Self-Balancing Bike with Reaction Wheel

Author Names: Sam Thomason, Shah Rifat Alam Bhuiyan

***Abstract*—Self-balancing robots are classic examples of underactuated dynamic systems and are widely used to study control theory in robotics. This project aims to build and control a two-wheeled self-balancing robot using a proportional-integral-derivative (PID) controller running on a microcontroller. The goal is to maintain upright stability by adjusting reaction wheel motor outputs in response to pitch angle measurements from an inertial measurement unit (IMU). The approach was to build a modular prototype to initialize and calibrate the sensors and actuator individually before integrating the full program, including PID tuning. An empirical approach was taken for tuning where the team tuned a P-only controller first to get the bike to almost balance, with small constant oscillations, and then added the integral and derivative controls to smoothen the response. The team was then successful in having the bike balance itself without any interference as well as with added external forces. Although the relatively simple PID-control was sufficient, it is not perfect and cannot recover from large lean angles. More advanced control algorithms such as linear quadratic regulator (LQR), model predictive control (MPC) or non-linear outputs can make the balancing more robust as well as better IMU filtering such as Kalman filtering can reduce long term tilt bias.**

I. Introduction

This project paper presents the design, construction, and testing of a miniature self-balancing bicycle that uses a reaction wheel controlled by a PID (Proportional–Integral–Derivative) controller. The project explores whether such a setup can effectively stabilize an otherwise unstable system, focusing on feedback control and mechatronic design principles.

A reaction wheel is a rotating disk with most of its mass concentrated at the rim. The principle relies on the law of conservation of angular momentum — when the wheel accelerates in one direction, it exerts an equal and opposite torque to the bike. The reaction wheel is mounted horizontally across the bike's frame. When the bike begins to lean to one side, the controller spins the reaction wheel in the same direction which creates a counteracting torque to balance the bike by rotating it back toward its upright position.

The team modeled the bike as an inverted pendulum system, governed by equations describing angular acceleration, torque, and moment of inertia. The key control principle is feedback stabilization, where sensor data from an IMU (accelerometer + gyroscope) provides the current tilt angle, and the PID controller adjusts the motor driving the reaction wheel to restore balance. The program also includes signal filtering using a complementary filter for sensor fusion, combining accelerometer and gyroscope data.

The team tested the PID controller on a small 3D-printed bike equipped with an Arduino UNO with a motor shield, brushless DC motor, rotary encoder and a 9-DOF IMU. PID tuning was performed via an empirical/heuristic method, with iterative fine-tuning for stability.

Objectives:

- Design, assemble, and program a two-wheeled robotic platform capable of standing upright on its own with assistance from the reaction wheel.
- Implement and tune a PID controller to maintain balance using real-time IMU data.
- Demonstrate successful balancing of the robot for 30 seconds under small disturbances.

II. Related work

Prior work on self-balancing two-wheeled systems largely treats the bicycle-at-rest problem as an inverted pendulum. The work typically stabilizes it through closed-loop feedback using an IMU for tilt sensing and an actuator that can generate corrective roll torque. In the specific niche of reaction-wheel stabilization, Demir and Tewolde Berhane built and evaluated a miniature stationary bike stabilized by a PID controller driven by IMU tilt estimates and show that adding helper logic such as dynamic reference/target shifting can substantially improve robustness compared to PID alone [1]. Their report also highlights practical failure modes such as sensor noise, gyro drift, and actuator saturation, motivating the use of sensor fusion, filtering and windup management. Closely related work by Soryani investigates a reaction-wheel self-balancing bike prototype in contrast to more common bicycle stabilization approaches that manipulate steering and use state-space modeling [2]. He also uses a recursive estimator (Kalman filtering) that tries to get the optimal value of a certain noisy data point, given that it has seen the previous data point and its associated noise.

