Bird Simulator

Anthony Amella

Emily Liu

Eli Yang

Final Report for ECE 445, Spring 2026

TA: Shiyuan Duan

6 May 2026

Project No. 2

Abstract

The Bird Simulator is a bird-inspired FPV drone system designed to overcome the limitations of traditional FPV control by enabling full-body motion control. Current FPV systems offer immersive visuals but lack the physical control necessary for quick, reflexive responses, which is critical in high-risk scenarios like search-and-rescue. Our solution integrates multiple IMUs into a wearable suit to capture the pilot's arm motion, head orientation, and torso tilt, converting these movements into drone maneuvering instructions transmitted via a 2.4 GHz transceiver. A drone with a custom carbon fiber body that houses a flight controller, motor controller, and video transmitter receives these signals for control. The pilot receives analog video through 5.8 GHz FPV goggles. This design also includes a traditional controller with an emergency override, as well as a drone flight simulator to allow for pilot training without flying in open air. This system aims to create a more intuitive and responsive human-machine interaction, enhancing both entertainment and safety-critical applications by leveraging natural human reflexes.

Contents

1. Introduction

1.1 Problem

This project addresses the limitations of current first-person view (FPV) drone systems, which can create an immersive visual experiment, but lacks the ability to allow for physical control through the human body through motion cues or body orientation. This results in an experience with a realism factor for people who want an even more exhilarating experience.

Beyond entertainment, FPV drones are very useful in situations where it is not feasible to place humans directly in the environment, such as search-and-rescue operations, disaster response, or other hazardous settings. In these high-risk scenarios, rapid situational awareness is critical. Human reflexes naturally respond to perceived threats by instinctively moving their head or body to shift attention, for example, ducking when approaching an obstacle, or turning your head when there is movement in your peripheral vision. These reflexive movements are much faster than the conscious decision to move your hands to control the sticks to reorient the drone to face the threat. Combined with traditional stick control, human reflexes can be leveraged to create a more responsive tool valuable for applications where quick response time is essential.

1.2 Solution

To address this problem, we designed and implemented a bird-inspired FPV drone system that allows the pilot to control flight using full-body motion rather than relying on a traditional controller. Multiple IMUs mounted on a suit capture human movements such as arm motion, head orientation, and torso tilt. A microcontroller then reads the data from the IMUs to convert them to instructions transmitted by a 2.4 GHz transceiver that the drone receives and uses to maneuver in its environment. The pilot is able to see what the drone camera sees through FPV goggles, which receives analog video transmitted over 5.8 GHz radio. We also made a traditional drone controller that can be used to override suit inputs and take over control in case the drone starts behaving erratically. The controller also transmits signals with the 2.4 GHz transceiver.

To allow development of the drone, suit, and video transmission in parallel, we also built a web-based drone simulator using JavaScript. This simulator provides a safe testing environment for validating controls and other algorithms without the risks of real-world flight, lowering the risk of injury and damage to hardware during development, and allowing more iterations for trial and error.

1.3 Visual Aid

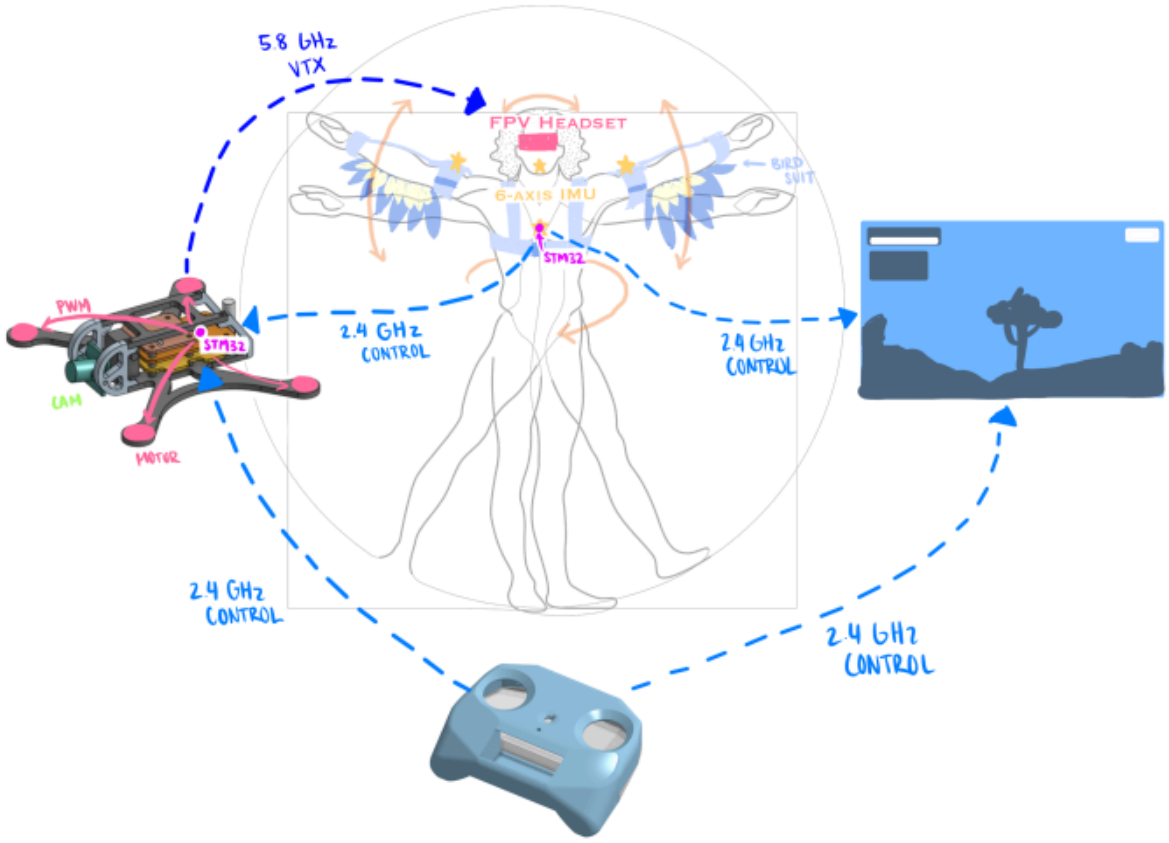


Figure 1. Visual aid for bird simulator

1.4 High Level Requirements

- The bird suit must measure values from a set of IMUs, and transmit data over 2.4 GHz to the drone, allowing for control similar to joystick controller inputs.
- The drone must receive 2.4 GHz data from the bird suit or controller, and fly through the air with assisted control using an onboard IMU.
- The drone uses a camera to generate and transmit a 5.8 GHz analog video signal, which an off-the-shelf FPV headset receives.

