D.E.A.R. Drone: Autonomous Delivery Drone

Daniel C. Biller, Ellie A. Lane, Austin T. Perkins, and Raymond A. Chenoweth

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

Abstract — **The D.E.A.R. Drone is an autonomous delivery drone that has been created with the intent to understand the scalability of delivery drones. In the past few years, through the progression of machine learning and processing powers continually increasing, it has allowed for the idea of delivery drones to become a reality. This project utilizes machine learning/computer vision to navigate autonomous flight in order to drop a package of up to 0.5 pounds.**

Index Terms — **Propellers, payloads, Microcontrollers, MOSFET circuits, Power MOSFET, Printed circuits, Computer Vision, Sensors**

I. Introduction

Our project was conceived to make a scalable proof of concept automated delivery drone. This can and has helped solve supply chain issues across the globe and helped businesses get product to their consumers quickly and safely. Our D.E.A.R delivery drone is a quad copter, with 6000mAh battery and will be able to carry a half a pound payload to its destinations safely. This idea was conceived when we looked upon the world during Covid's peak and understood delivery to customers remotely was to be the future of fast and most importantly, safe, delivery. We took inspiration from Amazon's drone program, which, while not large, is still a working drone delivery service. We planned on beginning with the simplicity of drone flight, working towards autonomous flight, and then incorporating other features that would lead towards a more robust and workable drone delivery platform at the end of the project.

II. Motivation & Goals

D.E.A.R Drone is a style of drone the world is steadily heading to in terms of convenience and innovation. Amazon has looked into such things, medical companies such as MatterNet are already using such drones to save lives and delivery blood and other such materials to hospitals when needed and may others will eventually come into the market [1]. The quadcopter drone is just a tool that is steadily reaching its technological maturity and as such is ready to be used widely in the market. Our motivation stem from this observation and wish to mitigate problems that will arise from the incorporation of such tech into the wider world. D.E.A.R Drone is the not necessarily a drone, as much as it's a drone with the ability to navigate and react to things that our group believe will hinder its progress in the future, such as the fact that cities (mostly downtown areas) are the most likely areas that will routinely use drones for deliveries. And as such, buildings, birds, pedestrians, and several other things will sometimes cause accidents that must be avoided if possible. Our sensor suite and flight controller's configuration, with data processing center added to it, provide but we think is a good answer to these problems.

The end goal of this project is to make a sensor suite and program format that can let drones take to the sky in cities or other crowded airspaces and react in real time without human interference to objects in their way while making deliveries [2]. Such as birds like pigeons that are known to be everywhere in big cities, or just buildings in general. We know GPS guidance for drones is already realized in automatic flight, however new buildings, cranes, and other big objects can be erected in short order that is sometimes not seen by GPS. Such instances need to be accounted for and that is the end goal of our D.E.A.R Drone project.

III. System Components

Each teammate in this project provided a different skillset and outlook to the design to the completed drone. So, in that respect we further divide this section into not just its components, but let each teammate contribute to each component of the drone and what they applied to the success of the project.

A. Frame

The frame for our delivery drone was chosen to be the plastic YoungRC F450 4 axis 450mm quad copter farms as shown in Figure 1. The reason for this frame is due to the fact that it's plastic rather than carbon fiber, because carbon fiber absorbs RF signals and that might have disrupted a few of our sensors. As well as 450mm frame would allow us to have more room for our sensors and to have larger propellers for more lift for our 1/2pound package. Additionally plastic when it breaks its snaps apart, while carbon fiber splinters, so in the need to repair we believe that plastic would be the best alternative.

This gram design also allowed us to have landing gear that would be 4 separate legs rather than "T-shape" stands

