Proceedings, Bicycle and Motorcycle Dynamics 2019
Symposium on the Dynamics and Control of Single Track Vehicles,
9 – 11 September 2019, University of Padova, Italy

Design of an Electric Bicycle Speed Controller

Trevor Z. Metz[#], Jason K. Moore[#]

[#] Mechanical and Aerospace Engineering
University of California Davis
1 Shields Ave, Davis, CA
email: jkm@ucdavis.edu
email: tzmetz@ucdavis.edu

ABSTRACT

Speed control is not a prevalent feature found in electric bicycles. Many electric bicycles implement a pseudo speed controller that does not include feedback based on sensing speed. As with automobiles, speed control can be desirable for driver comfort and safety. Additionally, accurate speed control is also very helpful when validating dynamic models of single-track vehicles, which is our motivation. This paper describes a low cost feedback speed controller for an instrumented electric bicycle. To achieve this, we used grey box system identification to fit a second order linear model of the longitudinal dynamics of the bicycle to a measured step time response. The resulting fitted plant model was used to design a robust PID controller. We implemented the controller with a custom Arduino-based microcontroller. The resulting implementation was able to maintain the interquartile range of measured speeds at steady state within ±0.1m/s of a desired setpoint speed.

Keywords: bicycle, electric, control, speed, mechatronics

1 INTRODUCTION

In prior work, a theoretical model of a bicycle-rider control system was developed based on previous pilot modeling efforts [1]. This model includes a Handling Quality Metric (HQM) that, based on physical parameters and speed of the bicycle, predicts the handling quality of the bicycle. The quantification of bicycle handling based on its geometry can enable designers to create safer bicycles potentially reducing bicycle related accidents. This predictor relies on the Whipple-Carvallo bicycle dynamics model [2] that is linearized about specific operating speeds. It is also well known that the dynamics [2] and handling [1] are very sensitive to speed, especially when the speed is low. This handling prediction model carries the assumption that longitudinal speed does not vary during lateral maneuvers, thus it is helpful if the speed of an actual bicycle can be both accurate and precise (non-varying).

Figure 1 shows attempts at maintaining 2.2 m/s with a factory electric bicycle pseudo speed controller, which we call “throttle lock”, during six short runs from a set of experiments on level ground in which the rider rides the bicycle as slowly as they can over a marked line [3]. At these low speeds, inaccuracy of hitting the 2.2 m/s mark and the variation in speed on reasonably flat ground is too high for model validation purposes. Uneven surfaces, wind, larger lateral maneuvers, and variations in rolling resistance all exacerbate this error further.

