MEMORANDUM

To: Charlie Refvem, Department of Mechanical Engineering, Cal Poly SLO
crefvem@calpoly.edu

From: Antonio Ventimiglia — ventimig@calpoly.edu
Caiden Bonney — clbonney@calpoly.edu

Date: 10/30/2025

RE: **Lab 0x03: Closed Loop Control**

The implementation of closed loop control for Romi's motors consisted of an incremental complexity approach. The controller was made sequentially in the following configurations:

- P: proportional controller,
- PI: proportional + integrator controller
- PID: proportional + integrator controller + derivative controller
- PI & K_ff: PI controller with feed forward
- PI & K_ff & V_droop: PI controller with feed forward and battery voltage droop

The following information is formatted in a "story" format to illustrate the development of the team's closed loop controller functionality. It is not mentioned, but all gain values are changeable from the terminal on the interfacing PC. All plots are also created automatically.

P-Controller:

Initially, only proportional control was implemented. With an initial proportional gain of $K_p = 1$, Romi exhibited a rapid forward and backwards motion that we call "jittering." Figure 1 shows the data gathered for such experiment. From here on out, all data presented will be the left wheel's data. The team will ensure that all qualitative data is consistent across both wheels.

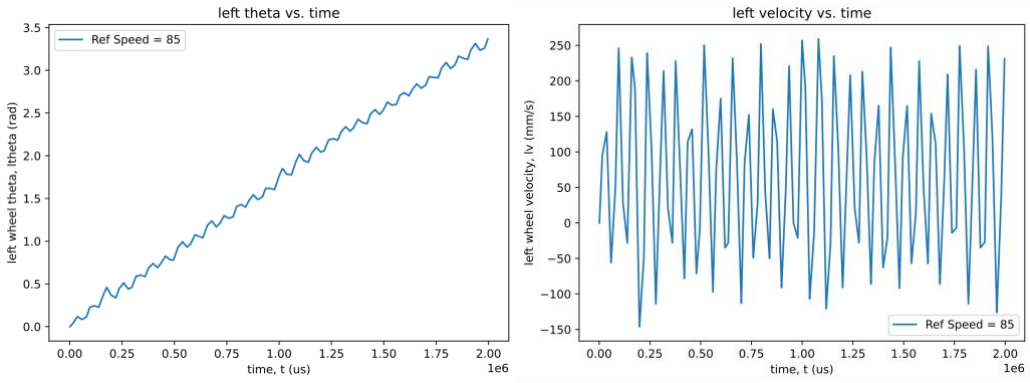

Figure 1. Proportional Controller: $K_p = 1$

It was determined that the jittering was occurring because the proportional gain was too high, resulting massive corrections swinging past the sought after reference value. The corrective action was to reduce K_p

and test iteratively until jitter was fully removed. At a final value of $K_p = 0.25$, the Romi exhibited no noticeable jittering and performed to our subjective satisfaction. Figure 2. shows the data collected for the new settled upon proportional gain.

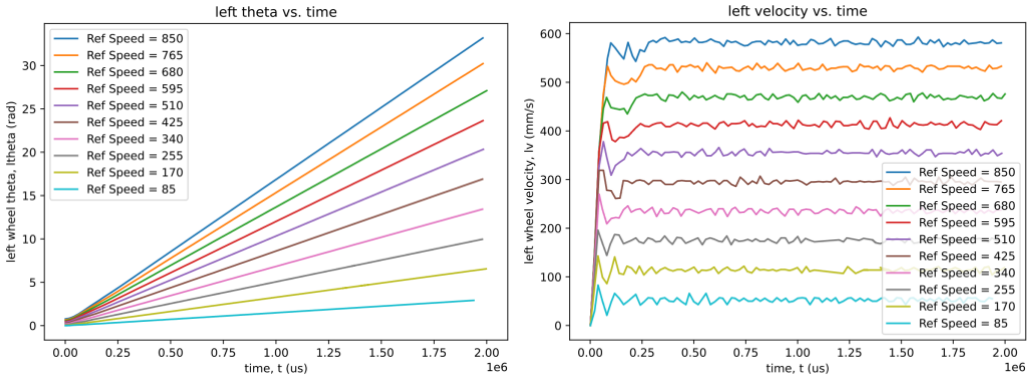

Figure 2. Proportional Controller: $K_p = 0.25$

For the following tests, the proportional gain will remain $K_p = 0.25$ unless otherwise specified.