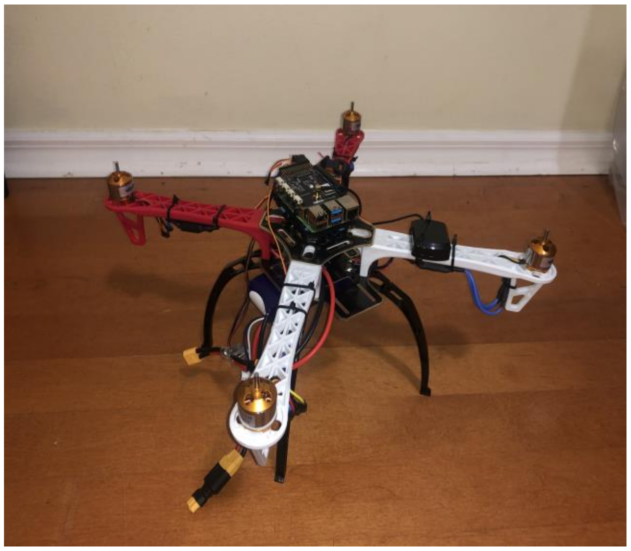
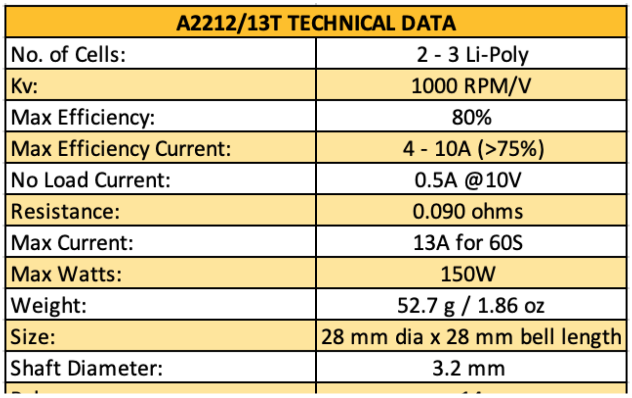


Figure 1: Current Drone Frame

that would allow our drone to be more stable on uneven ground and also allowing room for our battery and package.

B. Motors

The motors we used were a bundle package with the drone frame itself to save on cost as we did have a constrained budget. However, we made sure that the motors specifications were up to par with what the drone needed to do before we bought the package. This meant that the frame size and the interior motor size had to be 2212 or higher, which is a term used to describe how much lift can be applied to the frame from the motor. We used the QWinOut 2212 motor for our drone with specifications of that motor shown in Figure 2, and it was sufficient in its needs for our purposes [3]. The only issue with the motors were their durability when under duress. As our drone is in the



A2212/13T TECHNICAL DATA	
No. of Cells:	2 - 3 Li-Poly
Kv:	1000 RPM/V
Max Efficiency:	80%
Max Efficiency Current:	4 - 10A (>75%)
No Load Current:	0.5A @10V
Resistance:	0.090 ohms
Max Current:	13A for 60S
Max Watts:	150W
Weight:	52.7 g / 1.86 oz
Size:	28 mm dia x 28 mm bell length
Shaft Diameter:	3.2 mm

Figure 2: Motor Specs

prototype phase, crashes and problems of this nature are bound to happen, and they did. We found that after only one crash the inside of the motor's components had parts that were dragging on the inside reducing lift ability, thus making it harder for stability to be achieved when the flight controller took over. This led to the next crash that ended the drones flight time. We think if we were to continue the project further, we would have better motors that could handle further abuse, most likely by being made out of steel or also keeping the components on the inside completely inaccessible from dirt and mud.

The motors for our delivery drone are the A2212/13T 1000KV brushless motors. We bought theses in a pack with our ESCs and frames to save money. As you can see in Figure 3s graph, The A2212/13T is capable of producing 7,380 RPM's at 7 Volts and .6 Amps, 8,460 RPMS at 8 Volts and .65 Amps, and 10,500 RPM's at 10 Volts and .75 Amps. With a every increase in 1 Volt equaling about 1000 RPM's. In the graph below you can see the efficiency over Amps at 7.2 volt in blue. and 11.1 volts in yellow. [3]

C. Battery

Our battery is a typical Lithium Polymer style that is seen in multiple drone types across the industry. It is used for its robust design, capable of swelling (being dropped and bumped and slightly degraded) and still operating at capacity and being recharged in multiple cycles. These aspects give the LiPo battery the edge with drones, who always have the potential to crash. Our battery is the Tunrigy 6000 mAh, chosen for its capacity and C rating (the

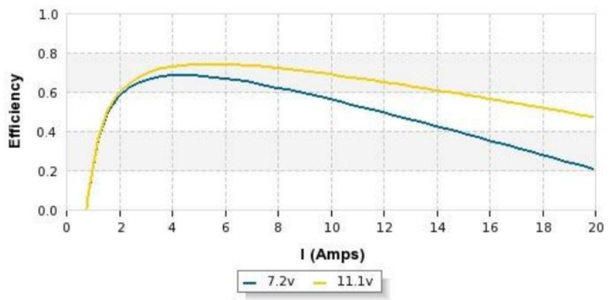


Figure 3: Efficiency over Amps for our Motor

rate with which it discharges and can be recharged). With this battery we predict a battery life for our drone during full operation to be 15-20 mins, depending on speed, wind, size of package and several other factors [4]. This battery is placed on the center of the bottom frame component of the drone so as to provide a stable center point of weight for the drone's flight controller to easily adjust too.