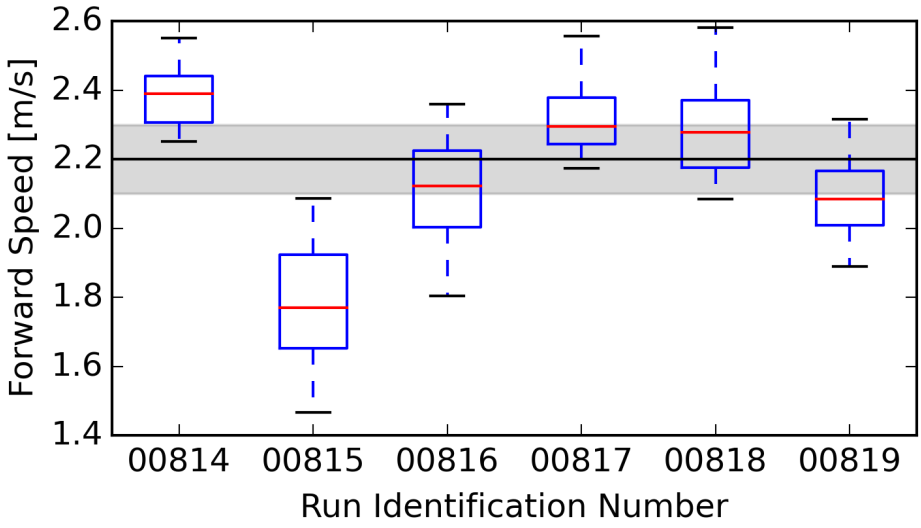

Figure 1. Bicycle speed variation during previous experiments using the throttle lock for maintaining speed [3]. The box plots indicate the variation in speed during 7 seconds of data collection. The horizontal black line indicates the speed at which the rider attempted to maintain using the throttle lock and the shaded gray bar shows a ±0.1 m/s range that we desired the speed variation to fall between. The average root mean square error, RMSE, of each run is 0.21 m/s and the average standard deviation about the mean is 0.12 m/s.

The purpose of this work is to design and implement a speed controller on an electric bicycle such that the mid-spread of the data falls within a ±0.1 m/s range of the desired set point speed. The mid-spread, or interquartile range, contains 50% of the data and is shown clearly with boxplots, as in Figure 1.

2 CONTROLLER DESIGN PROCESS

2.1 Plant Model

The longitudinal dynamics of an electric bicycle on flat ground can be modeled using a point mass acted on by rolling resistance, air drag, and propulsion. This point mass is coupled to a rotating body that represents the inertial effects of the wheels. The driving motor of the wheels can be modeled by a simple DC electric motor with some resistance and inductance. Figure 2 provides a schematic of this model. Brushless DC motors of more complexity typically drive electric bicycles, but this model is sufficiently complex for our purposes.

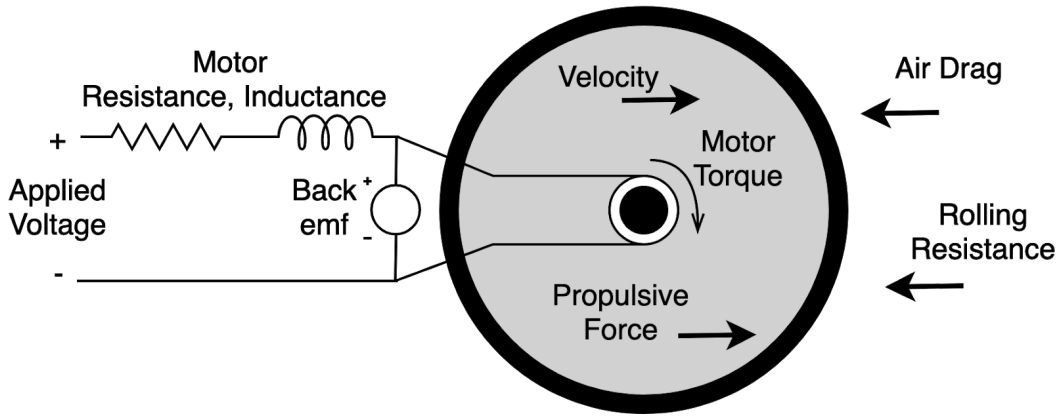

Figure 2 . Free body diagram and electrical schematic of the electric bicycle motor. The longitudinal dynamics are modeled by the acceleration of the coupled point mass and rotational inertia and the forces and torques are generated by the motor by applying a voltage as well as the drag and rolling resistance. A second order model can be derived with voltage as an input and position and velocity as states and outputs.

If linearized about an operating speed, the resulting relationship between the applied voltage and forward speed is a simple second order system with a gain taking the form of the transfer function in Equation (1):

$$\frac{v(s)}{V(s)} = \frac{d}{as^2 + bs + c} \tag{1}$$

where v(s) is speed and V(s) is voltage. The constants a, b, c and d represent the combinations of the linear model parameters defining the bicycle's dynamics. s denotes the frequency domain variable.

We determined the values of the constants a, b, c, and d using system identification. We collected voltage and speed measurements from the actual bicycle with a 61 kg rider for 20 seconds with a step input of 4.27 V (the maximum input voltage). Figure 3 shows the measured speed plotted against time in red.

