Project Report
on

GYROGLIDER

EKLAVYA MENTORSHIP PROGRAM AT

SOCIETY OF ROBOTICS AND AUTOMATION
(SRA)

VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE, MUMBAI - 19

ACKNOWLEDGEMENT

We extend our heartfelt thanks to Shankari Anandakrishnan and Atharva Kashalkar for your invaluable guidance and support. Your mentorship has been instrumental in our growth and learning, and we truly appreciate the time and effort you've invested in us. Thank you for being such inspiring mentors.

We also like to express our sincere thanks to all the members of SRA VJTI for their continuous support and encouragement. Their collaborative spirit and the opportunity provided by the Eklavya project have played a major role in shaping our work and enabling us to pursue our work with dedication and enthusiasm.

Soham Kute
sohamuk87@gmail.com

Atharva Khare
atharvakhare707@gmail.com

Ameya Tikhe
ameya.tikheofficial@gmail.com

Kesar Sutar
Kesarsuthar4716@gmail.com

TABLE OF CONTENTS

INTRODUCTION

GOAL:

The goal of Project Gyroglider is to revolutionise the Evobourne bot by equipping it with self-balancing capabilities, by harnessing the power of two BLDC motors as thrusters and two N20 motors as wheels.

In this project, we designed a chassis with two BLDC motors and propellers on one side and a hinge mechanism on the other. Using PID (Proportional-Integral-Derivative) control, we aimed to help this setup achieve and maintain balance at a specific angle.

By dynamically adjusting the throttle of the two BLDC motors, we aimed to tackle the challenge of stabilising the plate at the desired angle, ensuring a perfect balance through real-time corrections.

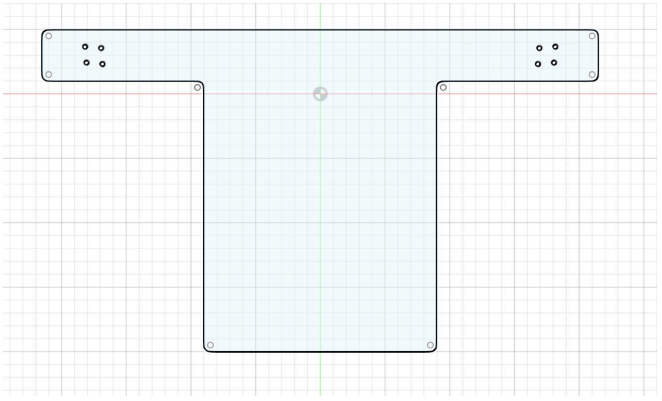
Here are the stages we followed while working on our project:

- **Stage 1:** Chassis Designing in Fusion 360.
- **Stage 2:** Chassis strength analysis on Ansys.
- **Stage 3:** BLDC control using Dshot protocol.
- **Stage 4:** Hardware Assembly
- **Stage 5:** PID implementation on hardware.

Stage 1: Chassis Designing in Fusion 360

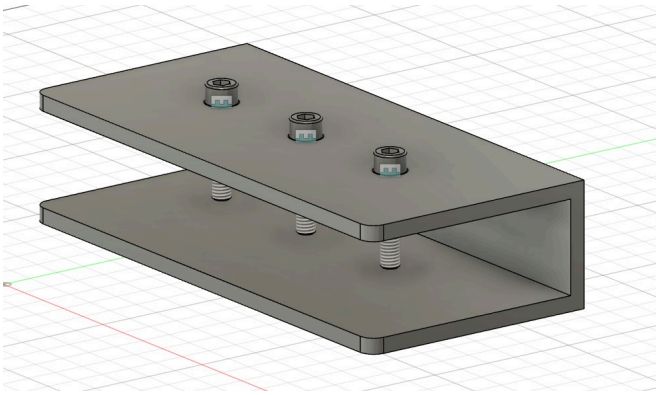
We used Fusion 360 software and designed a chassis that would acquire the SRA board, MPU6050, ESC/DSC, 2 BLDC motors, and a 24V LIPO battery.