D. ESC

ESC stands for Electronic Speed Controller and from the name comes the goal of the device. The device's goal is to control the speed of the drone, and this is being given by telling the motor to rotate and essentially acts as a translator from the commands being given through the pilot and what voltage/current needs to go through the motor to accomplish it. The ESC adds a lot of control to the motor starting with how the wires are connected, which determine what direction the motors turn in. We see this example in Figure 4. Add-on components are optional when choosing an ESC. This includes selecting ESC with BEC and a 4 in 1 option versus an individual. A BEC is a battery elimination circuit that provides more control to the power given to other components. For drone projects, this is not a necessary option as the Power Distribution Board (PDB) takes care of this operation [5]. Another optional component is purchasing individual ESC vs 4 in 1. The 4 in 1 is a board that has 4 ESCs already put together and ready to use, whereas individual ones need to be put together. With drone enthusiasts, it might be more cost-effective to purchase the 4 in 1 option, but for beginners, if one ESC is broken then the whole 4 in 1 board needs to be replaced. Hence, why for this project, we wanted to take the individual route, but through researching our desired frame and motor design, we found a kit that includes the ESCs needed to operate. A picture of the ESC is below in Figure 4 and has a working current of 30A for the motor. While purchasing this separately would be more expensive, finding it with a kit helped with those costs.

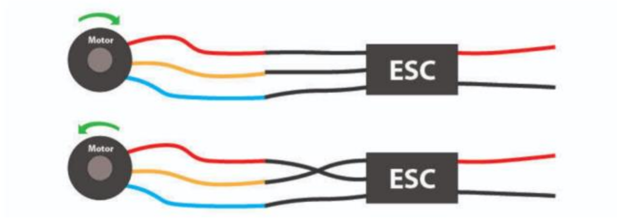


Figure 4: Diagram of ESC Wiring

E. Sensors

Sensors are a key component to collecting data points being processed by everything within the drone. Initially we started looking at sensors that would help with detecting objects and distance. This led us to look at 4 different type, Ultrasonic, IR, LiDar and Time of Flight sensors. When comparing the sensors in Table 21 and trying to decide what will be need for our project, Ultrasonic was the best for the job. Most IR would be influenced by ambient light and with LiDar and Time of Flight being a little expensive, then ultrasonic was best for the job. Ultrasonic can detect objects and be very low cost to the project. Ideally, if there was a larger budget then LiDar is the next buy to make it more accurate for the job. Because of choosing the Navio and the Raspberry pi then it included the Wi-Fi, GPS and altitude sensors and eliminated having to search for other options. We will be using the Wi-Fi for security and communicating with the drone, GPS will be used for navigation and location tracking, and altitude sensors will detect how low to an object or ground the drone is at.

F. Flight Control Board

The purpose of the flight controller is to ensure the safe and level flight of the aircraft. It does this by intaking data from various sensors and using this information to calculate the drone's position and motion. The flight control board uses this estimation to correct alter the individual motor speed to correct for perturbations in heading and position.

We chose to use a Navio2 flight control board paired with a Raspberry Pi as our flight control board. This combination works perfectly for our application because it allows us to run a full Linux operating system on the board, as well as our own software. Having our flight control board double as our data processing board simplifies software design and reduces power consumption, making the drone more power-efficient [6].

The Navio2 has inputs for multiple sensors and a microprocessor to handle streaming the sensor data to the Raspberry Pi. The sensors on the Navio2 include an accelerometer, gyroscope, barometer, compass, and GPS. It also has multiple I2C, SPI, and UART ports for connecting additional sensors.

The Raspberry Pi 4 B runs a modified image of Raspbian which configures the GPIO pins to accept input from the Navio2. On top of that, it runs our flight control software, PX4. Additionally, on the main OS, we are running our computer vision software to generate the 3D point cloud used for obstacle avoidance.

