Controls Final Project

Colin Snow, Meagan Rittmanic

July 26, 2022

1 Overview of Model and Calibration

1.1 Introduction to Model

In order to determine a course of action for stabilizing the magnet, we made some assumptions about the model, and then collected as much data as possible about the system dynamics to validate those assumptions. We wanted to understand what assumptions we could make about the system, how those assumptions would affect the results, and build a simulator to determine if our controller would be effective. We began by creating a model of the system that we thought would be effective and then tested each part by taking measurements to verify that our assumptions were sound.

Figure 1: Our measurement setup. We connected the magnet to a scale and moved the scale up and down to measure force at different distances.

1.2 Model Overview

When modeling our physical system, we split our model into two components: physical and electrical. The electrical component modeled the relationship between voltage input and force while the physical component modeled the relationship between force and vertical position of the magnet.

1.3 Physical Model: Force vs Distance

Our model assumes that the magnetic force between the electromagnet and the magnet when the electromagnet is off is inversely proportional to the distance between them squared, which is an

assumption we took from basic electromagnetic reasoning. To verify this assumption, we took force measurements every .25 mm from 0 to 13 mm below the sensor and plotted these with respect to distance. We then fitted an inverse square model and found that it fit extremely well, suggesting that this model effectively captured their interaction (Figure 2).

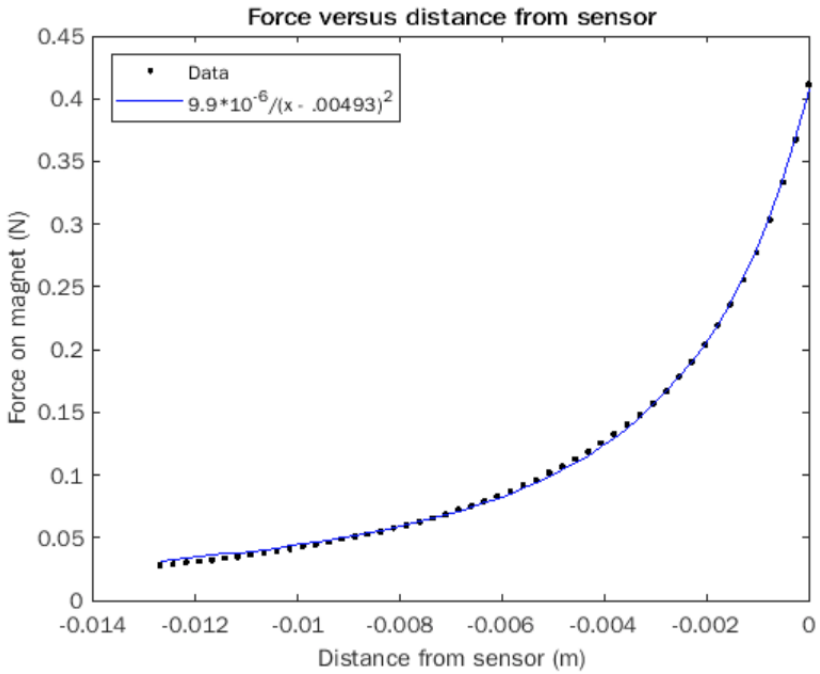


Figure 2: Measurement of force as compared to distance from sensor, taken at 0.25mm intervals.

We then needed to determine how this interaction would vary with current flowing through the electromagnet. We ran the previous experiment again several more times with different amounts of current flowing through the electromagnet. We found that increasing the current caused the force graph to shift downward and at high currents to sort of bend around a point at -4 mm. Given that we did not expect to see currents this high, we decided that a model which represents current as a constant force offset would be appropriate (Figure 3).

