Helicopter Collective Control for Flight Simulators

David Green, Mark Green, Sven Hall,
Joseph Pergola
Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — The Helicopter Collective Control for Flight Simulators is a consumer-directed product for use in casual flight simulation software. Traditionally, most consumer controls are intended primarily for airplanes rather than helicopters, and therefore the availability of realistic and affordable helicopter specific controls is quite limited. To combat this, we have designed a collective control which is designed to fill this market niche. Our collective control can be powered via a battery or a wired connection, connects wirelessly to a computer to operate flight simulation software, and replicates the look and feel of existing helicopter collectives to provide an enjoyable simulation experience.**

***Index Terms* — Aerospace Simulation, Embedded Software, Bluetooth, Calibration, Mechanical Systems.**

I. INTRODUCTION

Consumer flight simulation products are widely available for purchase and there exists a significant market for these products in order to replicate the experience of flying for the casual user. However, the vast majority of products available today are oriented towards airplane simulation rather than helicopter simulation. The primary example of these products is the HOTAS, which stands for “Hands on Throttle-and-Stick”, a class of flight simulation products which consist of a joystick and throttle lever as well as an assortment of generic buttons and switches to control a variety of aircraft. While these controls are affordable and widely available, they are generic and therefore do not accurately represent the look and feel of the controls of any specific aircraft. Also, while controls such as HOTAS devices, joysticks, and pedal controls may translate well between the two kinds of aircraft, helicopters have a unique control known as a collective lever which does not have a direct counterpart in an airplane. The collective lever is responsible for adjusting the pitch angle of all the helicopter blades simultaneously which therefore alters the amount of lift produced by the blades. This is used to both ascend and descend as well as adjust acceleration when flying forward, back, or side to side. Also, a set of additional controls is sometimes present on a collective head at the top of the lever to manage other helicopter functions. Typically, integrated onto the collective lever itself is also a throttle control to adjust total engine revolutions per minute (RPM).

Our collective control intends to replicate the look, feel, and function of those found in real helicopters while also providing conveniences which make it easy to use for a casual user. The collective control is designed as a portable unit that can be powered via a battery as well as operate via a wireless connection so that it can be moved easily or integrated into a larger simulation environment. Also, our collective control has a modular head system which allows for different collective heads to be quickly swapped out in order to accurately replicate different real-life helicopters. Finally, our collective control aims to be affordable for the casual user while still providing high accuracy via the implementation of precision axis sensors to report the operation of the lever and throttle to the connected computer and flight simulation software.

II. SYSTEM OVERVIEW

From an electrical hardware perspective, the collective control is divided into three major subsystems: power, logic, and input/output subsystems. The power system is responsible for voltage regulation, battery charging, and power passthrough in order to facilitate both a wired and battery power source. The logic subsystem is responsible for processing input data, calibrating the axis sensors, establishing a wireless connection with a remote device, and forming and transmitting gamepad-style input data to the remote device. The input/output subsystem is responsible for collecting input data from the collective head, lever axis, and throttle axis as well as providing a means for the user to interact with and configure the device.

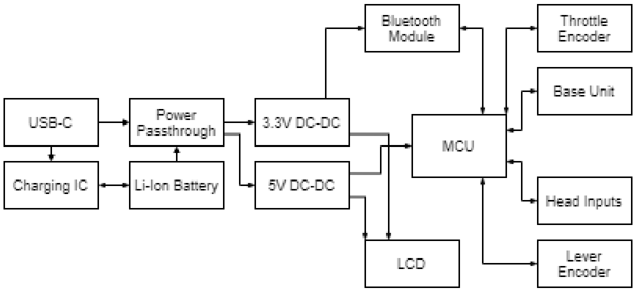

Fig. 1. Total electrical hardware system overview depicting the interconnection of major components.

III. POWER SYSTEM

A. Battery and Charging

A 3.7V lithium-ion 18650 rechargeable battery was chosen because of its modern features, such as long battery lifetime and fast charging capabilities. 18650 lithium-ion batteries are extremely common and allow for our device to be portable and the battery to be easily replaced by the user. Furthermore, 18650 batteries are available in a wide range of capacities up to 4000 mAh which satisfies one of our major project goals of a long runtime off battery power. 18650 batteries are low cost, low profile, rechargeable, and widely available, making them an ideal choice for this project.

