EE 16B Fall 2016
Final Project

SIXT33N Project: Written Report

Rahul Balakrishnan: ee16b-afa
Rohan Suresh: ee16b-aeu
Kailas Vodrahalli: ee16b-abq

Circuit:

Final Design:

The goal of the circuit is to 1) act as a lowpass filter to remove frequencies that arise from aliasing during sampling and 2) condition the output for the MSP (restrict the voltage range to [0V, 3.3V]). Our approach was to break up these functions into multiple steps, and add buffers in between the steps to remove loading effects and allow us to treat each substep individually. (Note: in our circuit, we used 2 potentiometers to help make our voltage dividers more accurate.) Please see last pages for diagram.

1) Level Shift Signal
    a) Here we use a simple voltage divider that moves the center of the signal to ~3.2V followed by a buffer to prevent loading from the next part of the circuit
2) Lowpass filter
    a) From the level shift, we have a low pass filter with cutoff frequency: using a resistor-capacitor voltage divider
    b) We buffer the output to prevent loading effects
    c) Identical to (a); allows us to have a “sharper” lowpass filter that attenuates quicker
    d) We buffer the output to prevent loading effects from the following stage
3) Condition output for MSP
    a) We buffer the output, because of the MSP’s requirements

Gain and Frequency Response:

We implemented the lowpass filter using two identical low pass filters (simple RC circuits; R ~ 120 Ohm, C ~1uF, to get a cutoff frequency a bit below the desired 1,500 Hz) connected by a buffer. By using the buffer, we were able to make the circuit quite simple and use two first-order low pass filters.

In our circuit, we did not gain the output at all. We found that the voltage level of the signal (relative to the noise) at the output was sufficient and did not require additional gain.

PCA Classification:

Commands:

We settled on the commands {laptop: turn right, vi: go straight, kane: go straight fast, aguero: turn left}. We tested 13 different words in total before settling on these four. We found that 1) words with distinct sounds, 2) words with different lengths, and 3) words with different numbers of syllables and different accent placements gave us the best results.

Processing:

We played around with the parameters until getting our final values. The values we chose are Snippet Length = 60, Prelength = 10, Threshold = 0.4. We did not use any additional processing to improve our classification as we found the basic PCA classification technique gave good results.

Controls:

Open Loop Model:
Please see last pages for diagram.

The open loop model is: $d_i[k+1] = d_i[k] + u_i[k]\,\theta_i - \beta_i$ where $i = R, L$ for the right, left wheel respectively. We can use this model to find an appropriate u_i given a desired v^*.

Closed Loop Model:
Please see last pages for diagram.

The closed loop model is : $d_i[k+1] = d_i[k] + u_i[k]\,\theta_i - \beta_i + \delta[k]\,\kappa_i$ where $i = R, L$ for the right, left wheel respectively and $\delta[k] = d_L[k] - d_R[k]$. As before, we use the open loop model to find u_i give the desired v^*.

The closed loop model is necessary for the car to drive straight because the linear model we use is inaccurate -- the actual system is not fully linear. So in order to straighten the car out when it does turn, we can use closed loop feedback that adjusts for any difference in the distance the wheels have traveled.

We were able to implement turning by intentionally creating an offset between the two wheels, so the closed loop model will correct for the offset and so turn the car. This took a bit of time experimenting with different offset values, but the end result was reliable and consistent.

Choosing Controller Values:

Initially we tried rather large k-values (~0.7-0.9 in magnitude). After experimenting more, we found that small k-values (~0-0.2) gave us much better, less erratic results. In addition, we found that setting the k-value of our "weaker" wheel (determined by our data-collection) to 0 improved our results significantly, because this way we are not over-relying on a motor which may not be able to generate the necessary wheel velocities determined by our control setup.