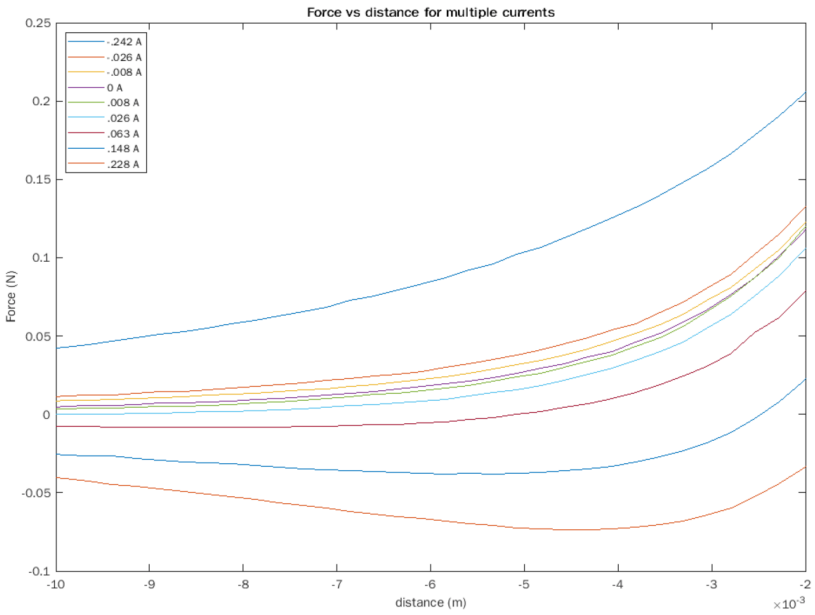


Figure 3: Measurement of force as compared to distance from sensor, taken at 0.25mm intervals, at various currents through the electromagnet

We then checked the model we created against the real data and found that it performed quite well across a range of currents. We saw that the model we created fit quite well but had a constant offset of .0125 N so we simply subtracted this amount to get a model that was accurate within .01 N over the distance range from -2 mm to -10 mm and currents from -.25 A to .25 A. Since we expected our real values to live within this range, we determined that this model was acceptably close for use in our simulator (Figure 4).

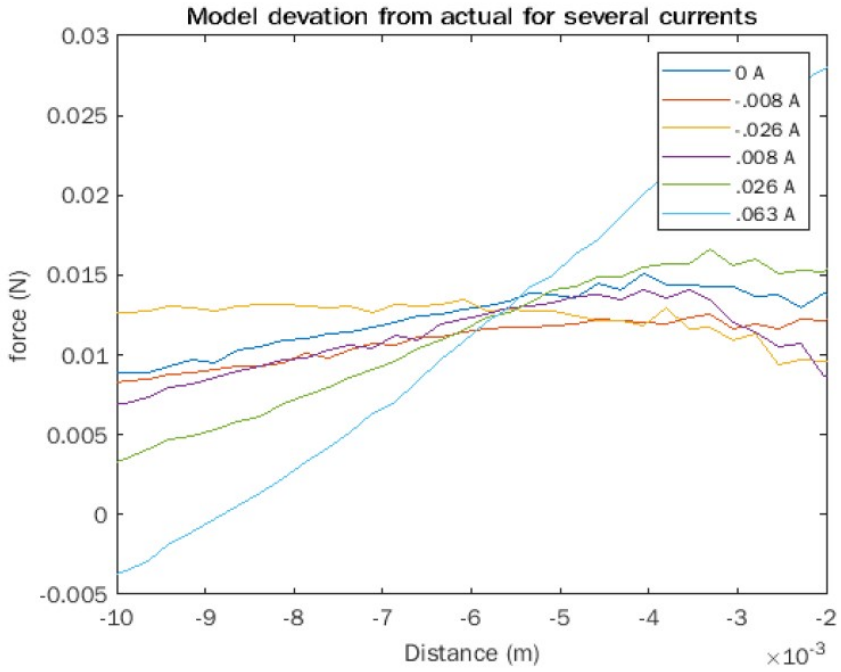


Figure 4: Error between model and actual force for varied currents and distances from sensor.