The upkeep of our battery was equally important as its performance, therefore, the MCP73831 was chosen as the charging IC to safely manage the charging profile of the battery. This IC features an input voltage range from 3.75V to 6V, a programmable charging current range from 15mA to 500mA, and safety monitoring capabilities like power-down and thermal regulation. The IC is programmed via an external resistor to charge the battery at a rate of 500mA, to a safe voltage just below its maximum rating, at around 4.185V.

B. Voltage Regulation

The overall electronic system has two voltage requirements of 3.3V and 5V, each of which are unique to specific components. Therefore, the TPS63020 Buck-Boost DC-DC Converter was selected because it offers the ability to take a range of input voltages and step it up or down to our system's needs. By selecting the TPS32020 we were able to reduce circuit design time and cost by reusing the same regulator for both the 3.3V and 5V line. The converter can handle input and output voltage ranges from 1.8V to 5.5V, and 1.2V to 5.5V, respectively, and up to 4A of output current, which is more than enough for our device's parameters. The output voltage is determined using an external resistor divider which provides feedback that indicates the desired output voltage to the regulator. Also, due to the TPS63020's nature as a DC-DC converter, it boasts extremely high efficiency of up to 90%. This is achieved through the implementation of a power-save mode which dynamically lowers the switching frequency of the regulator during low load current conditions in order to maintain consistently high efficiency.

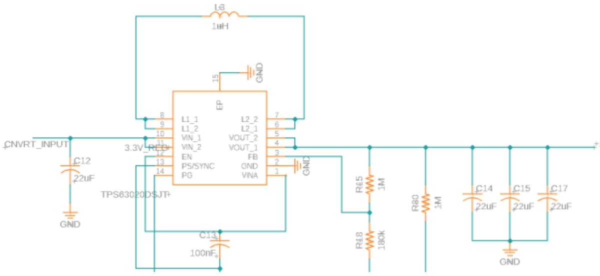

Fig. 2. Circuit diagram of TPS63020 DC-DC converter implementation.

C. USB-C Power

Our device is equipped to accept a wired power source via a USB-C connection. USB-C was chosen for its advanced technology providing reversible connectivity and support for USB Power Delivery to supply ample current to our device. Due to the USB-C's capabilities, we were able to program it to provide our system with up to 3A so that enough current was allocated to charge the battery and power the entire system simultaneously. Another important factor in selecting USB-C as our wired power method was its ubiquity; most modern devices now utilize USB-C and therefore it is commonly available and utilized by consumers.

D. Power Passthrough

Efforts were made towards improving battery management and user-experience, which led us to equip our system with power-passthrough capability. As our device is designed to be operated via either a wired power source or through battery power, the system needs to be able to recognize which power source is currently in use. This is due to the fact that the wired power source must simultaneously power the system while also charging the battery, whereas during battery operation the battery must solely provide power to the device. Power passthrough is achieved through the use of a P-Channel MOSFET and a Schottky diode. The P-Channel MOSFET detects the presence of a wired power source in order to cut off the battery from powering the device while the Schottky diode prevents current flow back into the USB power source.

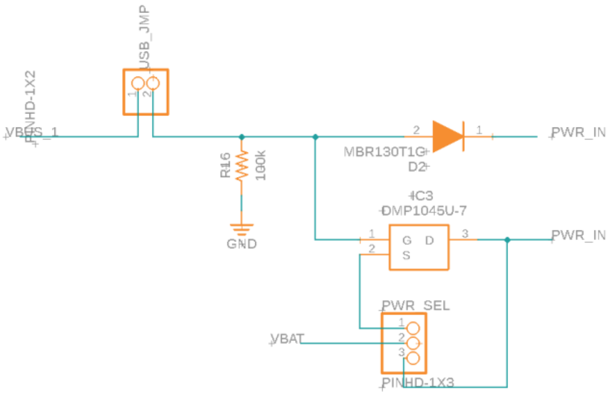

Fig. 3. Power passthrough circuit diagram, illustrating P-Channel MOSFET and Schottky diode used to switch between wired and battery power.