2 Design

2.1 Block Diagram

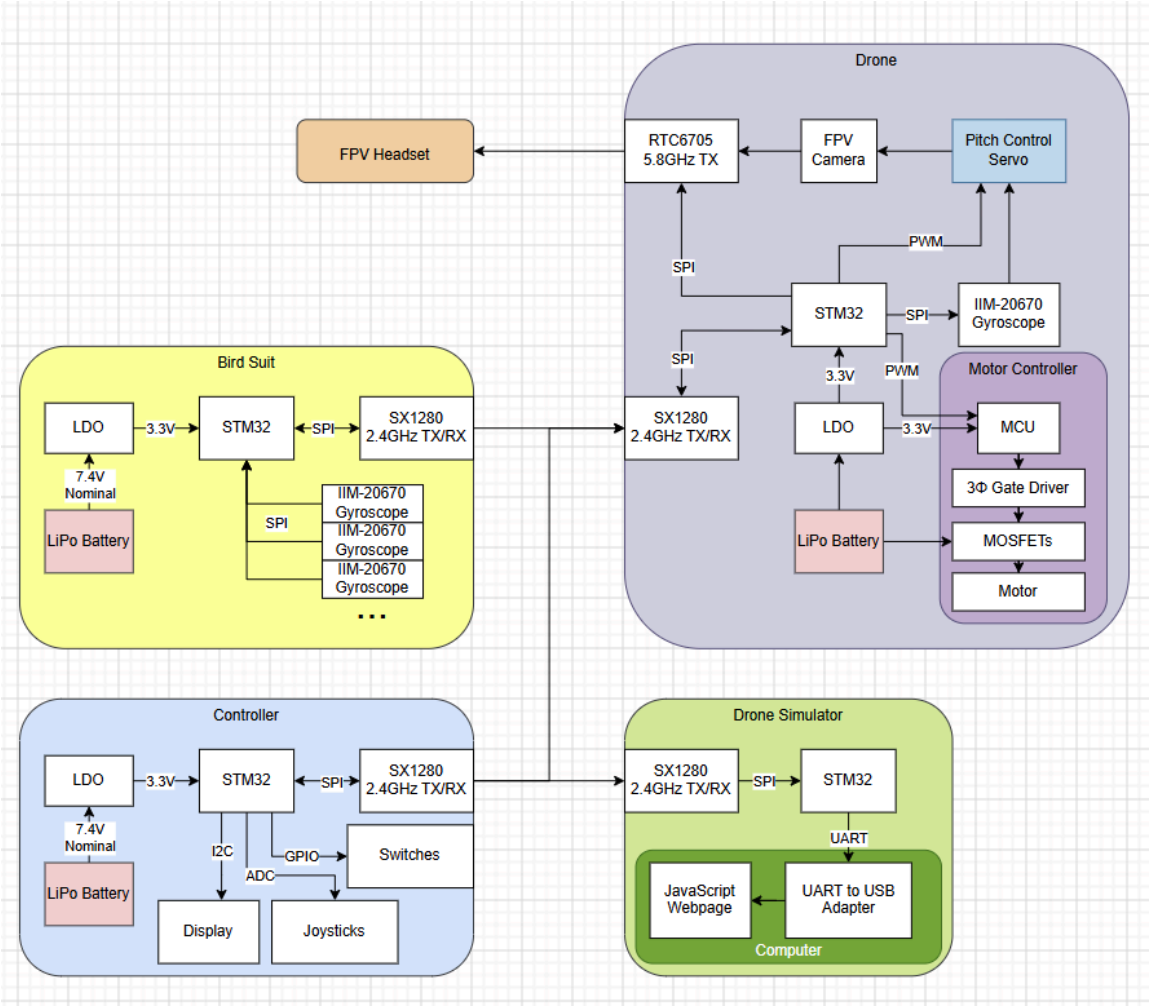


Figure 2.1. Block diagram for bird simulator

2.2 Subsystem 1: Drone

2.2.1 Flight Controller

2.2.1.1 Hardware

The control of the drone is centered around an STM32F410CBT3 microcontroller. This STM32 communicates with an SX1280 to receive control signals from either the bird suit or the controller [3]. These signals contain the raw inputs for the throttle, pitch, yaw, and roll of the drone. The STM32 then uses these signals, as well as the signals from the onboard IIM-20670 IMU, to generate four PWM signals for the four motor controller boards. It also generates a PWM signal for the pitch control servo to keep the camera level.

The flight controller has two connectors to attach to other boards. The motor controller board communicates with the flight controller using a 2x8 connector with 1 mm pitch. This connector carries battery power and ground to power the flight controller, since the battery plugs into the motor controller. The connector also carries SPI signals (MISO, MOSI, SCK), four chip select signals, which are used when configuring the gate drivers, as well as four PWM signals which are used during flight to control motor speed. The video transmitter board also communicates with the flight controller using a 1x8 connector with 1 mm pitch. This connector carries 3.3 V for the video transmitter chip, as well as 5 V to power the camera. SPI signals are sent to communicate with the video transmitter and the magnetometer on the auxiliary VTX board. These connectors line up with matching connectors on the motor controller and video transmitter board, allowing the boards to stack directly without any cables.

The flight controller board has connectors for programming the STM32 and communicating with the board over UART for debugging. A 3-pin servo connector is also present to provide PWM signals to control the camera leveling servo.

By constantly integrating the angular velocity for pitch over time, we can estimate the absolute angle of the drone. The camera is kept level with the servo, which is driven by this data. The camera signal from either the controller or the bird suit is then added, allowing the camera to look up or down according to the pilot, while keeping a constant pitch as the drone flies through the air.

Two LEDs and a pushbutton are also present. The drone's status is displayed on the LEDs, and the control loop parameters are changed using the pushbutton for flight mode versus frame mode.

The footprint of the board relies on a 20 mm grid, which is standard for existing drone parts. This allows us to test flight with off-the-shelf components, such as a motor controller, while we develop our custom boards.

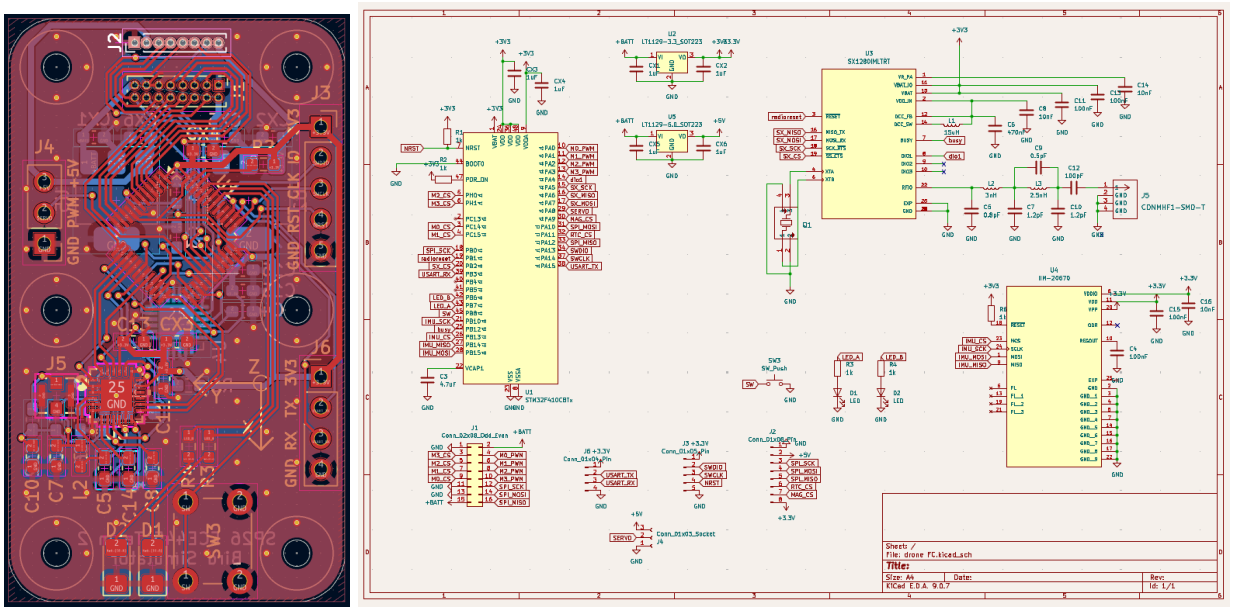


Figure 2.2: Schematic and PCB for drone flight controller

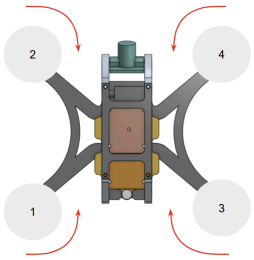
2.2.1.2 Software

A gyroscope (IIM-20670) is equipped on the drone to determine its orientation during flight. The flight controller uses this information to generate motor commands that match the drone flight with the movement of the bird suit, as well as account for turbulence that may cause the drone to lose control. Drone flight control, at its core, uses stick inputs to control angular velocity in each of the three rotational axes. The setpoint for angular velocity in one of the axes, $SP(t)$, is determined by the movement of the roll, pitch, and yaw sticks on the controller, or the corresponding movement of the pilot's body with the bird suit. The drone then measures the actual angular velocity of the drone in that axis, $PV(t)$, and calculates an error function as the difference between these two measurements. A PID control loop is used to determine the signal to send to the motor controller. Three constants, K_p, K_i, and K_d, are tuned to determine the relative effects of the error, the integral of the error over time, and the instantaneous derivative of the error on the motor output.

$$e(t) = SP(t) - PV(t)$$

$$Out = K_p e(t) + K_i \int_0^\tau e(t)d\tau + K_d \frac{de(t)}{dt}$$

Roll, pitch, and yaw are computed through three separate PID loops. These calculations, combined with the throttle input, determine the PWM output for each of the four motors.



$$M1 = T + P - R - Y$$

$$M2 = T - P - R + Y$$

$$M3 = T + P + R + Y$$

$$M4 = T - P + R - Y$$

Figure 2.3. Drone top view with propeller rotation drawn

The drone runs a separate PID loop to calculate values to zero angular position when the pitch and roll sticks of the drone are centered and the drone is not level. Angular position is calculated using small angle approximation with the accelerometers, since the drone remains at relatively small angles relative to level. The flow chart for the two control loops is shown in Figure 2.4.