A second thread of related work focuses on reaction-wheel inverted pendulums as a canonical benchmark. Gräsberg and Lavebratt study a "reaction wheel stabilized stick" and emphasize state-space modeling/control-theory framing (commonly LQR/state feedback in this class of systems) as a systematic alternative to purely heuristic PID tuning, which is what we did [3]. On the estimation side, complementary filtering is widely used in practice because it is simple and effective at combining accelerometer and gyroscope signals; Higgins' classic comparison discusses complementary filtering in relation to Wiener/Kalman filtering, helping motivate why more model-based estimators can outperform ad hoc filtering when drift/noise characteristics matter [4]. In the team's project, this directly supports the design choice of fusing gyro/accelerometer with complementary and low pass filtering to produce a stable lean estimate for feedback, while leaving open a path toward EKF/Kalman methods if better bias/drift handling is needed. Beyond academic theses, open-source and

hobbyist work has strongly influenced practical implementations. The ReM-RC reaction-wheel bike ecosystem provides comprehensive examples for reaction-wheel balancing on 3D-printed platforms [5]. This lineage motivates the team's emphasis on live tuning via serial parsing and rapid iteration. Where the team's approach contrasts is in the actuator/control pipeline: the BLDC gimbal motor is driven using voltage control rather than treating the motor plus driver as a coarse PWM "speed command", the tuning is explicitly structured around filtered lean estimates and on-the-fly gain updates, aiming for smoother torque authority and easier experimental characterization. More recent robotics research demonstrates how model-based optimal control and learning-based tuning can improve performance under constraints (i.e., torque/current limits, wheel-speed saturation) [6]. These works underscore a key limitation observed across many PID-centric builds: continuous disturbances can demand sustained torque, causing wheel speed to ramp until saturation, at which point balance is lost. Our project addresses part of this gap pragmatically through filtering, careful loop structure, live tuning, motor voltage control, and anti-windup controls.

III. METHOD AND APPROACH

A. Control Objective

The goal of the controller is to maintain the lean angle of the robot close to upright position. The input to the system is the measured lean angle, compiled from the gyroscope and accelerometer outputs from the IMU. The output is the torque generated by the motor through the reaction wheel.

Below is a flow diagram showing the control logic for the self-balancing bike.

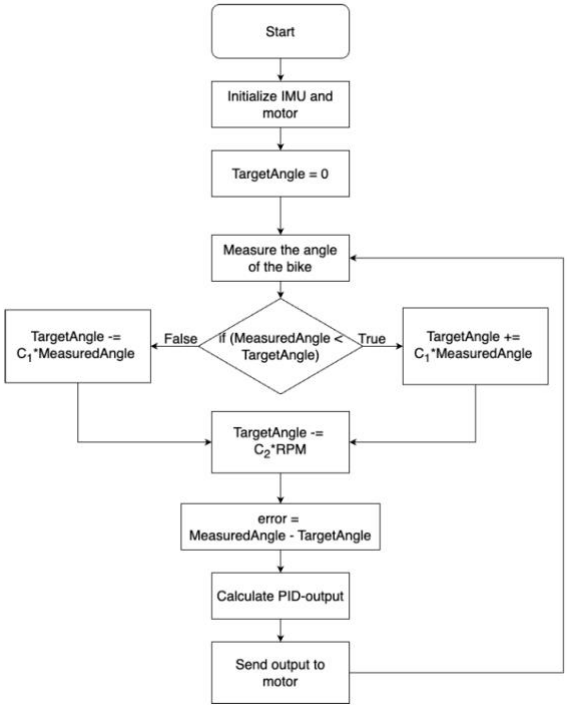

Figure 1: Flow diagram of control logic

B. PID Control Law

To maintain stability, the system uses a PID feedback loop that calculates a control signal u(t):

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau)\,d\tau + K_D \frac{de(t)}{dt} \quad (1)$$

Where:

- $e(t) = \theta_{ref} - \theta(t)$: the **error** between the desired and measured lean angle.
  - θ_{ref} is the target angle (usually 0°, or a dynamically adjusted value).
- K_P: proportional gain — strength of immediate correction.
- K_I: integral gain — removes long-term bias.
- K_D: derivative gain — anticipates motion and adds damping.

