Lockheed Martin Robocopter Drone Project

Kelsey Cameron, Trisha Singh, Justin Bates, and Jacob Hazelbaker

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

***Abstract* — Lockheed Martin has expressed interest in developing autonomous, aerial drones which are capable of identifying other drones ("prey drones") in order to deliberately collide with the prey drones, causing the prey drones to fall to the ground. Our team, referred to as "Blue Team", has developed a prototype to meet the design requirements provided by Lockheed Martin. The prototype utilizes a convolutional neural network (CNN) to identify the prey drones. Autonomous flight algorithms were developed to provide various flight modes, including a "pursuit mode" which will guide the prototype drone towards an identified prey drone, in order to collide with the prey drone. Various sensors were integrated with the on-board embedded system and the flight controller to provide situational awareness. First Person Video (FPV) streaming functionality was implemented, enabled individuals on the ground to view a live stream video feed from the perspective of the prototype drone with a Heads Up Display (HUD) providing relevant information.**

***Index Terms* — Quadrocopters, drones, convolutional neural network, first person video, heads up display**

I. Introduction

Aerial drones are becoming increasingly available on the consumer markets, presenting unique challenges not faced by previous generations. On October of 2016 a drone impacted a commercial airliner in Quebec City, Canada near the Jean Lesage International Airport. [1] The collision did not cause any injuries, but has raised concerns about the potential risks which readily available consumer drones may pose to commercial aircraft. Military installations are also at risk, as was demonstrated by the January of 2018 attack on a Russian base in Syria by a swarm of drones constructed from plywood, small engines, and rudimentary electronic components which are readily available on the consumer markets. [2]

To address the growing risks posed by commercially available aerial drones, Lockheed Martin has provided funding for several teams engineering students at the University of Central Florida (UCF) to develop autonomous drones which can identify target drones and deliberately collide with the target drones to knock them out of the sky. This approach does not require any projectiles to take out the target drones, henceforth referred to as "prey drones". Moreover, with a budget of $1500 for the prototype and an additional $500 for initial development, the final product will be a relatively inexpensive solution, as opposed to having personal and weapons on hand to shoot or jam the electronics of the prey drones.

Known simply as "Blue Team", we are a group of UCF engineering students who have taken on this challenge presented by Lockheed Martin. The solution we have created is a drone which can fly around autonomously as it searches for prey drones within the vicinity. Utilizing a high-definition, wide-angle camera, our drone can detect prey drones by processing camera footage through a convolutional neural network. All computational processing is performed on-board the drone via an NVIDIA TX1 Jetson module. Various sensors provide vital readings, including LiDAR for measuring relative altitude from our drone to the ground. Autonomous flight algorithms we developed run on the TX1, which sends rotational and translational flight commands to the Pixhawk flight controller, thus moving the drone accordingly.

Our team, Blue Team, is but one of many teams which are being sponsored by Lockheed Martin to produce a viable prototype. To determine which team has produced the most effective prototype, Lockheed Martin will be hosting a competition.

II. Lockheed Martin Drone Competition

On Saturday, April 14, 2018 at 7:00 a.m. Lockheed Martin will host a competition to determine which team has produced the most effective drone prototype. Located between the Engineering building and the Business Administration building, the competition will be held outdoors. The competition area will be marked with red tape on the ground, indicating the boundary region in which in the drones should remain within.

The prey drones in the competition will be of two main body types: a small rectangular drone approximately 3 inches wide and a slightly larger dome-shaped drone approximately 5 inches wide. While each of the two prey drone types come in only a few colors, Lockheed Martin has indicated that the prey drones may be painted different colors not typically available for those drone models. There will also be decoy versions of the prey drones in the form of printed pieces of paper depicting an image of the various prey drone types.

During the competition each team will have a set amount of time to have their prototype drone in the arena without the other team's drone prototype drones in the competition

arena. Prey drones will then be introduced into the arena and will be flown by Lockheed Martin employees. The prototype drone will then be expected to fly around autonomously within the arena and locate the prey drones in order to intentionally collide with the prey drone. Points will be awarded for colliding with a prey drone, and more points will be awarded if the prey drone falls to the ground as a result of the in-air collision. Points will be deducted if the prototype drone targets one of the printed paper prey drone decoys, which will be placed around the arena. Points will also be deducted if the prototype drone wanders outside of the arena boundaries, including going above the 40 feet high upper boundary.