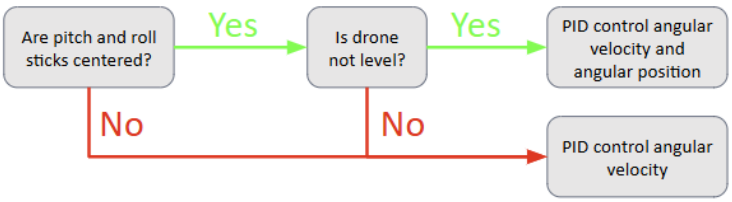


Figure 2.4. Dual PID flow chart

2.2.2 Motor Controller

2.2.2.1 Custom Controller

The motors for the drone are controlled on a separate board, placed under the flight controller. The motor controller board is directly connected to the battery, and this power is passed on to the flight controller. Four TI MCT8329A trapezoidal BLDC gate drivers are present, each driving six SiSHA14DN MOSFETs to individually control the speed of each motor. Each of the three phases of the motor are connected to two MOSFETs, one to ground and one to VCC, which are then controlled by the gate driver to drive the motor. This gate driver is programmed over I^2C to non-volatile EEPROM, so it only has to be programmed once for the operation of the drone. The motor controller receives four separate PWM signals from the flight controller to determine the speed for each motor.

To spin the motor, we needed to program the gate drivers via I^2C. We used a USB I^2C device to interface with the driver; successful communication would have allowed us to test its basic functionality on the shadow registers. This testing would cover motor control behavior, including speed control, current limitations, and acceleration/deceleration profiles. After determining the final parameters, we planned to store them using EEPROM writes. Unfortunately, the custom controller failed when the individual gate drivers would not accept I^2C programming. We made many attempts to secure register acknowledgment, such as checking target addresses, performing simple reads and writes, but the driver acknowledged nothing. We probed the USB I^2C device with an oscilloscope to confirm it transmitted the correct data payload. Additionally, we attempted debugging methods like changing the control word, ensuring the correct operating state, replacing the chip, and desoldering and resoldering the entire channel. Ultimately, we could not bring up a working motor controller.

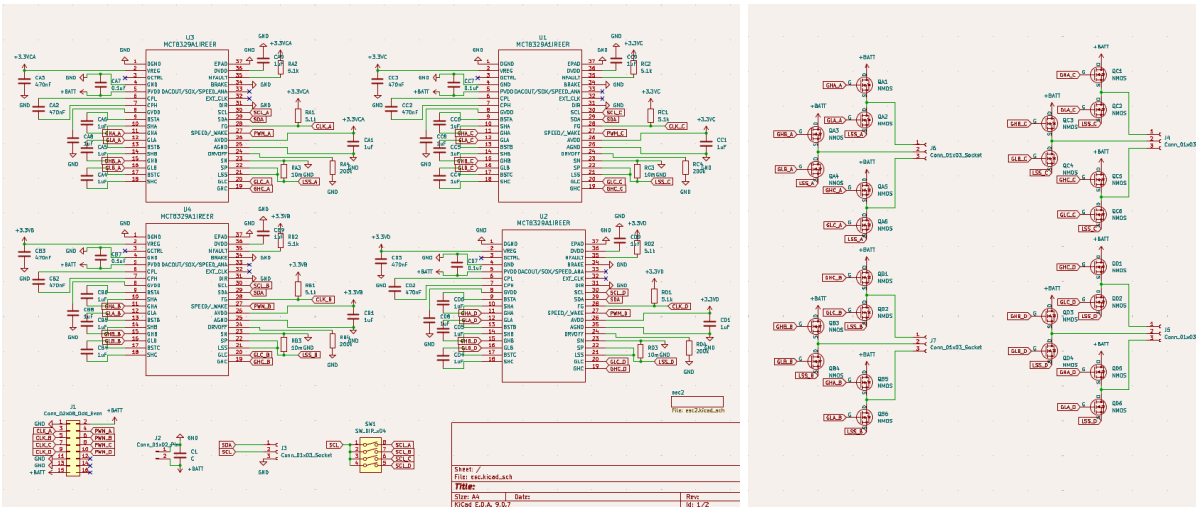


Figure 2.5. Motor Controller Schematic

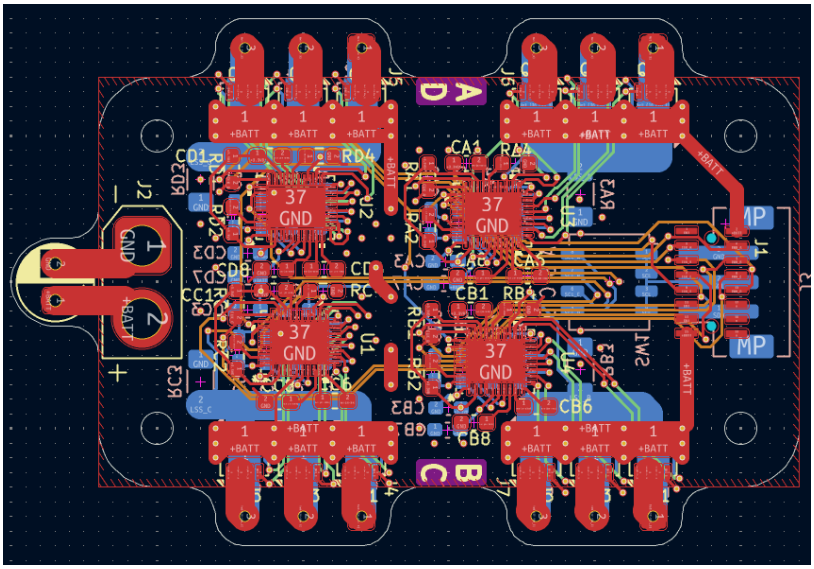


Figure 2.6. Motor Controller PCB

2.2.2.2 Off-the-shelf Replacement

Since we were not able to successfully program the custom motor controller, we bought an off-the-shelf motor controller to take the place of the custom motor controller. Similar to the custom board, the off-the-shelf replacement takes four separate PWM signals from the flight controller and uses them to independently drive the four motors. This motor controller also mounts on the 20 mm grid the drone was designed around, so it is effectively a drop-in replacement.

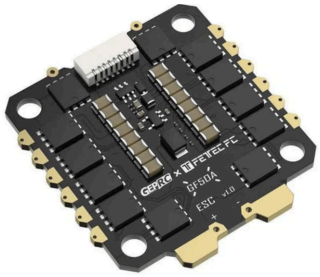


Figure 2.7. Replacement motor controller

2.2.3 Video Transmitter

We used 5.8 GHz radio to transmit video data from the drone to FPV goggles using the RTC6705 transmitter module. This RF module handles amplifying, mixing, and modulating/demodulating signals, and the STM32 configures and programs the module using SPI. The camera outputs analog video, which the RTC6705 transmits.

