Self Balancing Vehicle

Alec Hasegawa, Mechanical Engineering
Mentor: Dr. Nolan Tsuchiya
Kellogg Honors College Capstone Project

Abstract

The emergence of self-balancing, single rider vehicles such as the Segwey, Hoverboard, or One Wheel, have introduced a clear example of control systems into everyday life. The goal of this project was to design and construct a self-balancing vehicle using commercial off the shelf materials and an Arduino UNO microcontroller to display the use of proportional-integral-derivative (PID) control. This project will show the use of control loops to stabilize a single degree of freedom, unstable plant. The design of the self-balancing vehicle was similar to that of the commercially sold Hoverboards or Segweys that allowed the users to move horizontally depending on the angle of their tilt. The design will use a MPU-6050 inertial measurement unit to measure the angle of tilt from horizontal, as well as brushless DC (BLDC) hub motors. Electric bicycle speed controllers were used to drive the BLDC motors, as they were the only commercially sold hardware that could handle the motors large power draw.

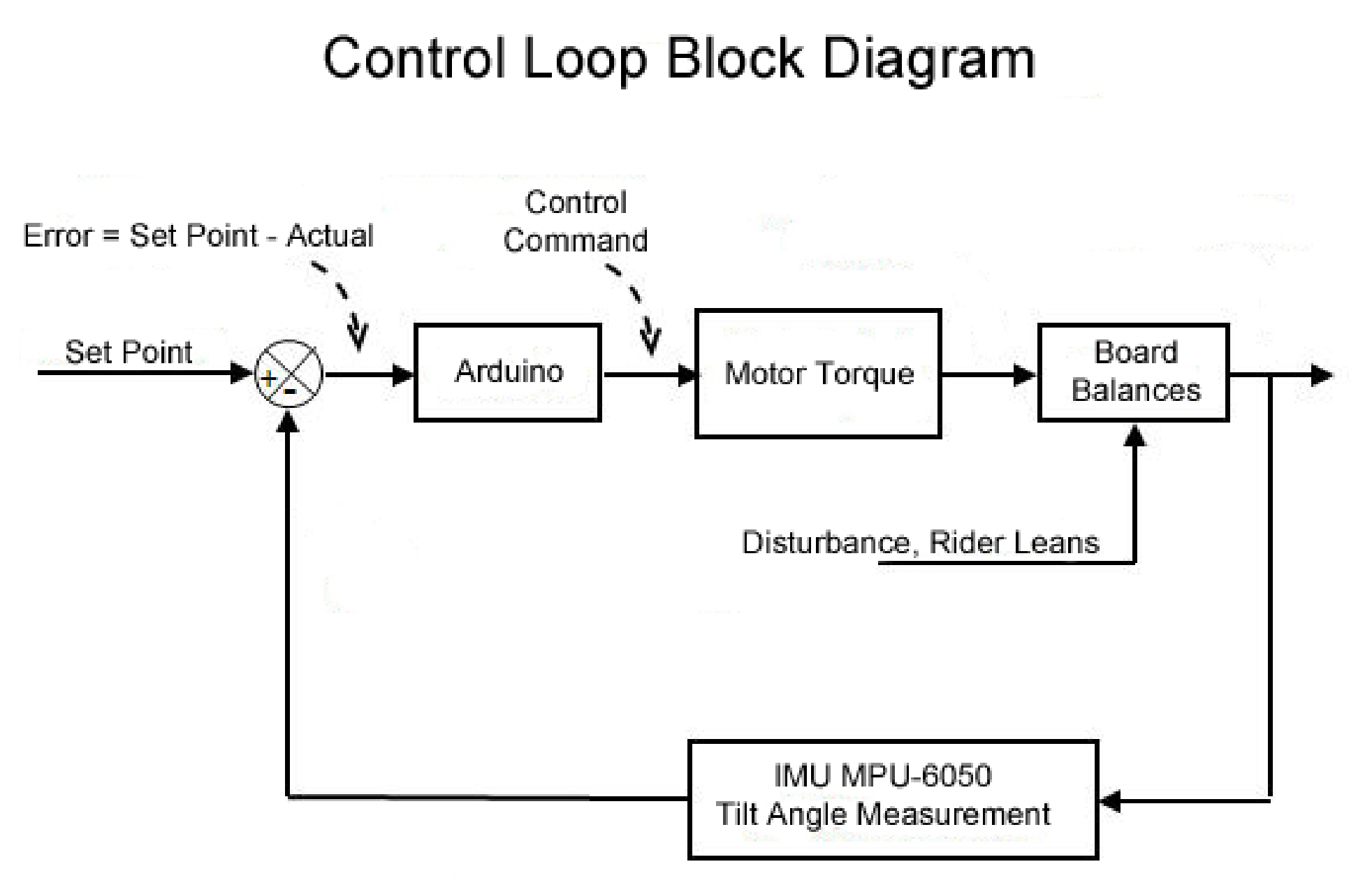
Control Loop Block Diagram
Error = Set Point - Actual
Control Command
Set Point
+
-
Arduino
Motor Torque
Board Balances
Disturbance, Rider Leans
IMU MPU-6050 Tilt Angle Measurement