IV. LOGIC SYSTEM

The two primary components of the logic system are the microcontroller and Bluetooth module. The microcontroller collects and interprets input data from the collective control and forms commands which are sent to the Bluetooth module. The Bluetooth module then connects wirelessly to a host device and appears as a Human Interface Device (HID) which allows the remote device to interpret the Bluetooth data as peripheral inputs. These inputs are then bound to actions within the flight simulation software so that physical inputs on the collective unit are reflected in the virtual environment.

A. Microcontroller

We elected to utilize the ATmega2560 microcontroller in our project for a number of reasons. The ATmega2560 has a total of 86 general purpose I/O lines which is crucial to our project due to the input-oriented nature of a helicopter collective. Also, the ATmega2560 operates at 16 MHz which provides an ample processing speed to interpret and report data to the remote device with seemingly no delay. The presence of 4KB of EEPROM also means that the collective has the ability to persistently store calibration data through power cycles which reduces the burden of the user to calibrate the device with every use. The ATmega2560 is also available on the Arduino Mega development board which means that software design and testing was able to occur much earlier in the design process without the need for a completed PCB. The ATmega2560 is programmed directly on our PCB via an exposed ISP header which connects to an external AVR programmer.

B. Bluetooth Module

The BM70 Bluetooth module was selected in order to facilitate wireless communication between the collective and the remote device running the flight simulation software. The BM70 is a self-contained, pre-shielded module which can be integrated directly into our PCB design. Also, the BM70 supports Bluetooth Low Energy and is Bluetooth 5 qualified which means that we are using modern Bluetooth standards and technologies which are less power-demanding than classic Bluetooth. The BM70 has a range of up to 50m which is more than satisfactory for our purposes and provides access to the full Bluetooth Low Energy software stack, which is necessary to implement HID over GATT and appear as a gamepad to the remote device. The BM70 communicates with the ATmega2560 over UART and is configured directly on the PCB via header pins which connect to an external USB to UART converter and a computer.

V. INPUT/OUTPUT SYSTEM

A. Collective Head

In order to accurately replicate a variety of helicopters, a modular collective head system was designed which allows for different heads to be quickly swapped in and out. This means that rather than requiring an entirely different collective unit to simulate a different helicopter, only the small collective head piece need be switched. This is achieved via a 17-pin female Molex connector which is present in the end of the collective lever itself, which allows a corresponding male connector to be embedded in each collective head. Each collective head can therefore vary in physical shape, size, and number of inputs but still connect to the same lever. Within this connector is also a signal line which is interpreted by the microcontroller to determine which head is attached. This is achieved via the microcontroller utilizing its analog-to-digital converter to read the voltage level present on the signal line and therefore determine which head is present. Inside of each head, a resistor of varying value simply needs to be soldered in place to provide a different voltage for each head. Once the microcontroller determines which head is present, it can then interpret the inputs from the 17-pin connector correctly and report them to the flight simulation software. The input from the 17 pin Molex connector embedded in the top of the lever is then connected to the PCB via an 18 pin Molex Connector in the base unit.

B. Axis Sensors

The two most essential inputs present on the unit are the throttle axis sensor and lever axis sensor. These need to measure the angular position of both the collective lever and twist throttle and report these back to the microcontroller. As a collective control is at its most basic a lever and a throttle, the accuracy, durability, and feel of these sensors are essential in creating a satisfactory final product. Thus, for this purpose, we opted to use EMS22A50-B28-LS6 absolute encoders. The fact that these are absolute encoders is crucial; a traditional rotary encoder simply indicates which direction it is spinning and how many positions it has spun since startup, without any knowledge of its current angular position. However, for our purposes, the sensor must be able to consistently report its angular position regardless of power cycling. The EMS22A50 achieves this by measuring the magnetic field within the device to determine its position, which provides for smooth operation and increases the life of the encoder. Furthermore, the EMS22A50 has a resolution of 1024 pulses per revolution and has a nominal accuracy of 0.7 degrees or better, which provides the necessary accuracy for our project's requirements. The EMS22A50 encoders connect to our PCB via two 6-pin Molex connectors.