The VTX board houses the RTC6705, as well as a magnetometer, which share an SPI channel but have separate chip selects. This board connects to the flight controller board from which it receives ground, +3.3 V to power the RTC6705 and magnetometer, +5 V to power the camera, SPI clock and data shared between RTC6705 and magnetometer, and two chip selects. The magnetometer was not used in the final

design, since we found that the autoleveling was sufficient with just the 6-axis IMU data. An external antenna connects to the PCB through a U.FL connector, which transmits and amplifies the RTC6705's RF output. The initial π-network design was based on existing designs including the RTC6705 [10].

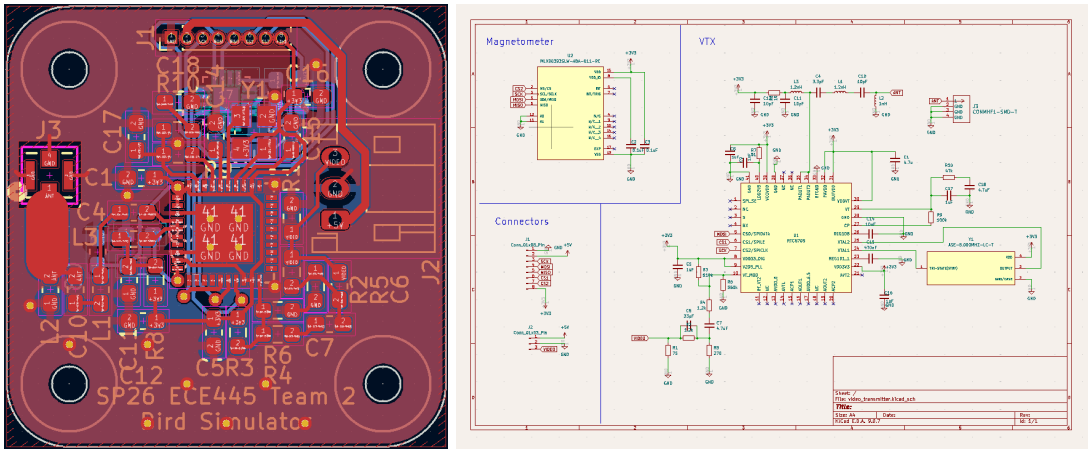


Figure 2.8. Schematic and PCB for video transmitter

The RTC6705 datasheet [5] specifies that it takes half-duplex SPI, 25-bit commands, with LSB first. This is difficult to implement properly using the built-in SPI HAL library, since it sends data in bytes. Any data transferred that is not part of the RTC6705 command, even if it is all zeros, can cause unexpected behavior. To mitigate this, SPI communication was emulated by manually toggling GPIO pins, also known as "bit-banging". This meant that the MOSI pin, which is now a GPIO data pin, needed to switch between input and output modes for each write or read operation. Like with normal SPI, data is transferred every time the clock pin is switched high. With bit-banging, the clock cycles do not have a consistent 50/50 duty cycle, but SPI does not require a consistent clock, and the STM32 was able to read the module's status and configure it for operation.

The initial design of the VTX board produced no RF output when the headset scanned the available frequencies. This is because the impedance matching π-network design was not compatible with 5.8 GHz transmission with 50 Ω antenna impedance. An s-parameter simulation, which computes the frequency-domain characteristics of an RF circuit across a frequency sweep, was performed on the initial design using the Qucs-S circuit simulation software [12]. Figure 2.9 shows the simulation results. The Cartesian plot shows that for a frequency sweep of 5.7 GHz to 5.9 GHz, 100% of signal transmitted is reflected back to the input and 0% is transmitted to the antenna. This means that the initial π-network design is seriously detrimental to the signal strength and integrity of the RF output.

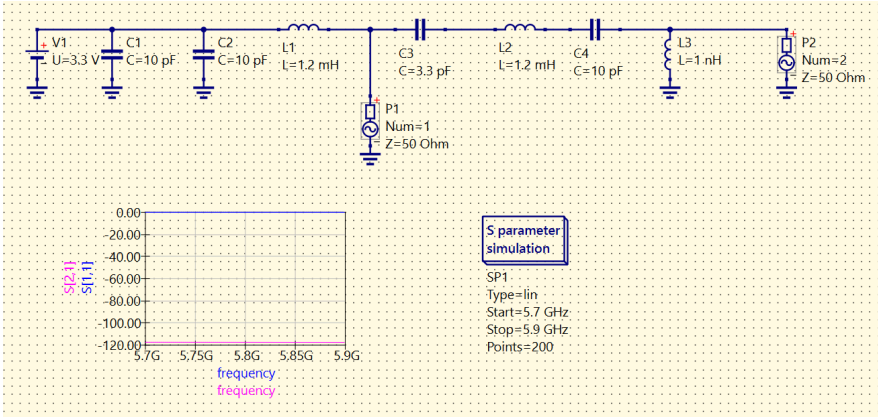


Figure 2.9. VTX π-network s-parameter simulation with original design

The π-network was redesigned using a π-network calculator and subsequently resimulated [9]. Figure 2.10 shows the results of this simulation. The updated Cartesian graph reveals a pronounced dip in signal reflection near 5.8 GHz, accompanied by a small increase in transmission power. This indicates that a greater portion of the signal is being effectively delivered to the antenna. The Smith chart further supports this observation, showing very short arcs that correspond to low impedance variation around 50 Ω across the swept frequency range. Due to component value and time constraints, we adapted the existing hardware design rather than fully redesigning. Unused pads were left unpopulated or fitted with 0 Ω resistors, while the required capacitance and inductance values were achieved by stacking components in parallel to approximate the calculated values.

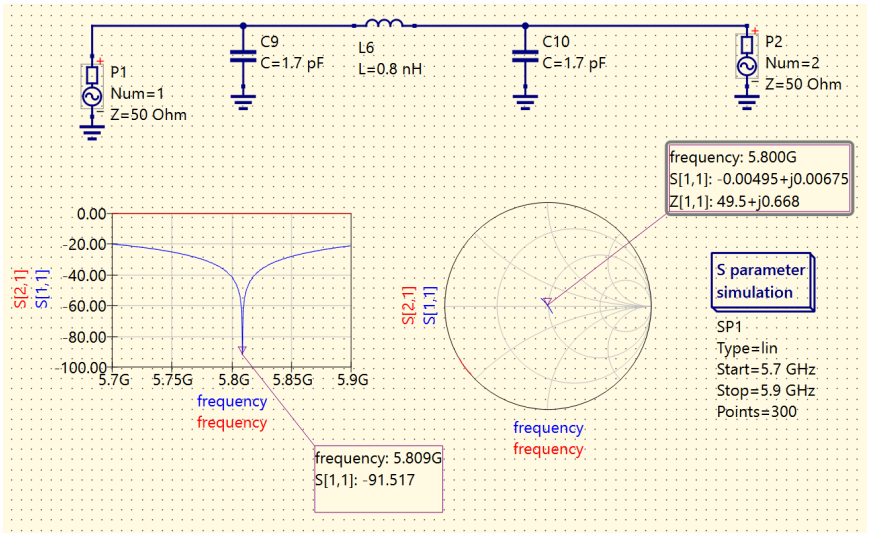


Figure 2.10. VTX π-network s-parameter simulation with revised design

Another problem with the initial design is the transmission line between the π-network and the antenna connector. This trace should have a controlled impedance of 50 Ω to minimize reflections. According to the KiCAD built-in trace impedance calculator, the trace on the board is approximately 12 Ω. This impedance mismatch is another source of signal loss. To fix this issue, we disconnected the trace using a

hacksaw to cut into the PCB. We then removed the connector, stripped the antenna, and soldered the exposed signal and ground wires directly to the output of the impedance matched π-network.

The camera also initially received +5 V from the flight controller for power. During testing, it was found that when the video output was choppy, the servo movements synced with when it cut out. This implied that the camera was not getting enough power when the servo was moving, which was fixed by resoldering the camera's wires directly to the battery terminals. With the changes to the π-network, transmission line, and camera power applied to the board, we successfully generated a signal with enough strength and integrity to see video through the headset at a range of 10 m.