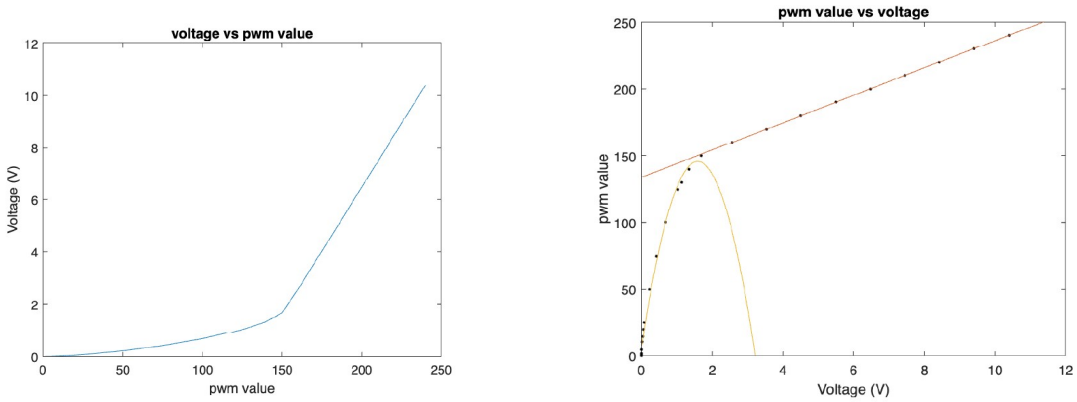
In order to integrate this nonlinear model into our linear controller, we modeled the force as linear, but then had the conversion from force to current modeled outside of our controller (Controller Model section).

1.4 Electrical Model: Voltage vs pwm

The electromagnet creates a magnetic field proportional to the current flowing through it. In order to set the current accurately, we created a model of the electrical system which treated the electromagnet as an LR circuit. In order to properly model this system, we needed to determine the resistance and inductance of the electromagnet. The resistance was easily measured with a multimeter, but the inductance was slightly more difficult. We ended up connecting a multimeter measuring current in series with the electromagnet and connecting one channel of an oscilloscope across the leads of the multimeter and the other channel across the the leads of the h bridge. The channel across the h bridge would then measure the input voltage while the one across the multimeter would measure the voltage drop across the small resistance of the meter. By inputting a step voltage and watching the current increase, we measured the time the current took to get to 63.2% of its final value. This gave us the time constant of the system which along with the resistance allowed us to calculate the value of the inductor to be about .0936 H.

We suspected that this value was sufficiently small that it would not have a very large effect on our system, so we assumed that the transition between voltage application and current was instantaneous in our controller model and modeled the actual inductor dynamics in our simulator to verify that this assumption would not lead to instability. This allowed us to ignore the pole that the LR circuit created in the model of our system (Controller Model section).

We then had to relate the pwm value input to the h bridge to a voltage and therefore a current in the electromagnet. We cycled through a range of pwm values while measuring voltage with a



(a) Voltage vs pwm value

(b) pwm value vs Votage fit

Figure 5: Measured voltage vs pwm value and fit for pwm value versus voltage. By using this fit we can produce a desired voltage with a high degree of accuracy. The voltage equation is fit with a quadratic below two volts and a line above two, capturing the weird transition between these two ranges reasonably well.

multimeter and found that the relationship between pwm and voltage was relatively linear above 150 but became rapidly less steep as the pwm approached 0. We decided the easiest way to model this behavior was to fit a linear model if the desired voltage was above 2 and a quadratic if it was less than 2. This produced a relatively accurate model over the full range of possible voltages (Figure 5).

1.5 Calibration: Sensor vs Distance

Our model relies on accurate positional data to determine how much force to apply. This means we needed a way of converting sensor readings to distance that was more complex than linearizing about a point. In the case where the electromagnet is off, this is relatively easy. We measured the sensor value vs distance every .25 mm below the sensor and fit an inverse square model to it. This model predicts the distance from the sensor to within +- .1 mm up to a distance of 10 mm (Figures 6 and 7).