G. Drop Mechanism

A key feature of our aircraft is being able to drop our package at its destination. Having the drone carry the package once it is attached is not an issue. However, it was a challenge to ensure that our drone is able to be equipped with a variety of packages sizes and drop them with minimal modifications to the packaging. Rather than a claw mechanism or a hook, we felt the best option for this was a sliding pin release mechanism, as seen in Figure 5. Any shape of packaging can be secured to the drone using a harness that attaches to the drone and fits through the sliding pin. Additionally, in the event of a power failure or other error, the mechanism stays closed rather than releasing the package, which is another safety feature.

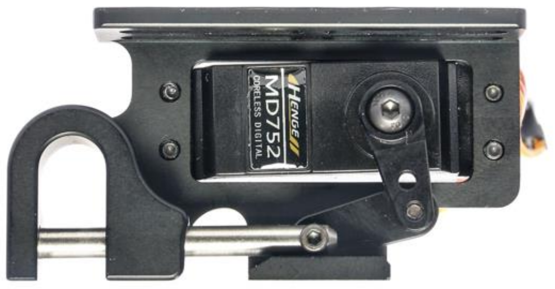


Figure 5: Payload Release Mechanism

When the package is ready to be released, the pin slides out, releasing one side of the harness, while the other side stays attached to the drone. The means that the harness can be reused and is not wasted, which would be a problem for a system with extremely high usage volume.

H. Camera

The camera is the heart of our obstacle avoidance system. The industry standard for collision avoidance in drones is to use expensive sensors such as LIDAR or depth-sensing cameras. By contrast, our system uses a single-point camera and calculates the 3D point cloud by performing calculations with motion data from the drone. Using the cheaper camera reduces the overall unit cost by around 40%. The tradeoff is that our system uses extra processing power to perform the 3D point cloud calculations, reducing the system's power efficiency and flight time. However, this is not nearly as impactful as it sounds, since the computer vision only needs to run during takeoff and landing. As there are no obstacles at the flying altitude, it is safe to turn off the collision avoidance system to save power and rely on GPS for coarse navigation.

The camera we chose for our project has a 5 Megapixels (2592 × 1944 pixels) OV5647 sensor. It is able to capture still images at full resolution, 30fps video at 1080p, and 60fps video at 720p. This particular sensor provides a good balance between cost, resolution, low light sensitivity, and framerate for fast-moving images. This is what allows us to decrease the cost of our drone by so much compared to current offerings.

IV. Printed Circuit Boards

A.PDB

Our power distribution board (PDB) was first measured out by hand and then with those measurements we made a

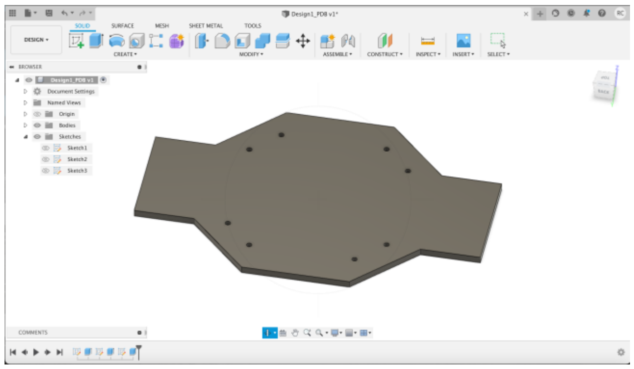


Figure 6: CAD File used for Power Distribution Board

CAD model in Fusion 360, that match commercial products designs to insure it would work correctly without taking away lift from the propellers, also with the proper through hole to match our frame. We then exported the CAD DXF file and import it into EasyEDA where we had designed a schematic with xt60 connectors that will evenly distribute 11V to all of our ESCs [7].

B. Power Module

Our power module was designed in EasyEDA, our PCB schematic builder, and we based it off the parts available from the EasyEDA database that is directly linked to the JLCPCB store that we used to purchase the build board, thus, in theory, getting rid of the problem of not having parts in stock when needed. We knew our battery would be delivering 11 Volts DC power and we needed 5 of those 11 volts to be converted from our power module to our flight controller. We had our battery connect to our board through a XT60 connection, as seen in figure 7 and be split with one path being stepped down to 5.3V and 2.25A that would have connected to our flight control through an 8pin FCC connection. While the other path went to a different XT60 connection that would pass our 11v to our power distribution board and on to power our other systems like our motors directly.

