David Camacho noedavidcamacho@berkeley.com
Michael Campos michael_c55@berkeley.edu
Group ID: 111 - 004

1. Midterm Lab Report Review

1. The majority of this lab was getting acquainted with working with circuits once again through using op amps as buffers and then creating an inverting op amp. It primarily served as an introduction to the concepts that would be fundamental to creating SIXTEEN and would later be applied to our following lab's creation of the biasing circuit with an op-amp and a non-inverting amplifier with an op-amp.

2. This lab focused on creating a 3-bit SAR ADC followed by a 4-bit version. This would ultimately help us create the conversion of DAC voltages to Analog which would be necessary for our SIXTEEN car's signals for parts such as the micboard. Once again, we learned how to utilize lab equipment such as the potentiometer and voltmeter which would become essential in the following labs.

3. Motion's lab would prove being essential to getting SIXTEEN to turn as it involved the motor setup for both wheels. Additionally, it involved setting up the encoders which would be the source of our data for least-squares regression, wheel velocities, distance traveled, and more.

4. Sensing Part 1 involved the creation of our 5 volt regulators, 3.3 volt regulators, and mic board circuitry. It would end up being the entire system supplying our micboard readings, its buffer, and regulators for our motors as well. Finally, we would have to tune the microphone such that it could filter out any noise and only pick up our voice for our integration final demo test to work smoothly and so that our commands could be classified accurately later on in the course.

5. Sensing Part 2 was centered on creating our low-pass filter for our microphone, our low-pass signal amplifier, and the high-pass filter. This culminated into the creation of the band-pass filter that avoided any issues the lack of buffer created such as the second filter was not loading from the first filter; thus ruining our mic's data collected. The lab's connection to SIXTEEN was that it improved our mic board's readings significantly to ultimately read our inputs without any interferences that could come from loading between the low-pass filter and the high-pass filter.

2. System ID

Summary: In this lab we collected data to ultimately run the least squares on our car with the appropriate variable so that it could drive straight. We initially wrote data into our MSP without any PWM values set. Later on, we specified both low and high PWM values such that our

velocity graph could become more linear. With that, we used least squares regression to find our desired variable values (theta/beta left and right) to evaluate our line of best fit. The encoders were utilized to gather data and compute the wheel velocity. Through this line, it was possible to find the operating point, our velocity ranges, and a v_star value to use for our next lab.

:
1) Our goal is to model our linear system as accurately as possible so we collected data that was appropriate for least-squares to be used. We selected a small PWM range such that we returned a linear coarse plot such that our wheels would have the same velocity such that the car can drive straight. We then applied least-squares regression to this new data. It is important to choose a region to run the least-squares on so that we can accurately approximate the free model parameters for our real system.

2) Θ is the relation between the change in the input to the change in velocity with the units of measurement ticks/(timestep*duty cycle). β is the constant offset in velocity with units in ticks/timestep. This offset can be impacted by external physical factors.

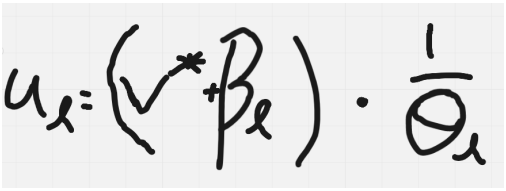
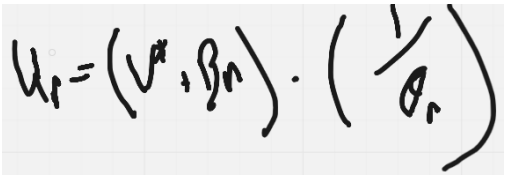
3) Each motor we have has a different velocity which impacts the wheel's rotations and velocities per each side of the car. Additionally, external factors such as weight distribution can impact the data on each respective wheel (left or right). In order to make our calculations easier when trying to make the car go straight, these different sensitivities can be accounted for with variables for left and right.

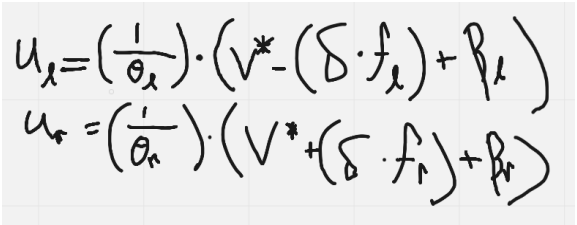
4) v*, our target velocity, is the best set in the midpoint so that we could have the largest possible margin of error for both sides. If we set v* outside of the overlapping velocity range, our margin of error would be very small to the point that we have no safety net in our calculations making our physical car not behave as expected. One of the consequences could be that the car would never go straight.