PI-Controller:

After the proportional controller was calibrated adequately, the integrator was incorporated. Integrator gain was initially (and arbitrarily) set to $K_i = 1$. This resulted in the response recorded in Figure 3

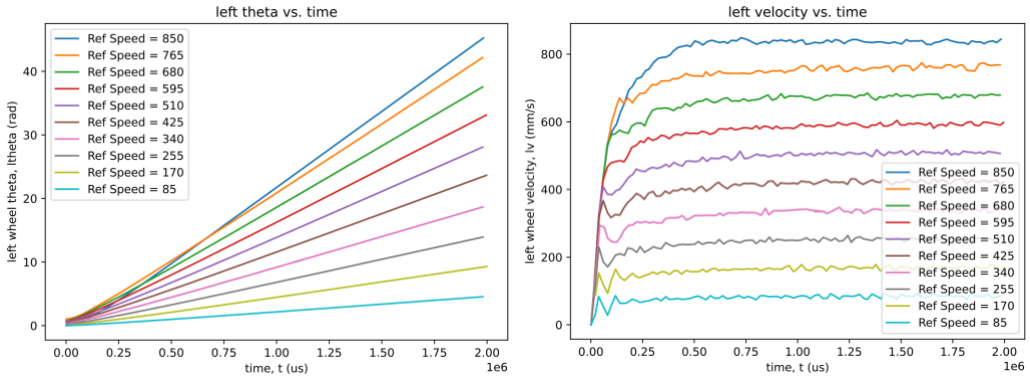

Figure 3. PI Controller: $K_i = 1$

Since no overshoot was noticed in this test, it was determined that $K_i = 1$ was a gain that may have been too small for our use case. Ideally, each wheel reaches the reference speed as fast as it possibly can. Without seeing overshoot from the integrator, it was reasoned that the K_i was too small. The corrective action was to find an upper bound at which overshoot was unacceptable. This unacceptable overshoot was noticed at $K_i = 10$ with the behavior shown in Figure 4.

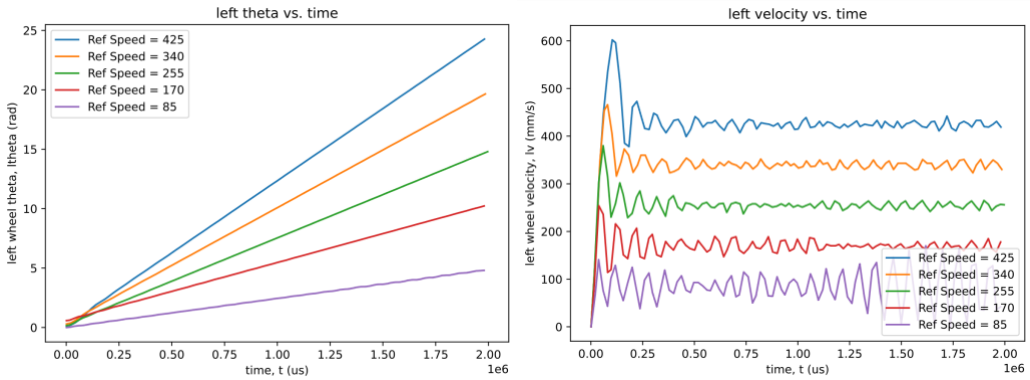

Figure 4. PI Controller: $K_i = 10$

Similar to the proportional controller process, an intermediate K_i value was determined with iterative testing. $K_i = 2.5$ resulted the maximum K_i value obtainable with minimal overshoot. Figure 5 shows the behavior of this system.