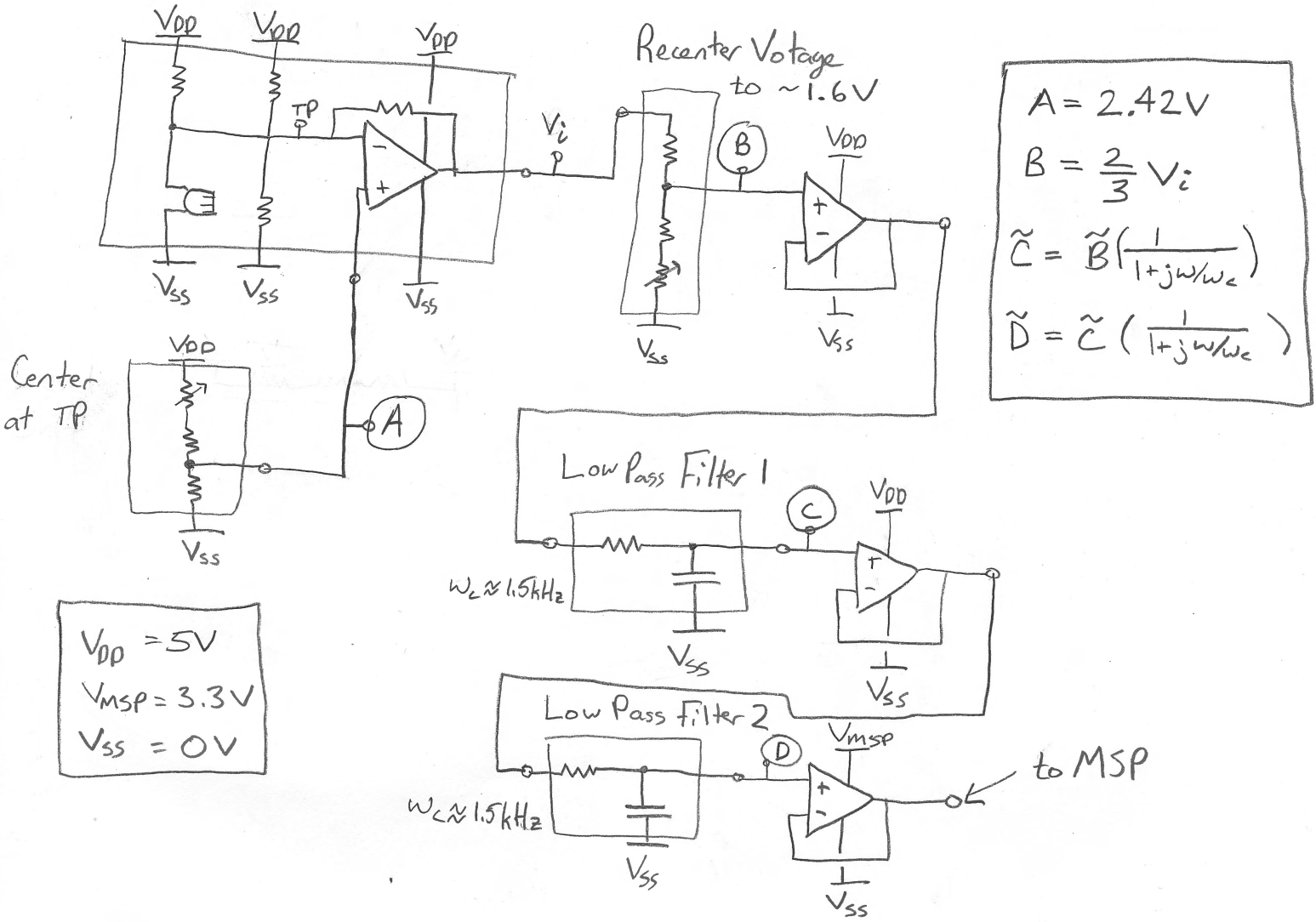
General:

Through this project, we learned how to apply design principles and relevant techniques in order to build a properly functioning device. It added a dimension to the course in which we were able to combine the knowledge of circuits and linear algebra signal processing techniques and see their relationship for our device. We enjoyed how the project built on itself week by week, adding new elements and functionalities at each checkpoint.