This control signal u(t) is converted to a voltage signal that commands the motor driver to spin the reaction wheel with the needed magnitude and direction.

C. Hardware

The team used the following hardware for the prototype implementation:

- Microcontroller: Arduino Uno
- IMU: LSM9DS1 9-DOF
- Motor: 90 kV BLDC gimbal Motor
- Battery: LIPO 3 Cell 850 mAh
- Motor Controller: FOCShield for Arduino Uno
- Rotary Encoder : AS5600

Below is a flow diagram showing the control logic for the self-balancing bike.

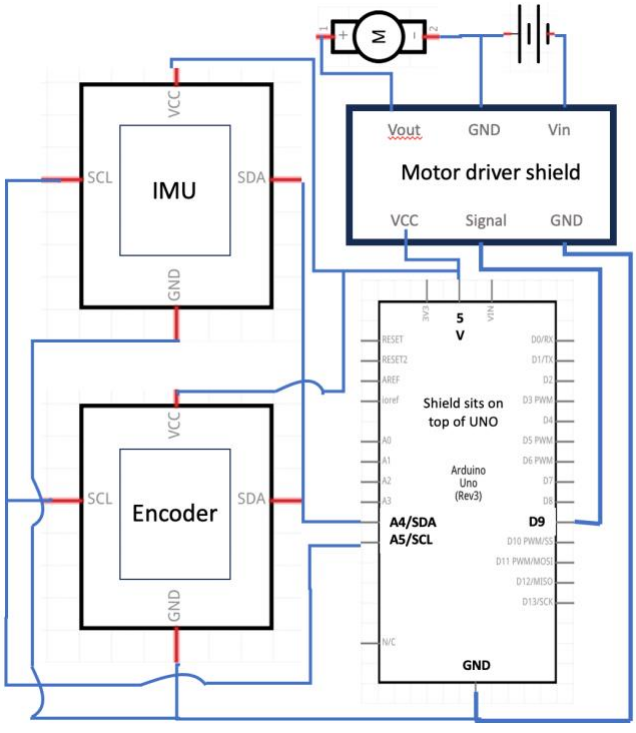

Figure 2: Circuit diagram

The team was able to collect raw data from the encoder and use code to calculate the wheel angular velocity. For the IMU, the team first had to find the correct orientation of the sensor to find the axis along which the bike would lean. Using trial and error to switch axes in the code, the y-axis of the IMU was determined to be the lean axis. The raw data from the accelerometer is used to calculate the lean angle from gravity and the gyroscope data tracks fast angle changes. The accelerometer data is noisy, and the gyroscope angle tends to drift, so a complimentary filter, with a high gyro coefficient was used to fuse both results to provide a more stable reading of the lean angle. The team struggled initially to control the motor using the ESC, especially trying to make it go in either direction. A configurator was used to flash the ESC for bidirectional mode, but the motor was spinning too fast. An open-source firmware was used to reflash the drive to achieve reasonable speeds. It was later determined that a high-speed RC plane motor was not suitable for this application, and the team switched to a BLDC gimbal motor with an Arduino Uno motor shield driver from SimpleFOC.

The mechanical structure consists of three main subsystems: the bike assembly, the reaction wheel, and the control electronics. The team successfully 3D printed all the CAD files and completed the physical assembly. The design was made modular to enable easy removal of reaction wheels, actuators, and sensors for testing and calibration.