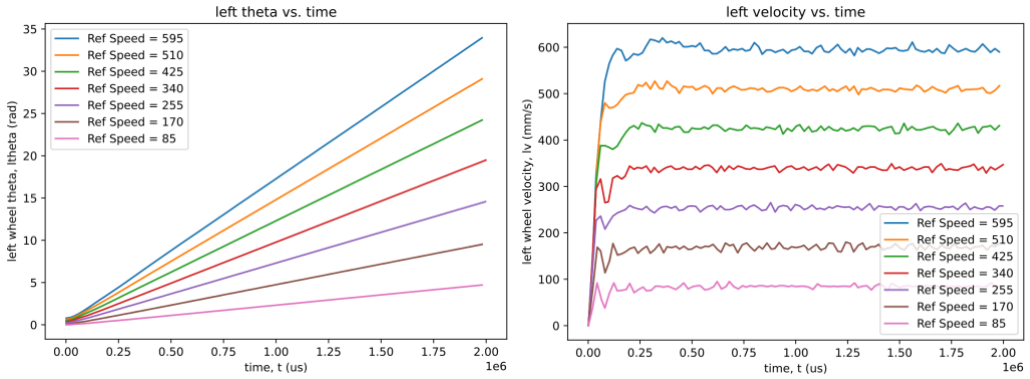

Figure 5. PI Controller: $K_i = 2.5$

For the following tests, the integrator gain will remain $K_i = 2.5$ unless otherwise specified.

PID-Controller:

After the PI controller was calibrated adequately, the derivative portion was attempted. Derivative gain was initially (and arbitrarily) set to $K_d = 1$. This resulted in the response recorded in Figure 6.

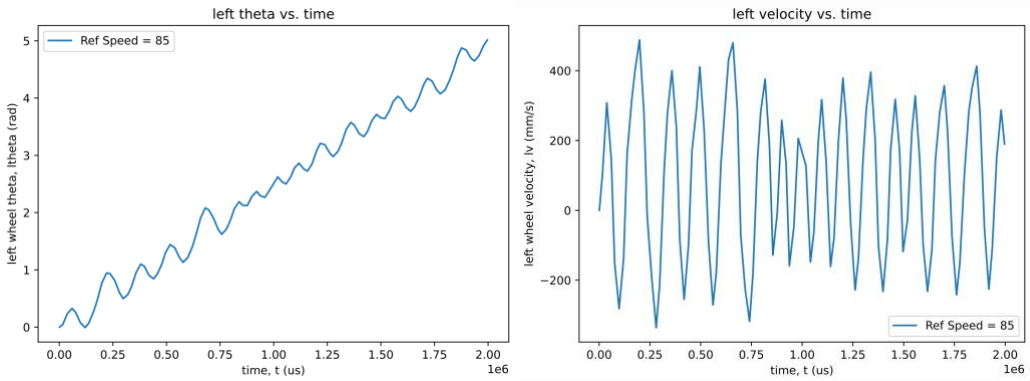

Figure 6. PID Controller: $K_d = 1$

A similar behavior as seen in Figure 1 is seen in Figure 6. It is concluded that the same fundamental reason, the gain value being too large, was the root of the unfavorable behavior. The corrective action was to decrease the derivative gain by orders of magnitude until favorable behavior was exhibited. Figures 7 and 8 show an attempt at this process.