2.2.4 Mechanical Design

The drone body is made primarily from 3 mm carbon fiber sheets. There is one large body frame at the bottom of the drone, where the battery, motors, and PCBs attach, as well as a smaller top frame. The two carbon fiber frames are connected by two aluminum pieces, which hold the camera as well. The aluminum was manufactured by the ECE machine shop with a CNC mill, and the carbon fiber parts were cut using a waterjet.

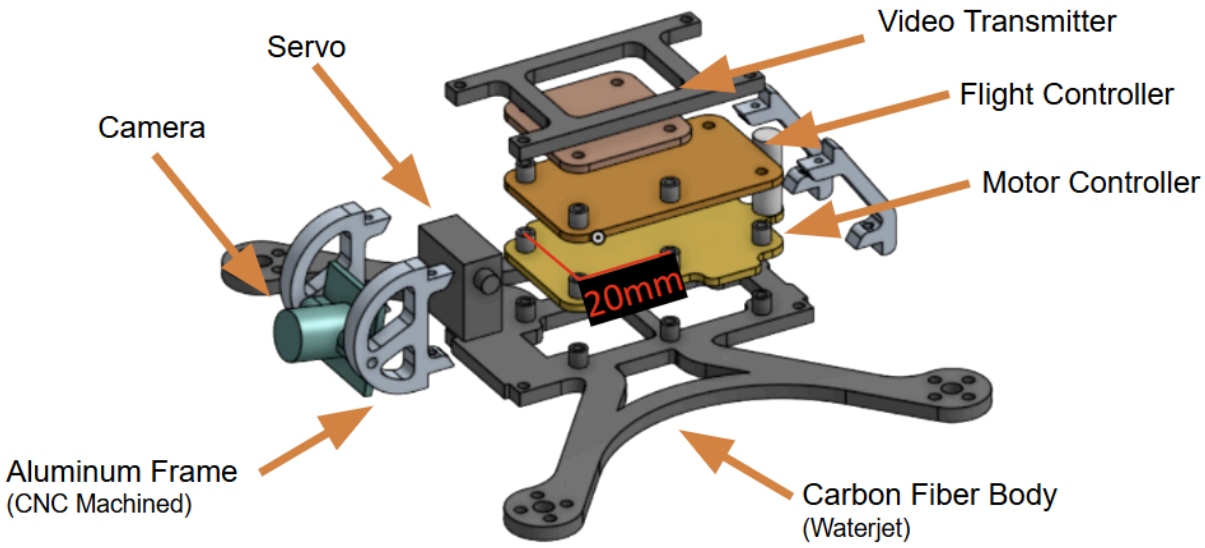


Figure 2.11. Drone frame CAD (exploded view)

Each of the three PCBs can be stacked directly on top of each other with the matching mechanical frame with a 20 mm mounting grid that can also support potential off-the-shelf components. The pitch of the camera is controlled by a servo that is located directly behind the camera.

Table 1. Drone Requirements, Verification, and Results

Component	Requirement	Verification	Results
Motor Control	Motor is able to spin at a maximum speed of at least 5000RPM. The speed of the motor can vary according to a PWM signal by a factor of at least 5 (e.g. if the max speed is 5000RPM, the motor can spin as low as 1000RPM)	Use a slow-motion camera to capture the motor while it runs at the maximum and minimum speed. Count camera frames to calculate RPM	Used tachometer to verify motor speed range of 3.2k-60k rpm using off-the-shelf motor controller
Camera Levelling	Camera is continuously kept within 10 degrees from level with the ground, across a maximum camera tilt range of at least 90 degrees.	Move the drone in the air across a range of acceptable angles. Use the iPhone Measure app to measure camera pitch.	Drone is able to measure absolute angle and camera servo can be moved to keep camera level.
Drone Flight Control	Drone is able to measure pan, tilt, and roll from IMUs, and individually adjust motor speeds to move towards a more level position, within a maximum range of +/- 30 degrees in the tilt and roll axes.	Drone is held in the air, and moved across a range of angles. Capture motor speed using a slow-motion camera, which must attempt to compensate for movement, such that the drone tends towards a level position.	IMUs can measure angular velocity and linear acceleration, allowing for a control loop to compensate for movement with less than 30 degrees of range. Tachometer used to verify.
VTX	Transmit a signal with RSSI value from receiver of -60 dB or stronger within a radius of 10 m	Read the RSSI value on the receiver headset and ensure that it is greater than -60 dB with the transmitter 10 m away	Headset is not able to give exact RSSI value, but video is visible from 10 m
Mechanical Structure	Drone is fully functional after 10-foot drops on a rigid floor at any angle. No significant damage occurs and all above requirements are still met.	Drone is held 10 ft in the air, verified with a tape measure, and dropped at a wide range of angles.	Dropped from ~50 ft multiple times and remains intact and fully functional
Emergency Shutoff	If the drone does not receive any control signal for 2 seconds, cut off power from all motors.	Some input device will initially control the drone before powering off. Use a timer to ensure the drone shuts off in time.	Removing controller battery causes no radio packets to transmit and motors turn off after 0.5 seconds

2.3 Subsystem 2: Bird Suit

There are 4 IIM-20670 modules embedded in a wearable suit that collect data used to determine the motion and orientation of the user: one on each arm, one on the back of the head, and one on the back. The IIM-20670 module transfers gyroscope and accelerometer data to the microcontroller through SPI. [7]. Movements such as head rotation, wing flapping, body orientation translate to stick inputs on a drone controller. This information is organized into a series of bytes, sent to the 2.4 GHz transceiver chip (SX1280), and transmitted over radio to the drone. This subsystem is powered with a LiPo battery by a 3.7 V nominal voltage, which is stepped down to 3.3 V by an LDO.

To extract meaningful body movement and orientation from the raw gyroscope and accelerometer data collected by the IIM-20670 modules, we scaled the raw data to convert angular velocity into units of dps and linear velocity in units of g's. This scaled angular velocity and accelerometer data is used to approximate human movement and translated to drone input controls which we assigned as follows:

1. Throttle: The angular velocity for the left and right arm is subtracted to determine synchronous movement, and then ignored if they are negative. This leads to a value that is positive when both arms move down at the same time, and zero otherwise. A small constant subtracts from this number, and finally added to the current throttle. This leads to a value that increases when the pilot 'flaps' their wings, and decays over time. This causes the bird suit to slowly descend, requiring occasional flapping.

$$\Delta T = max(0, (\Delta\theta_L - \Delta\theta_R)) - C$$

$$T^+ = T + \Delta T$$

2. Pitch: angular displacement of the torso in the sagittal plane from the neutral position determines the angle of the drone's pitch
3. Yaw: angular displacement of the head in the transverse plane from the neutral position determines the rate of yaw
4. Roll: angular displacement of the arms in their extended position in the coronal plane from the neutral position determines the angle of roll
5. Camera pitch: angular displacement of the head in the sagittal plane from the neutral position determines the angle of camera pitch

Table 2. Bird Suit Requirements, Verification, and Results

Component	Requirement	Verification	Results
Wing flapping	Arm IMUs measure flapping both arms in phase, reflected by increasing the throttle sent. Each flap must give a clearly noticeable, positive change in vertical velocity	Bird suit will transmit to bird simulator, which will clearly reflect drone flight based on IMU movement. Bird simulator allows for	All movements of bird suit are successfully translated to their corresponding stick inputs and control both the drone and the

	according to the intensity of the flap.	inputs to be more obvious, since there will be no wind or issues with flight control	simulator successfully
Wing rolling	Keeping both arms parallel, and changing their angle from normal with the sagittal plane will cause linearly proportional changes in the roll of the drone		
Speed control	Leaning forwards or backwards with the body IMU changes the speed of the drone, proportionally to the angular displacement from the coronal plane		
Camera pitch	The angular displacement between the coronal plane and the head IMU causes the drone camera to pitch the same angle, with a maximum error of 10 degrees, within the range of motion of the camera.	Bird suit will transmit to drone. Measure the head and camera angular displacement with the iPhone Measure app.	Head pitch angle corresponds to the pitch angle of the camera with an error of <10 degrees within the working range of the servo

2.4 Subsystem 3: Drone Simulator

A web-based drone simulator was developed using JavaScript to simulate flight. 3D environment models imported from sketchfab.com are rendered using Three.js. To simulate FPV flight, the camera transforms and rotates around the environment accordinging to the user's controls. The simulator is compatible with both the controller and the bird suit in conjunction with either a dedicated receiver board or the drone's UART pins. The STM32's UART output sends the received packet to the computer through a UART-to-USB adapter, and this output is read and processed by the computer to generate the next frame. The flight physics are based on Newton-Euler equations and are scaled by constants depending on the size of the environment. With the ability to easily modify the backend of this simulator, it is a better tool for the development of the bird simulator than a closed-source existing flight simulator.

