Facial Recognition and Data Aggregation for Multi-Camera Person Tracking

E. N. Minor, A. Sankaralingam, S. Naidu, N. Mitchell, T. Krause, University of Colorado Boulder, Boulder, CO, 80309, USA;

ATLAS INSTITUTE

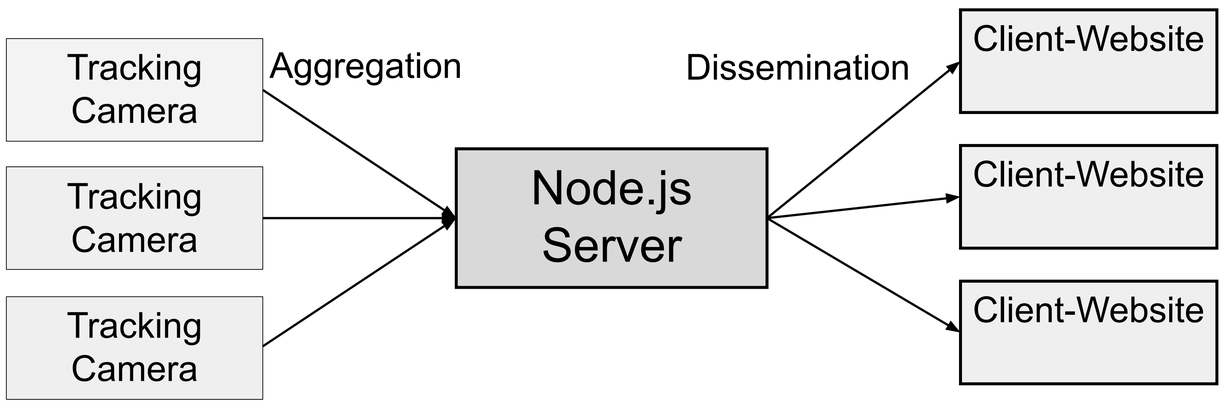
Abstract

Collaboration focused workspaces rely upon being able to quickly and easily access the expertise of your coworkers. Messaging apps allow for quick digital communication, but it is often more efficient to simply have a conversation in person. The system we developed allows for consenting individuals to train a webcam system to identify and track their face in order to create a map of everyone's location in a collaborative workspace.

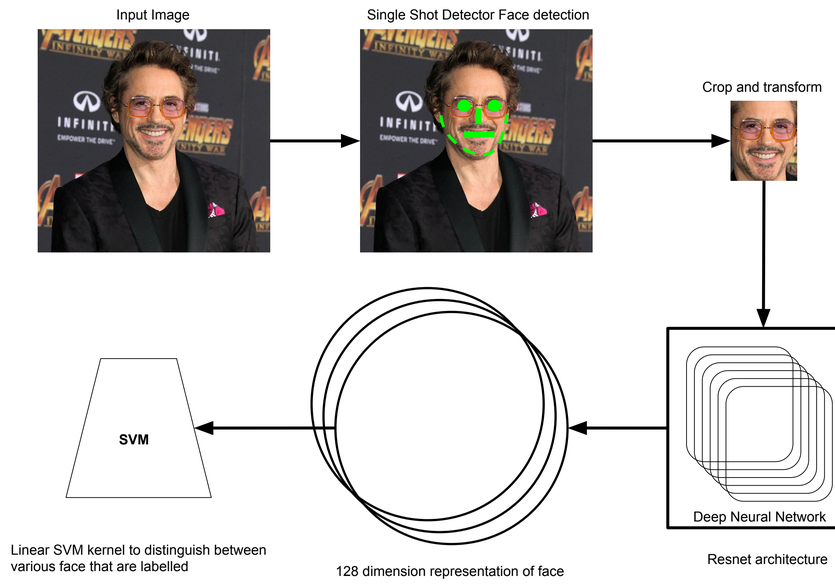
Using a simple interface, a python script records images of a person's face, extracts features, and uses transfer learning to create a recognizer. This can be done with multiple computers and cameras, which all connect via websockets to a Node.js server that aggregates the data and keeps a dictionary containing everyone's position. The node server also serves a website that displays a map of everyone's location.

Node.js as a Tool for Data Aggregation and Interpretation

In order to aggregate the facial tracking data from multiple cameras, a Node.js server was implemented and hosted on Heroku. Each machine running the face tracking script creates a websocket connection to the server and transfers the data from each face detection as a JSON object. The server maintains a dictionary with the last known location of each person, the last detection time, the direction the person was seen traveling in, and the camera the person was detected in. If a person is not detected for a short period of time, they are removed from the dictionary. This allows tracking data to always be available on-demand. The server also provides the tracking data to a website that displays each person's location on a map.