5) Unlike a linear model, polynomial/higher order functions can be interpolated in a way to encompass a general, nonlinear trend of our data. For example, if a cluster of data points cluster around a parabolic shape, the best approximation would be to use a second order polynomial function. However, there can be cases where two different, district shapes are formed by our data points. Say a set of data follows a parabolic trend, and at some time T the data points cluster around a flatline for the rest of the time, then a piecewise function is needed for the best model: a polynomial function before time T and linear/constant function on and after T.

3. Controls Part I

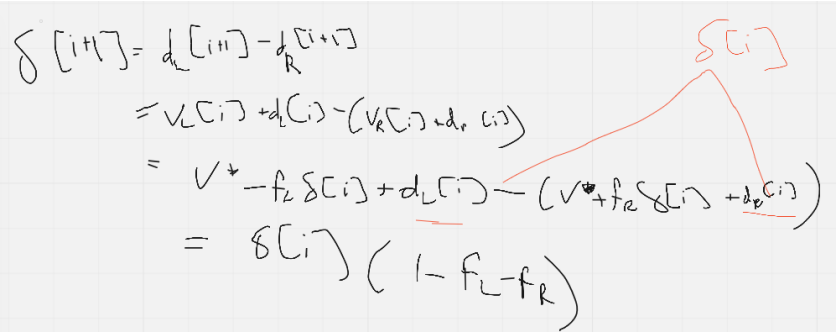
Summary: This lab builds off of the previous lab's results and utilized both open-loop control and closed-loop control to make our car actually drive straight. We begin utilizing our variables from the System ID lab (beta, theta) in our open loop equation and calculate the appropriate pwm jolt for each wheel as it varies. We noticed that our car was not driving straight as we were not incorporating any feedback into our equation. This is where the closed-loop equation comes into play. A delta value is used to calculate the difference between the distances traveled by the wheels and that is fed back into our equation to take the steady error into account such that our $\delta[i]$ is stable. We simulated model plots to find appropriate f-value pairs which will determine our eigenvalues and stability of our car. Once we implemented the closed-loop equation, we were able to drive our car straighter, and the steady state error correction (that uses δ_ss) ended up allowing our car to drive straight.

1. 


2. Given the fact that our motors do not have the same velocity offset and react differently to change in PWM duty cycles, this implies that they will respond differently when at rest to actively moving. For that reason, we use different jolt values to start moving once we overcome static friction. The left and right jolts might be different because the motor PWM duty cycles are different.
3. Our open-loop control fails as it is essentially built off of the idea that our environment is an ideal world without any unforeseen variables. Open-loop fails because its predetermined input will never be adjusted during operation as we assume no mismatch, disturbances, or noise, and assume an ideal behavior. Closed-loop is essential because it utilizes feedback such that our car can account for these mentioned external factors. Closed-loop guarantees that velocities will remain the same between both wheels through the use of feedback.
4. 

δ is the difference in the distance traveled between the left and right wheels at a given time step used to calculate velocity and guarantee our car will go straight. This should remain constant. It is a state variable.
v* is our target velocity. It is a system variable.
Our **fL** and **fR** are feedback gain values that create our state-space controllers that should create eigenvalues that are in a stable position, $|\lambda| < 1$. These will represent the possible configurations of our system.
Θ is the relation between the change in the input to the change in velocity with the units of measurement ticks/(timestep*duty cycle) and its purpose is to help us translate the inputs into wheel velocity.
β is the offset in velocity with units in ticks/timestep. This offset can be impacted by external physical factors and is meant to help calculate the velocity of the wheels by subtracting any offset in comparison to the input.

5. System Eigenvalue = 1-fL-fR



In theory, the system is stable when the equation | 1-fL-fR | < 1 (alternatively written as $|\lambda| < 1$).

