EE16B Fall 2018
Profs. Alon and Lustig

William Sheu: ee16b-amp
Michael Wang: ee16b-aio

SIXT33N Project Report

Circuit

Final Design

The front end circuit is composed of the micboard, DC offsets, and a band-pass filter. The integral part of the micboard is the electret microphone, and the first part of the circuit converts the current source into a voltage signal. Next, a buffer and capacitor deal with any DC offset and noise. Since we did not have a negative power source, the biasing circuits are necessary to set the DC offset of our signal to the midpoint of the usable range, 1.65V. This task is performed by the OS1 circuit. Next, we amplify the signal from the micboard; however, the DC offset will be amplified as well. Thus, the OS2 circuit, or level-shifter, is necessary to set 1.65V as the reference.
Finally, the output of the micboard is passed through the band-pass filter. This component of the circuit minimizes noise, recording signals from only the 250 to 2500 hertz range. It is important to note the buffer between the lo pass and high pass filters and the connection between the high pass filter and OS2. These prevent damage to the LaunchPad and keep the voltage centered at 1.65V, respectively.

Gain and Frequency Response

Micboard: $\textbf{gain} = 1 + \frac{R_{potentiometer}}{10^5}$

OS1: $\tilde{H}(\omega) = 0.5$

OS2: $\tilde{H}(\omega) = \frac{R_{potentiometer} + 5.1 \cdot 10^4}{1.52 \cdot 10^5}$

Low Pass Filter: $\tilde{H}(\omega) = \frac{1}{(6.32 \cdot 10^{-5})j\omega + 1}$

High Pass Filter: $\tilde{H}(\omega) = \frac{6320}{6320 + \frac{1 \cdot 10^7}{j\omega}}$

PCA Classification

Commands

The commands we settled on are "Hello", "Rock", "Paper", and "Eighty-one", which correspond with straight long, turn right, straight short, and turn left respectively. We wanted the words to be very distinguishable from each other, and so we choose words with different number of syllables. The two words with the same number of syllables ("Hello" and "Paper") sound different enough so that we thought it wouldn't be a problem.

Processing

We preformed PCA with the first and second PCs of our recorded data. From there we found the centroid for each cluster of words. With these centroids, we would compare them to the de-meaned sample collected and see which centroid the sample was closest to (euclidean distance). We made the classification more robust by adding a k-means threshold, which would not classify any input that has a distance to all centroids greater than this threshold. This prevents misclassification by not classifying at all.

Controls

Open Loop Model

With the open loop model, we collected data for each of the motors in order to find out their difference in velocities, and how we can take that into account for driving straight. From there, we hard-coded in θ, β, and the $V*$ into the program and had it run solely based on this information, and without the help of sensors to correct for any deviance from a straight line.

Closed Loop Model

In the closed loop model, the program takes continuous data from the sensors to adjust the system so that they follow a straight line. This takes the form of delta, or the difference between the wheels' distances. The k_left and k_right are the gains for each motor. This is necessary for the car to adjust if it starts drifting left/right.

Choosing Controller Values

The thetas and betas are derived from running the dynamic data, and thus calculating the correct model the car should follow. $V*$ is calculated by finding the middle velocity in the range for where the velocity/power seems most linear. As the car reaches $V*$, the car may deviate from the model and thus a right/left_jolt is necessary to allow the car to go straight before $V*$.The k values are the gains to adjust the car if the delta (the difference between the wheels' distances) begins to increase. The k values must satisfy this equation to be stable: $|1 - k_l - k_r| < 1$[1]. The closer the value is to 0, the quicker it will try to return to a straight line, but may cause an oscillatory response. We found experimenting for the best k values to be the best approach. Lastly, the delta_ss value is the known steady state error, which s the model better respond to unexpected perturbations affecting delta which increases the model's control strength. We calculated this by allowing the car to run to steady state, and found the delta in its steady state.

General Comments

This lab really helped put what we learned in class into practical use, and thus made these subjects more memorable. One example would be implementing our own PCA model and using it to classify real signals. I found doing this very interesting and really cemented my knowledge for PCA/SVD.

There have been trying times, such as when we didn't know we were supposed to tape our back wheel and thus our car didn't go straight for the first couple of labs, and when we couldn't find out why our classifying system was not working. However, it all paid off when we finally got our robot up and working.

