ENGS 26 Control Theory Final Project Report: Inverted Pendulum

James Monaco and Seth Loomis

Table of Contents

Introduction

Achieving a balanced inverted pendulum, or balancing a stick on its end, is no easy task for a person and is difficult implement in a machine without proper utilization of control theory. This project attempts to do just that, with a stick attached to a motorized car via a hinge, with a sensor to indicate the stick's deviation from the vertical. So, by utilizing the tools of control theory by using the sensor, feedback, and a compensator, the desired system behavior of a balanced stick can be easily achieved.

The compensator chosen was a proportional-differential (PD) compensator. A proportional compensator is a given, and its contribution results in a small rise time, but a large overshoot. A differential controller is also implemented in order to decrease the overshoot and settling time. Overall, this gives a system that reacts quickly (small t_r), settles quickly to its steady state (small t_s), and hopefully has only a small amount of overshoot. The only downside to this PD controller is it might have a significant steady-state error given a step disturbance, but that can be compensated for by adjusting the reference voltage until the desired steady-state is achieved. We used car #3.

System Modelling and Characterization

Our first objective was to develop a system model for our car-pendulum-sensor system. To accomplish this, we utilized different methods to characterize the various components before combining these characterizations into an entire system model. A high-level block diagram can be seen below in figure 1.

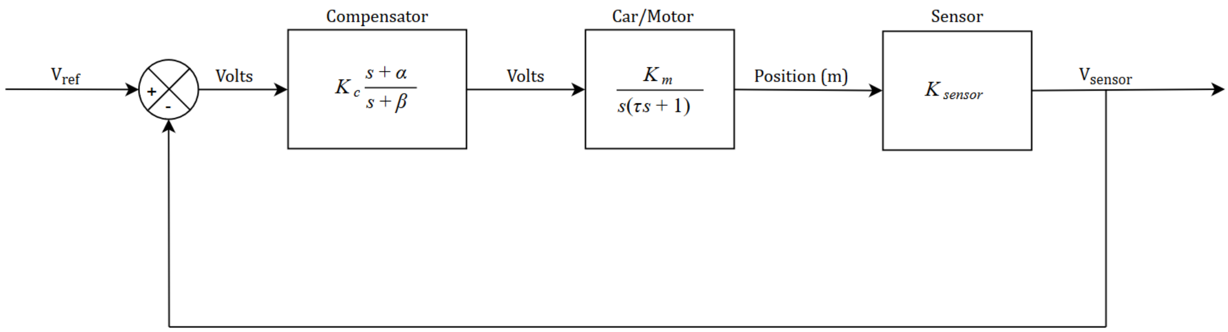

Figure 1: A block diagram of the closed-loop compensated system, with general transfer functions for each block. The system is not yet characterized, so variables are put in place. The uncompensated system simply has the compensator equal to a unity multiplier.

Sensor Characterization

The sensor had a range of -8.5V to 8.5V, depending on the angle of the stick. This range did not cover the full arc that the stick could trace out; the sensor became saturated at either extreme well before the stick hit the stops on either side. But, when unsaturated, the sensor gives a linear response of angle vs. voltage. With some minor trigonometry, this can be translated to a position in, voltage out sensor. In order to characterize the sensor, the slope of volts/meter had to be calculated within the linear region (see figure 2 below).

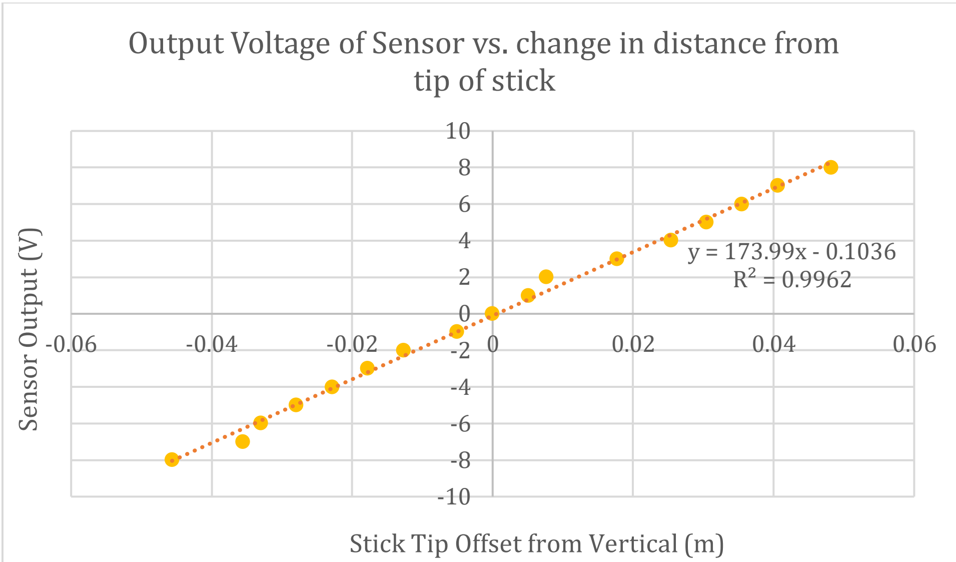

Figure 2: Calibration curve used to characterize angle/position sensor. Shown here, the sensor gives 174 V of difference for every meter that the tip of the stick moves.

The position referenced was that of the tip of the stick. In order to create this calibration curve, measurements were taken with changing angle of the stick at every 1V change of sensor voltage. The horizontal change in position of a fixed point on the stick was measured (0.394 m from pivot point) as the stick was moved, and simple trigonometry was used to then translate this to a position of the tip of the stick.

$$\text{Tip Position} = sin(\, arctan \frac{\text{Horizontal offset}}{1m})$$

As seen in figure 2, this analysis gave us the the sensor output 174 V per every meter offset of the tip of the pendulum.

Motor and car Characterization

We also needed to characterize our car and motor to figure out how the system reacted to a supplied voltage. To do this, we supplied a step voltage input directly to the motor by creating a voltage divider across a power rail. The magnitude of our step input was 3.20 V. We videotaped the response of our car to this step voltage and loaded the video into Open Source Physic's "Tracker Video Analysis and Modeling Tool" software, seen above in figure 3.