6. If our value is positive, the left wheel has turned more than the right wheel, and the car is turning left. If the value is negative, the right wheel has turned more than the left wheel, and the car is turning right.
7. Having zero on the f values is the equivalent of having no feedback whatsoever so you're basically running the open loop equation instead which won't account for the external factors previously mentioned. Non-zero f-values give the necessary feedback so that our vehicle can account for external factors and drive straighter and more consistent.
   Non-zero f values are necessary because they utilize the main part of being closed-loop, which is utilizing feedback to maximize consistency and consideration of external factors.
8. Using negative f values would result in our system eigenvalue: 1 - fL - fR always being greater than 1. Our goal is to make the eigenvalue < 1 to guarantee stability in our car. If we wanted to always use negative f-values our closed-loop model equations would have to swap so the equation for uL would be the current equation for uR and vice versa.
9. Delta_ss is the known steady state error added to every calculation of error delta to correct make the car drive straight. A zero delta_ss value means that our car must be

driving straight for that timestep and there is no error to add and correct the car for. A non-zero delta_ss value means that our car's trajectory is not straight. With delta_ss our error term becomes $\delta[i] = d_L - d_R + \delta_{ss}$. so it corrects steady state such that it converges to a value; ultimately,driving straight.

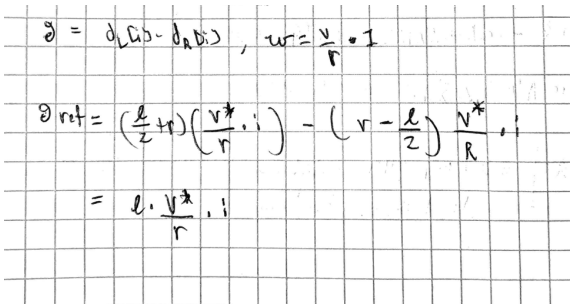
10.

4. Control 2

Summary: Stemming from control lab part 2, we continue on with the linear model for velocity of each wheel, but with the goal of trying to make the vehicle move in the straight line. Unlike the previous lab which used open loop control, we modified the linear so a closed-loop control can be achieved. This involved introducing a state variable delta which is the difference between the distance traveled of each wheel and new coefficients fL and fR. In order to drive straight, the delta needs to be constant. So, the whole process involved tuning the feedback gain f-value on energia to find which finds causes the vehicle to drive straight.

1) Delta reference is used as some error value added to the system in order for the car to turn and correct itself. So if delta reference is negative, it will cause the robot to think that it has moved left. So by our closed loop model, it will compensate for the error by turning right thus executing the turn right command. Likewise, when delta reference is positive, it causes the robot to think it moved right, so it will move left to correct itself. And, to move straight the delta reference is 0.

2) **Answer:** l * vstar/r * i



If delta reference is constant, the added error will cause the robot to interpret it as a one time fix error. The car will begin moving in one direction and then turn to a stable direction over time due to the error. However, if we need the car to turn at a 90 degree angle for example, then we will have to continuously add a new delta reference at each time step. Therefore, delta reference will depend on distance between the two wheels, the radius of the curve, the target velocity, and time.

3) Since the control loop and data collection have different sampling periods in order to update the trajectory every 10th second, they conflict with each other when turning. For example, if the data collection has a slower sample rate than the controller, there isn't enough data to process during control loop. So we need to reduce the velocity of the controller (ticks/sec). The variable "m" is defined as the ratio of sampling frequency (Fc) of the control to that of the data collection(Fd). The data collection and control loop also have their own velocities (ticks/sample), Vd and Vc respectively. The product of Fc and Vc must equal to the product of Fd Vd since we want constant velocity which also means constant distance (delta). If we solve for Vc, we see that it is equal to the product of Vd and the inverse of "m". So if Tc = 100ms and Td = 500, the respective frequencies are Fc = 10 and Fd = 2. M is equal to 5, so the velocity of the controller is a reduction of the velocity of data collection by a factor of 5.

4) Even if the added delta is constant, mechanical issues like imperfect wheel sizes and/or axle wobble causes the system to think it is moving straight even though it's not. So unlike delta ss from Controls Part 1 which depended on the data from the sensors, the straight_correction uses the radius of straight_radius in order to correct the trajectory.