Feedback:

The part of the project we had most trouble with was getting the car to drive straight. Eventually, after a good number of hours in lab, we figured out with the help of a TA that the wires connecting the motors to the power source created lots of noise when overlapping with the wires from the sensor, and so the readings sent to the MSP were distorted. Had we known about the noise from the motors earlier, it would have saved us a lot of time debugging; so, this might be a useful tip to include for next semester. Overall, the project was a great experience.

Mic Board



$A = 2.42V$

$B = \frac{2}{3} V_i$

$\tilde{C} = \tilde{B}\left(\frac{1}{1+j\omega/\omega_c}\right)$

$\tilde{D} = \tilde{C}\left(\frac{1}{1+j\omega/\omega_c}\right)$

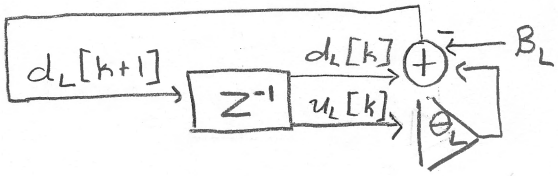
$V_{DD} = 5V$

$V_{MSP} = 3.3V$

$V_{SS} = 0V$

EE16B Mic Circuit

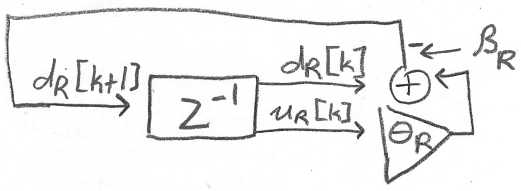
Left Wheel



$\theta_L \approx 0.1657$

$\beta_L \approx 6.692$

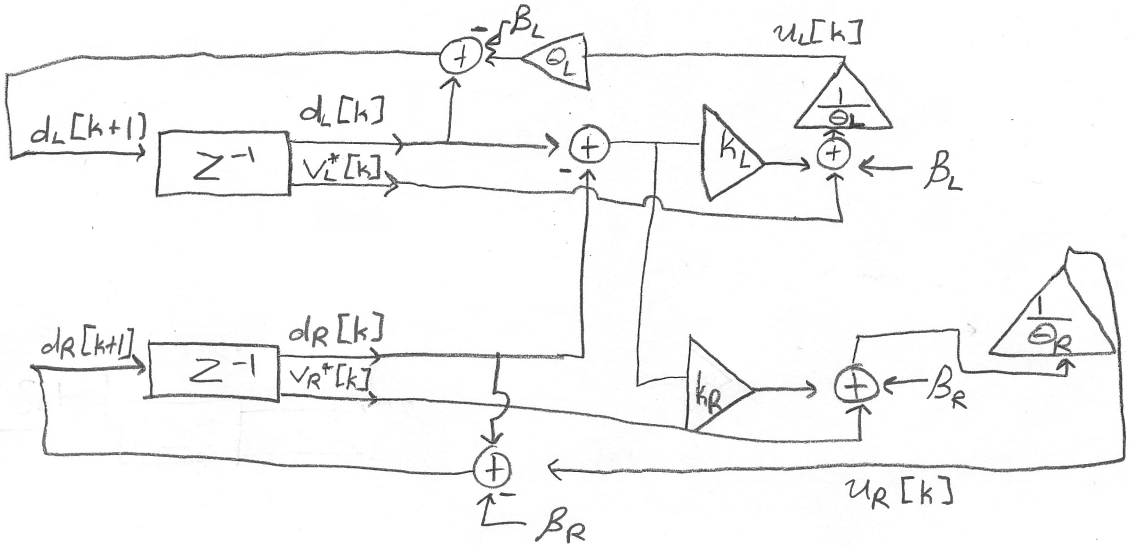
Right Wheel



$\theta_R \approx 0.1217$

$\beta_R \approx 1.508$

Open Loop Model



$k_L = 0.0$

$k_R = 0.2$

Closed Loop Model