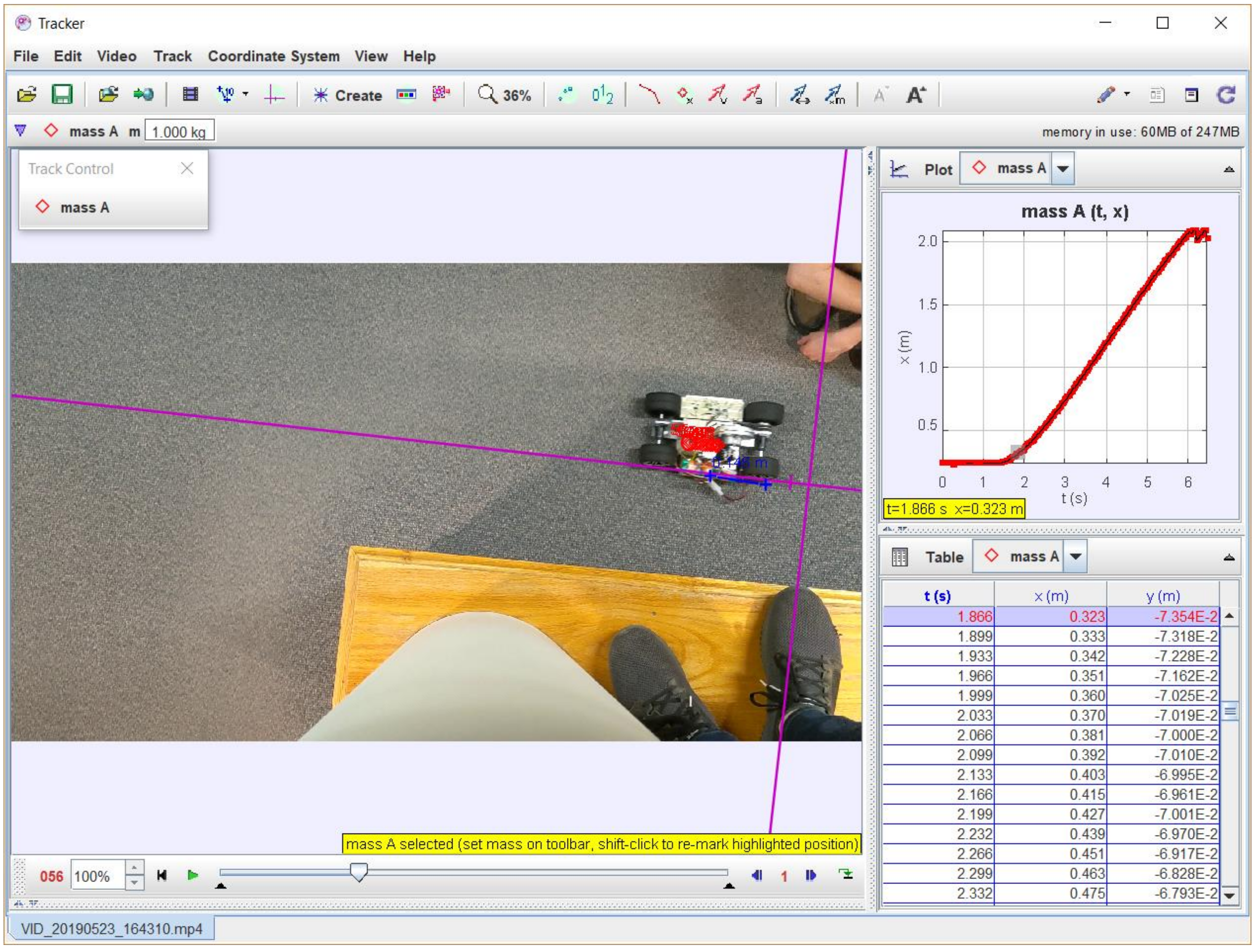

Figure 3: Tracker software used to characterize the motor and car

This software gave us the frame-by-frame position of the car in response to the step input. In order to characterize this response, we decided to analyze the velocity output of the car as it is more closely related to the voltage input than the time-dependent position output. Using MATLAB, we took the derivative of this data to get the velocity and plotted the output (see figure 4 below).

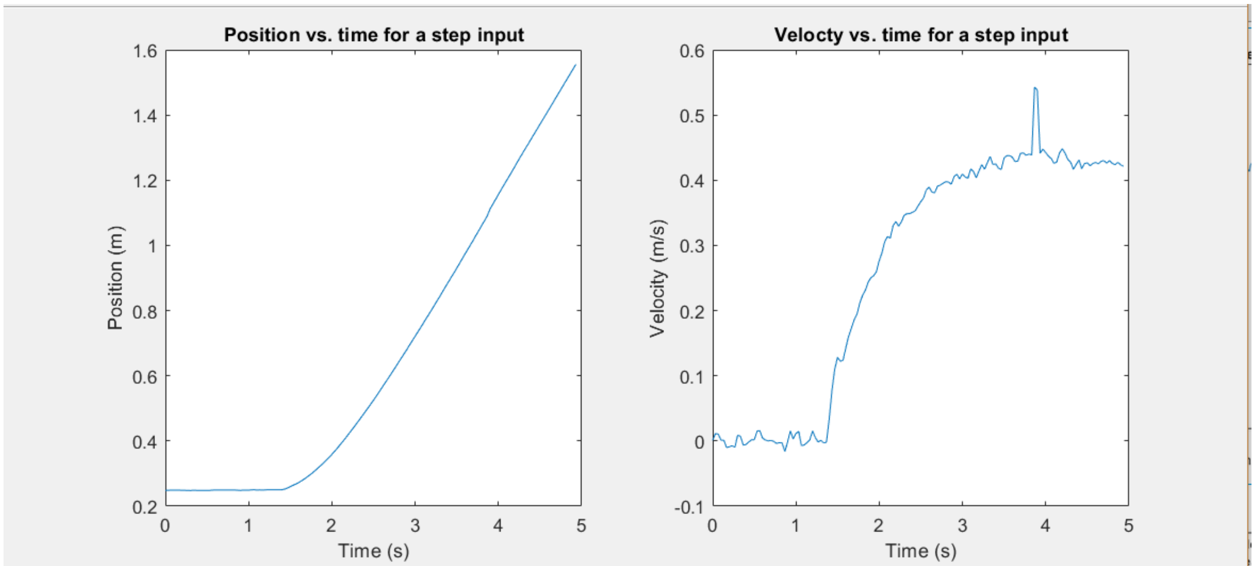

Figure 4: Position and velocity plotted over time for a step input of 3.20V to the car. Data was taken from the Tracker software and plotted in MATLAB.

Without considering the significant noise towards the end of the plot, this response appears to approximately be first order. That is, we estimate the transfer function of the motor to be of the form:

$$K_m/(Ts+1)$$

From the plot we see that the motor steady state for a step input of 3.20 V is approximately 0.4412 m/s (see figure 5 below). If we gave the motor a step input of 1 V, then K would equal our steady-state value. Thus, for our response, K = 0.4412 m/s / 3.20 V = 0.1379 mV/s.