Table 3. Drone Simulator Requirements, Verification, and Results

Component	Requirement	Verification	Result
Control	Given stick inputs from a drone controller, the simulator must receive each input and convert them to a value between -1 and 1, representing stick position.	Each command (throttle, yaw, pitch, roll) from the controller results in an appropriate camera movement in the simulator.	Both controller and bird suit send appropriate command to simulator that allows control to feel natural
Latency	To ensure realistic and	A timer will be added to the	Timer added to record time

	responsive control, the time between receiving a packet and calculating/rendering the simulated response should be less than 10 ms.	software to track time between when the simulator receives the packet to when it is finished processing the data.	between frame updates shows that frames are updated fast enough to keep up with display's 60 Hz refresh rate

2.5 Subsystem 5: Controller

The traditional drone controller consists of two switches, two joysticks, a display, and button/joystick, and is powered by a LiPo battery with a 7.4 V nominal voltage, which is stepped down to 3.3 V by an LDO. The microcontroller reads the joystick values with an ADC to be converted to throttle, pitch, roll, and yaw values, which sends the data to the RF transceiver (SX1280) to transmit over 2.4 GHz radio. One of the switches arms the drone, and the other switches the drone to actively auto-level angular position. Camera angle, throttle, yaw, pitch, and roll each use one byte. There is also an identification byte, which allows the drone to know if signals are sent from the suit or the controller. The final byte is a CRC. The value of this byte is an exclusive-or of the first seven bytes, which confirms the packet was received correctly. If the CRC byte is incorrect when received by the drone, it is ignored. The information is sent continuously in packets, which organize the data in this order, with each element being represented by one byte:

`[Arm/Level][Camera][Throttle][Yaw][Pitch][Roll][Identification][CRC]`

The identification byte allows for the controller to override signals sent by the bird suit. As soon as the drone receives a packet with an armed signal from the controller, it ignores all packets from the bird suit until it is power cycled. This allows for flight to be stabilized, and for the drone to be safely landed with a controller, should the drone flight be too unstable from flight commands from the suit.

The controller is equipped with a LCD screen (DOGM204-A 4x20) that interfaces with the STM32 over I^2C that displays a cursor which can be controlled by the small button/joystick [8]. This is used to adjust controls, parameters, etc. without having to reprogram the drone's flight controller. This controller was designed in Fall 2025 and was adapted to fit the needs of the bird simulator.

Table 4. Controller Requirements, Verification, and Results

Component	Requirement	Verification	Result
Control override	Controller signals will take precedence over bird suit signals. If the controller is armed, the drone will ignore bird suit signals.	Bird suit will initially give signals. Once the controller is armed, the drone must behave according to exclusively controller inputs.	By using a transmitter identification system, drone will ignore all bird suit controls once an override command is received

2.6 Tolerance Analysis

The most significant risk to successful completion of the project is the latency of the IMU data processing, both for the bird suit and the autoleveling PID control loop.

The polling of IMU data for the bird suit must be done fast enough so that four IMU's can be polled and a radio packet is sent every 20 ms in order to maintain control of the drone that feels responsive. The latency of the bird suit is tested using the HAL timer function. The IMU packet reads in the bird suit are optimized to only read the axis relevant to the movement that the module is assigned to capture. We discovered that one IMU read takes 1 ms to complete, which is done 4 times per loop. A 10 ms delay is inserted in between packets to avoid overloading the receiver. In total, the period and frequency of the bird suit operation is:

$$T = 4 \cdot 1 + 10 = 14ms$$

$$f = \frac{1}{0.014} = 71.5\,Hz.$$

This operating frequency is above the 50 Hz needed for drone controls to feel responsive.

The flight controller's control loop also depends greatly on the loop frequency. Again, using the HAL timer function, we found that one IMU read takes 1-2 ms, which is slower than the bird suit because the flight controller reads all six axes of data. The flight controller also receives a radio packet every 10 ms, and the time spent on calculations and sending PWM is negligible. In total, the period and the frequency of the drone's flight controller operation is:

$$T = 2 + 10 = 12\,ms$$

$$f = \frac{1}{0.012} = 85\,Hz$$

3. Cost and Schedule

3.1 Cost Analysis

3.1.1 Bill of Materials

These part totals sum to $121.08. We bought at least triple for most of the required part quantities, to account for component loss. We also required some parts with express shipping as our final demonstration approached. This brought our total parts cost to $467. A full bill of materials can be found in Appendix A.

3.1.2 Labor Costs

Assuming a reasonable salary, similar to what a graduate from ECE at Illinois might typically make, the total labor cost can be estimated by multiplying the hourly cost by the number of hours worked. The average ECE graduate from the University of Illinois has a starting salary of approximately $95,000 [11]. Assuming each of our team members works an average of 10 hours per week across the 16-week semester, and multiplying by a factor of 3 for overhead, the total labor costs can be calculated as follows:

$$3 \times 10 \times \frac{95{,}000}{2{,}080} \times 16 \times 3 = \$65,769$$

3.1.3 Total Cost

Summing the cost of materials and labor cost for each team member, the total cost is:

$$\$65,769 + \$467 = \$66,236$$

3.2 Schedule

Table 5. Semester Schedule, broken by week and team member

Week	Anthony	Emily	Eli
1-3 (1/19-2/2)	Project idea formulation		
4-5 (2/9-2/16)	Details of overall project and individual subsystems finalized		
6 (2/23)	Bird suit and flight controller boards ordered	VTX board research	Motor controller research
7 (3/2)	Motor controller board ordered	VTX board designed and ordered	
8 (3/9)	Drone frame designed and prototype 3D printed	Initial libraries written for IMU and VTX	
9 (3/16)	Spring Break		
10 (3/23)	Flight controller board assembled and tested	IMU library troubleshooting	Motor controller tested
11 (3/30)	Bird suit code written, drone frame refined	Initial flight controller code written	
12 (4/6)	Final drone frame manufactured	IMU library finished and functionality tested	
13 (4/13)	Drone PID written to work with controller in test frame		
14 (4/20)	VTX board heavily modified, drone PID tested in open air with controller as well as in frame with bird suit		
15 (4/27)	Drone PID refined to work with bird suit in open air		
16 (5/4)	Final presentation, demonstration, awards ceremony		

4. Ethics, Safety, and Societal Impact

4.1 Legal Issues

This project posed three potential legal issues; transmission on the 2.4 GHz band, 5.8 GHz band, as well as operating a flying object in public, outdoor airspace. While the SX1280 is FCC Part 15 certified, our use

in a custom board without FCC testing and certification will require an amateur radio license [2]. Similarly, both of the 5.8 GHz chips we used for video are not FCC certified. To legally proceed with this project, Emily tested for and received a Technician Class Amateur Radio License (HAM License) under FCC part 97. This allowed for legal use of custom radio boards without FCC certification [2].

Recreational flight in public airspace requires an FAA remote pilot license, which Eli currently possesses. This allows for flight of any object below 250 grams, without any FAA registration [6]. Our final drone has a weight of approximately 220 grams. During any outdoor flight of the drone, Eli was in control of either the bird suit or the controller, allowing him to take control of the drone at any point, should the flight pose any hazard.