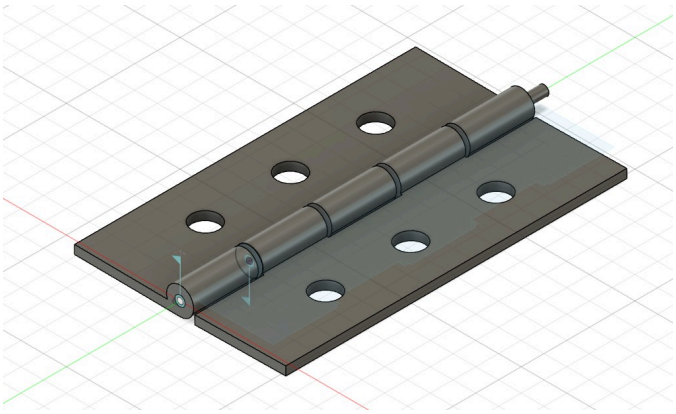
Below are the pictures of Chassis:



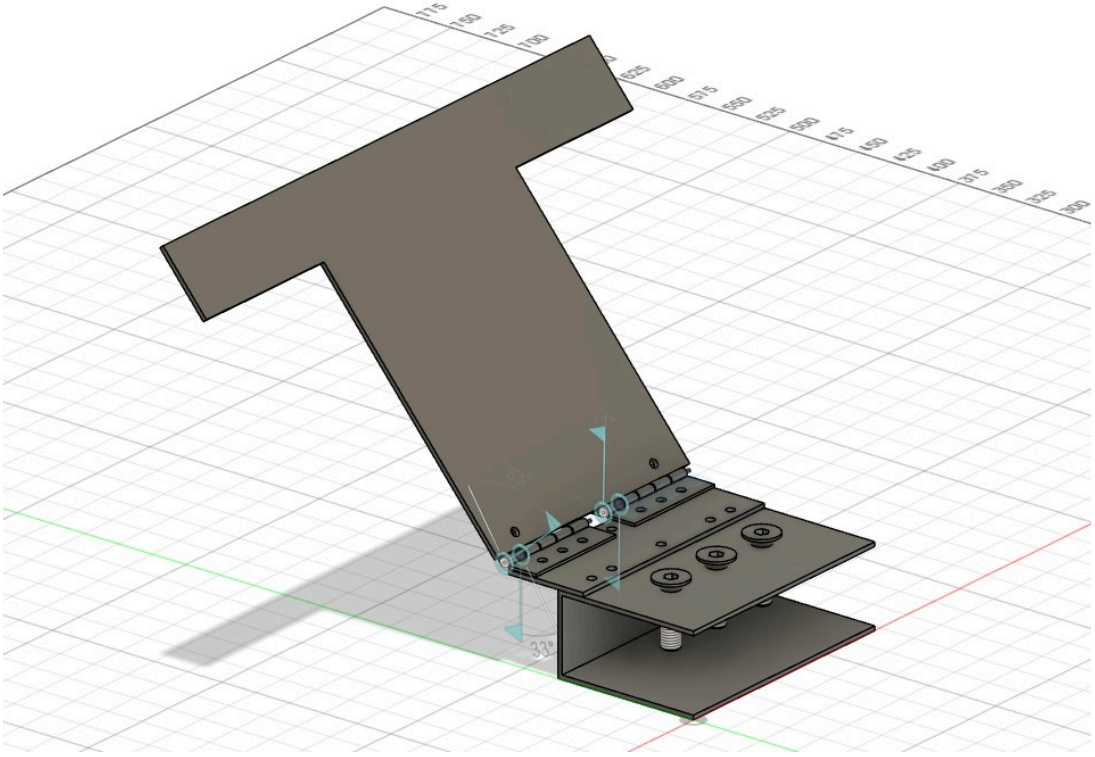
File is provided in .dwg format to laser cutting vendor

We also designed a clamp to interconnect the table and hinge mechanism to one side of the chassis.





And then this is what the final assembly would look like:



Stage 2: Chassis Strength analysis on Ansys

Before manufacturing the chassis, we also conducted a strength analysis to ensure it could withstand the load applied by the BLDC motors.