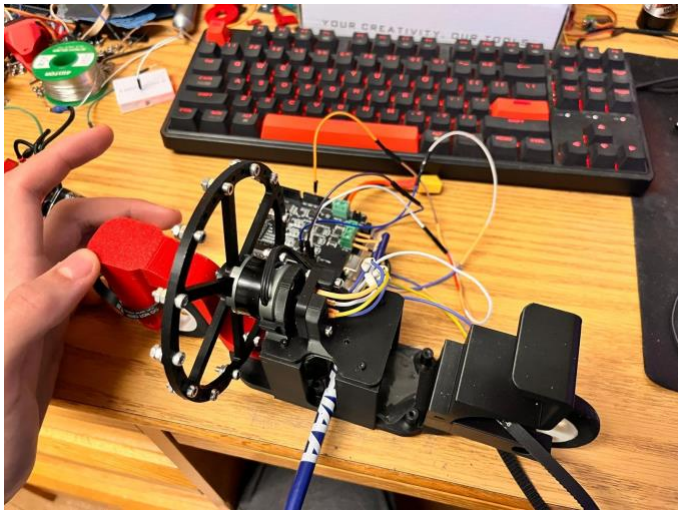

Figure 3: Top view of fully assembled prototype

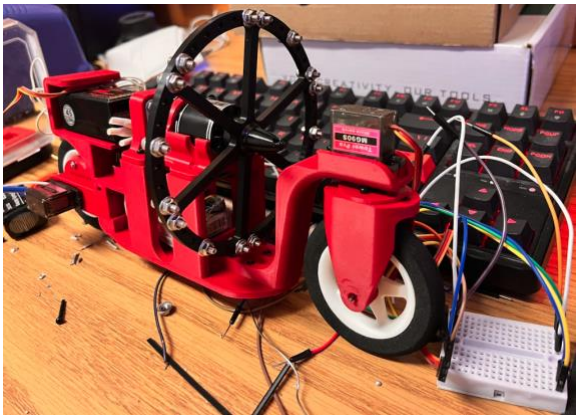

Figure 4: Side view of prototype

D. Justification of Methodological Decisions

Switching from the 850 kV BLDC motor to a 90 kV BLDC gimbal motor was necessary to achieve the level of torque control required for this system. The 850 kV BLDC motor is designed for high rotational speeds and relatively low torque, which is the opposite of the requirements for a reaction-wheel balancing application. In contrast, the low-kV gimbal motor provided significantly higher torque at low speeds, enabling more precise and stable control of the reaction wheel.

Reducing the amount of mass located at the periphery of the bike was also critical to achieving stable balancing behavior. Initially, the reaction wheel spooled up too slowly; however, as angular momentum accumulated, the system became increasingly difficult to decelerate or reverse. This caused the bike to “fight itself” during direction changes, resulting in a sluggish and unstable response. These dynamics were incompatible with the fast, responsive control required for balance. By removing excess peripheral mass, the system was able to respond more quickly and exhibit the sharp, controlled corrections necessary for stable operation.

Improving the rigidity of the wheel mounting was another essential factor in achieving successful balance. Even small amounts of mechanical wobble proved highly detrimental, as they caused the system’s reference zero point to drift depending on the magnitude and direction of the oscillation. This drift severely degraded control performance. Ensuring that the wheels were firmly clamped and mechanically stiff relative to the bike frame was therefore crucial to maintaining a consistent reference and achieving reliable balancing behavior.

Electrical shorts were the most severe and costly issue encountered during the project. A total of three motor driver boards were used, shown in Figure 4 The first board was destroyed by an unintentional short at the AS5600 angular encoder module, which caused both the encoder and the motor driver to stop responding. The second board failed due to a stray M3 nut on the workbench, which shorted two resistors on the underside of the board and permanently damaged it. After these failures, strict precautions were taken to isolate the final motor driver from all conductive debris. As a result, the final board has remained fully functional.