4.2 Ethical and Safety Issues

In accordance with the IEEE Code of Ethics, our primary focus is keeping our project safe, both for us, as well as people in the areas where our drone would be flown. The primary safety risk is a loss of control of the drone in flight. During testing of the drone, rotating propellers and potential high velocity movement pose a safety hazard. Operating on 2.4 GHz and 5.8 GHz without FCC certification poses an ethical risk of interfering with other communications. The use of LiPo batteries in multiple parts of our system also poses safety risks.

4.3 Mitigation Procedures

To uphold IEEE Code of Ethics I.1 [1], we implemented an 'override' system, where a traditional controller can take control over the bird suit's inputs. This allows for a certified remote pilot to take control of the drone, and allow it to land safely. We are mitigating this by earning a Technician Class Amateur Radio License (HAM License) under FCC Part 97 [2], ensuring we operate legally, within any power or frequency band limitations. To minimize physical risk, we used a drone simulator for initial testing of the bird suit. This allows for us to validate the signals generated by the bird suit in a virtual environment, preventing hardware damage or injury from the drone flying erratically. When initially testing the drone, we used a test frame to hold the drone in space, preventing it from flying erratically in open air before our control system was refined.

When testing drone functionality in open air, we maintained a safe distance from the drone and wear appropriate eye protection when necessary. All flights were tested in a controlled environment, whether that is indoors with no people in the immediate vicinity, or outdoors in open fields, far from people or public spaces. In the case of RX loss (no control signal received for more than 0.5 seconds), the motors were disabled, preventing further loss of control.

We also adhered to standard charging and storage protocols to prevent thermal runaway or fire hazards. This was done by storing the batteries in battery storage containers when not in use, charging them only in open air under active supervision, and regularly monitoring the voltage of the batteries to prevent over- and under-voltage.

4.4 Societal Impact

This project is broad in its applications and potential to impact society. In this iteration, the bird simulator is purely for entertainment purposes, giving FPV pilots a new, more immersive way to explore environments using a drone. Beyond entertainment, this project explores how leveraging natural body movement and reflexes for control can lead to more intuitive and responsive ways for humans to operate remote systems. This enhanced-human machine interaction can have benefits from a safety perspective, reducing risk to humans in hazardous situations. Applications such as search-and-rescue, disaster response, and inspection of dangerous environments could benefit from faster, reflex-driven control that mimics natural human instincts, potentially reducing risk of damage and injury when reaction times matter.

5. Conclusion

5.1 Accomplishments

Our project met nearly all of the requirements we set. The drone is able to fly in the air, under controls from either the traditional controller, or the bird suit. The bird suit is also able to fly in a simulator, which we also designed. The drone is able to have remarkably stable flight, both in a test frame, indoors, and outdoors in open air. We were able to abide by all relevant laws, including the FAA restrictions on flight, by flying under the control of a certified remote pilot, as well as the FCC requirements for radio transmission, by earning a HAM radio license.

5.2 Uncertainties

The one major flaw of our design was the inability to program the motor controller; if this had been successful, every component of our drone would have been custom. Fortunately, we designed with the possibility of off-the-shelf parts in mind by keeping our boards on a 20 mm grid. This allowed for simple substitution of the non-functional board so the rest of our design was still possible, highlighting the modularity of our overall design.

5.3 Future Work

Beyond redesigning the motor controller board to be custom, we can further refine this design by more precisely tuning the PID values on the flight controller. More tuning will enable safe, reliable indoor flight. Furthermore, a magnetometer can improve absolute angle measurements compared to current methods like integrating angular velocity or using the small angle approximation with linear acceleration, which both introduce considerable error. We could improve on the VTX board by using correct impedance matching designs, increasing signal strength and integrity. Finally, better packaging for the bird suit could transform this project into a commercial product for anyone from first responders in crisis situations to children at birthday parties.

5.4 Final Words

We are very proud of the outcome of our project—in the words of Professor Arne Fliflet, "Drones designed in this class have a habit of not getting off the ground"—we are elated to have proved him wrong, and honored to receive the award of best project this semester.

References

[1] IEEE, "IEEE Code of Ethics," *ieee.org*, Jun. 2020. https://www.ieee.org/about/corporate/governance/p7-8.html

[2] "Federal Communications Commission FCC 25-60." Accessed: Feb. 14, 2026. [Online]. Available: https://docs.fcc.gov/public/attachments/FCC-25-60A1.pdf

[3] "SX1280/SX1281," *Mouser.com*, 2026. https://www.mouser.com/datasheet/2/761/sx1280_81-1107808.pdf?srsltid=AfmBOooDbVxLfQr3AzSAIDq3zJD_xqX2DoAVFrCTwrAqbstYnwREkYcq (accessed Feb. 14, 2026).

[4] "RTC6715 CMOS 5.8GHz Band FM Receiver," 2007. https://opendevices.ru/wp-content/uploads/2015/02/7058r.pdf

[5] "RTC6705 CMOS 5.8GHz Band FM Transmitter Product Description," 2007. Accessed: Feb. 14, 2026. [Online]. Available: https://wildlab.org/wp-content/uploads/2015/07/RTC6705-DST-001.pdf

[6] "FAA-G-8082-22 Remote Pilot -Small Unmanned Aircraft Systems Study Guide," 2016. Available: https://www.faa.gov/sites/faa.gov/files/regulations_policies/handbooks_manuals/aviation/remote_pilot_study_guide.pdf

[7] "IIM-20670 Datasheet SmartIndustrial™ 6-axis MotionTracking® MEMS Device for Industrial Applications." Accessed: Feb. 14, 2026. [Online]. Available: https://invensense.tdk.com/wp-content/uploads/2023/07/DS-000183-IIM-20670-v1.1.pdf

[8] "DOGM204-A," *Mouser.com*, 2026. https://www.mouser.com/datasheet/3/711/1/dogm204e.pdf?srsltid=AfmBOooq4qkRDof1eHDmkp5DgJaeL9aE2SEPhDB7MTqIqsARJZv0SLzp (accessed Feb. 14, 2026).

[9] W. Storr, "Pi-pad Impedance Calculator and Interactive Tool," Basic Electronics Tutorials, Jun. 11, 2024. https://www.electronics-tutorials.ws/tools/pi-pad-impedance-calculator.html (accessed Apr. 20, 2026).

[10] fishpepper, "tinyOSD & tinyFINITY – a tiny opensource video tx with full graphic OSD – fishpepper.de," *Fishpepper.de*, Mar. 21, 2019. https://fishpepper.de/2019/03/11/tinyosd-tinyfinity-a-tiny-opensource-video-tx-with-full-graphic-osd/ (accessed Feb. 27, 2026).

[11] G. E. O. of M. and Communications, "Salary Averages," *ece.illinois.edu*. https://ece.illinois.edu/admissions/why-ece/salary-averages

[12] V. Kuznetsov and M. Brinson, "Qucs-S," ver. Qucs-S-26.1.0, April 2026, [Online]. Available: https://ra3xdh.github.io/.