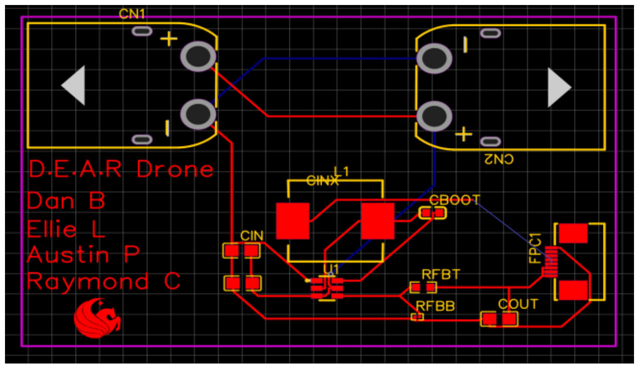


Figure 7: PCB of the Power Module

C. DC to DC convertor

Our DC-to-DC converter was designed in EasyEDA and we designed it based off the parts EasyEDA and JLCPCB said where in stock and available. We start with a Vin of 19v and current of 3.42A that would be stepped down to the voltage to 12v and a current of 15mA that was needed to power our battery management system. After we sent through a few design changes and secondary options we learned that even though the part says it's available on EasyEDA when you go to order on JLCPCB they say out of stock and then when we tried to find the parts ourselves, we would have wait times of 52-72weeks.

D. Battery Management IC

The third type of PCB that our group has made is what we collectively call the Battery Management System, which uses chips from the BQ24 family of chips (Texas Instruments), with our BM IC system using the BQ24133RGYR for its main component.

This chip was used because not only is it excelling in the ability to charge LiPo batteries, but also included in the typical design using such chips, is a protection scheme that focuses on overvoltage and over current protection, which is essential for LiPo batteries because overcharging or charging at rates higher than the batteries C rating can handle can lead to catastrophic destruction of the battery and the release of hazardous material. This is done through a combination of Zener diodes included into the input of the chip itself and the battery structure, and three different MOSFETs as seen in figure 8, that let the flow of current from the chip structure to the battery and the input voltage to the chip structure be controlled thus providing overlapping protection. Another aspect that we chose this chip for is its very useful ability to choose how the configuration of the charging sequence shall be done. In other words, we can choose how many cells we want to charge instead of a generic one cell charge. This is perfect because our battery is a 3-cell type, which is good for battery capacity, but when charging, if we do not charge equally across all 3 cells at the same time, we could damage the battery over time and could potentially over charge one cell thus causing the same problem the over voltage protection scheme is there for to

prevent. Our main chip comes with a pin input that lets us choose between low (1 cell), float (2 cell), or High (3 cell)

which is excellent for our design. The final and most important of the features of this chip, and the reason we choose it in the first place, is that the program we used to design our PCB's, EasyEDA, has an in-built library that tracks what is in stock and what is not, and includes

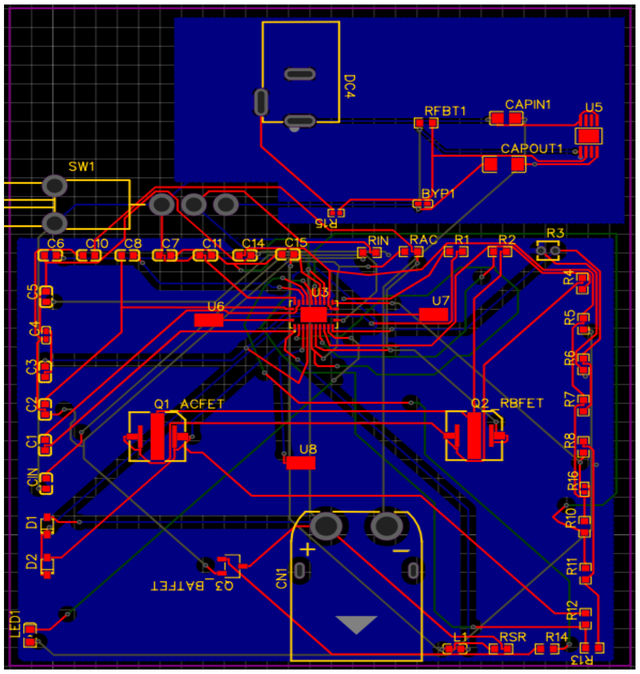


Figure 8: PCB Layout of the Battery Management IC

footprints for schematics for our chosen chip, thus providing us with the ease of mind in the supply chain issue that is currently plaguing the world. However, even EasyEDA could not keep up with the rapidly changing and depleting supply, and so when we actual checked out without design after designing it fully, we were disappointed that our main chip and DC to DC converter were both out of stock and would not be back in stock for some time. If such a time that it comes back into stock however, we expect to receive the full board with all components, test this board, and possibly have to adjust some values for the resistors used in deciding the charging current if simulated current does not end up matching real charging current.