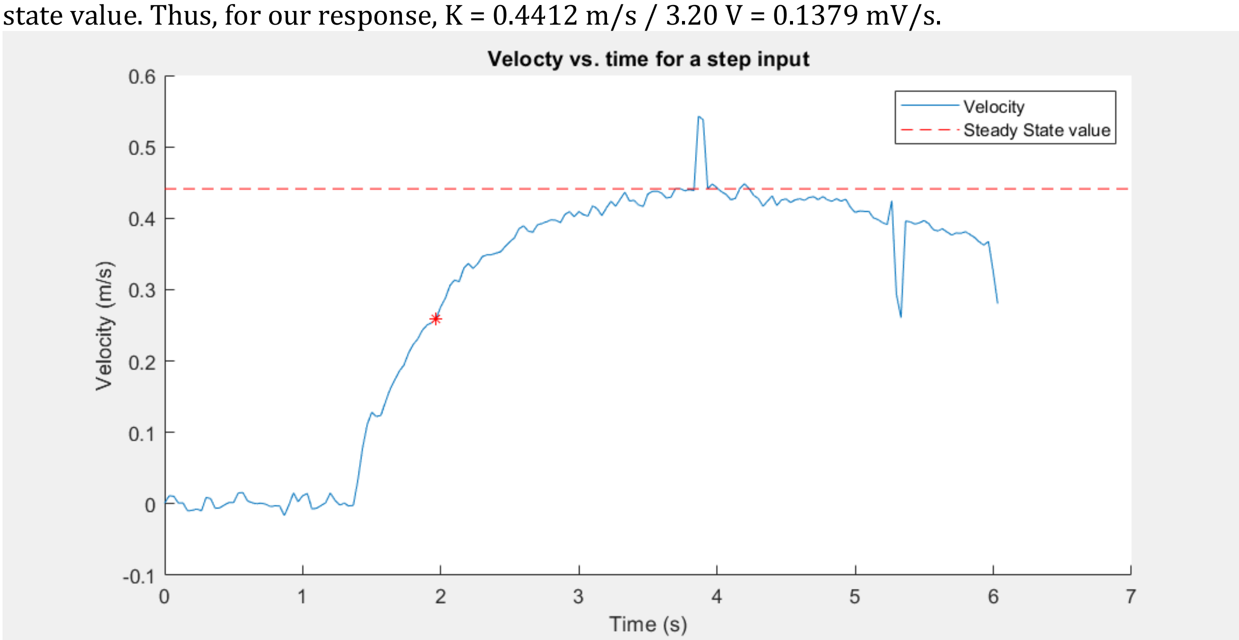

Figure 5: The car's velocity to a step input. The steady state is 0.4412 m/s, marked in a red dotted line, and 63% of that is value of marked in a red dot.

For a saturating first order response, the time constant, τ, is equal to the time at which the response is equal to 63% percent of the steady state value. As our response has a steady-state value of 0.44 m/s, T is the time at which the response reaches 0.63*0.4412 m/s = 0.278 m/s. Using

MATLAB we determine that this happens at about t = 1.9992 s. Considering that the step response is delivered at t = 1.3662 s, we can find our time constant τ = 1.9992 s – 1.3662 s = 0.633 s.

So, our motor transfer function, taking a voltage and outputting a velocity is given by 0.1379/(0.633*s + 1).

Our end goal was to determine how the position of the car changed with the voltage applied to the motor as to ensure that the position of the car was always under the position of the tip of the pendulum. To do this, we need the motor's transfer function to convert the voltage input to a position output. Position is the derivative of velocity, and fortunately the s-domain has a convenient way to perform integration in the time domain: dividing by s. So, our final motor transfer function, taking an input of voltage and outputting a position is given by

$$G_m = \frac{0.1379}{s(0.633s + 1)}$$

After all characterizations have been made, the block diagram can be filled in, seen in figure 6.

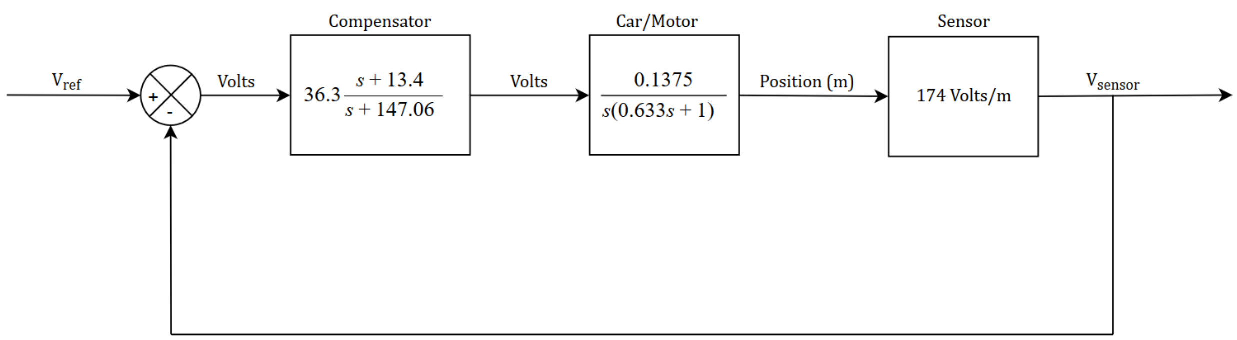

Figure 6: A block diagram of the closed-loop compensated system, with transfer functions for each block. The final compensator is shown in the compensator block.

With the characterization of all the components of the system complete, we were able to construct a complete block diagram. Our system took a reference voltage, the desired output of the sensor, in the "Implementation of Compensator" section we discuss how this reference voltage was set. From this reference voltage was subtracted the actual sensor voltage to generate our error signal, the voltage that is passed into our compensated, which generates a motor voltage. The motor transfer function coverts the motor voltage generated by the compensator into a meter position of the car. Assuming a frictionless connection between the pendulum and the car, the change in position of the car will equal the change in position of the pendulum tip relative to the car, generating a new sensor voltage to feedback into our system.

Analysis of Stability and Performance of uncompensated system

Time domain analysis

We attempted to characterize some aspects of the uncompensated system. To analyze the uncompensated system's settling time, we pushed the pendulum to the maximum disturbance position where its weight was supported by the sensor casing, then turned on the motor and intended to time how long it took the car to get under the tip of the pendulum. However, we discovered the uncompensated system was not able to recover from such a disturbance. As the uncompensated system is unable to reach a steady state, the settle time is an irrelevant measurement, as well as the steady-state error, as there is no steady-state of an unstable system.