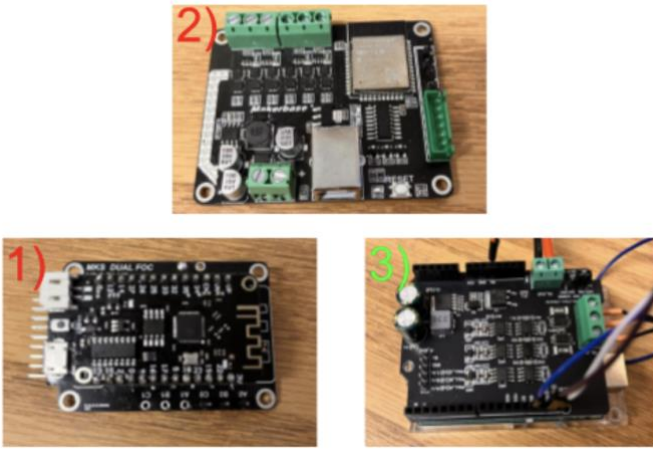

Figure 5: Motor driver boards used

E. *Tuning Strategy*

The input to the closed loop system is the lean angle and the output is the motor speed and direction. The team's tuning strategy was to take a sequential approach; the system was first tuned only as a P-only controller. The team adjusted Kp until the system reached the equilibrium between non-responsive and overshooting. Once the bike was relatively balanced with small constant oscillations, Kd was adjusted until the oscillations were dampened and the balancing became smoother. Ki was only ever added to try to offset any long-term bias.

IV. EXPERIMENTS AND RESULTS

The prototype, including the electronic components, weighs approximately 350 grams The reaction wheel is designed to be light, with most of the mass concentrated on the periphery using metal screws. The reaction wheel can be viewed in Figure 7.

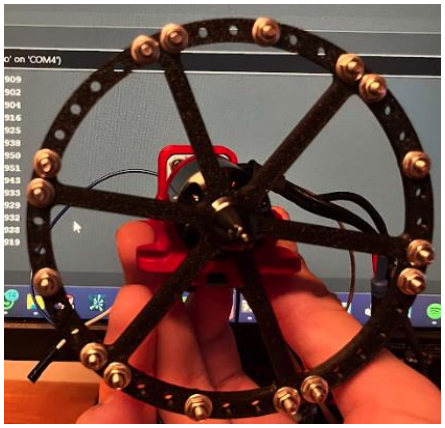

Figure 6: Front view of reaction wheel

The AS5600 encoder can sense a magnetic field up to 3mm away from the chip body. The fixture (shown in figure 6) holds the motor shaft exactly on top of the magnetic sensor. The IMU is placed directly under the motor, perpendicular to the encoder.