V. SOFTWARE

A. Coding Language

The coding languages being utilized for the drone will be Python and C++. Python will be utilized for the main software of the computer vision and controlling the drone. Python provides a style standard to follow and lots of documentation for completing projects of this magnitude. It is very popular in the machine learning and computer vision community for its feasibility of understanding and being extremely open source [8].

C++ will be utilized for a lot of the code between the hardware and firmware and lower language items. This will communicate how to set the hardware to certain numbers of performance. C++ is an algorithm-based language with object-orientated programming. This allows us to have a lot of flexibility in terms of things we can do with it because it will adapt to our software programs or with communicating to hardware.

B. QGroundControl

QGroundControl is the application that allowed us to control every aspect of the drone, from parameter tuning and sensor calibration, to flight path planning and even controlling the drone during flight. It is an open-source application made to run on the ground control laptop and interface with the drone through Wi-Fi.

We first used QGroundControl to perform the initial setup of the drone. First, we selected the airframe that matched our physical platform layout. We then calibrated all of the sensors in the inertial measurement unit (IMU) as well as the GPS receiver. Next, we tuned the flight parameters such as landing decent rate, maximum attitude adjustment, and differential gain. These settings were very important to tune to our specific drone, because each airframe has different flight characteristics, and these require different software settings [9].

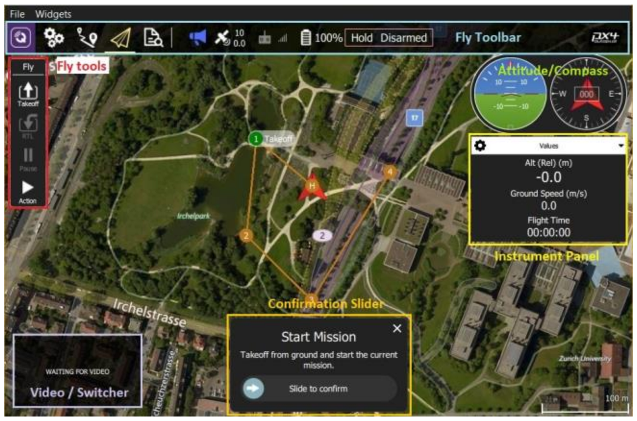


Figure 9: QGroundControl

Finally, the last thing that we used QGroundControl for was mission planning as, as seen in Figure 9. We were able to select a launch zone, drop zone, landing zone, and customize all of the actions in between. We instructed the drone to autonomously take off to a safe altitude, fly to the drop zone, descend to a safe drop distance, drop the package, ascend, fly back home, and land autonomously. This is what really allowed us to implement the delivery feature of our drone. The computer vision provided safety and ensured that crashes would not happen, but QGroundControl was the backbone of the delivery feature.

C. Flight Control Software

The flight control software takes in sensor data from the drone and calculates adjustments to keep the drone flying steady and level. We chose to use PX4 as our flight control software. It is open source with a very active community and supports high-level functions that make it great for integrating with a computer vision platform. Additionally, PX4 allows the drone to communicate directly with the ground station during flight, sending back telemetry data and real-time, as well as receiving commands and updated flight plans [10].