C. Base Unit Controls

While the collective head inputs and the axis sensors are the inputs which are transmitted to the flight simulator, some controls are necessary in order to configure the collective unit itself. For this purpose, a power switch, 3 buttons, and an LCD1602 module were designed to be integrated into the base of the device. The 3 buttons are used to select different functions such as calibrating the absolute encoders, viewing the battery voltage, or pairing with a remote device. The information of what function is currently being performed is presented on the LCD1602 for the user to read. The LCD1602 module was chosen due to its low cost, ease of use, and the small amount of information required to be presented on the device at any given time. The base unit controls connect to the PCB via a 24 pin Molex connector. Also, directly mounted on the PCB but exposed out of the back of the unit is a potentiometer to adjust the contrast of the LCD display.

VI. PRINTED CIRCUIT BOARD

We decided early on in the design process that we wanted as many electronics as possible to be contained on the PCB itself rather than utilizing breakout boards or off-the-shelf development boards. We designed our PCB in Fusion 360 Electronics and decided to order our PCBs through JLCPCB. JLCPCB was selected due to their extremely quick turnaround time as well as their ability to assemble all of the surface mount components on the board. However, as JLCPCB only assembles components available in their in-house inventory, this also presented a unique challenge of designing the collective based on what parts were made available by JLCPCB. Furthermore, our PCB design was limited in size due to the physical constraints of the base unit, which further complicated the PCB design process. All through hole components were hand soldered by our team members after receiving the boards.

Fig. 4. 3D Rendering of completed PCB Design.

VII. SOFTWARE DESIGN

The software design for the collective control unit involves creating software on the ATmega2560 microcontroller that seamlessly integrates all of the hardware components involved into a functioning and cohesive final product. The software is thus broken down into three high-level logical sections: user interface, sensor input processing, and Bluetooth communication. Each of these sections of have unique design considerations which will be henceforth explored.

A. User Interface

In order to maintain simplicity of use for the end user while still allowing for complex functionality, a menu system was designed for the user interface which the end user can interact with via buttons on the collective base. This menu system is implemented as a state machine. This allows for the user to switch between various states of functionality, for example between Bluetooth setup and a state in which the user can calibrate the encoders for the collective lever and throttle. Information regarding the current state is presented to the user via the LCD present on the collective base. The primary benefit of this form of user interface, besides simplicity, is the strict compartmentalization from a software perspective. This division into various menu states allows for each form of functionality to be designed and programmed nearly completely separately from one another and take control over any hardware component as necessary whilst performing its function.

B. Input Processing

The most crucial software consideration for the helicopter collective control is the input processing logic used to take in and process data from the various forms of sensor input. For the purposes of this portion of software, sensor input refers to the two major forms of sensor input used in the collective control: angular input from the encoders on the collective lever/throttle and button-based input from the collective head. Each of these two major forms of sensor input have unique software requirements in order to properly take in and process them as data that can be transmitted as gamepad input to a host device.

C. Encoder Input

The EMS22A50 encoders used in the collective lever and throttle to detect angular position require special software design in order to function. One of the considerations regarding the encoders is the process for taking in encoder input. This is because, as opposed to something like a button that sets a pin high or low, the encoders do not passively provide an input that can be read by software. Instead, a specific technique is required to both receive and process data from the encoders. This technique involves manual manipulation of pins on the encoders utilizing specific timing provided by the manufacturer in order to receive data from the encoder. The process begins by signaling an encoder to begin the process of sending data, and then managing a clock signal by toggling a pin on the encoder such that data from the encoder is sent back, bit-by-bit, to the microcontroller for processing. These bits of data are then manipulated via binary operations to construct an end value in binary representing an integer in the range of 0-1023 that corresponds to a specific angular position of the encoder.