Stage 3: BLDC control using the DSHOT protocol

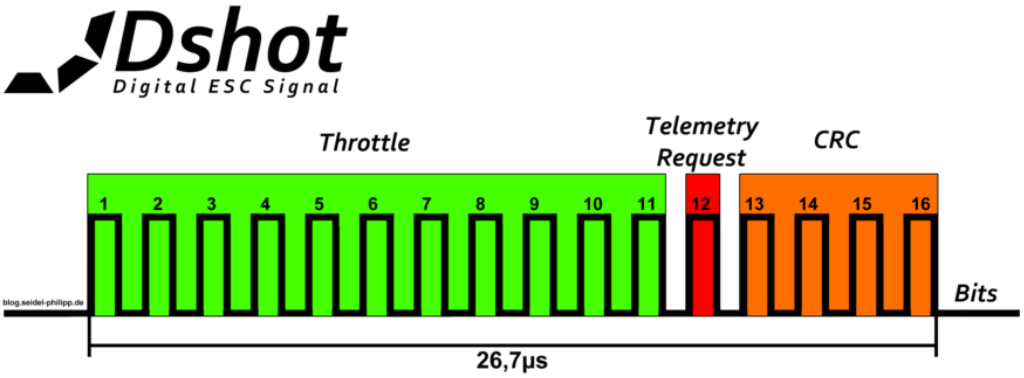
DSHOT (Digital SHOT) is a high-speed digital communication protocol designed to improve the precision and reliability of control signals sent to

electronic speed controllers (ESCs) in drones and other remote-controlled vehicles. Unlike traditional PWM signals, DSHOT transmits commands as digitally encoded pulses, reducing noise and signal degradation.

The DSHOT protocol consists of transmitting 16-bit packets to the ESC: an 11-bit throttle value, a 1-bit to request telemetry and a 4-bit checksum. There are three major protocol speeds: DSHOT150, DSHOT300, and DSHOT600.

DSHOT	Bitrate	TH1	TH0	Bit Time µs	Frame Time µs
150	150kbit/s	5.00	2.50	6.67	106.72
300	300kbit/s	2.50	1.25	3.33	53.28
600	600kbit/s	1.25	0.625	1.67	26.72
1200	1200kbit/s	0.625	0.313	0.83	13.28

The 16-bit data packet in DSHOT is structured as follows: the first 12 bits (0-11) set the throttle value, ranging from 0 to 2047. Bits 12 and 13 carry telemetry or additional information like error status, while the last 2 bits (14–15) handle error-checking to ensure reliable communication between the flight controller and ESC.



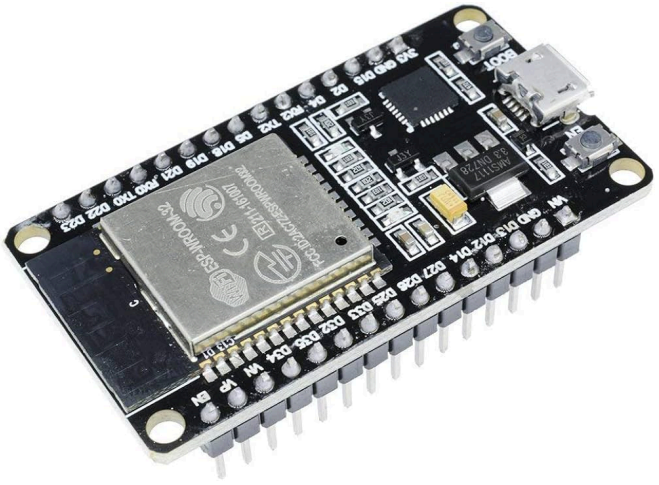
To control BLDC motors using the DSHOT protocol, we needed to generate the DSHOT signal with the ESP32. To achieve this, we wrote code that precisely creates these digital signals, ensuring accurate and reliable motor control.

Stage 4: Hardware Assembly:

In this stage, we assembled the ESP32, SRA board, 4-in-1 Aria ESC, BLDC motors, and MPU6050 onto the aluminium chassis. Here is a brief overview of each component:

1. ESP32 Wrover:

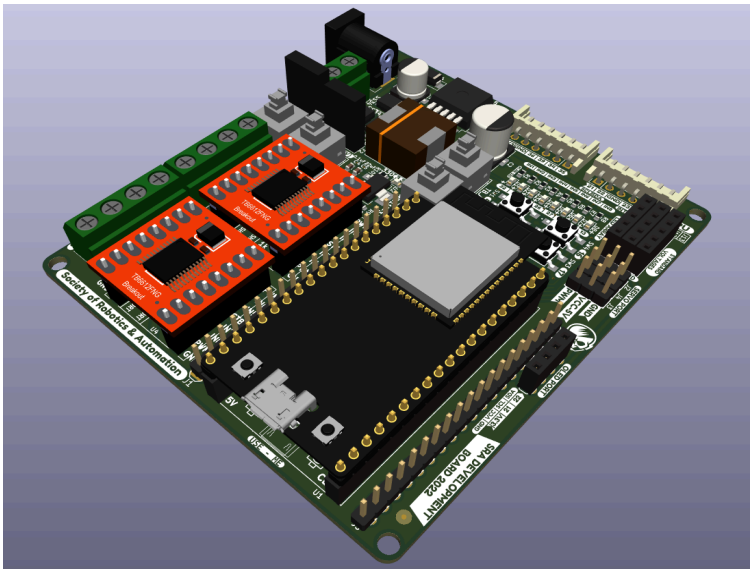
The ESP32, developed by Espressif Systems, is a powerful System-on-a-Chip (SoC) used in wireless communication, IoT, home automation, and robotics. It features dual-core processors, 448 KB ROM, 520 KB SRAM, multiple I/O pins, WiFi, Bluetooth, and LED control. Its versatility and high performance make it a popular choice for developers across diverse applications.



2. SRA board:

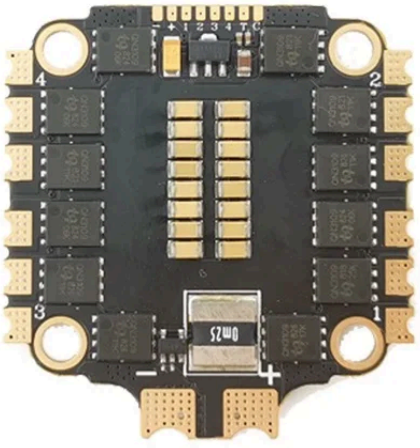
The SRA board is a development board built around the ESP-32 microcontroller. It is designed with various built-in peripherals, including programmable LEDs, switches, sensor ports, and protection circuits for

over-current and reverse voltage. Additionally, the board features motor drivers, enhancing its functionality and versatility for a wide range of projects and applications.



3. DYS 60A Brushless Speed Controller ARIA 4-in-1 ESC

The DYS 60A Brushless Speed Controller ARIA 4-in-1 ESC is a high-performance DSHOT-based electronic speed controller designed for multi-rotor and drone applications. It integrates four independent ESCs into a single compact unit, streamlining the wiring and reducing weight. Each ESC is capable of handling up to 60 amps of current, providing precise and reliable power management for brushless motors.

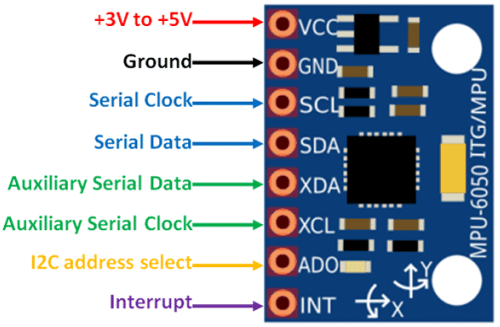


The ARIA 4-in-1 features advanced firmware with programmable settings for throttle response, brake, and other parameters to optimize

performance based on the user's specific needs. It supports various protocols and is compatible with popular flight controllers, ensuring seamless integration into a wide range of drone setups. This ESC is built to handle the demands of high-speed and high-performance flight, making it ideal for both racing and professional-grade aerial photography

.

4. MPU6050:



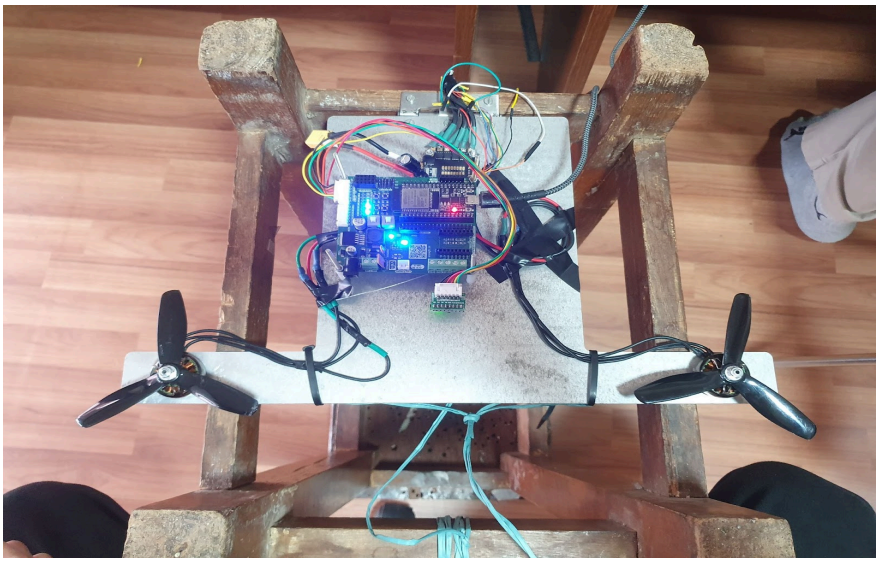
The MPU6050 is a 6-axis sensor that integrates a 3-axis gyroscope and a 3-axis accelerometer. It provides real-time measurements of rotational motion and linear acceleration along three axes. Using I2C for communication, it includes a Digital Motion Processor (DMP) to manage advanced motion tracking functions. This sensor is widely used in robotics and drones for precise stabilization.