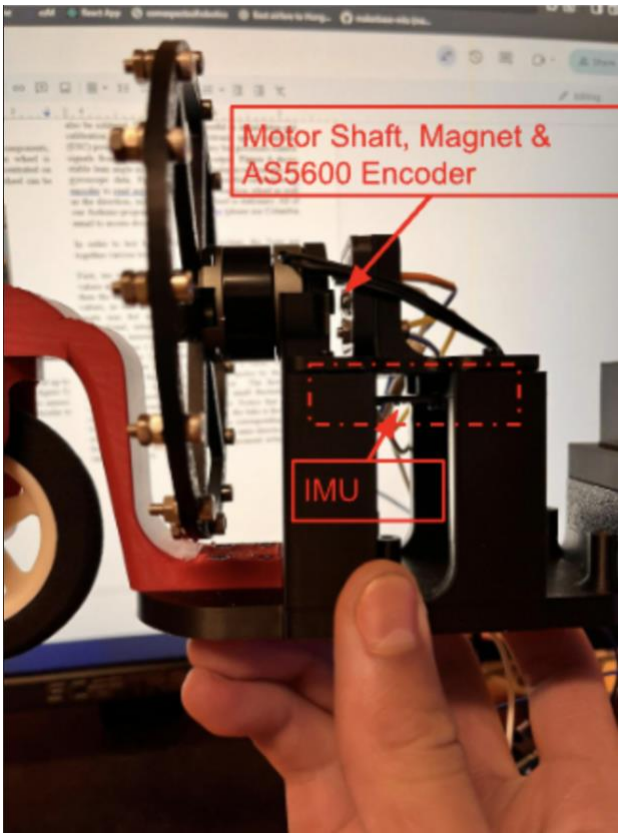

Figure 7: IMU and encoder

All the wires to the IMU and encoder were soldered on. During the testing period we were using a breadboard to connect the sensors to Arduino (both sensors use the SCL and SDA pins) but later those connections were also soldered. The team was successful in initial setup and calibration of both sensors. The motor driver SimpleFOC shield powers the motor from the battery but processes control signals from the Arduino for motor output.

To evaluate the validity and performance of the self-balancing system, the team conducted a series of structured experimental tests designed to verify both signal correctness and overall system stability.

The first test verified that the system produced appropriate control outputs in response to known physical inputs. The complete bike assembly was manually tilted while the onboard IMU measured the resulting lean angle. The lean angle was computed using a complementary filter that fused accelerometer and gyroscope data. This estimated tilt angle was then provided as the input to the PID controller, which generated a control signal intended to minimize the deviation from the upright setpoint.

Figure 8 illustrates this behavior. In this plot, Value 1 represents the estimated tilt angle obtained from the IMU, while Value 2 represents the PID controller output, u. The vertical axis corresponds to the controller output voltage commanded to the motor driver, which is proportional to the torque applied to the reaction wheel. When the bike is at rest, the small fluctuations observed in the tilt signal are attributable

to sensor noise. During this period, the PID output remains near zero, indicating that the controller does not react unnecessarily to small measurement noise.

Figure 8: Lean angle vs motor output (u)

In Figure 9, the bike is deliberately tilted first to the right and then to the left. The corresponding controller response is shown by the change in the PID output. In both cases, the motor response acts in the same direction as the lean, producing a reaction-wheel-induced torque that generates a restoring moment on the bike frame. This behavior confirms that the sign conventions, sensor fusion, and control logic are consistent and functioning as intended.

Figure 9: During bike oscillation

A second validation test assessed the system's ability to maintain balance without external correction. The bike was required to remain upright for a continuous period of at least 30 seconds. To ensure repeatability, the team standardized the initial conditions for each trial. The bike was placed on the same support block for initialization and then lowered to the table surface in an identical manner for every run. A total of ten independent trials were conducted, and a 30-second timer was used to determine whether the bike remained balanced for the full duration of each test.

The results of these trials are summarized in Table 1. Successful completion of this test demonstrates that the system can achieve sustained dynamic stability under consistent initial conditions, meeting the primary performance objective of the project.

Table 1: Trial results

Test Condition	Successful Trials
Undisturbed balance	9 / 10
Balance with repeated disturbances	6 / 10

During testing, disturbances were introduced by gently nudging or tapping the bike with a finger. This approach was useful for checking whether the system could recover from small pushes, but it was not a precise or repeatable way to apply disturbances. The strength, direction, and timing of each push varied from trial to trial, which makes it difficult to compare results or draw strong quantitative conclusions.

It was also observed that if the disturbance was too strong, the bike could be pushed past a critical lean angle. Once this angle was exceeded, the system was no longer able to recover, regardless of the control effort applied by the reaction wheel. On average, the team observed the critical angle to be around 15 degrees tilt beyond which the system was unable to recover balance. This behavior indicates that there is a physical limit to the dynamic stability of the system, imposed by factors such as reaction wheel torque, motor speed limits, motor saturation and mechanical geometry.

In future versions of the system, it would be helpful to use a more controlled method to apply disturbances. For example, a small mechanical device such as a spring-loaded plunger or a simple impact arm could be used to apply the same disturbance each time. This would allow tests to be more consistent and make it easier to evaluate how well the controller handles disturbances under repeatable conditions.