Figure 1: “Block Diagram of Self-Balancing Vehicle Control Loop

System Requirements

- Reach a speed of at least 5 mph or 8 km/h on smooth, level ground
- Have an acceleration of 2 ft/s^2
- Support a maximum user weight of 250 lbs
- Vehicle must be able to travel in both directions
- Turn in both directions when user leans that direction
- Function for at least 1 hour on a full battery charge
- Charge battery pack using a typical 120V wall outlet
- Withstand light impacts and gravel abrasions

Arduino Code

The Arduino code was completed using the open source Arduino IDE software. The purpose of the code was to facilitate the IMU measurement of the tilt angle, ensure that the measurement was accurate, and then calculate a corresponding motor torque signal to send to the brushless DC hub motors. This was completed using the MPU-6050 library created by Jeff Rowberg and shared through Github. Included in this library is a script for calibrating the MPU-6050. The offsets received from this calibration are shown below in table 1.

Table 1: “Inertial Measurement Unit MPU-6050 Offsets”

	Gyroscope			Accelerometer		
	X-Axis	Y-Axis	Z-Axis	X-Axis	Y-Axis	Z-Axis
Offsets	-58	-138	-35	-3624	-990	1765

With these offsets the MPU-6050 can start measuring the angle of tilt using yaw pitch and roll angles. The further the roll angle is away from horizontal the higher the motor torque. The Arduino sends a pulse width modulated (PWM) signal using the analogwrite function in the Arduino IDE. This converts a digital signal into an analog signal that the throttle control wires can read. This loop is repeated at a baud of 115200.

Project Platform

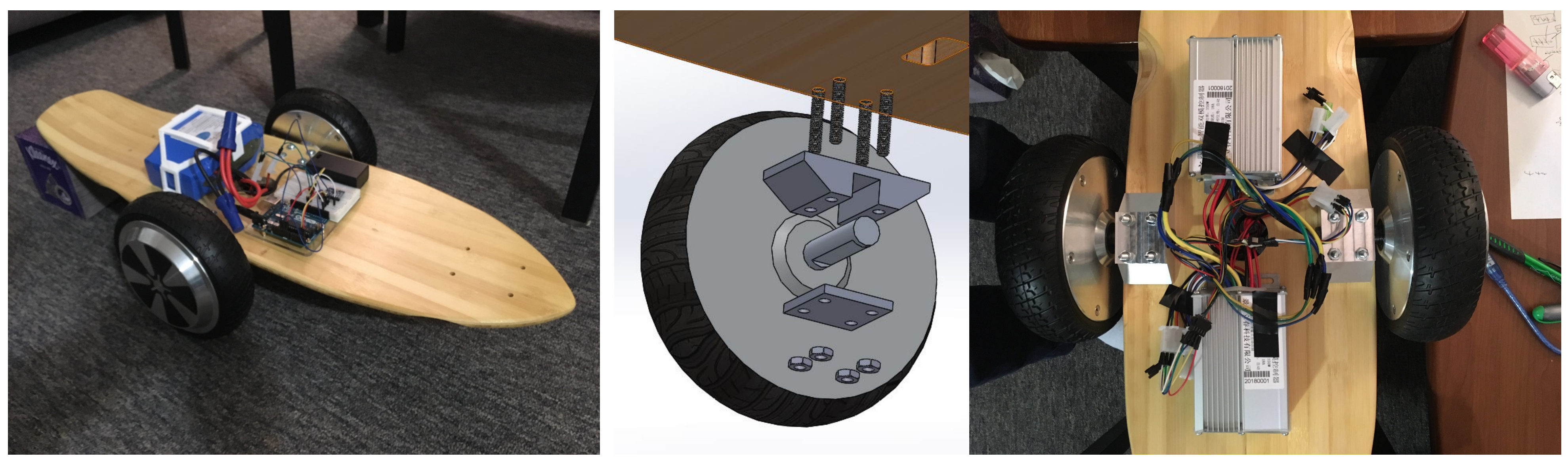

Figure 2: “(a) Completed Project Platform. (b) Motor Mounting. (c) ESC Mounting and Wiring

The project platform is shown above in figure 2. The motor mounts were machined out of aluminum 6061 and mounted to the board using 2 inch long bolts. The skateboard deck is made out of bamboo for the lightweight, yet durable material design. A 36V, 4400mAh lithium ion battery was used, as this battery was relatively inexpensive and met the voltage and capacity requirements. 36V brushless DC hub motors were used due to the price, the high torque at low speeds, and the convenience of having a wheel and motor all-in-one. 36V, 350W electronic speed controllers normally used for electronic bicycles were used to drive the three phase brushless DC motor. These were mounted to the bottom of the skateboard deck.