We computed values for the four parameters in the transfer function by solving a nonlinear least squares curve fit based around a step input simulation providing the fit seen in Figure 3 (blue curve). The resulting numerical values from this fit are shown in Equation (2):

$$\frac{v(s)}{V(s)} = \frac{15.32}{s^2 + 70.81s + 7.34} \tag{2}$$

.

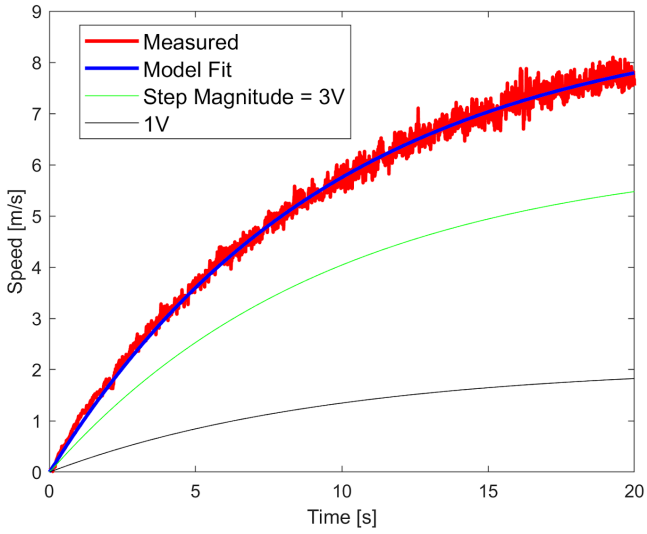

Figure 3. Step response of the fitted model over a measured step response. Also shown are step responses of the fitted model with smaller step inputs to show the open loop behavior.

Additionally, Figure 3 shows the step response of the fitted model for slower speeds to verify that the model behaves reasonably. This provides a realistic linear plant model for our controller design efforts.

2.3 Controller design

A unity-feedback control architecture was formed with the identified plant model and a PID controller. The controller was tuned (K_p = 1.03, K_i = 0.145, K_d = 0.05) using MATLAB's Control System Toolbox to achieve zero steady state error and a transient behavior similar to the one observed in the measured step response shown in Figure 3 above.

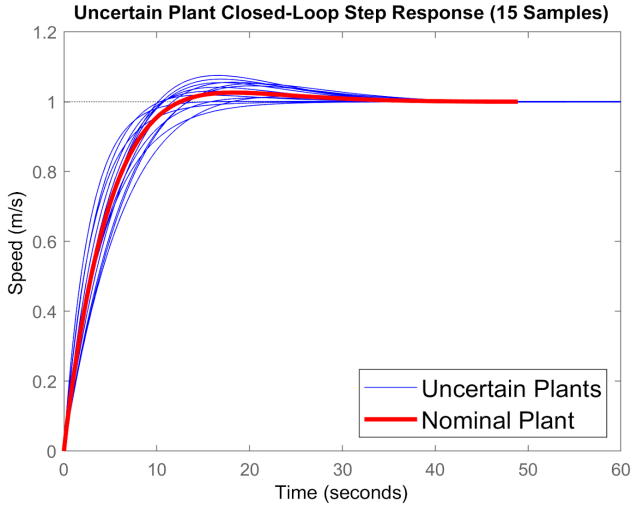

Figure 4.Step responses of both the nominal closed loop system (green) and 15 random samples of the uncertain closed loop system (blue).

To account for uncertainty in the plant model, MATLAB's Robust Control Toolbox was used to simulate the performance of the closed loop system with plant model constants having percentage-based uncertainties about their nominal values. Figure 4, above, shows the nominal closed loop step response with 15 random samples of the uncertain plant model with no issues in stability or performance.

3 CONTROLLER IMPLEMENTATION

The derived PID controller was implemented digitally on an Arduino Nano microcontroller integrated into the powertrain of an instrumented electric bicycle. The electric bicycle (Figure 5) consists of a steel frame Surly 1x1 converted to an electric drive using an Amped Electric Bicycle conversion kit. The kit consists of a brushless direct drive hub motor driven by a motor controller and a 36V Lithium-ion battery. The input to the motor controller is a Hall effect sensor-based thumb throttle.