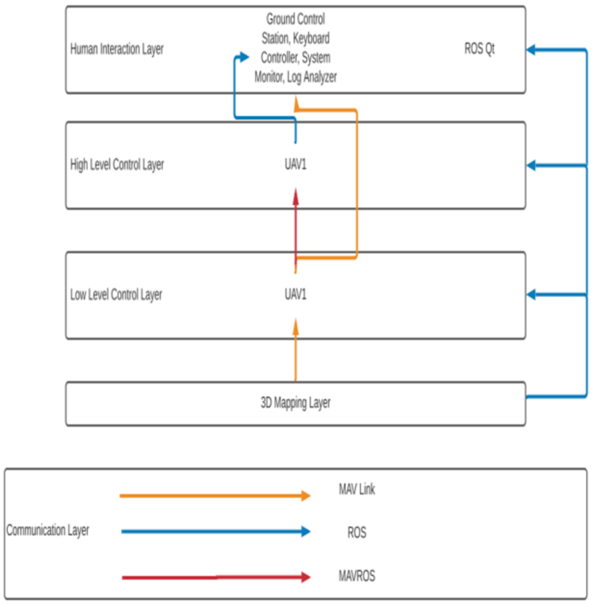


Figure 10: Software Communications Flowchart

As seen in Figure 10, our software involved a complex combination of many layers. On the bottom we have the computer vision layer which sends data about the environment to the control layers. On the low-level control layer, PX4 ensures that the drone has a stable and level flight, despite disturbances. Above that, in the high-level control layer, the software uses GPS data to ensure that the drone is following its planned path. It also uses the computer vision data to ensure that the drone does not collide with any obstacles. Finally, at the top is the high-level control layer. This is the ground control application that allows the user to send commands to the drone, as well as the manual control input for overriding the drone during emergencies.

D. Computer Vision Software

The responsibility of the computer vision software is to identify obstacles in the drone's environment. It does this by using shape, size, and color of objects to estimate the object type, as seen in Figure 11. The position of these objects is then passed to the flight control software to navigate around any obstacles and avoid collisions. Additionally, the computer vision software will be responsible for identifying the package drop zone, marked by a red rectangle, so that the drone knows where to deliver the package.

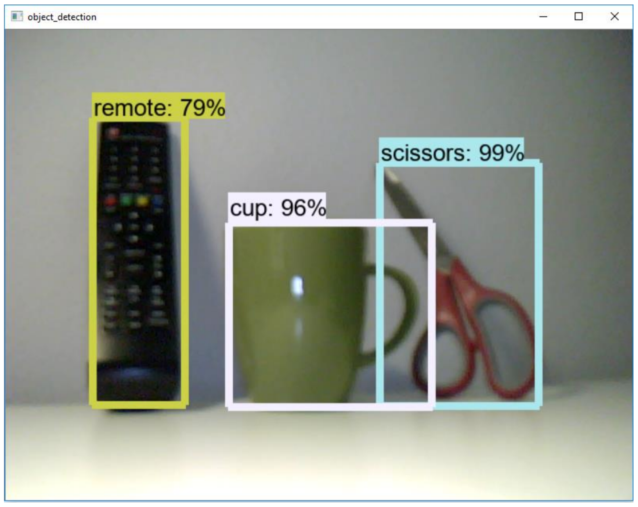


Figure 11: OpenCV Object Detection

Our drone uses OpenCV as our computer vision software. This is a widely used and supported computer vision library which has many different models such as object detection, depth mapping, and visual odometry [11]. All of these features mean that our drone will not get lost or confused during the two most critical times of the flight: takeoff and landing.

E. Path Planning Software

The software used to divert the drone and plan a new path around an obstacle is an extension of PX4 called Obstacle Avoidance. It sits in between our computer vision software and our flight control software. The path planning software takes in the 3D point cloud from ORBSLAM3 to calculate how far the obstacles are and how quickly they are approaching. It also takes in information from PX4 such as the current flight plan and the drone's orientation and velocity. By combining these two datasets, the path planning software can determine the optimal path for the drone to continue on its mission, while also avoiding any obstructions and dangers.

F. MAVLink

MAVLink is the communication protocol used by PX4. It is lightweight and very versatile. The protocol is used in all aspects of controlling the drone, from inter-process communication in PX4, issuing commands from the ground station to the drone, and even communicating with external ROS nodes. The latter is what we use for our obstacle avoidance system. The computer vision software, ORBSLAM3, runs inside a ROS node on the same Raspberry Pi controlling the drone. Using MAVLink, it then publishes the 3D point cloud as a data stream to which the Obstacle Avoidance extension subscribes. If this extension detects an obstacle, it will then calculate the optimal path around it. The Obstacle Avoidance plugin sends this updated flight path to PX4 via MAVLink. As seen in this example, MAVLink is vital to the flow of information in our software.

VI. Limitations

During our project we encountered numerous problems from day one. We overcame many such problems through careful planning or adjusting our goals to be more realistic to our future, but somethings are formed as more limitations than obstacles to overcome except through other avenues such as more funding. Some such limitations we need to stipulate here so as to not be found wanting of reaching out goals.

We found through our batteries being used daily and many times in succession, that they depleted to a "low voltage" setting, which actually couldn't be charged by the micro charger we had on hand as it was dangerous to charge our multicell battery from such a low voltage. This was fixed by charging one cell at a time till they reached minimum voltage needed and continuing on from there, but this could be a limitation when our drone is running multiple missions a day and needs to have constant recharging sessions for its batteries.

