Andrew Matthews & Steve Mendoza
UCLA ECE M202A - Embedded Systems
Final Project
3/21/2019

Problem Description and Architecture:

The goal of this project was to create a robot that will keep a certain distance from a person, and follow them indefinitely. The user interacts with the robot through the Google Assistant API, in which the user says specific commands such as “move forward” or “come to me”. The primary way the robot finds the person is through a webcam and the OpenCV API, which uses a neural network that runs on the edge and is able to detect a person in an image. The Matrix Creator board and API is used for signals that control the motors, as well as used for sensors which help guide the robot. The goal is to have the robot keep track of one person and keep a certain distance and orientation from the person. In our implementation, the robot is able to detect a person from a distance of up to 8 meters away and as close as 2 meters away. The robot also has the ability to rotate to detect a person in the event that a person is not directly in front of the robot’s camera. The high level hardware and software architectures are shown in Figures 1 and 2 respectively.

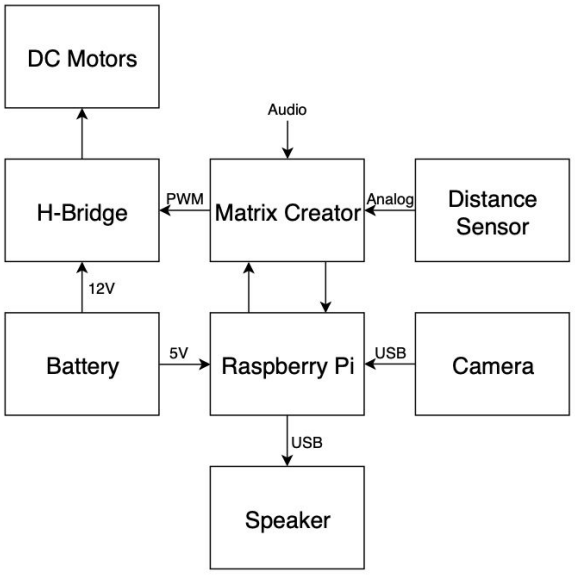

Figure 1: Hardware block diagram

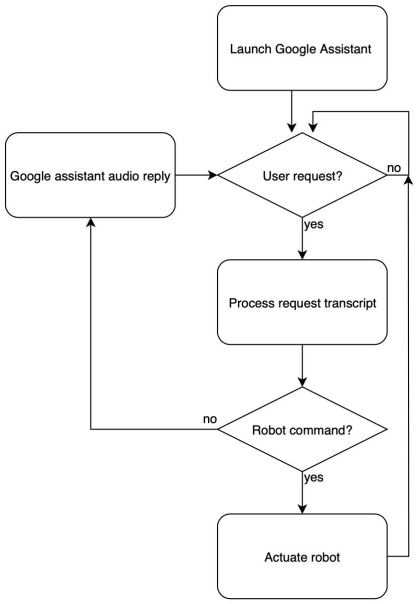

Figure 2: Software Block diagram

Prior Work:

To detect a person, there are other approaches such as haar classifiers [1] and support vector machine (SVM), combined with histogram of oriented gradients (HOG).

[2] The problem with many of these prior methods is that they require more manual tuning and they may not be reliable, also these methods assume a certain configuration of the person, ie if the person is looking straight at the camera. We want an algorithm that is as automated as possible, and works on as many test cases as possible. This is why we went with the neural network approach for person detection.

Technical Approach

1.) Person Detection:

We have a pre-trained neural network that runs on the Raspberry PI itself. It uses a single shot detector (SSD) framework [3] that detects the person within the image, implemented with the MobileNet neural network architecture [4]. In its default configuration, using the Raspberry Pi 3B+ it runs in constant time taking about 2 seconds to accurately classify an image as a person. We have an input image of 1920x1080 pixels, then the algorithm resizes it so that it becomes 400x300 pixels using binning so that the field of view is preserved. The sample output of this algorithm is shown on Figure 3.

To calculate how far away the person is from the robot, we did not use a range sensor but solely the camera. To calibrate our range algorithm, we needed to map a certain straight-line distance from the camera to the person. For instance, we measured the person's height in pixels when they stood 8, 5, 4, 3, and 2 meters away from the robot. We then extrapolate for other distances. Figure 4 shows the distance from the robot is proportional to 1/pixel height, as well as the mathematical mapping from our measured data. This is how we determine the person's distance from the robot.