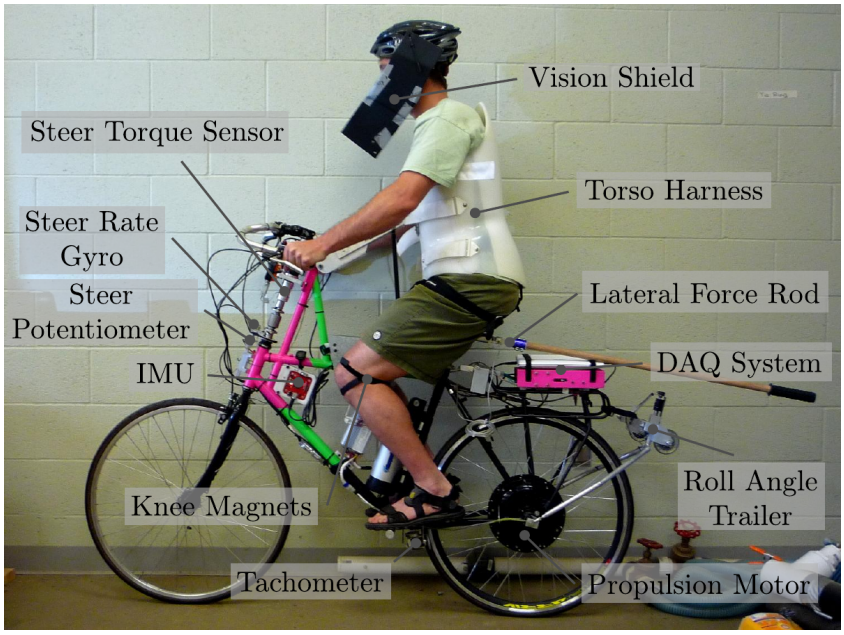

Figure 5. The instrumented electric bicycle. The rear wheel hub contains the electric motor and both the battery and motor controller are mounted mid-frame.

We place an Arduino in line with the voltage signal coming from the Hall effect sensor in the throttle to allow the control system to take over the throttle from the human rider and modulate it to maintain a desired speed. The current speed is measured by a simple generator based speed sensor at the rear wheel and is fed back into the microcontroller. The schematic in Figure 6 shows a detailed view of how the microcontroller is integrated into the existing electronics platform.

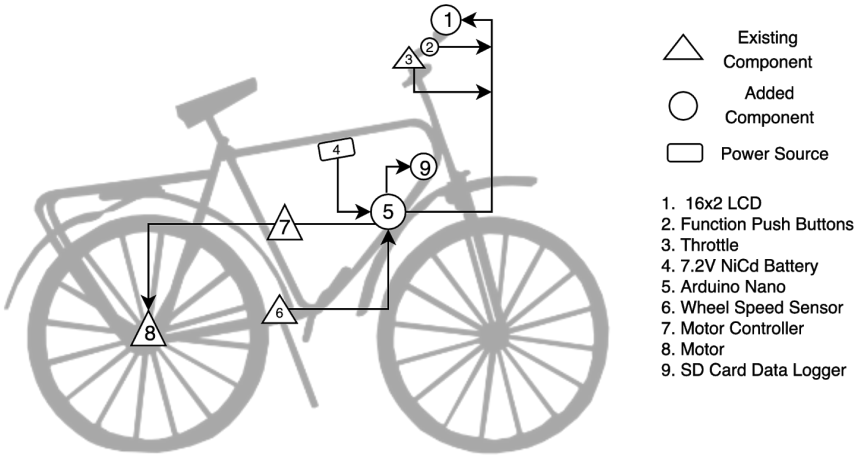

Figure 6. Layout of the control system components showing location, information flow, and type of each component. Components called out with a triangle are existing components on the bicycle. Components called out with a circle are the new additional components to support the speed control system.