We also discovered a limitation in the receiver and transmitter of our drone's communication system. This made it so the ground control system had difficulty addressing the drone and connecting for a mission input. This is not something that wasn't foreseen, but budget played a role in limiting our purchase of which components for the communication suite we could buy.

VII. Conclusion

The DEAR Drone was a complicated project that combined aspects of Aerospace, Electrical, and Software Engineering. We were able to draw from our academic experience for many of the components of the project, but it also involved quite a bit of on-the-spot learning to bridge the gaps in knowledge and experience. This was the first major engineering project for many of us and thus, we learned so much in terms of integrating complex subsystems into a whole, setting realistic requirements and goals, and figuring out the best way to test every aspect of the project.

Overall, the DEAR drone is a proof of concept that shows the feasibility of producing low-cost payload drones at high volumes. It is still effective in many circumstances as most deliveries happen within a short distance from last-mile warehouses and most packages have a low weight. Improvements can still be made could be made in terms of power efficiency, range, and payload capacity. However, these improvements are related to the drone body and power system. These systems can be fitted to any drone body which is why the dear drone is more than just a proof of concept, but also a system with the potential to reduce the cost and complexity of all delivery drones going forward.

The Engineers

Ellie Lane is a 23-year-old Computer Engineering senior that has maintained specific interest for the Software and Embedded systems industry. Ellie is taking a job for Abbott in Boston, MA to work on the embedded systems of class 3 next-generation heart devices.

Daniel Biller is a 22-year-old Computer Engineering student. He has an interest in machine learning and computer vision, with hopes to continue his education in these areas. Daniel's major career goals are to focus on software engineering for robotics applications such as self-driving cars and other autonomous vehicles.

Raymond Chenoweth is a 25-year-old graduating Electrical Engineering student who is taking a job with Boeing in Orlando as an Electromagnetic Effects Engineer, ensuring commercial aircraft are protected from lightning strikes and high intensity radio frequencies.

Austin Perkins is a 26-year-old graduating Electrical Engineering student who is taking a job at Duke Energy in Orlando, FL, as a substation engineer, specializing in designing power grids and their infrastructure

Acknowledgement

The authors wish to acknowledge the assistance and support of the Professors Richie, Wei, Sun, Li, and DeMara for their willingness to see their students during their final presentation and for helping us in our academia careers to their completion.

References

[1] Y. B. D. B. A. M. E. P. M. Chen, "A Case for a Battery-Aware Model of Drone Energy Consumption," *research gate*, vol. 1, no. 1, pp. 1-8, 2018.
[2] M. Irving, "New Atlas," 19 March 2020. [Online]. Available: https://newatlas.com/drones/drone-dodgeball-obstacle-detection-system/. [Accessed 25 April 2021].
[3] NA, NA. *A2212/13T Technical Data - Rhydolabz.com*. https://www.rhydolabz.com/documents/26/BLDC_A2212_13T.pdf.
[4] J. Reagan, "dronegenuity," [Online]. Available: https://www.dronegenuity.com/lipo-drone-batteries-users-guide/.
[5] "Drone Nodes," [Online]. Available: https://dronenodes.com/drone-esc-electronic-speed-controller/. [Accessed 1 February 2021].
[6] "Introduction," *Emlid Docs*. [Online]. Available: https://docs.emlid.com/navio2/. [Accessed: 28-Nov-2021].
[7] NA, "DroneQuadCopterX.blogspot.com," NA, 2 January 2016. [Online]. Available: https://dronequadcopterx.blogspot.com/2019/01/quadcopter-power-distribution-board.html. [Accessed 10 April 2021].
[8] "PLCGURUS.NET," [Online]. Available: https://www.plcgurus.net/drone-programming/. [Accessed 10 February 2021].
[9] NA, "docs.qgroundcontrol.com," 1 January 2020. [Online]. Available: https://docs.qgroundcontrol.com/master/en/SetupView/Parameters.html.
[10] "# PX4 Autopilot User Guide (master)," *PX4 User Guide*. [Online]. Available: https://docs.px4.io/master/en/. [Accessed: 28-Nov-2021].
[11] I. Y. S. L. J. P. Jaehoon Jung, "Object Detection and Tracking-Based Camera Calibration for Normalized Human Height Estimation," *NA*, vol. 1, no. 1, p. 10, 20