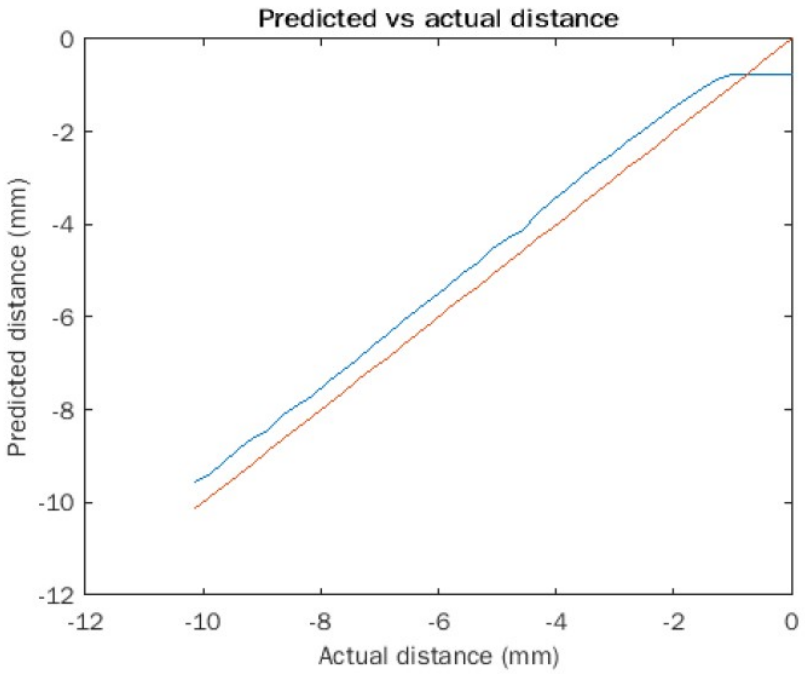


Figure 6: A model of our predicted as compared to measured distance, testing our sensor. The constant offset was accounted for in future models.

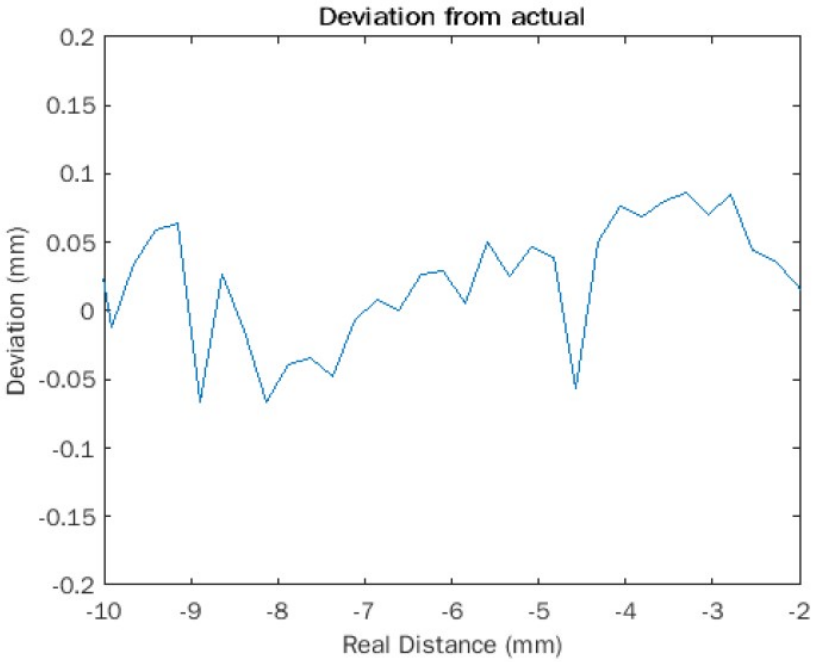


Figure 7: Error between predicted and measured distances.

However, the problem with using a linear hall effect sensor for measuring magnet position is

that it is affected by the magnetic field from the electromagnet. This appears as a constant offset in the sensor value as a function of the current pwm value (Figure 8).