The other form of software design regarding the encoders is the algorithm required to bridge the gap between the physical limitations of encoder movement and the full range of axis-based input that is sent to the simulator. This gap exists because the absolute encoders report 1024 values across their full range of motion, which directly corresponds to the 1024 values of axis-based input sent to the simulator, but the physical movement of the encoders via the collective lever or throttle would result in only a small portion of these values ever being seen by the host device. This is handled through a calibration process that is implemented in software. This calibration process allows the user to set a lower and upper bound for each of the encoders using their physical controls. These bounds are then put through an algorithm which determines the proportion of this limited range to the overall range of 1024 values that the encoder can produce. After this proportion has been calculated, an array containing 1024 values has its contents inside of the limited range modified via an algorithm that expands values in the limited range to represent values in the full range. For example, if the difference between the upper and lower bound of an encoders physical movement encompasses only half of the actual range of the encoder, any movement from one value to the next that is reported by the encoder will be treated as though it was a change of twice the magnitude. This array is then utilized essentially as a lookup table in which the encoder value read in corresponds to an index in the array containing the modified value that should be reported to the host device. This allows for quick translation between raw encoder input in a small range of values to modified input in the full range to be sent to as gamepad input to a host device.

D. Collective Head Input

The collective heads created for use as part of the helicopter collective control unit consist of a few types of physical inputs, however all types of inputs present on the collective head must be treated as purely button-based input for the purpose of emulating gamepad input. In order to achieve this, some amount of software design is required. The first, and most important, is the logic to treat toggle switches as buttons, logic that is necessary for two reasons: lack of a signal for certain states of the switch as well as issues when binding physical controls to controls in the simulator. The first of these issues stems from the "off" state of toggle switches not producing a signal that can be read by the microcontroller directly. This issue, in isolation, could simply fixed in software by a conditional statement which checks if any of the other states of the switch are on, and if they are not, consider the switch to be in the "off" state. However, this alone is not sufficient, as it would result in certain forms of input constantly outputting some signal to be sent to the simulator. This would mean that when trying to bind a switch on the collective head to a control in a simulation environment, the simulator would not be able to determine which control is currently active. In order to remedy this, software is implemented such that the current and previous state of controls that may have this issue (such as toggle switches) are recorded upon change of the state of the control. This allows for a check to make sure that input is only sent to the simulator if the switches have actually changed state, effectively turning a constantly active switch into a button-like input.

E. Bluetooth Communication

Communication between the BM70 and ATmega2560 microcontroller requires a very specific software implementation. As opposed to some Bluetooth modules which are communicated with simply using ASCII or other well-known standards, the BM70 platform utilizes a very specific command format. Some commands are static and are simply defined as byte arrays in the software to be sent. Dynamic gamepad input, however, must be generated according to a specific format in order to send them to the BM70 for wireless transmission. From a software perspective, this encompasses two functions: one to concatenate values read in from the collective lever, throttle, and head into command parameters and one to calculate a checksum that is required for the BM70 to accept and acknowledge the command being sent.

On a somewhat higher level, the Bluetooth process of pairing a device must be handled very specifically in order to ensure proper connection before sending of data. In order to achieve this, Bluetooth pairing functionality is implemented as a state in the overall menu-based state machine. Whilst in this state, the software is designed to send a pairing command to the BM70, wait for a response, parse the response, and then either begin sending data or prompt the user for some action if pairing fails. This implementation is essentially a loop which waits for some response from the BM70 in order to maintain expected states in software.

VIII. MECHANICAL DESIGN

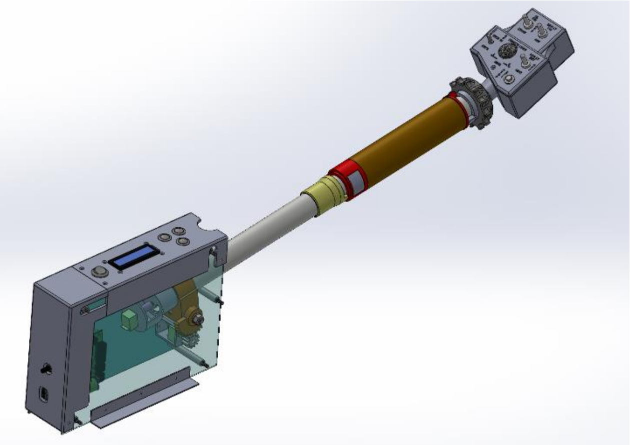

Fig. 5. Complete mechanical design of collective control, shown with UH-1 Huey head attached.