Transfer Learning for Face recognition



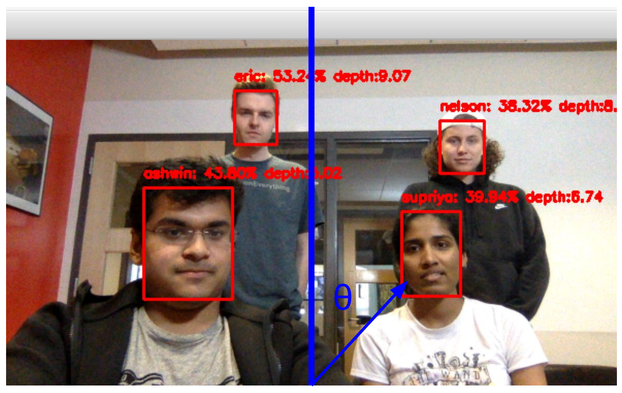
The model uses Single Shot detector to detect faces, and cropped face is selected as input for neural network. The cropped face is sent through the pretrained resnet architecture, and the 128 dimension representation generated is sent to SVM for classification amongst the trained classes.

Centroid Tracking to increase confidence

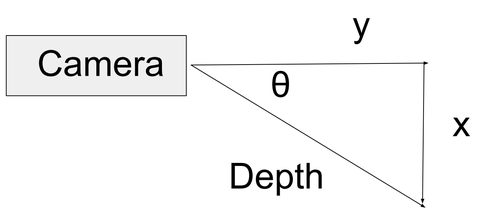
The face recognition algorithm was very sensitive and was required to run the forward propagation followed by SVM every frame, for every face to classify who the face belong to every frame. In order to avoid the need to rerun the face recognition algorithm each time, we ID each bounding box associating a face with a name. We use a centroid tracking algorithm to identify where the current center of the bounding boxes are, and when a person moves in a subsequent frame, the euclidean distance will be smaller compared to other objects in the scene. If there was a new object in a subsequent frame, the system will assign a new ID. If an object is not present in the scene, after some buffer time, it will lose its ID and will be assigned a new ID when it enters the scene.

Depth Estimation and Positioning

Depth ∝ (Actual Face Width)/(Face Width in Frame)

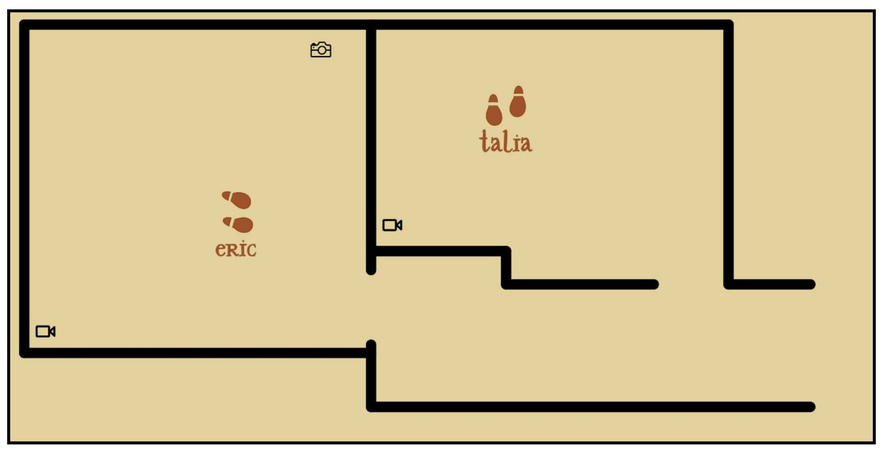


Depth can be used with the angle from central of a face to calculate coordinates in 2D space. When combined with camera position coordinates and angle, this allows the absolute position of each detected person to be estimated.



Multi-Camera Map

A single camera often cannot see each face in an area, so multiple cameras are necessary to accurately detect every face in a room. The position of each camera is manually set so it can calculate the absolute location in the real world. After absolute coordinates have been calculated for each detected face, the data is aggregated on the node server and passed on to clients, which connect to the server via a website. The website uses websockets to constantly update positions.



Acknowledgements

We would like to thank ATLAS, the Human Computer Interaction Class, the ATLAS BTU Lab and Professor Ben Shapiro for making this possible.

References

[1] Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.

[2] J. C. Nascimento, A. J. Abrantes and J. S. Marques, "An algorithm for centroid-based tracking of moving objects," 1999 IEEE International Conference Proceedings. ICASSP99 (Cat. No.99CH36258)

[3]https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition/

[4]https://www.pyimagesearch.com/2018/09/24/opencv-face-recognition/

[5] https://nodejs.org