The software on the Arduino implements the control architecture shown in Figure 7 and operates the LCD display and push button array that allows the user to both operate and monitor the status of the speed control. To engage the cruise control, the user simply presses both push buttons at the same time. The Arduino then takes the current speed and sets that at the desired speed which the user can adjust up or down by pressing the pushbuttons. When the cruise control is engaged, the LCD displays for the user both the current and desired speed. To disengage the cruise control, the user simply needs to press the throttle. The software is open source and available at https://github.com/mechmotum/eBikeSpdController.

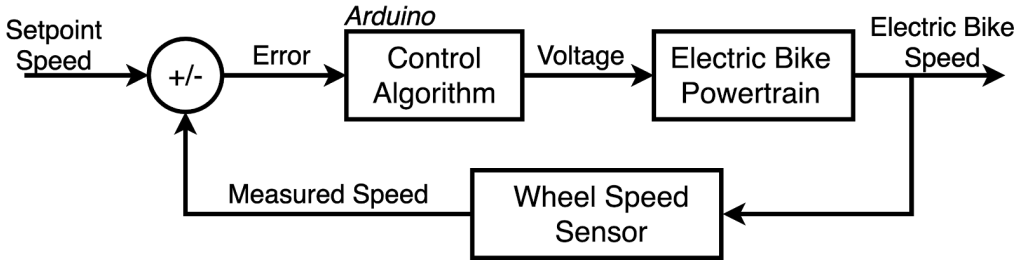

Figure 7. Control architecture implemented in the microcontroller..

4 TESTING AND RESULTS

To evaluate the performance of the control system, we tested the speed controller's performance in straight-line flat ground riding as well as its ability to maintain speed through a slalom course, a typical testing maneuver for single track vehicles.

4.1 Straight-Line Test

This test involves riding the bicycle in a straight-line out and back along a reasonably flat bicycle path on the UC Davis campus. The total distance of the path is approximately 800m (0.5

miles). For each trial, the test rider began by ramping up to the desired speed. At this speed, the cruise control is engaged and held at the same set point speed for the duration of the course. Throughout the entire run, the speed of the bicycle is measured. Figure 8 shows an example set of collected speed data.