By simply watching the response of the uncompensated we were able to determine that it was unstable, as it never came to any steady state. It either ended in wild oscillations until it was manually stopped or ended with the stick slammed to one side and the car careening down the hallway at its maximum velocity, unable to recover.

Develop Specifications

Time domain specifications

The main specification is that the controller implemented allows the pendulum to become balanced in a reasonable amount of time (small settle time) and reacts quickly to disturbances (small rise time). The implicit specification here is that it creates a stable system. Additionally, we require that the pendulum never slam into the edge of its casing and saturating the sensor voltage when a disturbance is encountered, meaning the overshoot should be kept small enough to prevent this.

To achieve this balance, we decided to make a specification that the rise time of our response, from 0% to 100%, must be less than the time it takes for the pendulum to fall from equilibrium position to hitting the sensor casing when the motor is off. This ensures that the system is always able to catch a falling pendulum. To measure this, we placed the car on a flat surface and filmed the pendulum being dropped from its equilibrium position and then analyzed the video to find the time it took to fall. From our analysis we determine that time is 0.385 seconds, thus our maximum rise time specification is:

$$t_r < 0.385 \text{ s}$$

Besides a fast rise time, steady state error can be another system specification. One approach is to configure the system to have zero steady-state error and set the reference voltage equal to the sensor voltage at equilibrium. This would likely require a PID controller; the integrator would help reduce the steady-state error.

An easier approach, although less elegant, is to simply use a PD controller and accept some stead state error. The reference voltage can be adjusted to compensate for the steady-state error. Although the reference voltage would not be equal to the sensor voltage at equilibrium, the steady state error would ensure the steady-state sensor voltage is still equal to the equilibrium voltage, giving the desired balanced behavior. In other words, our primary goal is to make sure that the steady state of our system occurs when the pendulum is directly above our car, this can either be achieved by setting the reference voltage to the pendulum's equilibrium position and achieving a zero steady state error, or it can be achieved by setting the reference voltage offset from the pendulum's equilibrium position amount and achieving a steady state error equal to that same amount. So, we consider zero steady state error to be an unnecessary specification and can calculate a range of acceptable steady state errors for which we can adjust our reference voltage to make sure that this still puts the systems steady state at the pendulum's equilibrium position.

With a PD compensator implemented and steady-state error accepted, it is important that the steady-state error achieves the desired sensor voltage at steady state without an unreasonable reference voltage. The reference voltage required must be within the linear range of the sensor (-8.5V to +8.5V) and must be reasonably centered to give a symmetric motor response. If the reference voltage is biased too far to one side, that would cause the error signal to have more room on one side than the other. For example, if the sensor voltage at equilibrium was -1.1V, a system with an unreasonable amount of steady state error might require a reference voltage of -7V to achieve a -1.1V steady state at the sensor. This leaves an error voltage of 1.5V on one side and -15.5V on the other. This would cause the motor to have much more power in one direction than the other and is not an acceptable compensator. Ultimately, the motor must have enough power to recover the stick from both sides. From our measurements, it takes about 5V on either side for the motor to be able to recover the stick from its most extreme position to equilibrium. Therefore, the reference voltage should ideally give at least 5V on either side of the equilibrium voltage of the sensor.

So, in order to characterize how much steady state error is too much, causing an asymmetric error signal to the motor, the equilibrium voltage of the sensor must be known. With the stick balanced straight up, the sensor reads ~ -1.1V. With a 5V error signal required, this gives a reference voltage range of -3V to +3V. Below, the acceptable range of steady state error is calculated, normalizing the sensor's range:

$$\frac{(-1.1+8.5)-(+/-3+8.5)}{-1.1+8.5} = -55\% \text{ to } +27\%$$

Another factor we considered was the maximum overshoot of our system response. Our reference voltage acts as a step input to which our system responds with position, as dependent on our sensor. Our only concern with the overshoot is that we must make sure that the system does not overshoot so far that the pendulum slams into the sensor casing, causing the pendulum to no longer be balanced, or push the pendulum into the non-linear region of the sensor. However, if we ensure that the maximum overshoot is under 100%, then the response for any pendulum position in the linear-region of the sensor will remain within the linear region, and any response to a pendulum position that is in the non-linear region will eventually die down until the pendulum settles into the linear region of the sensor. Thus, our only specification for the maximum overshoot to make sure that the pendulum remains balanced is

$$M_p < 100\%$$

However, a lower percent overshoot will give a smoother response and is therefore more desirable. Note that for a second order system as we are estimating ours to be, the percent overshoot is given by

$$PO = 100 \cdot e^{\left(\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}\right)}$$

So, if a second-order approximation is made, in order for us to have a percent overshoot above 100%, the damping ratio must be negative which results in an unstable system. System stability is a criterion that we consider in the frequency domain specification and so, we consider system stability to be a more fundamental specification than the maximum overshoot as a stable system implies that our percent overshoot specification is met.

A final specification is settling time. The stick must be balanced in a reasonable amount of time. Qualitatively, we determine that amount of time to be about 1 second or less. Any time significantly more than that we would consider to be an inadequate response as we are hoping for a fast and smooth system, not one that oscillates significantly.

Frequency domain specification

We require that our pendulum is always balanced, so our system must be stable, as an unstable system would result in the cars correctives oscillations growing, eventually slamming the pendulum back and forth between the sides of the sensor casing. Thus, we require that our compensated system has no poles in the right-hand side of the s-plane to avoid instability.

As an additional precaution, we sought to make sure that no portion of the root locus for our system entered the right-hand side of the s-plane. This allows us to freely adjust the gain of our system without fear of triggering instability.

We can also translate our time domain specifications into frequency domain specifications. If a second-order system approximation is made, a settling time of 1 second can be translated to values of damped frequency (ω_d), relating to both the damping ratio (ζ) and natural frequency (ω_n). Thus:

$$t_s = \frac{\pi}{\omega_d} = \frac{\pi}{\omega_n\sqrt{1-\zeta}} < 1 \text{ sec}$$

As explained previously., we require that our system has a rise time of less than 0.385 s. The rise time of a second order system is given by

$$T_r = \frac{1}{\omega_d}\tan^{-1}\left(-\frac{\sqrt{1-\zeta^2}}{\zeta}\right)$$

Thus, we require that for our compensated system

$$tan\,0.385 * \omega_n\sqrt{1-\zeta^2} < -\frac{\sqrt{1-\zeta^2}}{\zeta}$$

We can solve for inequalities of both ζ and ω_n if we assume a second-order approximation of the system. Solving for these inequalities gives: $\zeta < 0.89, \omega_n < 6.94$ rad/s.