Appendix A: Bill of Materials

Part Description	Mfr	Board	Price	Qty	Total
100pF Cap, 0603 SMD	Murata	Drone FC	$0.03	1	$0.03
4.7uF Cap, 0603 SMD	KEMET	Drone FC	$0.15	1	$0.15
100nF Cap, 0603 SMD	TDK	Drone FC	$0.02	4	$0.08
0.8pF Cap, 0603 SMD	Murata	Drone FC	$0.05	1	$0.05
470nF Cap, 0603 SMD	Samsung	Drone FC	$0.04	1	$0.04
1.2pF Cap, 0603 SMD	TDK	Drone FC	$0.07	2	$0.14
10nF Cap, 0603 SMD	KEMET	Drone FC	$0.02	3	$0.06
0.5pF Cap, 0603 SMD	Murata	Drone FC	$0.06	1	$0.06
1uF Cap, 0603 SMD	TDK	Drone FC	$0.10	6	$0.60
LED, 1206 SMD	Wurth	Drone FC	$0.25	2	$0.50
Conn 02x08, 1.00mm P	GCT	Drone FC	$4.59	1	$3.50
Conn 01x05 Socket, 2.54mm P	DuPont	Drone FC	$0.85	1	$0.85
Conn 01x03 Header, 2.54mm P	DuPont	Drone FC	$0.55	1	$0.55
CONMHF1-SMD-T	Hirose	Drone FC	$1.20	1	$1.20
Conn 01x04 Header, 2.54mm P	DuPont	Drone FC	$0.70	1	$0.70
Conn 01x08 Header, 1.00mm P	GCT	Drone FC	$1.22	1	$1.22
15uH Inductor, 0805 SMD	Coilcraft	Drone FC	$0.45	1	$0.45
3nH Inductor, 0603 SMD	Coilcraft	Drone FC	$0.30	1	$0.30
2.5nH Inductor, 0603 SMD	Coilcraft	Drone FC	$0.32	1	$0.32
52MHZ Crystal, NX2016SA	NDK	Drone FC	$2.80	1	$2.80
1k Resistor, 0603 SMD	Yageo	Drone FC	$0.01	5	$0.05
SW Push Button, THT	E-Switch	Drone FC	$0.40	1	$0.40
STM32F410CBTx MCU	STMicro	Drone FC	$4.50	1	$4.50
LT1129-3.3 LDO, SOT-223	Linear	Drone FC	$1.80	1	$1.80
SX1280 IMLTRT RF Chip	Semtech	Drone FC	$3.20	1	$3.20
IIM-20670 gyroscope	InvenSense	Drone FC	$5.50	1	$5.50
LT1129-5.0 LDO, SOT-223	Linear	Drone FC	$1.75	1	$1.75

100nF Cap, 0603 SMD	TDK	gyroscope	$0.02	2	$0.04
10nF Cap, 0603 SMD	KEMET	gyroscope	$0.02	1	$0.02
Conn 01x07, 2.00mm P JST	JST	gyroscope	$1.10	1	$1.10
IIM-20670 gyroscope	InvenSense	gyroscope	$5.50	1	$5.50
100pF Cap, 0603 SMD	Murata	Suit	$0.03	1	$0.03
4.7uF Cap, 0603 SMD	KEMET	Suit	$0.15	1	$0.15
100nF Cap, 0603 SMD	TDK	Suit	$0.02	4	$0.08
0.8pF Cap, 0603 SMD	Murata	Suit	$0.05	1	$0.05
470nF Cap, 0603 SMD	Samsung	Suit	$0.04	1	$0.04
1.2pF Cap, 0603 SMD	TDK	Suit	$0.07	2	$0.14
10nF Cap, 0603 SMD	KEMET	Suit	$0.02	3	$0.06
0.5pF Cap, 0603 SMD	Murata	Suit	$0.06	1	$0.06
1uF Cap, 0603 SMD	TDK	Suit	$0.10	4	$0.40
LED, 1206 SMD	Wurth	Suit	$0.25	3	$0.75
Conn 01x07, 2.00mm P JST	JST	Suit	$1.10	3	$3.30
Conn 01x02 XT30PW-F	AMASS	Suit	$1.50	1	$1.50
CONMHF1-SMD-T	Hirose	Suit	$1.20	1	$1.20
Conn 01x04 Header, 2.54mm P	Molex	Suit	$0.70	1	$0.70
Conn 01x05 Socket, 2.54mm P	TE Conn	Suit	$0.85	1	$0.85
15uH Inductor, 0805 SMD	Coilcraft	Suit	$0.45	1	$0.45
3nH Inductor, 0603 SMD	Coilcraft	Suit	$0.30	1	$0.30
2.5nH Inductor, 0603 SMD	Coilcraft	Suit	$0.32	1	$0.32
52MHZ Crystal, NX2016SA	NDK	Suit	$2.80	1	$2.80
1k Resistor, 0603 SMD	Yageo	Suit	$0.01	6	$0.06
SW Push Button, THT	E-Switch	Suit	$0.40	3	$1.20
STM32F410CBTx MCU	STMicro	Suit	$4.50	1	$4.50
LT1129-3.3 LDO, SOT-223	Linear	Suit	$1.80	1	$1.80
SX1280 IMLTRT RF Chip	Semtech	Suit	$3.20	1	$3.20
IIM-20670 gyroscope	InvenSense	Suit	$5.50	1	$5.50

4.7u Cap, 0603 SMD	TDK	VTX	$0.12	1	$0.12
10pF Cap, 0603 SMD	Murata	VTX	$0.04	3	$0.12
10nF Cap, 0603 SMD	KEMET	VTX	$0.02	1	$0.02
470nF Cap, 0603 SMD	Samsung	VTX	$0.04	1	$0.04
0.1uF Cap, 0603 SMD	Murata	VTX	$0.03	2	$0.06
3.3pF Cap, 0603 SMD	TDK	VTX	$0.06	1	$0.06
1uF Cap, 0603 SMD	KEMET	VTX	$0.10	4	$0.40
33pF Cap, 0603 SMD	Murata	VTX	$0.05	1	$0.05
4.7uF Cap, 0603 SMD	KEMET	VTX	$0.15	2	$0.30
1nF Cap, 0603 SMD	TDK	VTX	$0.03	1	$0.03
Conn 01x08 Header, 1.00mm P	GCT	VTX	$1.60	1	$1.60
Conn 01x03, 2.00mm P JST	JST	VTX	$0.90	1	$0.90
CONMHF1-SMD-T	Hirose	VTX	$1.20	1	$1.20
1.2nH Inductor, 0603 SMD	Coilcraft	VTX	$0.35	2	$0.70
1nH Inductor, 0603 SMD	Coilcraft	VTX	$0.38	1	$0.38
75 Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
47k Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
1.2k Resistor, 0603 SMD	Yageo	VTX	$0.01	2	$0.02
510k Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
270 Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
560k Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
51 Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
10 Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
100k Resistor, 0603 SMD	Yageo	VTX	$0.01	1	$0.01
RTC6705 VTX	Richwave	VTX	$6.50	1	$6.50
MLX90393SLW	Melexis	VTX	$7.50	1	$7.50
8.000MHZ XTAL	Abracon	VTX	$1.50	1	$1.50
MCT8329A	Texas Instruments	ESC	$4.00	4	$16.00
SiSHA14DN	Vishay Semiconductor	ESC	$1.01	24	$24.24

Appendix B: Symbols and Abbreviations

Unit or Term	Symbol or Abbreviation
Analog-to-digital converter	ADC
Brushless direct current	BLDC
Computer-aided design	CAD
Computer numerical control	CNC
Cyclic redundancy check	CRC
Decibels	dB
Degrees per Second	dps
Electrically Erasable Programmable Read-Only Memory	EEPROM
Electromotive Force	EMF
Federal Aviation Administration	FAA
Federal Communications Commision	FCC
First-person view	FPV
Feet	ft
Gravitational acceleration, 9.81m/s^2	g
Gigahertz	GHz
General Purpose Input/Output	GPIO
Hardware abstraction layer	HAL
Hertz	Hz
Inter-integrated circuit	I^2C
Institute of Electrical and Electronics Engineers	IEEE
Inertial measurement unit	IMU
Liquid-crystal display	LCD
Low-dropout (regulator)	LDO
Light emitting diode	LED
Lithium-ion polymer	LiPO
Long range	LoRa
Least significant bit	LSB
Meters	m
Megahertz	MHz
Master In/Slave Out (SPI)	MISO
Millimeter	mm
Metal oxide semiconductor field effect transistor	MOSFET
Master Out/Slave In (SPI)	MOSI
Printed circuit board	PCB
Proportional Integral Derivative	PID
Pulse width modulation	PWM
Radio frequency	RF
Revolutions per Minute	RPM
Received Signal Strength Indicator	RSSI
Receiver	RX
Serial Clock (SPI)	SCK
Serial Peripheral Interface	SPI
Transmitter	TX
Universal Asynchronous Receiver/Transmitter	UART
Universal Serial Bus	USB
Volts	V
Video transmitter	VTX
Microsecond	µs
Ohm	Ω