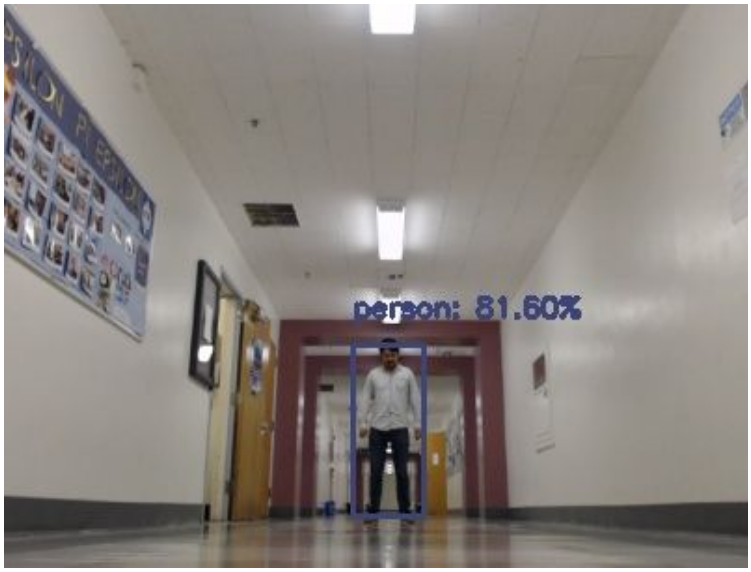

Figure 3: Subject standing 8 meters away from the robot

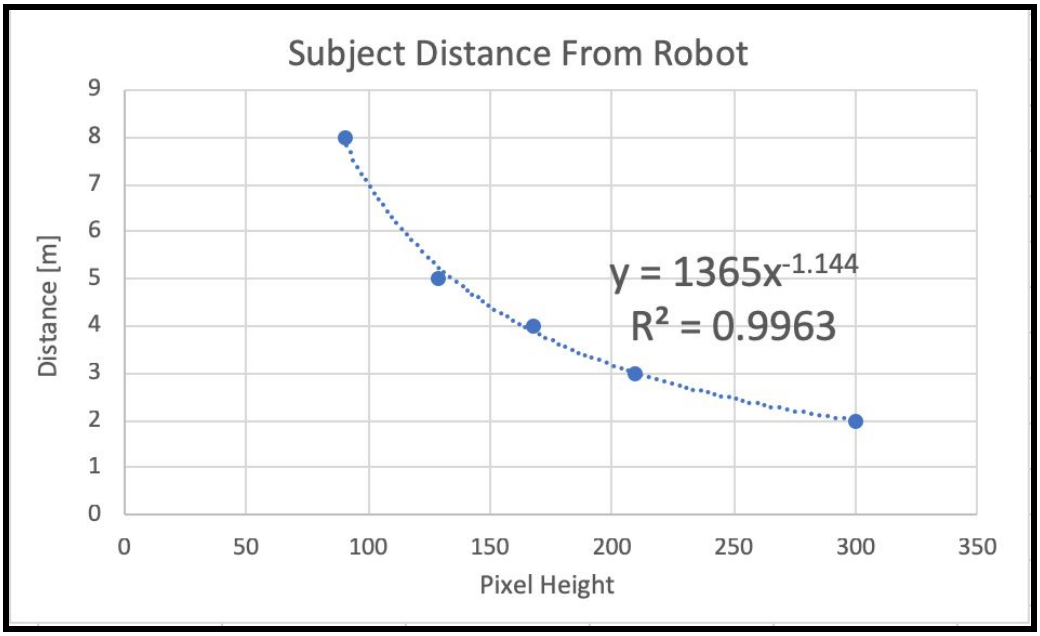

Figure 4: Mapping height of the person in pixels to straight-line distance away from the robot.

2.) Driving towards the person:

Once we have the person in the frame, as well as the lateral and longitudinal positions relative to the robot, we then want to reposition the robot such that it keeps the person in the center of the field of view. If the subject appears to the left of the center of the frame, we turn the robot to the right. Similarly, if the subject appears to the right of the center of the frame, we turn the robot to the right. Using the field of view of the webcam, which is 60 degrees, and the difference between the center pixel of the bounding box around the person and the center pixel of the frame, we approximate the angle the robot must turn, and turn the robot accordingly. Using the previously mentioned distance of the person from the robot, we the move the robot longitudinally. The algorithm is shown as figure 5.