When initially considering the mechanical design elements and constraints that would be required for this project the main factors examined were our goals of producing a highly-realistic appearance, an end product with high mechanical accuracy that maintains acceptable tolerances, and a final design that could be predictably manufactured. In an effort to assist in covering all three of these goals, a decision to use 3D printing for all custom components was made. This decision, coupled with reference images and measurements, allowed us to accurately create nearly identical outside geometry when compared to the real-world collectives we based our designs on. The use of CAD programs allowed for both the basic design of each component as well as the ability to see the entire end product, which aided greatly in the creation of our physical assembly steps and procedures. The mechanical design has been broken up into the four main functional aspects of the end product: the collective head assembly, the twist throttle assembly, the collective lever assembly, and the base of the unit.

A. Head Assembly

The collective head assembly is fundamentally broken down into the categories of common and unique design components based on the collective head being emulated. The unique components are different between the Bell 206 and UH-1 "Huey" as they must approximate the physical form of their respective head as well as allow for the implementation of the appropriate number of human inputs. These unique components are the shell of the head, the face plate, the set of human interface components, and the outside geometry of the bottom captive nut.

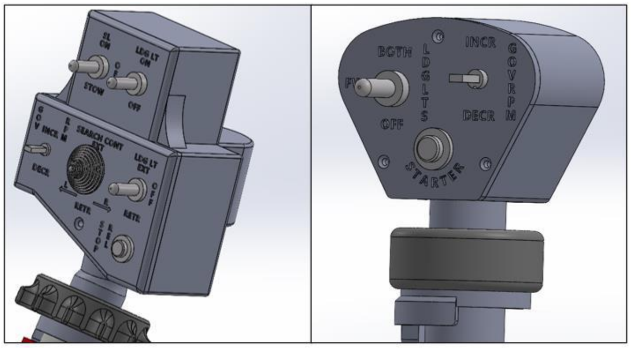

Fig. 6. Two swappable heads which can be attached to the collective lever. Left: UH-1 Huey Right: Bell 206.

The common components shared by both heads includes the bottom neck geometry and the inside geometry of the captive nut. These two shared aspects of the collective heads facilitate the ability to swap heads on the collective lever arm. While the common physical connection between the lever and head is facilitated by the neck, lever end cap, and captive nut, the common electrical connection to the head controls is facilitated by a common 17-pin circular Molex connector.

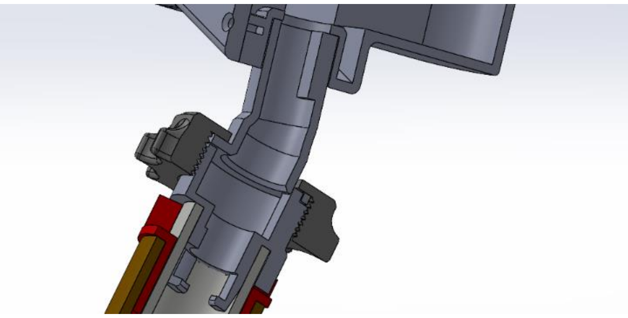

Fig. 7. Quick-detach mechanism cutaway illustrating screw-on captive nut and threaded end cap.

B. Throttle Assembly

The main objective of the throttle assembly is to translate the rotational motion of the physical twist throttle to a rotation on the throttle encoder. The collective twist throttle assembly consists of the external throttle control, throttle limit bearings, throttle limit collar, a mounted plate that revolves centrally inside the lever arm, a brass connecting rod, an encoder coupler, and the encoder itself. The external control is connected to the lever arm via two bearings, which are both captive along the lever via a bottom collar and the lever end cap, thus limiting the throttle's movement about the length of the lever arm. The lever end cap and bottom collar also physically interface with the two bearings inserted inside the throttle in order to limit the rotational movement of the throttle to 140 degrees. The bottom of the brass connecting rod is held centrally inside the lever arm via a bottom end cap that holds both the connecting rod, encoder coupler, and encoder. The top of the connecting rod is connected to a plate extended from the outside throttle via a coupler and hex pairing that mates with a hexagonal shaped hole inside the plate arm.