Compensator Design and Design Process

Root locus method

To assist us in the design of our system's compensator we used MATLAB's Control System Designer app. We loaded the uncompensated open-loop transfer function into the system. From figure 6 with a unity compensator, the open-loop uncompensated response can be expressed as:

$$TF = 174 * 0.1379 * \frac{1}{s(s*0.633+1)} = \frac{24.08}{s(s*0.633+1)}$$

In the Control System Designer interface, we set the previously determined specifications for rise time in the step response window so that as we adjusted the compensator we could visually see if the response met our specifications.

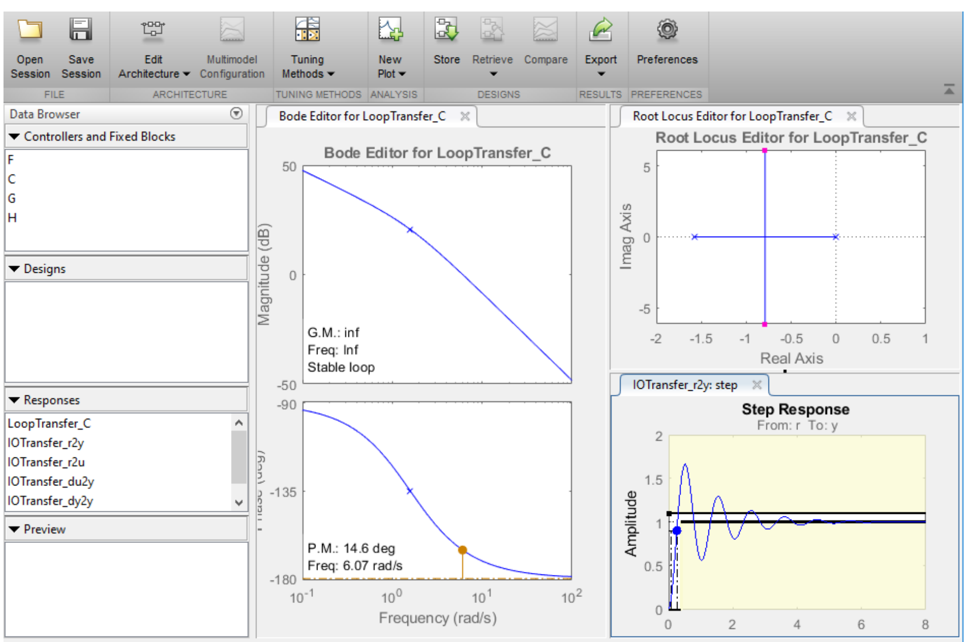

Figure 7: Uncompensated SISO tools window

Through experimenting in Control System Designer, we developed a sense for how the placement of poles, zeros and gain changes affected the system.
We found that a lead compensator of the form

$$\frac{3.8374e06\ (s+0.3162)}{(s+5.719e06)}$$

Gave an adequate response with a rise time of about 0.177 sec, which is within our 0.385 specification. This design also gave us an infinite gain margin and a phase margin of 92.8 degrees, giving us significant headroom to make adjustments without fear of instability.

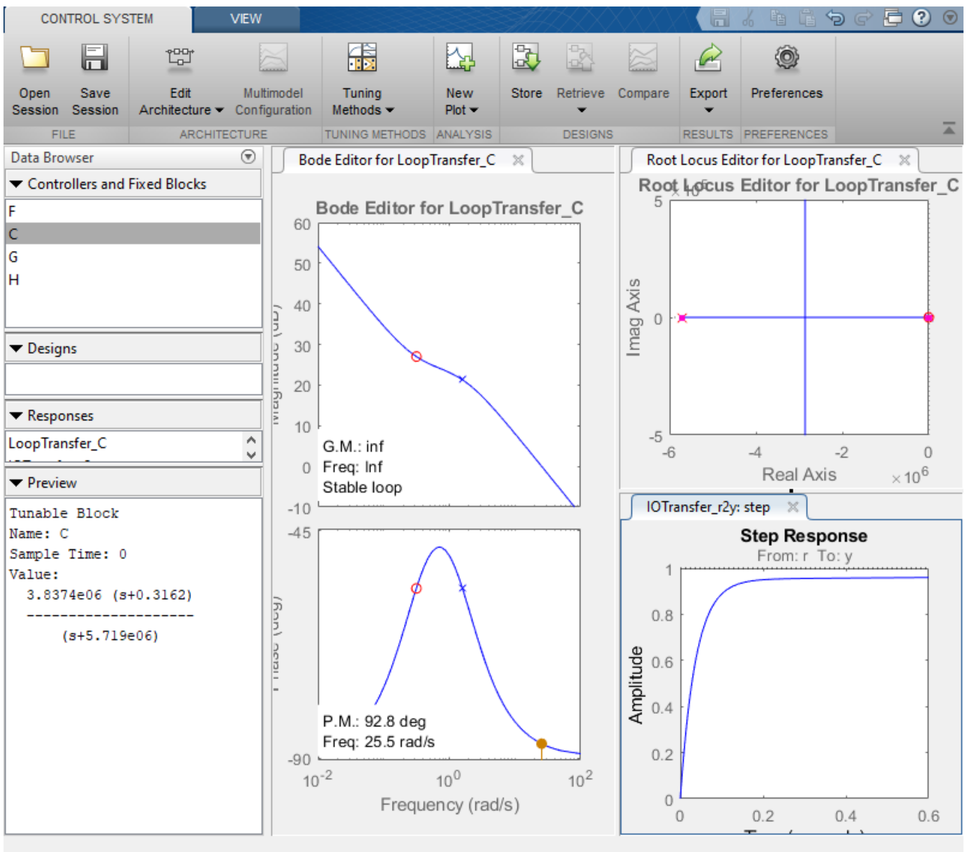

Figure 7a: Compensated SISO tools window

Additionally, this compensator gave a theoretical overshoot over 0%, and a stable response, meeting all of our design specifications. So we decided to implement this controller into our system as a starting point.

Implementation of Compensator

Circuit

Our compensator is a lead compensator, which can be achieved with a standard PD controller with a stop resistor, shown below.

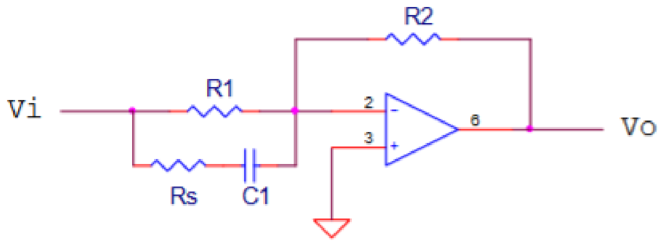

Figure 8: The circuit diagram of the PD controller with a stop resistor. This diagram is taken directly from Lab 5: Magnetic Levitation.

$$K_p = \frac{R_2}{R_1}, \qquad T_D = \frac{1}{R_1 C_1}, \qquad \alpha = \frac{1}{R_S C_1}$$

$$G_c(s) = -K_p\left[1 + \frac{s}{T_D}\left(\frac{\alpha}{s+\alpha}\right)\right] = -K_p\frac{(T_D+\alpha)s + \alpha T_D}{T_D(s+\alpha)}$$

Figure 9: The unsimplified transfer function of the compensator shown in the previous figure. The derivation of this transfer function was taken directly from Lab 5: Magnetic levitation and is therefore not shown here.