5. BLDC Emax 2000KV motor:

The Emax 2000KV BLDC motor converts electrical energy into mechanical motion efficiently, operating at 2000 RPM per volt. Its brushless design minimizes wear and enhances longevity. Controlled by an Electronic Speed Controller (ESC), it offers precise control and high-speed performance, making it ideal for fast-moving applications like drones and racing vehicles.

Final Assembly:

Our assembly process started with a laser-cut T-shaped base plate. We mounted two BLDC motors at the horizontal ends of the T base plate. At the center of mass, we attached the MPU6050 by drilling holes and securing it with screws and spacers between the plate and the MPU6050.

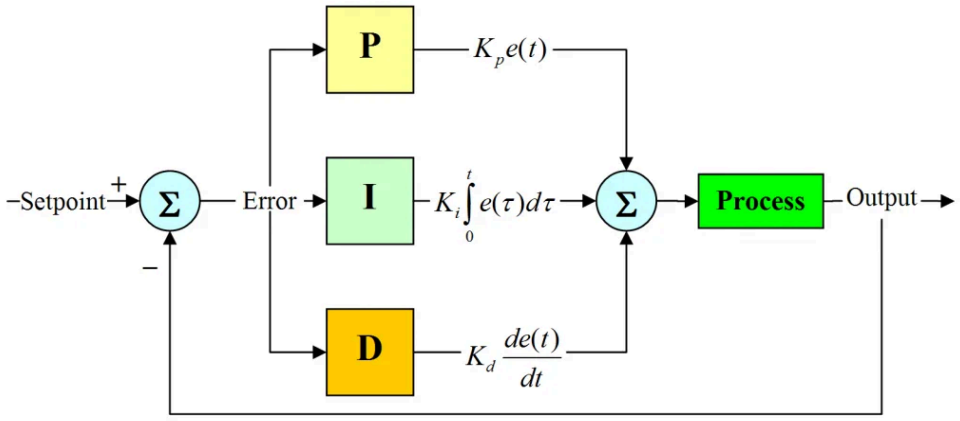


`

After that, we attached the SRA board (ESP32 microcontroller) and ESC on the centre of the T bone of the aluminium chassis. We used screws and spacers to keep them elevated to avoid any current flowing in the aluminium chassis damaging other components. Since the T-shaped base plate has a hinge mechanism at another end, we interconnected the stool and hinge mechanism with nuts and bolts. We also tied it to a pillar for added safety during testing.

Stage 5: PID implementation

A PID controller is a type of control system used to manage and regulate various processes. In essence, it is a feedback control system designed to adjust the output of a system to achieve a desired goal by continuously comparing the actual output to the desired setpoint. The PID controller uses three distinct control actions—Proportional, Integral, and Derivative—to calculate the control input required to minimize the error and stabilize the system.