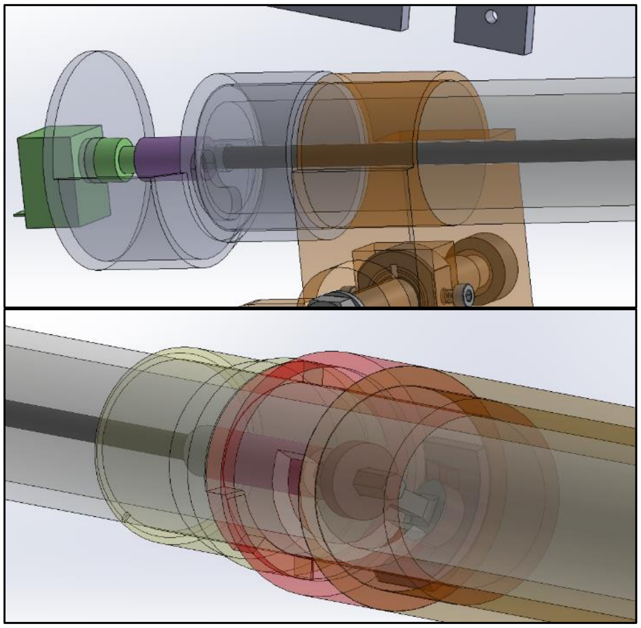

Fig.8. Overview of throttle subassembly and connecting rod. Above: Throttle encoder mount and coupler Below: Throttle shaft plate and hex coupler.

C. Collective Lever Assembly

The collective control assembly is primarily centered around the ability for the lever arm's movement around a central axis to be translated to a rotation of the collective encoder. The additional feature of being able to adjust the amount of resistance felt by the user was also implemented.

The physical interface between the collective lever arm and the collective encoder occurs at the base of the lever arm via a custom designed collar, bearing, and partial gear combination. This custom piece statically mounts to the collective lever arm and allows it to rotate around a central bolt. The gear segment facilitates the translation of the rotation on the lever arm to the encoder, which has a mating gear permanently affixed to it. The gear segment and the driven gear on the encoder maintain a 3:1 gear ratio, which was chosen as a method for increasing the number of usable steps on the driven encoder, thus increasing the overall accuracy of the collective lever control.

In order to facilitate the adjustment of resistance felt by the user, a shaft collar with a variable friction screw was used. This shaft collar is placed inside the collar and bearing combination at the base of the lever arm and tightens around the centrally placed shoulder bolt. In this configuration, the user is presented with a single hex bolt from the front of the unit that allows them to granularly set the felt resistance when moving the lever arm.

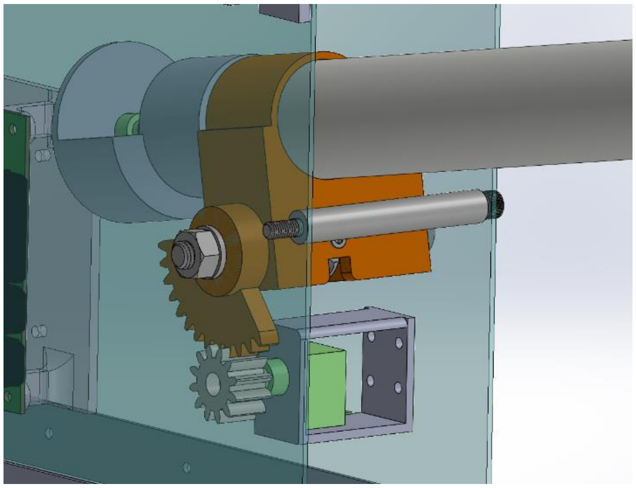

Fig. 9. Overview of collective lever subassembly illustrating 3 to 1 encoder gear ratio and friction adjustment mechanism.