The above transfer function can be simplified further:

$$G_c = \left(\frac{R_2}{R_1} + \frac{R_2}{R_s}\right)\frac{s + \frac{1}{R_s C_1}}{s + \frac{1}{R_s C_1 + R_1 C_1}}$$

So, to implement the controller we designed in MATLAB we require that

$$\left(\frac{R_2}{R_1} + \frac{R_2}{R_s}\right) = 3.84e6$$

$$\frac{1}{R_s C_1} = \mathbf{5.72e6}$$

$$\frac{1}{R_s C_1 + R_1 C_1} = \mathbf{0.3162}$$

To approximate this compensator with component values that are available, we chose:

C_1 = 1n

R_s = 150

R_1 = 22M

R_2 = 681K

These were the closest values we could get to estimate our compensator designed in MATLAB. This gave our actual compensator of the form.

$$G_c = \frac{40(s + 45.45)}{(s + 6.67e6)}$$

Our actual compensator sacrificed some gain from the theoretical value in exchange for a larger zero.

Deviation from theoretical compensator

When we tested the compensator designed above in our system, we realized that the response was not exactly what we had expected. Even though the compensator gave us an almost ideal response in MATLAB, the estimations we made in characterizing our system, the compromises we had to make in order to use reasonable component values in our circuit, and the unpredictability/non-idealities of real life made our response less than ideal. While the system was able to balance the pendulum with this estimated compensator, it was extremely jittery and appeared to be only marginally stable. That is, the system never reached a steady state, rather, the car rapidly moved from side to side. Alternatively, the gain may have been too high, resulting in fast oscillations while balancing. So, in order to achieve a more fluid response with a steady state of the pendulum directly above the car, we adjusted our compensator.

We determined that the form of a lead compensator was still adequate as in our initial design, but the exact position of our pole, zero, and gain were not appropriate. We developed our adjusted compensator through many iterations, primarily looking at much lower gains. We also iteratively adjusted component values and observed the new response behavior. We then debated whether the response had improved by demonstrating either a faster response, smoother behavior, or a stronger tendency to a steady state with the pendulum at equilibrium. With each change, the reference voltage was also adjusted to attempt for a balanced response. If the response had improved, we kept the previously made change and debated on what component change we believed would further improve the response. This process got better with time as we developed a sense for how changes in each of the component values affected the system response.

After repeating this process, we determined that the following component values gave an adequate response.

C_1 = 6.8u
R_s = 1k
R_1 = 10k
R_2 = 33k

These component values correspond to a compensator of the form:

$$G_c = \frac{36.3(s + 13.4)}{(s + 147.06)}$$

Other Considerations

When we first characterized our sensor, 0 V from the sensor corresponded with the pendulum being straight up above the car. However, as we continued testing, we noticed that our reference voltage (the voltage of the sensor when the pendulum is straight up) began to drift negative. The maximum voltage this reached was –8.5 V, before setting at –5 V. We weren't able to determine the cause of this, but we assume it was due to play in the sensor casing. Our initial solution was to set the reference voltage in our circuit to –5 V also. However, as the sensor voltage rangers from approximately –8.5 V to +8.5 V, this means that our error voltage would range from –3 V towards the negative side, but 13 V towards the positive side. This gave us an asymmetric response that causes the car to be unable to correct for large disturbances towards the negative side.

In order to solve this, we made several adjustments. First, we drilled a new hole in our pendulum to slightly adjust its angle to the vertical. This relocated the center off mass of the pendulum such that the stick rested at a position that was more centered to the sensor's voltage.

The final voltage of the sensor at equilibrium was -1.1V. We then adjusted the reference voltage to account for this change in the equilibrium position.

Results of Compensated Design

The compensated design was shown to be stable and had a fast response with not too much overshoot. A plot of the control effort voltage going to the motor best illiterates the stability of the system (see figure 10).