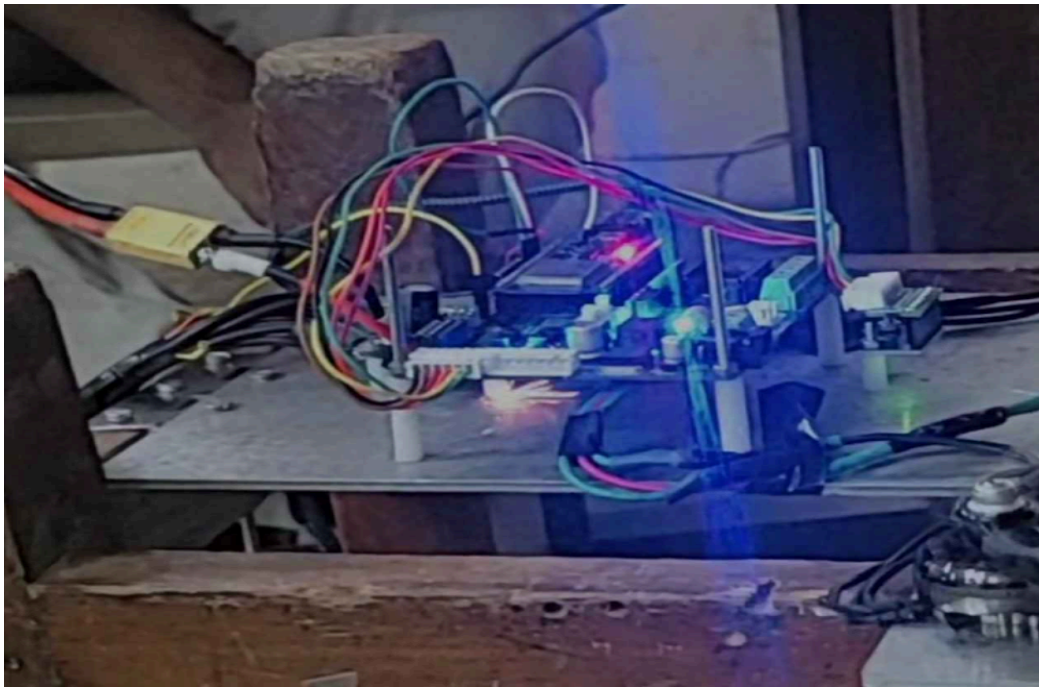


The proportional component adjusts the control output proportionally to the current error, which is the difference between the desired setpoint and the measured process variable. The integral component accumulates the error over time, addressing any residual steady-state errors by adjusting the control output to eliminate them. The derivative component predicts future errors based on the rate of change, providing a damping effect that helps to counteract rapid changes and stabilize the system. By tuning these three parameters, a PID controller can achieve desired performance characteristics.

CHALLENGES ENCOUNTERED:

Issues in hardware while testing:

1. Sudden sparks were observed through the capacitor. The possible reasons might be it may not be properly soldered to ESC or may



   be its positive and negative terminals shorted.

2. Later on, we discovered that some sparks were through screws holding the SRA board, so we changed the position of the SRA board.
3. When we were building our code and flashing in ESP32, we observed that the MPU was getting tripped because of voltage in the aluminum chassis, which was getting transported to the MPU via screw.
   But after replacing BLDC motors, the issue was resolved.

4. Lately, we too observed fluctuations in MPU readings because of the magnetic field from BLDC, as the MPU was near both the BLDC motors.

5. The stool too flipped due to wrong values provided through the websocket server, which were dependent on kp, ki, and kd. But later, for safety, we tied the stool to a pillar.

We too were in the learning and implementing phase, so we too made small mistakes and learned from them, which helped us to debug hardware issues.

CONCLUSION:

We achieved several milestones in our project:

1.Understanding PID: We gained a deep understanding of how Proportional Integral Derivative(PID) works and its application in the control system.

2.Understanding DSHOT protocol: Exploring DSHOT excelled us through an efficient way rather than the old traditional PWM method and implementation of DSHOT, as it is reliable and efficient communication between the flight controller and ESC in a drone.

3.Fusion 360 and Ansys: We really sharpened our 3D modeling, design, and analyzing skills by diving into Fusion 360 and Ansys and getting hands-on experience.

4.Embedded C Proficiency: We gained valuable experience in embedded C programming and got experience in integrating code for MPU6050, DSHOT, Websocket server, and PID.

RESULT:

Although we made significant strides, the intricate nature of the substantial challenges, and regrettably, we did not achieve our main goal of enabling the Evobourne to operate in semi-drone mode for climbing and flying in rugged terrains.

However, this project has equipped us with valuable insights and skills that will be advantageous for future projects. The difficulties we encountered have contributed to our development and enhanced our ability to handle complex projects with greater effectiveness.

It's important to underscore the importance of our system, which has extensive applications in the balancing and stabilization of dynamic systems. Such systems are essential for maintaining control and stability across a variety of uses.

BIBLIOGRAPHY:

-Linear Algebra
-ESP-IDF Framework
-Understanding PID Control
-DSHOT protocol