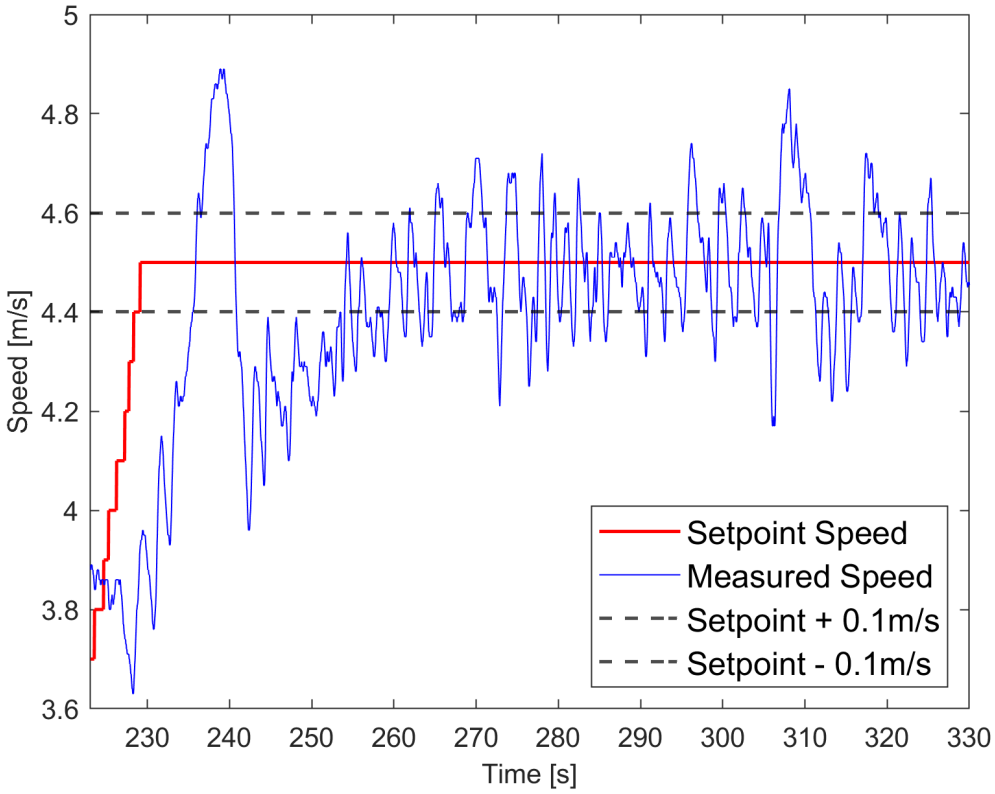

Figure 8. Speed data collected during a run where the set point speed was ramped to 4.5m/s.

Figure 8 shows a typical response observed during our testing. As the setpoint speed is ramped up, the measured speed follows with some delay and overshoot until it settles into a steady state where it oscillates about the setpoint speed. The jaggedness observed in the measured speed oscillation was also observed qualitatively by the test rider as the bicycle's accelerations felt jolty at times.

The performance of the controller is evaluated through its accuracy error and precision error. The accuracy of the controller is measured by the mean error between actual and desired speed in steady state. Precision error is measured using the standard deviation of these errors. Figure 9 shows a boxplot of the speed distributions for three trials of the straight-line test. Table 1 summarizes the controller's performance during these trials.

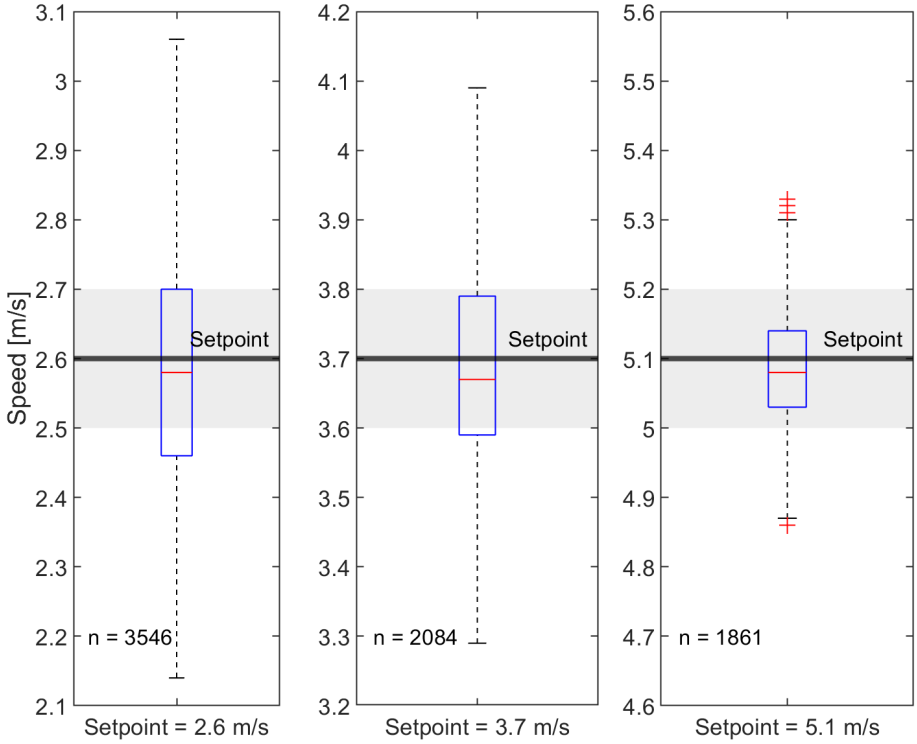

Figure 9. Boxplots showing the distribution of speed measurements while the controller is active and settled during the straight-line trials as compared to the rider's selected set point speed. The horizontal gray bars indicate the desired maxima and minima of the mid-spread speed (blue box in the box plots). n refers to the number of data points collected during each run.

