithms to see if we can reduce some of the limitations discovered with Tracktor++. Eventually, once we find a stable enough system, we would develop an app with a user-friendly interface to allow for its insights to be used for commercial and industrial application. The app could then be further expanded to a multi-device sync application that would allow for stereo application to improve overall accuracy.

REFERENCES

[1] Bergmann, Philipp and Meinhardt, Tim and Leal-Taixé, Laura ”Tracking Without Bells and Whistles” The IEEE International Conference on Computer Vision (ICCV) October 2019

[2] thekaeashow. ”That was a NBA 3!! hahaha SkyCAM 5’s ”, https://www.youtube.com/watch?v=JdWtMV8vtxM

[3] jpresa. ”HGoPro Hero 5 Basketball test (raw) ” https://www.youtube.com/watch?v=LGXXpYEbdcA

[4] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “MOT16: A Benchmark for Multi-Object Tracking, Multiple Object Tracking Benchmark, May 2016.” https://motchallenge.net/data/MOT17/