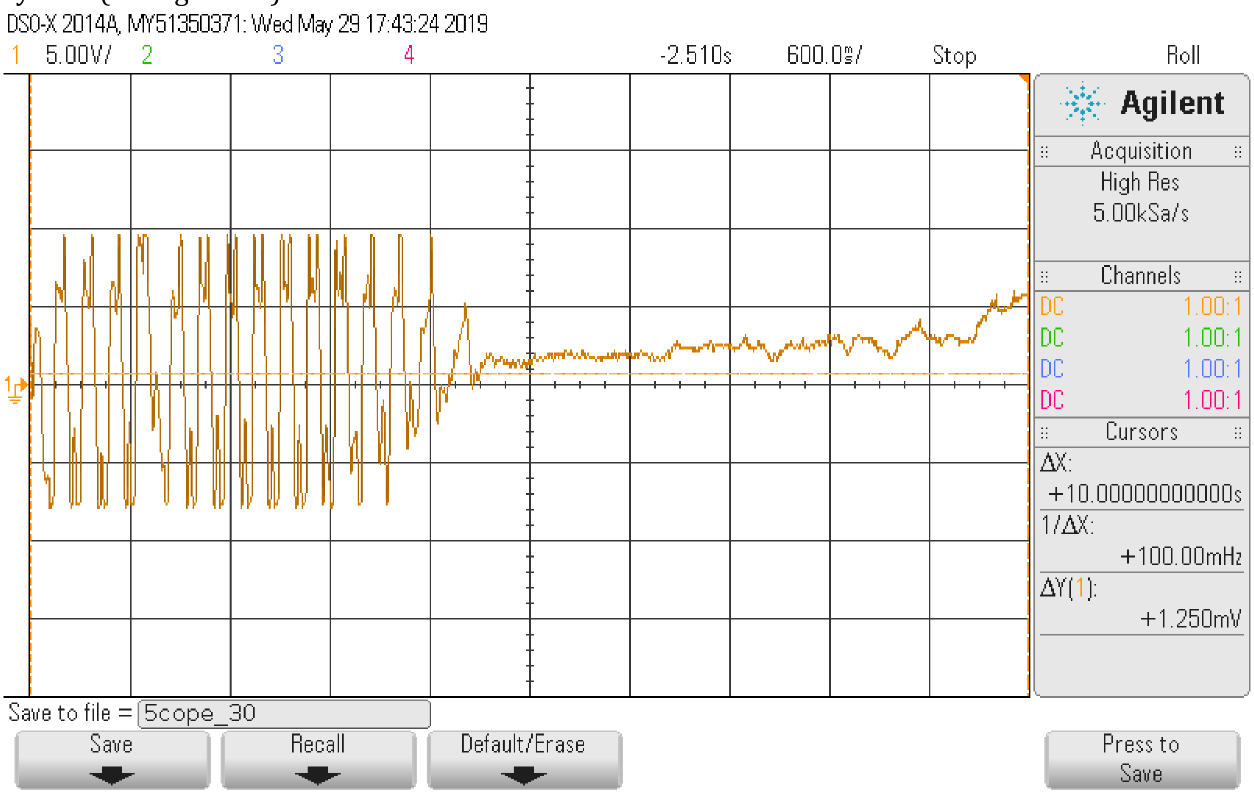

Figure 10: Control effort voltage going to the motor as it reaches steady state.

The above plot of control effort is the best plot we captured that shows the system dampening oscillations. The motor reaches a stable and clear steady state; very little drift is present, and the stick was balanced in when the control effort evened out.

However, this system did require a reference voltage of -3.7V to get the desired steady-state sensor voltage of -1.1V; this is a steady-state error of:

$$\frac{(-1.1+8.5)-(-3.3+8.5)}{(-1.1+8.5)} * 100 = 35.1\% = e_{ss}$$

This did lead to an asymmetric response, as we had 4.8 V of headroom for the error signal on the negative side but 12.2V on the positive side. However, we determined that for reasonably small disturbances, the system was still able to stabilize with this configuration. For large disturbances, it still recovered remarkably well one the side that had 12.2V of error signal to work with. This was just outside of our goal of $e_{ss} < 27\%$.

Below are scope measurements of the sensor's voltage for a step input

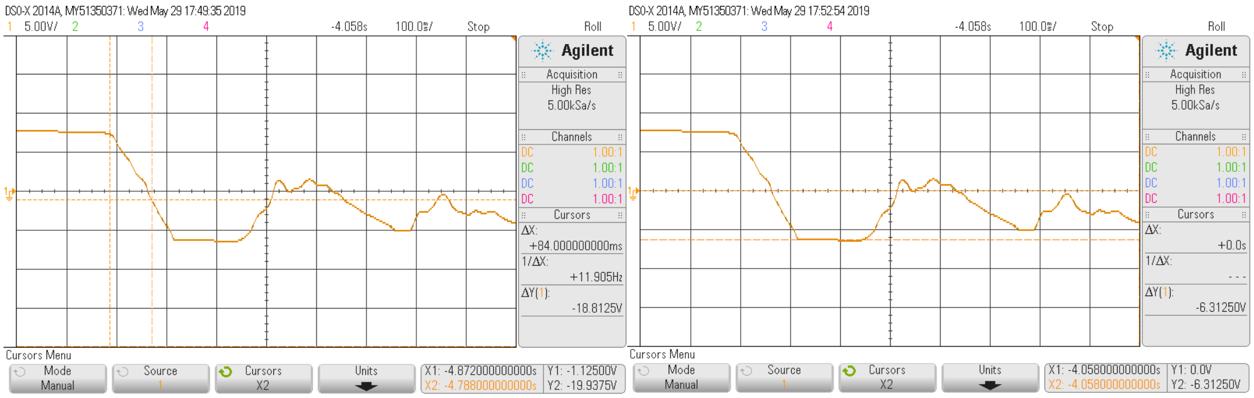

Figure 11: Scope measurements of the sensor's voltage for a step input, as steady state is reached. Cursors show 100% rise time and overshoot.

From the oscilloscope, 100% rise time was measured to be 84ms. That is, time it took the sensor to get to -1.1V from startup. This is much less than the 384ms specification, so the rise time specification has been met.

The sensor overshot to -6.3V. calculating the percent overshoot is as follows:

$$\frac{(-1.1+8.5)-(-6.31+8.5)}{-1.1+8.5} * 100 = 70\% = M_p$$

This is less than the 100% overshoot stated in the specifications, so the overshoot specification has also been met.

The settling time could not be captured by the oscilloscope due to limitations of the lab set up. Instead, the time it took for the car to reach a steady state was measured several times with a stopwatch, and the result was measured. It was found that settle time was 0.49 seconds. This is less than half the specified settle time of 1 second, and so this specification has been met.

Analytical Analysis of the Compensated system

Bode plots of the closed-loop system shown in figure 6 above are shown below.