D. Base Unit Assembly

The base of the total collective assembly acts primarily to store necessary hardware and offer mounting options to the user. The sides of the base unit are constructed from aluminum plates in order to supply a solid mounting surface for the bolts that are connecting the two plates together and the central axis that the collective control rotates about. The base of the plates includes brackets that enable the end-user to mount the final product as they see fit with pre-drilled mounting holes. The base unit houses the PCB and all required wiring to connect the encoders and base unit controls, including the LCD, power switch, and buttons that exist on the top of the base unit itself. Finally, the back plate of the base unit exposes a dial to allow the user to change the LCD contrast as well as the USB-C interface from the PCB for charging.

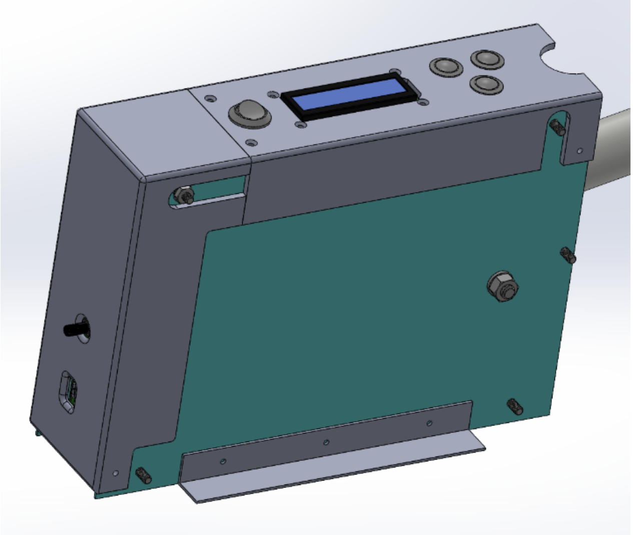

Fig. 10. Overview of base unit subassembly illustrating user control top panel, rear contrast control, and bottom mounting feet.

IX. THE ENGINEERS

Sven Hall is a 22-year-old Senior currently majoring in Computer Engineering at the University of Central Florida. He has prior industry experience in mechanical design and software development. He previously interned at L3Harris and has accepted a Software Engineering position at L3Harris immediately following graduation. Sven's primary responsibility was mechanical design and secondary responsibility was software design.

David Green is a 22-year-old Senior currently majoring in Computer Engineering at the University of Central Florida. His post-graduation plans involve looking for a job in the software engineering and development industry. His interests include .NET and desktop application development. He is expected to graduate Cum Laude in Summer 2021. David's responsibilities in this project include Bluetooth configuration, PCB Design, and mechanical design.

Mark Green is a 22-year-old Senior currently majoring Computer Engineering at the University of Central Florida. His post-graduation plans are undecided, but he is currently exploring options in the software development and defense industries. Mark's primary focus was on embedded software for this project.

Joseph Pergola is a 23-year-old Senior currently majoring in Electrical Engineering with a focus in Power Systems at the University of Central Florida. He has accepted a position at Cuhaci and Peterson doing electrical design. Joseph's focus was on the Power System and PCB Design of this project.

REFERENCES

[1] Microchip, "ATmega640/V-1280/V-1281/V-2560/V-2561/V Datasheet," May 2020. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf.

[2] Microchip Technology, "BM70/71 Data Sheet," 25 October 2017. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/BM70-71-Bluetooth-Low-Energy-BLE-Module-Data-Sheet-DS60001372H.pdf.

[3] Bourns, Inc., "EMS22A - Non-Contacting Absolute Encoder," 17 May 2018. [Online]. Available: https://www.bourns.com/docs/product-datasheets/EMS22A.pdf.

[4] Federal Aviation Administration, Helicopter Flying Handbook, Oklahoma City: United States Department of Transportation, Federal Aviation Administration, Airman Testing Branch, 2019.

[5] Microchip Technology, "MCP73831/2 Miniature Single-Cell, Fully Integrated Li-Ion, Li-Polymer Charge Management Controllers Datasheet," 28 Feburary 2020. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP73831-Family-Data-Sheet-DS20001984H.pdf.

[6] J. Smoot, "Rotary Encoder Options: Absolute or Incremental?," 20 November 2018. [Online]. Available: https://www.digikey.com/en/articles/rotary-encoder-options-absolute-or-incremental.