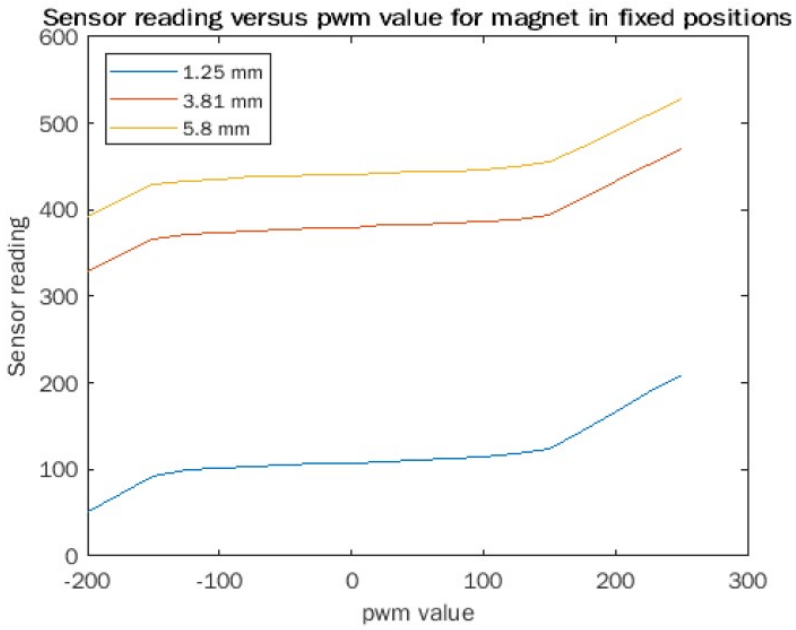


Figure 8: Sensor reading as compared to pwm value at varied magnet distances. Sensor reading seems to be affected by a constant offset, dependant on the pwm value.

We found that this offset was not affected by distance so could simply be subtracted from the sensor value before converting to a distance measurement.

2 Controller Model and Dealing with Nonlinearity

Given that the magnetic force is highly nonlinear in distance, any system which transforms position to voltage needs to account in some way for the system's deviation from a linear model. We originally tried the approach of linearizing about a point, running a linear model which took in the magnet position and output a voltage to the electromagnet. However, we found this to be very unstable in all but a very small region in both simulations and in the real system. Movement of only a few millimeters could easily cause the magnet force to increase by an order of magnitude, causing the magnet to fly away in one direction or the other. This was further exacerbated by the fact that the region of highest sensitivity of the sensor is right in the most nonlinear section of magnet pull, so moving further away to avoid the nonlinearity significantly degraded the positional accuracy, and made it much more susceptible to noise from the electromagnet. Given how poorly this model performed in simulation, we started thinking of whether there was a better approach.

We then realized that because the nonlinear part of the magnet force is in the conversion from force to current, we could run a linear controller that takes in position and outputs force and then simply map that force to a current value we knew to produce that force. From the data we collected on the magnet we were able to produce a relatively good approximation for that mapping equation and also account for the force required to counter gravity as part of it. The model we chose for this has voltage proportional to force and inversely proportional to the distance squared.

The final equation we used to relate desired force to voltage was

$$V = \frac{r}{b}[m(f+g) - \frac{c}{(d-x)^2}] \tag{1}$$

where r is resistance, b reltes current to force, m is the magnet mass, f is the desired force, g is the gravitational constant, c relates distance to force, and d is the offset between x=0 and the theoretical center of the electromagnet. This equation translates any desired force into a voltage that will hold the weight of the magnet plus apply the desired force.

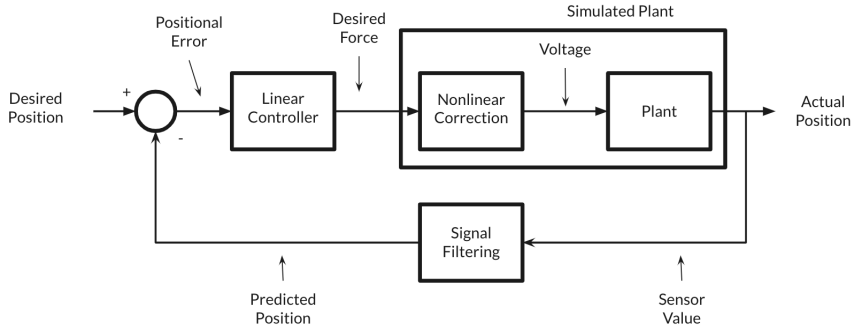


Figure 9: Block diagram of our system. In order to run a linear controller on the applied force we create a simulated plant with a nonlinear correction that makes the plant act linear despite being highly nonlinear. This allows the linear controller to control it more easily.