Figures

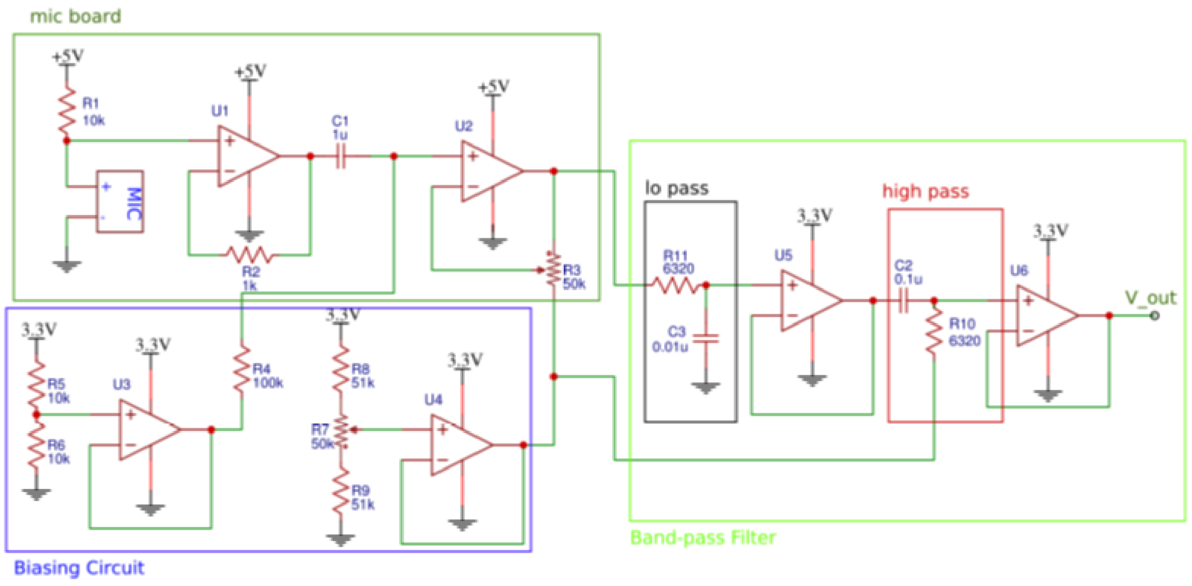

Figure 1: Full Circuit Diagram

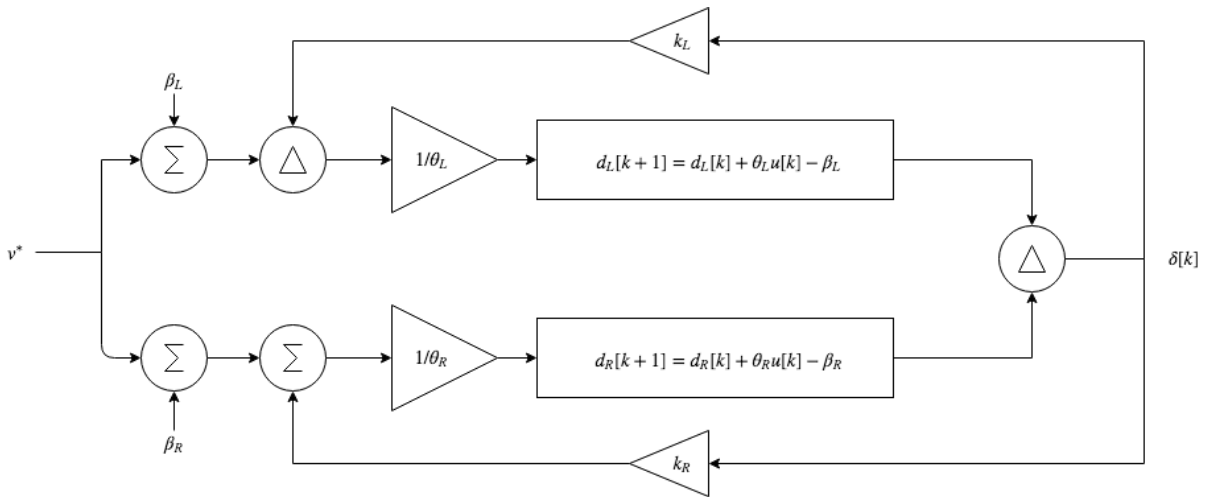

Figure 2: Closed Loop Control Scheme Block Diagram

References

[1] "Primer for the Upcoming Control Lab" Available from World Wide Web: (https://inst.eecs.berkeley.edu/ ee16b/fa17/proj/controls-primer.pdf).