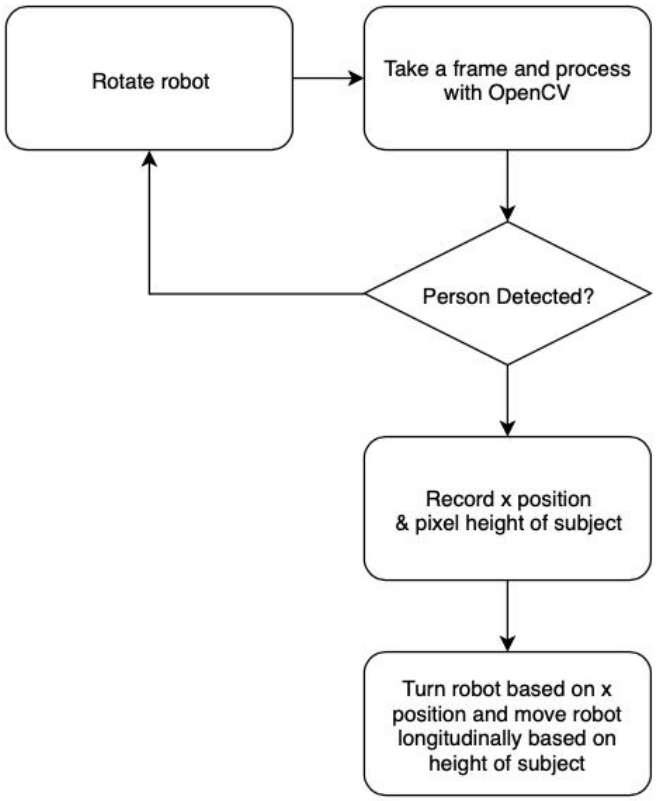

Figure 5: Driving Algorithm

3.) Google Assistant:

The main purpose of the Google Assistant is to listen for users requests and process the speech of the user. The Google Assistant API returns the transcript of the user request, and then the transcript is checked to see if the user requested a specific robot command. The speech processing of the Google Assistant API is cloud based and thus, the robot must have a WiFi connection to function. If the user request is a robot command, we actuate the robot based of the specific request. If the user request is not a specific robot command, the Google Assistant API provides and audio reply to the user.

Future Directions:

The main limitation of our robot is the latency of the process of taking a camera frame and detecting a person, with the default implementation of the neural network giving us an FPS of 0.5. This allows us to detect people from 8 meters away with high confidence exceeding 70%. However, once the robot gets closer to the person, the person gets larger in the frame, and therefore we can change the sensitivity of the neural network so that the algorithm can run faster. We can run the algorithm about 5x faster, by lowering the minimum confidence of the algorithm. If we can get closer to the

person and the person appears bigger with the same confidence level as before, at about 2 fps. This will allow the robot to detect a person quicker and track the person more accurately.

However, once the robot gets closer to the person, the height is no longer a reliable metric since the entire person does not get integrated into the field of view. So another thing we could do was to integrate a range sensor when the person is closer to the robot or incorporate width. However, the main flaw is that depending on how the person extends their arms the network may misclassify width. So height is reliable as long as the person does not jump.

Another thing we did not finish implementing was the integrating the distance sensor with the Matrix Creator board. The distance sensor would give us a more robust method of obstacle detection and avoidance for short range obstacles.

Other things that would increase the reliability of the project would be adding a hotword to the Google Assistant, so that the Google Assistant does not respond to any stray audio in the room. We also would like to implement audio direction of arrival to decrease the time it takes for the robot to locate the person. Knowing the direction of audio would allow the robot to calculate the precise angle that the person is relative to the robot. Finally, we would like the robot to be able to detect different people, and lock on to the person who issues the command. In its current state, the robot will lock onto the first person it detects, and may switch to other people who are in the camera frame.

References:

1.) Papageorgiou, Constantine, and Tomaso Poggio. "A trainable system for object detection." *International journal of computer vision* 38.1 (2000): 15-33.
2.) Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." *international Conference on computer vision & Pattern Recognition (CVPR'05)*. Vol. 1. IEEE Computer Society, 2005.
3.) Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision*. Springer, Cham, 2016.
4.) Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).

Links:
Methodology for the image classification:
https://www.pyimagesearch.com/2017/09/11/object-detection-with-deep-learning-and-opencv/