V. Conclusion

Beyond the control algorithms themselves, hardware integration and electronic control proved to be equally—if not more—challenging aspects of this project. The tight coupling between mechanical design, motor control, and real-time sensing required repeated iteration across both hardware and software domains.

The team successfully achieved dynamic stability of a two-wheeled reaction-wheel–stabilized bike for sustained periods of 30 seconds. This outcome met the primary project objective of demonstrating closed-loop balance control using a reaction wheel and validated the effectiveness of the chosen control strategy and hardware configuration.

A significant portion of the project effort was devoted to mechanical/electrical engineering challenges, including CAD redesigns, circuit assembly, debugging electrical shorts, and mechanical reinforcement. These issues required extensive troubleshooting and iterative refinement but ultimately led to a more robust and reliable system.

Through this process, the team gained substantial hands-on experience in PID controller tuning, BLDC motor control, and sensor fusion using IMU data. In particular, the sensitivity of the system to sensor latency and mechanical compliance highlighted the importance of tight integration between

software and hardware in real-time control applications. One clear takeaway from the project is that a higher-fidelity virtual simulation of the bike dynamics could have accelerated development. Simulating the system prior to hardware implementation would have enabled preliminary PID tuning and controller validation, potentially reducing the time spent addressing hardware-induced instabilities and unforeseen interactions.

Future extensions of this work could include implementing a full-state observer, improving sensor fusion through more advanced filtering, or transitioning the control architecture to a higher-performance embedded platform. Additionally, extending the balancing duration and adding locomotion functionality would further validate the robustness of the system.

REFERENCES

[1] David Demir and Noah Eriksson Tewolde Berhane. *Self-Balancing Bike with Reaction Wheel: PID-Controlled Self-Balancing Miniature Bike*. Degree Project (First cycle), KTH Royal Institute of Technology, 2024.

[2] Reman Soryani. *Reaction Wheel Control System for a Self-balancing Bike*. Bachelor's thesis, KTH Royal Institute of Technology, 2023.

[3] Pontus Gräsberg and Bill Lavebratt. *Reaction Wheel Stabilized Stick*. Bachelor's thesis, KTH Royal Institute of Technology, 2019.

[4] Walter T. Higgins, Jr. *A Comparison of Complementary and Kalman Filtering*. (Often circulated as a classic tutorial PDF), 1975.

[5] ReM-RC. *Self balancing bicycle (reaction wheel bike)*. YouTube video and associated open-source resources (e.g., Reaction-Wheel-Bike repository / build notes), 2022–2023.

[6] A. René Geist, Jonathan Fiene, Naomi Tashiro, Zheng Jia, and Sebastian Trimpe. *The Wheelbot: A Jumping Reaction Wheel Unicycle*. arXiv preprint, 2022.

TEAM MEMBER CONTRIBUTION

Shah Bhuiyan focused primarily on the algorithmic, theoretical, and software components of the project. This included developing the underlying control equations, researching and adapting sensor interfacing methods, and identifying suitable approaches for IMU data acquisition and processing. Shah was responsible for compiling and integrating the control code that enabled closed-loop balancing of the bike, including PID control logic and sensor data handling. Notably, these contributions were completed alongside full-time employment, requiring coordination across time zones and limited availability.

Sam Thomason was primarily responsible for the hardware and system integration aspects of the project. This included designing and iterating on the mechanical CAD models, coordinating fabrication and 3D printing, and assembling the physical bike and reaction wheel system. Sam also selected, sourced, and integrated the major hardware components, including the motor control boards, BLDC motors, encoders, IMU sensors, batteries, and supporting electronics. In addition, Sam led troubleshooting efforts related to mechanical stability, electrical shorts, mounting rigidity, and overall hardware reliability.

Both team members contributed to experiment design, system testing, iterative debugging, and report writing. The successful integration of hardware and control software reflects the combined efforts of both contributors across mechanical, electrical, and algorithmic domains.