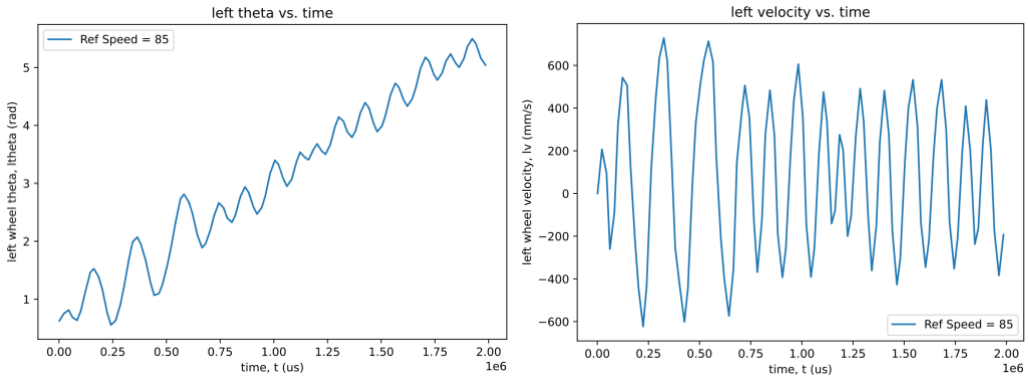

Figure 7. PID Controller: $K_d = 0.01$

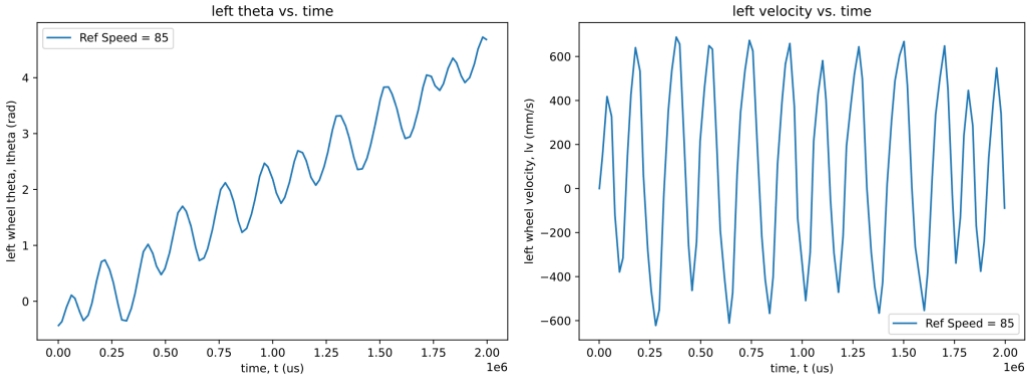

Figure 8. PID Controller: $K_d = 0.001$

Derivative controllers are susceptible to amplifying high frequency noise. Given the fact high frequency noise has been observed in all data collected with Romi, it can be assumed that derivative control would be ill-fit for this application. Noticing the behavior in Figures 6 through 8, this expected, nonideal behavior is observed. For this reason, derivative control has been abandoned and will not be implemented in any configuration of the closed loop control.

PI & K_{ff} Controller:

Following the PID controller failure, the next step was to integrate the feed forward loop into the PI controller. This time around, the feed forward gain was calculated rather than iteratively tested for. The feed forward gain, K_{ff}, was determined by analyzing the steady state data gathered in Lab 0x02. The steady state speeds for each PWM percentage delivered to the motor was plotted and the linear curve fit was determined for each wheel (Figure 9).

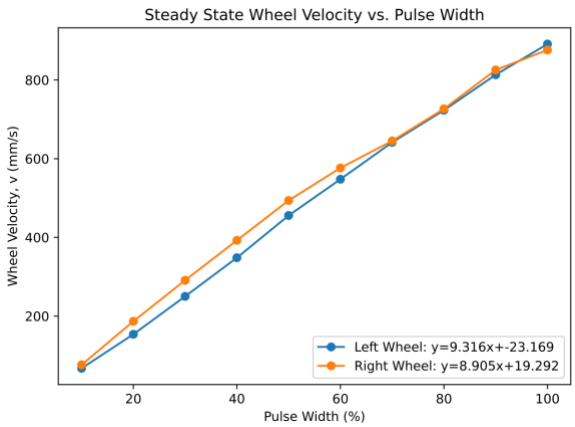

Figure 9. Steady State Wheel Velocities vs. PWM Percentage Given to Motor

The slope of each line in Figure 9 corresponds to the steady state gain from PWM percentage to wheel velocity. That means the inverse of these values provide the steady state gain from the wheel velocity to the PWM percentage, which is K_{ff}. Following this math, the left and right motor's feed forward gains were set to 0.107 and 0.112, respectively. Additionally, the x-intercept of the first order curve-fits in Figure 9 correspond to each motor's minimum required PWM to obtain movement (roughly). This value, which we call PWM_{start}, is calculated for the left and right motors to be 2.49 and -2.17, respectively. This results in feed forward protocol for both motors to follow Equations 1 and 2

$$\text{Left Motor Feed Forward Output} = 0.107 * V_{ref} + 2.49 \quad (1)$$

$$\text{Right Motor Feed Forward Output} = 0.112 * V_{ref} - 2.17 \quad (2)$$

It is noted that the PWM_{start} of the right motor being negative is not unexpected, since the right motor appeared to have a much higher speed then the left motor, however not ideal. The goal is that the disparity between PWM_{start} of each motor will result in more similar behavior between the left and right motors during operation. Figure 10 shows the behavior of Romi with the now added feed forward loop but with the PI controller turned off.