5. Classification

Summary: The classification lab involved identifying words spoken into the microphone. We recorded six different words, about 40 times each and manipulated the length, threshold, and pre-length parameters to clean up our training + testing data. By exploring PCA/SVD, we are able to compute principal component vectors to sufficiently reduce the training data matrix, using it to find the centroids. At the end, we used these centroids to classify each appropriate word in an energia file.

1) The different number (1, 2, 3, 4 syllables, 1 for each word) of syllables (amplitude and pal for example) and the way of enunciating each syllable (kilowatt and kilometer for example) are two characteristics that help create form unique graphs for each of the four words for classification.

2) Length is defined as the word's total number of samples it takes to complete a recording. Threshold is defined as the fraction used on the largest value in the recording to find the beginning of a word. Pre-length is considered the number of samples before the threshold is hit which will be included as part of the word.

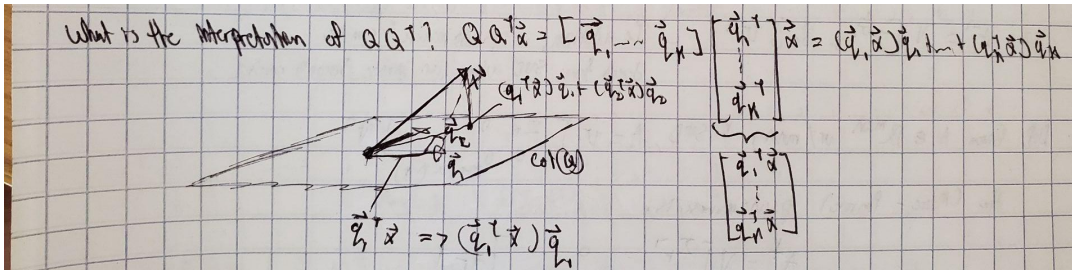
```
Best trial:
  Value:  93.75000000000001
  Params:
    length: 80
    pre_length: 9
    selected_words_arr: ['New', 'Action', 'Kilowatt', 'Amplitude']
    threshold: 0.33239721753621465
```

3) Since recordings are imperfect (started recording too late or ended too early), there will be some samples with noise or silence, making it difficult to identify important characteristics mentioned previously. So in order to clean it up, the data is preprocessed by manipulating the threshold, length, and pre-length size.

4) PCA/SVD is primarily used to compress the data (reducing the rank of the data matrix) since the launchpad has a limited amount of space. Therefore, only the first two to three principle components are considered to be used for the test data.

5) After stacking the data vertically, the rows of the data matrix are the different recordings of each different word and the columns are the different samples or time steps. Therefore, we only consider the row data since it maps to each recording. PCA uses the U matrix for column data, therefore we use V transposed.

6) After stacking the data vertically, the rows of the data matrix are the different recordings of each different word and the columns are the different samples or time steps. Therefore, we only consider the row data since it maps to each recording. PCA uses the U matrix for column data, therefore we use V transposed.



7) Steps:

1. Compute the PCA vectors from SVD, projected mean vector, and centroids
2. Project new data vector onto the new PCA basis by just computing the dot product
3. Demean the projections to reduce data size/save memory of the launchpad.
4. For each projection data point, compute the distance from the data point to the centroid. The smallest distance from the centroid that is less than the euclidean threshold, would be the classification/word.

8) For a data set whose graph is like a thin straight line, we only need 1 PCA vector since the space is 1 dimensional. On the other hand, a circle graph means that the data set spans a 2 dimensional space; therefore, 2 PCA vectors are needed. If we are dealing with a cylinder graph with a large base area and small height, then by PCA, we are trying to perform a low rank approximation on the data matrix. So, we ignore the height (reduced the rank by 1), and only consider 2 PCA vectors.

9) If the number of PCA vectors increases, starting from the number we currently have, then the increase in accuracy would not be very significant since the current principal components are already sufficient for classification.

10) Euclidean threshold is a value used to filter out samples whose distances to the closest centroid are too large. On the other hand, loudness threshold is a value used to compare the amplitude of the recorded data and filters the classification if the amplitude is too big or small.

6. Integration/Final Demo

Summary: The Final/Demo lab integrated all of the features from previous labs and allowed us to demonstrate our project in action. Using four different words: pal, action, amplitude, and kilowatt, we were able to change the direction and distance of the vehicle (left, right, close, far). Most of the lab involved copying and pasting code from previous labs and integrating it all in one file.