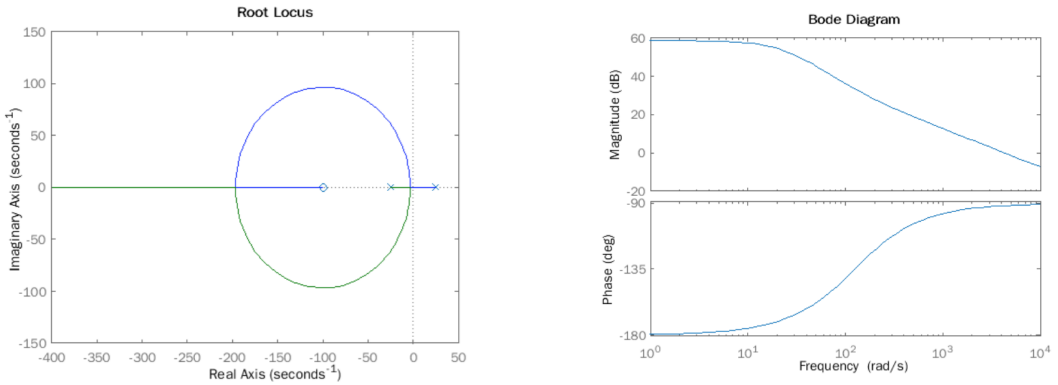
This approach reduces the problem for the linear controller to controlling a mass with no gravity by applying a desired force. We can group the real plant with this equation to form a simulated plant which takes in a desired force and putouts a magnet position. Our transfer function for this simulated plant, relating the distance of the magnet from equilibrium to the applied force was

$$\frac{X(s)}{F(s)} = \frac{N}{ms^2 - C} \tag{2}$$

with m being mass, $X(s)$ being position, in the complex domain, $F(s)$ being force, in the complex domain, and N and C being constants.

Since all the terms in this part are linear it is relatively trivial to come up with a controller that stabilizes the system. We chose to use a PD controller and place the zero to the left of the leftmost pole at about -80 on the real axis. This creates a locus which wraps around the two poles onto the real axis to the left of the zero (Figure 10). We chose a sufficiently high gain to get the poles to wrap around and become real again which provided a strong response on the time scales we desired (about .2 seconds of rise time in the simulation) while still being relatively overdamped so as not to oscillate too much and leave the relatively linear region. We found that with our simulator the PD controller was sufficient to achieve strong stability and little to no steady-state error, even though the linear system had some steady-state error.

This linear system performed extremely well in simulation, easily performing a step from -9 mm to -4 mm in less than half a second. Because the model accounts for the changing slope of force vs distance it is able to perform these maneuvers which take it across very nonlinear regions of distance (Figure 11).



(a) Root Locus

(b) Open Loop Bode Plot

Figure 10: Root locus and open loop Bode plot for our controller. By making the gain sufficiently high we are able to bring the poles very far to the left on the real axis and achieve a very stable system.