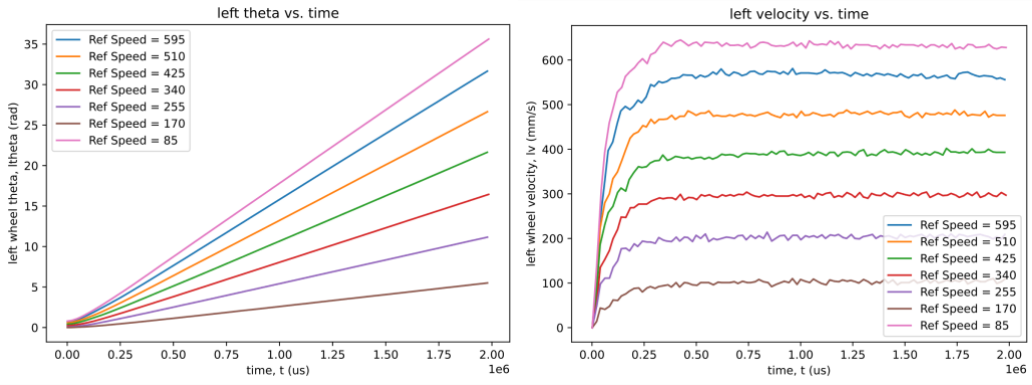

Figure 10. Only Feed Forward: Theoretical K_{ff} and PWM_{start}

The behavior of the Romi with just the feed forward was fairly exceptional. The Romi arced a little bit when it wasn't supposed to, but the arc was acceptable given the fact that PI was turned off. With the

original values of PI derived above, and the feed forward outlined in Equations 1 and 2, the Romi was tested again (Figure 11).

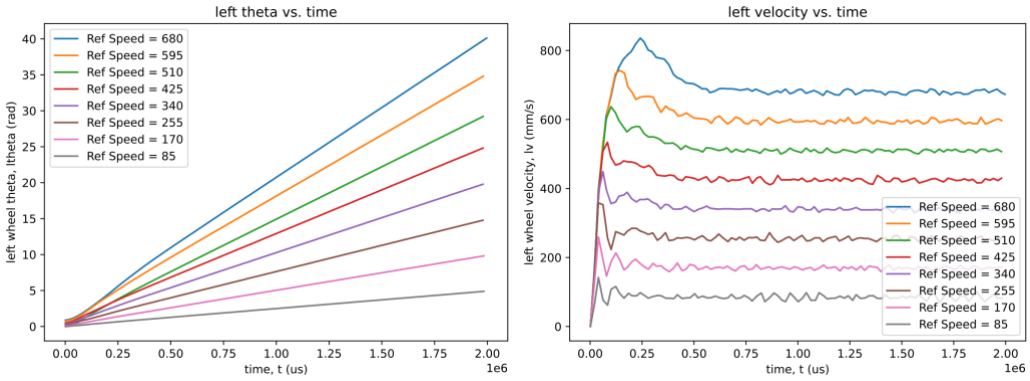

Figure 11. PI & Feed Forward: Previous PI with Theoretical K_{ff} and PWM_{start}

The Romi now exhibits remarkable straightness. However, the overshoot is not optimal. The proportional and integrator gains were not adjusted because the next section will rechange the dynamics of the closed loop control.

PI & K_{ff} & V_{droop} Controller:

The last addition made to the Romi was scaling the effort requested by the motors by the inverse of the battery voltage droop. As the battery dies, the steady state gain of the motors will decrease. This results in the effort given to the motors obtaining smaller velocities than intended. Although this error is corrected by the PI controller, it can be minimized by adjusting the effort request (PWM) by Equation 3

$$PWM_{adjusted} = PWM * \frac{V_{NOM}}{V_{BAT}} \tag{3}$$

where V_{NOM} is the battery voltage at full battery, and V_{BAT} is the battery voltage at the instance the effort is being set. With this scaling implemented, the Romi exhibited the behavior in Figure 12.