1)

Michael: Introduction to simulation helped give an insight on how circuits work without having to worry about real world bugs. Analog and Digital Interferences helped connect what we've

learned about low pass, high pass filters from lecture and improved our circuit debugging skills. Although it was cool seeing the microphone registering our sounds, hearing loud, high frequencies for three hours straight made it my least favorite lab. Motion allowed us to get a better feel for energia and had fun moving those wheels. The sensing labs gave us an insight of how to regulate voltage, understand microphones, and provided an experience in debugging circuits.The system ID and control labs helped consolidate our understanding of closed and open loop systems. In the classification lab, SVD/PCA was one of those topics that was hard to see in the real world from just reading the textbook/lecture notes, so the lab helped us see how these tools are applied. For example we used SVD to obtain a special matrix, V transposed, and then used PCA to appropriately reduce the size of data in order to fit in the limited space of the launchpad. Even though we reduced the size of the data, our contraption was still able to detect our words. Despite it taking the most time to complete, this was my favorite lab in the series. Probably, one of the easiest labs, the final demo gave us no major issues and smoothly finished on time.

David: Most of what I learned in the labs was mostly conceptual applications of what we were learning in class. There was always this abstract wall between the plug and chug / proof nature of the homeworks/discussions that was created and the lab resolved most of that by seeing practical applications to what we were learning. Although cliche, I did learn how to work with a lab partner and communicate my weaknesses with them. Seeing the car itself pass the final integration demo without ANY issues was honestly the highlight of the semester. After going through such a tedious process of debugging every minor error and busted part, I think we deserved a smooth checkoff and I am glad it was the last lab. My least favorite part was anything involving the mic board, but specifically the first lab in which we calibrated the mic board with the high frequencies. There were so many students who just didn't use any self-awareness and raised the frequency volumes to unnecessarily high. Apart from annoying, it was nauseating by the end because our mic board just wouldn't function properly until we realized that we had to get a new mic board. It was beyond frustrating that our mistake was not in our control. Overall, I appreciate the lab TAs so much, we were in the 5-8pm section on Wednesday and every lab assistant was a wonderful person who helped us with such kind attitudes. Respect.

2) **Michael**: The control and the classification labs were perhaps one of the hardest labs we had to deal with. It was not because they were conceptually difficult, but rather the debugging process consumed most of our time. Since we kept building off of the previous labs and some launchpads were unreliable , there were a lot of potential problems we had to deal with, especially when we cleaned up the circuit. For example, in the classification lab, we had a new circuit after cleaning the old one and we were having problems recording our words. At first we checked that our circuit was properly connected, and we found a couple of disconnected wires using the multimeter, but despite that, it was still not working and we were advised to check the microphone and the op amps.Ultimately, after a couple of hours of debugging, a TA came over

and figured that one of the pins was not working. Finally, with a sigh of relief, we began to make progress. We have to give special thanks to all the TAs who helped us, they're awesome! One of the important things we took away from this lab is to debug/check every modification before coming to the lab. Rewiring everything at home with no multimeter and Energia was perhaps the greatest contributing factor that led to our downfall (at the beginning of the lab). If we had carefully verified every modification, this would have allowed us to narrow down the possibilities of potential problems and quickly identify the problem.

David: I think the busted mic board we had was the worst bug this semester. It wasn't something that was in our radar at all and after probing all of our outputs multiple times and getting the right values, it was sad to see we had to rewire some stuff to get it working. I learned how to breathe in and not scream in public spaces.

7. Feedback

David: Offering parts for SIXTEEN that are fully functioning would be nice. I know this is mostly a byproduct of the budget issues, but if I was given the opportunity to pay 40 dollars for new parts knowing they would definitely work, I would have done that. I have no complaints or criticisms for the lab TAs just keep that up please, they are the best part of the lab since they are so approachable. I believe the strict grading on the lab reports is incredibly unreasonable after the instructions specified that a simple answer would suffice, and that should be changed in future iterations.

Michael: Despite spending a lot time debugging, I enjoyed the labs and was impressed to see the end result. The TAs were amazing and I appreciate their help, but importantly their patience. There isn't much to critique, but I do have to say that a lot of launchpads were out service or had some of their pins broken.

8. Collaborators and Sources

50% David
50% Michael