Figure 9 shows how well the measured speed distribution fits within our desired speed bounds. The midspread of each trial roughly lies within our desired bounds. The final trial, at a setpoint speed of 5.1 m/s, has the greatest spread of data lying within the desired bounds. The median of each trial lies within the desired bounds and near the setpoint speed.

Table 1. Accuracy (mean error) and precision errors for the three straight-line trials.

Trial number	Set point speed [m/s]	Mean error between set point speed and actual speed [m/s]	Standard deviation about the mean speed [m/s]
1	2.6	-0.0111	0.1658
2	3.7	-0.0041	0.1506
3	5.1	-0.0041	0.1057

Table 1 shows that the accuracy and precision error of the last trial is better (closer to zero) than the previous trials. Overall, Table 1 shows that the mean errors of each trial are near zero meaning that, on average, there is close to no error between the measured and desired speed for each trial. However, each trial contains a large precision error meaning the measured-desired speed errors are spread widely about the mean.

4.2 Slalom Test

This test involves riding the bicycle through 0.76m wide gates spaced 7.3m apart marked on the ground with tape in a slalom maneuver. The test was performed on a reasonably smooth level bicycle path on the UC Davis campus.

The test rider began the test by ramping up to the desired speed and engaging the cruise control at that speed before entering the first gate. Speed data was collected in the same way as the prior described test. Figure 10 shows a boxplot of the speed distributions for three trials of the slalom test. Table 2 summarizes the controller's performance during these trials.

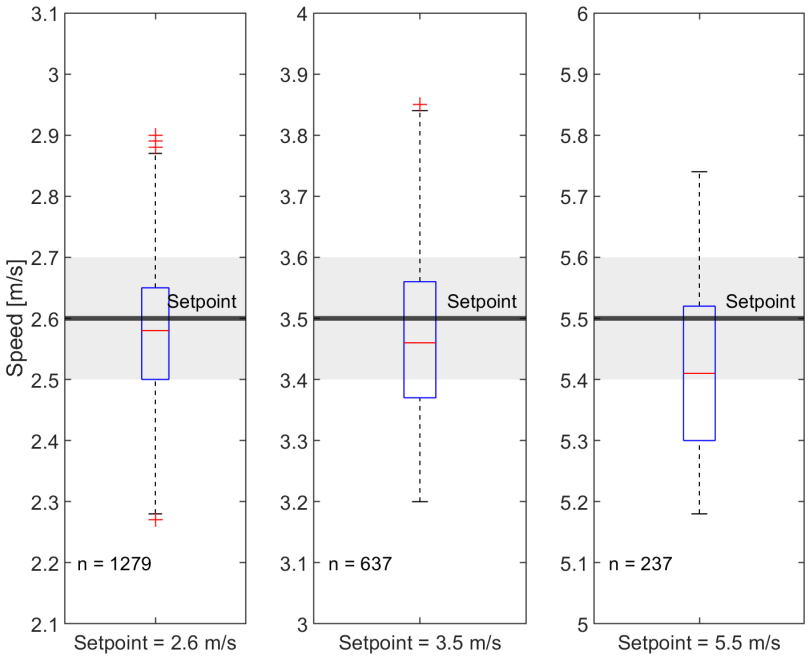

Figure 10. Boxplots showing the distribution of speed measurements while the controller is active and settled during the slalom trials as compared to the rider's selected set point speed. The horizontal gray bars indicate the desired maxima and minima of the mid-spread speed (blue box in the box plots). n refers to the number of data points collected during each run.

Similar to the straight-line test, Figure 10 shows the midspread and median of the measured data roughly falling within our desired bounds. The exception occurs with the third trial in which the median lies towards the edge of the desired bounds.

Table 2. Accuracy (mean error) and precision errors for the three slalom trials.