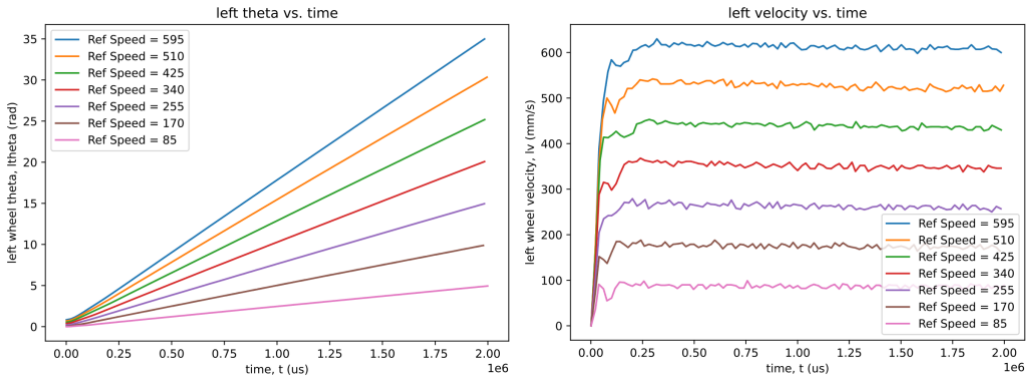

Figure 12. PI & Feed Forward & V_{droop}: All values nominal

After the testing from Figure 12, all gains were varied to some degree in an attempt to remove two undesired behaviors: initial unequal jerks from each wheel on startup causing sudden yawing and non-

straight paths. After over 30 iterations the final gain values are $K_p = 0.1$, $K_i = 1.5$, $K_{ff,left} = 0.1$, $K_{ff,right} = 0.1$, $PWM_{start,left} = 2.49$, and $PWM_{start,right} = -2.17$. The behavior is shown in Figure 13.

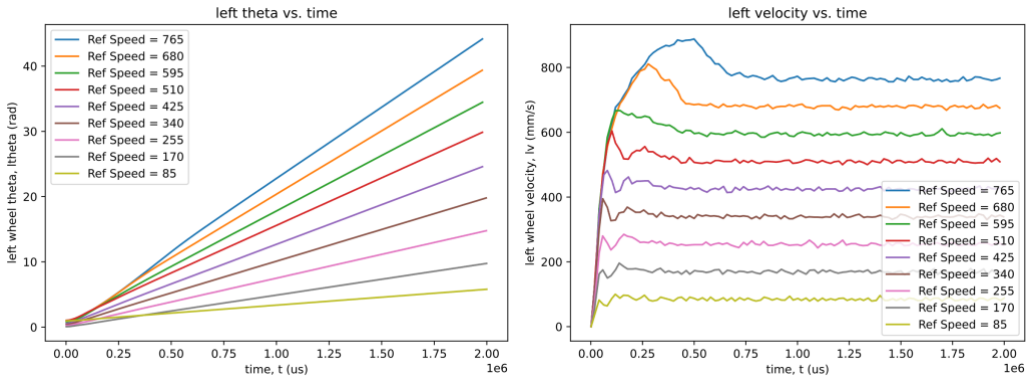

Figure 13. PI & Feed Forward & V_{droop}: $K_p = 0.1$, $K_i = 1.5$, $K_{ff,left} = 0.1$, $K_{ff,right} = 0.1$, $PWM_{start,left} = 2.49$, and $PWM_{start,right} = -2.17$

Despite how visually unappealing Figure 13 may seem, it provided the best behavior we could visually observe. This concludes the testing done.

References:

- Refvem, C. (2025). *Lab 0x03 Closed Loop Control* [Unpublished lab instructions]. Department of Mechanical Engineering, Cal Poly. Retrieved from https://canvas.calpoly.edu/courses/161863/assignments/1368603?module_item_id=4886829