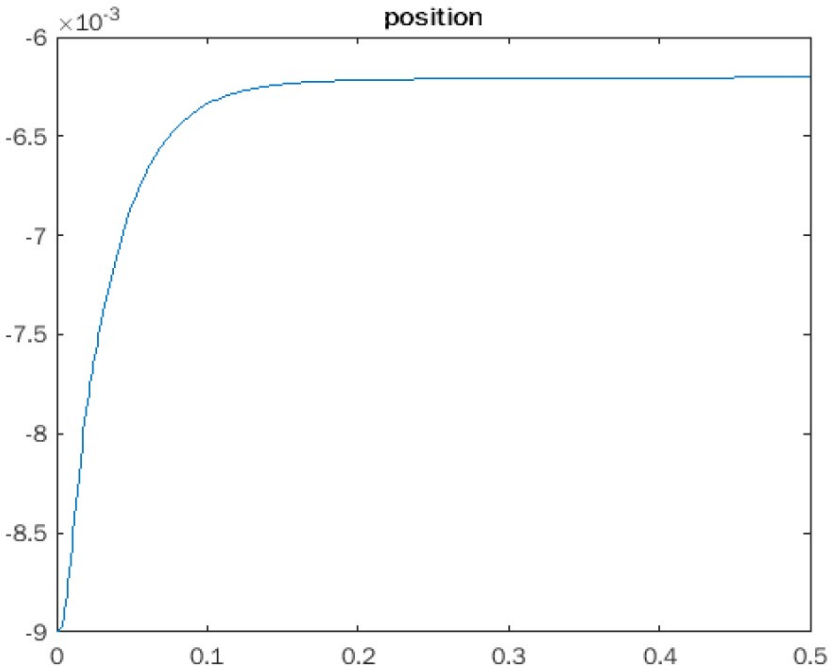


Figure 11: Position response of our simulation. The transient response stabilizes very quickly, even with initial values multiple millimeters away from equilibrium.

3 Tuning for Physical System

Given the amount of data that informed our simulation, we found that the real system performed very similarly to how we expected except for the difficulty of sensing position. We had determined earlier that the electromagnet applies a constant offset to position based on its pwm value and

accounted for that offset in our distance calculation. However, the biggest problem ended up being the incredible amount of sensor noise created by the electromagnet and the compounding effect on the position derivative. Even with three stages of filtering, the position signal tended to still be relatively rough which caused the derivative part of the controller to receive false information and react quite violently to jumps in sensor noise. This issue was further compounded by the fact that these spikes in the derivative would cause the magnet to react violently, creating another violent spike in position and yet another spike in voltage. This vicious feedback loop made it incredibly difficult to do any significant derivative feedback and made the system constantly at risk of devolving into increasing vibrations.

Additionally, the real positional error produced from this noise increased as the magnet got further away as a smaller change in the sensor voltage represents a larger distance change as distance increases. This meant that we needed to keep the magnet relatively close to the electromagnet which caused additional problems. While in the simplified model the electromagnet can push and pull the magnet, in reality any attempt to push the magnet away results in the magnet moving horizontally, getting out of the range of the sensor, then flipping over and shooting up onto the electromagnet. This is because the only thing holding the magnet in place horizontally is gravity pulling it down and some force from the electromagnet in the direction of its center. Once this force is directed away from the magnet it is no longer restoring and the magnet can move sideways. We tried several approaches to make the system work with push and pull control, but found it to be unfeasible for our setup and instead resolved the problem by preventing the voltage from going negative, effectively preventing pushing.

4 Reflection on this Approach and Practical Challenges

Given that this was a linear controls class, the easier approach to this model, which is the one we initially attempted, was just to linearize our model about a point. However, after running this through our simulation, we found that this model wouldn't allow it to respond to initial positions that were off by more than a fraction of a millimeter. This was further solidified by our first tests with the physical system, which, when we stabilized the magnet inside a very thin straw, were very stable, but only at very specific initial positions, and only with no outside disturbances.

That lead us to exploring the model we ended up adopting, which controlled the relationship between the distance and the necessary force, and then was able to incorporate part of the nonlinear relationship that was actually experienced to adjust the force output when we sent the actual current value to our electromagnet. This approach was more difficult and ended up being imprecise, though it gave us a much more stable system, capable of handling a much wider range in starting positions and possible disturbances. Still, with some tweaking, this model was much more generalizable to various target equilibrium points, and was more robust when considering disturbances than our original model had been.

That said, we were a little disappointed that we could never get our magnet to balance completely independently of the boba straw and stick. We expect this was due to not ending up with a perfect model of distance, as there was still some significant noise in the system, though it also could have been due to our setup, which was not exactly in-line with gravity, causing some force in the X-direction that we couldn't reconcile.