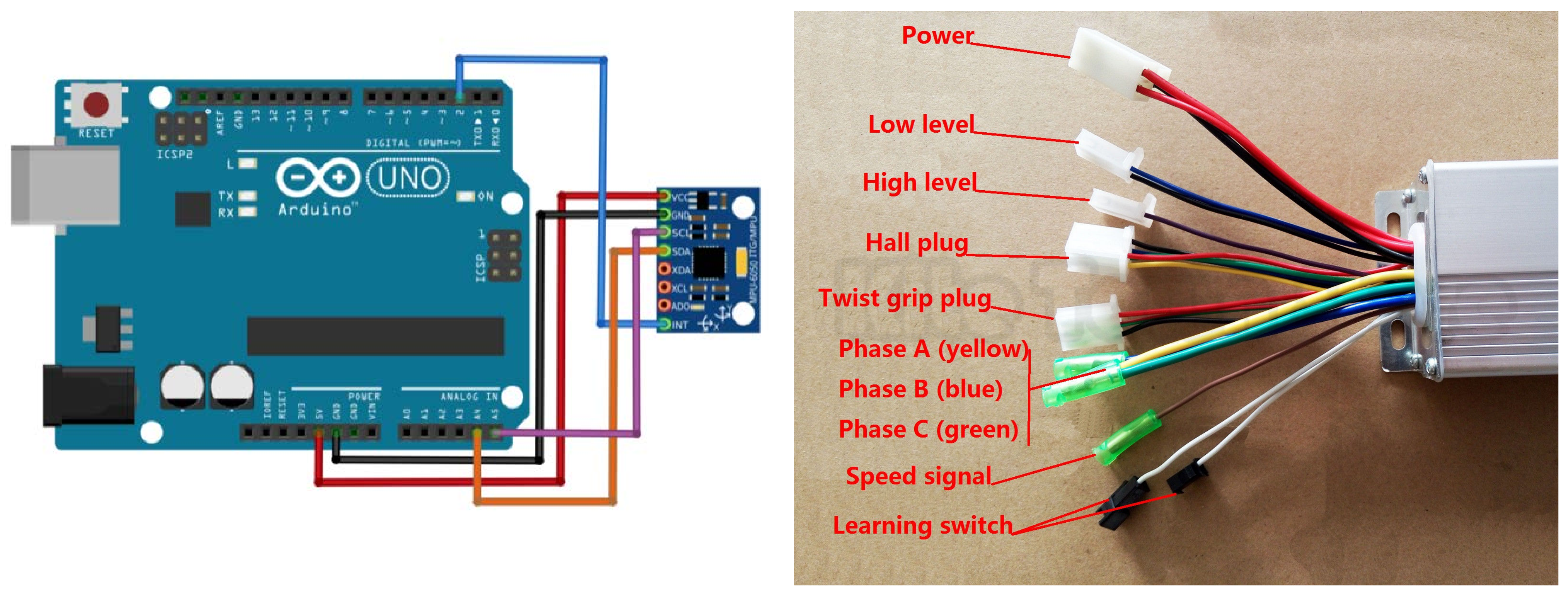

Figure 3: “(a) MPU-6050 Wire Connections. (b) E-bike Electronic Speed Control Wire Connections

The hardware was sized and chosen after the motor was chosen. The electronic speed control (ESC) had to be rated for the same power output and voltage as the motor. Thus, a 36V, 350W e-bike speed controller was chosen because it was the only commercially available product that could handle large power output. For the inertial measurement unit, the MPU-6050 was chosen simply due to the vast amount of resources available online in pairing the IMU to the Arduino. The MPU-6050 measured the angle of tilt relative to horizontal using an accelerometer and gyroscope. This measurement is then sent to the Arduino, which uses PID control algorithms to calculate a corresponding signal to send to the motor.

Conclusion

There are many ways this project could be improved, and also many ways this project was successful. This project was successful in using an IMU to accurately measure the angle of tilt of the skateboard deck, and sending a corresponding motor torque signal. However, a commercial off the shelf ESC could not be that could drive the motors at low speeds very well. The ESC also did not allow the motors to reverse rotation direction very quickly, which is a necessity for a self-balancing vehicle; thus, the motors could only spin the forward direction. Since the motors could not be driven at low speeds, the motors would lurch, quickly accelerating faster than the rider could handle on the board. This made the product almost impossible to ride. Future improvements would be to recruit an electrical engineer to design a custom ESC and power supply circuit in order to properly drive the motors.