Trial number	Set point speed [m/s]	Mean error between the set point and actual speed [m/s]	Precision error [m/s]
1	2.6	-0.0113	0.1473
2	3.5	-0.0086	0.1532
3	5.5	-0.0738	0.1456

The results of Table 2 are similar to those of the straight-line test in that the accuracy errors are close to zero (with the exception of trial three) and the precision errors show a large spread about the mean.

5 DISCUSSION OF RESULTS

Examining the percentage of data points that lie within our desired speed bounds helps to determine how often the speed controller is maintaining the speed of the electric bicycle within our acceptable bounds. Table 3 shows these values for each trial of our two tests.

Table 3. Percentage of data points that fall within +/-0.1m/s of the setpoint speed for each trial in both types of tests.

Trial number (Straight-line test)	Percentage of Data Points in bounds [%]	Trial number (Slalom test)	Percentage of Data Points in Bounds [%]
1	37.6	1	57.2
2	47.1	2	45.8
3	70.4	3	37.1
Mean	51.7	Mean	46.7

Table 3 shows that roughly 50% of the measured speeds for each test type fall within our desired bounds. This could be due to the amplitude of steady state oscillations about the setpoint speed being too large leading to more data points falling out of bounds (see example in Figure 8). Modifying the derivative gain in the PID controller may mitigate this effect.

Table 1 and Figure 9 suggest that, for the straight-line test, increasing set point speeds may lead to greater controller accuracy and precision. However, the slalom test results do not support this, as increasing set point speed does not appear to improve controller accuracy or precision. More testing is needed to further investigate this possible trend.

Furthermore, Tables 1 and 2 and Figures 9 and 10 show all non-zero mean errors having a value of less than zero suggesting that the controller is causing the speed of the electric bicycle to undershoot more often than overshoot the set point. The test rider also felt this phenomenon qualitatively during testing as the electric bicycle felt underpowered at times.

Aside from the last trial in the slalom test, the accuracy and precision of the controller are similar across both types of tests. This suggests that the controller is equally adept at controlling speed during straight-line riding as it is during slalom maneuvers.

The boxplots of Figures 9 and 10 show that the midspread of speeds for a majority of trials of each test type lie within our desired speed band of ±0.1 m/s about the set point. Table 3 supports this by showing that roughly 50% of data points in trials of each test lie within our desired bounds. This, along with the fact that the accuracy error of a majority of trials for each test type are below our target of 0.1 m/s implies that the PID controller implementation has mostly met our goals.

6 CONCLUSION

This work presented the design, implementation and testing of a PID based cruise control for an electric bicycle. A PID controller was tuned using a linearized model of an electric bicycle identified from measured data. This PID controller was implemented digitally on a consumer

microcontroller integrated into the powertrain of an electric bicycle. Initial testing of this implementation showed the speed controller performed in accordance with our goals for the project. Further testing is required to tune and improve the performance of the controller. Improvements to the cruise control system should include simpler integration into the existing electric bicycle platform, smoother speed modulation and improvement over the precision error in the existing controller.

7 ACKNOWLEDGEMENTS

We would like to acknowledge the help of Scott Kresie in the design of the slalom course. We would also like to acknowledge the help of Sejin Han with testing the speed controller.

REFERENCES

[1] R. Hess, J. K. Moore, and M. Hubbard, “Modeling the Manually Controlled Bicycle,” *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 42, no. 3, pp. 545–557, May 2012.

[2] J. P. Meijaard, J. M. Papadopoulos, A. Ruina, and A. L. Schwab, “Linearized dynamics equations for the balance and steer of a bicycle: A benchmark and review,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2084, pp. 1955–1982, Aug. 2007.

[3] S. W. Kresie, J. K. Moore, M. Hubbard, and R. A. Hess, “Experimental Validation of Bicycle Handling Prediction,” in *Proceedings of the 6th Annual International Cycling Safety Conference*, Davis, CA, USA, 2017.