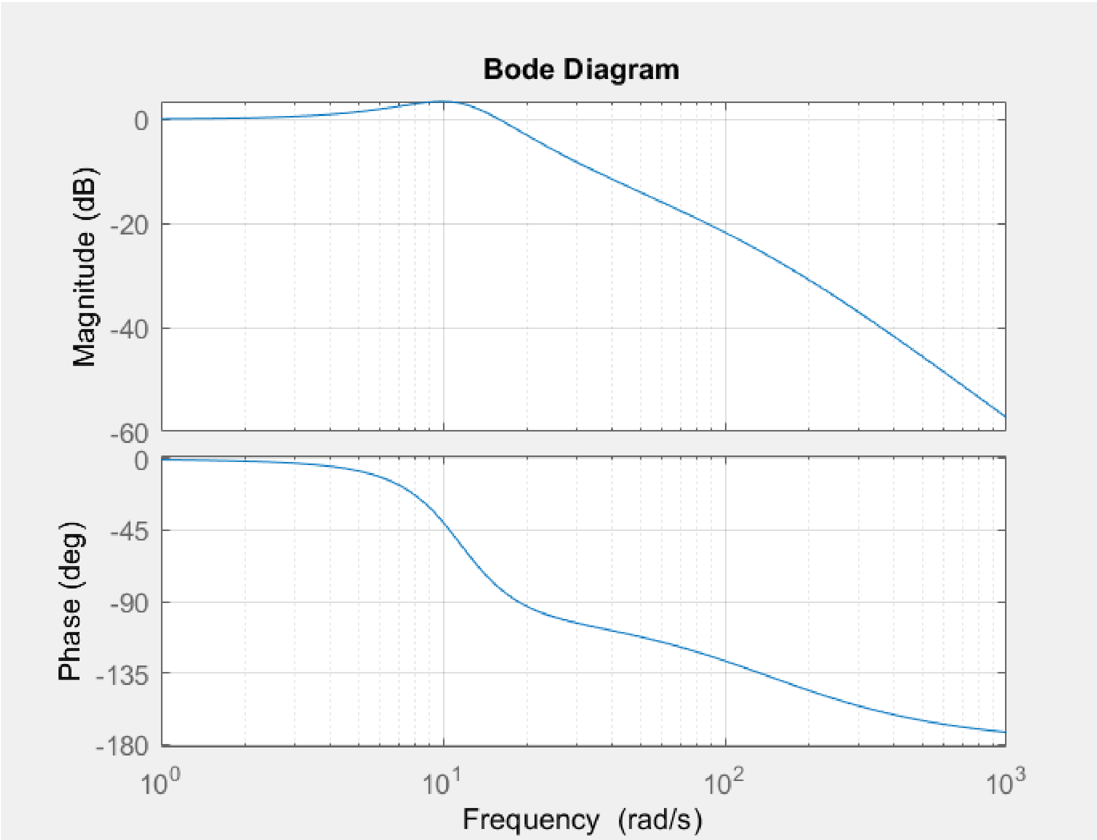

Figure 12: Bode plots for the closed-loop compensated system. It should be noted that the system has an infinite gain margin, a 98 ° phase margin, and a gain crossover frequency of 15.8 rad/s.

The system has a bandwidth from 0 to 20 rad/s, as found by looking at the bode plot. This system does not meet the criteria of a maximum 1dB maximum increase from DC gain. DC gain is 0 dB, while the peak gain is 3.4 dB at 10 rad/s. Despite having a peak above 1dB, our system was still able to successfully balance the pendulum without interference from this characteristic, so we do not consider this peak to be significant. The phase and gain margins do meet specifications, as the system has a 98° phase margin and an infinite gain margin, giving us significant headroom for adjustments.

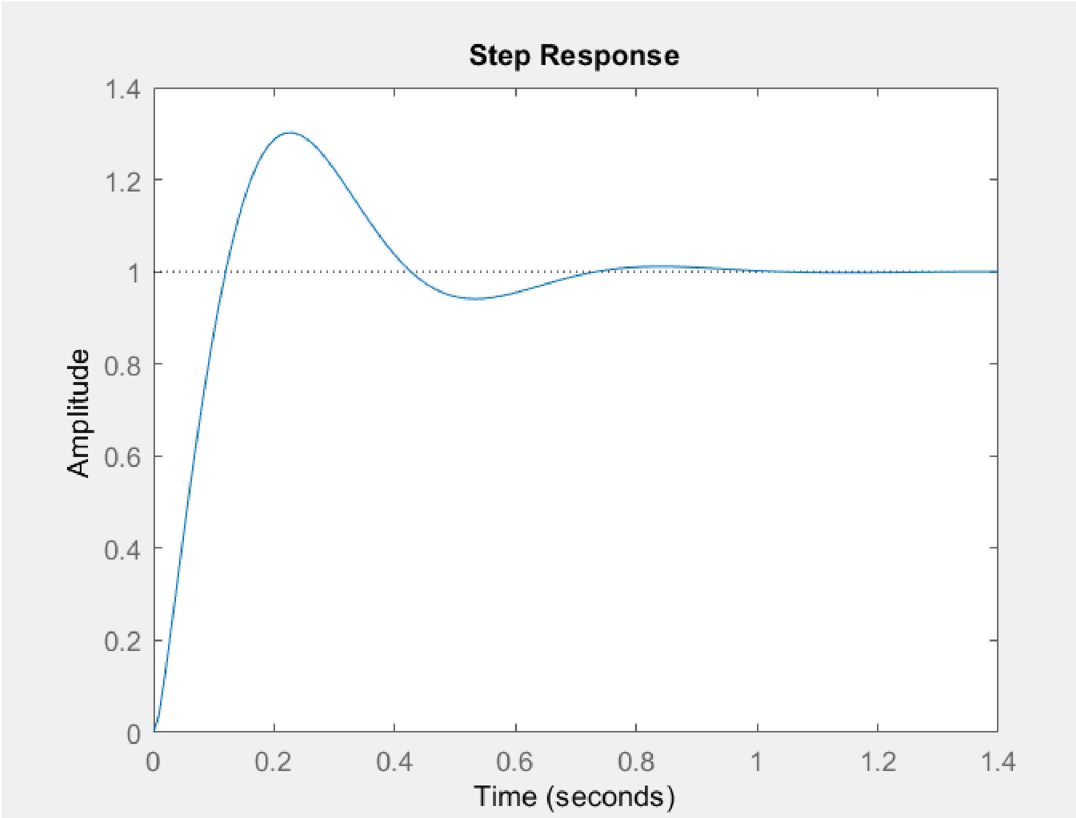

Figure 13: The unit step-response of the compensated system.

Above is the simulated closed-loop step response of the system. These results are only semi-consistent with the experimental results. Matlab shows a rise time of 88.2ms, which is remarkably close to the measured 84ms. Matlab also gives a settling time of 0.67 seconds, which is close to the measured 0.49 seconds measured considering the likely inaccuracies of measuring settling time with a stopwatch. But, Matlab gives a 30% overshoot and a 0% steady-state error. The discrepancies between Matlab and the real world are likely due to non-idealities such as backlash in the mechanical system, the asymmetrical behavior of the motor, slippage of the wheels on the carpet during fast movements, or other non-idealities that were not put in the system model.

Conclusions

This project was largely successful, with the settling time specification halved (t_s < 1 second specified, 0.49 seconds achieved), the percent overshoot specification achieved (M_p < 100% specified, 70% achieved), and the rise time specification well achieved (t_r < 385ms seconds expected, 84ms achieved). Both the gain and phase margin specifications were also achieved. The only blemish on this project would be the steady state error of 35%. This led to an asymmetrical response, with the car recovering the stick better in one direction than another. Another blemish is that the bode plot gave a peak gain ~2 dB higher than specifications required.

If this project was to be done again, an integral controller would have been added as well. This would have decreased the steady-state error, allowing the reference voltage to be set closer to that of the sensor's equilibrium voltage, and therefore allowed for a more symmetric response. The integral controller may have increased settling time and rise time, but that would have likely been affordable, considering how well those specifications have been met. The addition of this integral controller may also increase overshoot, but because the specifications are that overshoot is less than 100%, that may also be an acceptable loss if steady-state error is lessened.

Overall, this was a successful project in which both members of the group learned a lot about control theory. Application and hands-on learning are what solidified understanding of the material of this class, and iteratively working on the circuit is what gave much more intuition behind controller's effect on the response. It was also helpful to see the difference in simulating a system and implementing the system in real life, and witnessing the unexpecting discrepancies that occur.