III. Protective Cage Enclosing Drone

To protect the prototype drone from being damaged while colliding with a prey drone, a 3 foot by 3 foot by 1 foot cage has been constructed to enclose the prototype drone. The supporting frame of the cage is composed of PVC pipe, providing a sturdy, light structure. The sides of the cage are enclosed with lightweight, plastic chicken-wire fencing.

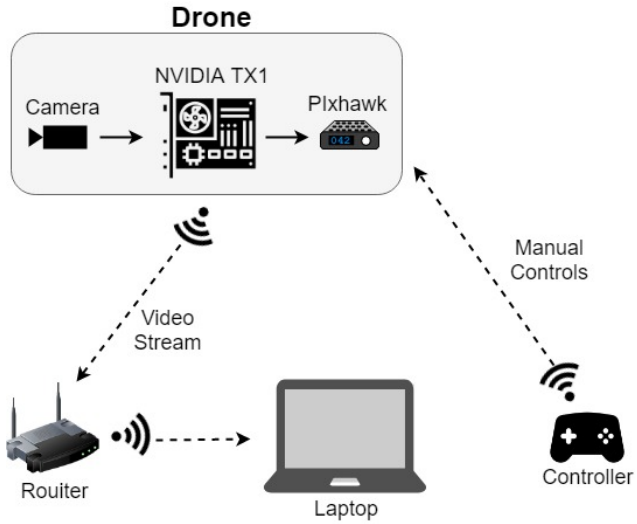
Fig. 1. Protective Cage Enclosing Prototype Drone

Initially, the protective cage was planned to be composed of carbon fiber tubes as the supporting frame and transparent polycarbonate tubes to cover the sides of the cage. While this design was anticipated to be high structurally sound, it was soon realized that these materials would drastically increase the weight of the drone. The design decision was therefore to go with the aforementioned PVC pipe and plastic chicken-wire design for the prototype drone cage.

IV. Network and Communication

The prototype drone can be controlled autonomously or manually. Autonomous control of the drone is performed via commands sent from the NVIDIA TX1 Jetson module to the telemetry port of the Pixhawk flight controller. This is the method by which all commands to the drone will be sent during the competition to ensure that the drone meets the autonomous flight design requirement of our sponsor, Lockheed Martin. Manual control of the drone can still be performed via a remote control which wirelessly sends flight commands to the radio port of the Pixhawk flight controller. Moreover, a flight control mode has been implemented into the remote control to provide a manual override command to ensure that an operator on the ground can take full command of the drone if needed, thus instructing the drone to ignore all autonomous flight commands sent to it via the NVIDIA TX1 Jetson module.

Fig. 2. Network Diagram of Drone Controls and FPV



First Person Video (FPV) is provided via the use of a high-definition camera which streams live video to the NVIDIA TX1 Jetson module, which then transmits the data over Wi-Fi to a nearby router. Bystanders on the ground within the vicinity may connect to the router to view the live FPV with a Heads-Up-Display (HUD) overlay depicting relevant information, such as altitude. If any prey drones have been detected then the HUD will also display a bounding box around the identified prey drone to indicate the location of the prey drone within the camera frame.

V. Hardware Design Decisions

Prior to purchasing any of the hardware components for the drone prototype, our team carefully compared the various options on the market for each of the major hardware components. Special consideration was granted to products which not only met our design requirements, but was also easy to integrate with the other hardware components, included thorough documentation, and could be shipped to our location within a few days.

Very early within the design phase, our team decided to create two drones: a large drone which will be used during the competition, and a smaller drone which will serve as a test bench for debugging and experimentation while the large drone is still being constructed. This design decision proved to be exceedingly useful, providing our 13 person team with valuable information from lessons learned while building and experimenting with the smaller test drone. When it was time to construct the large, main drone our team had a much more clear idea of what to do and what not to do, thus saving precious time and resources.

A. On-Board Embedded System

Considerable computational processing power would be required to detect the prey drones and to run the autonomous flight algorithms. The convolutional neural network (CNN) implemented to identify the prey drones is particularly taxing on the system resources and is highly GPU intensive. While a desktop or laptop computer with a dedicated GPU could provide more than enough system resources to smoothly run the CNN, such hardware is not an option for this project due to the design requirement that all processing be on-board the drone. The embedded system chosen must be light enough to easily be carried by an aerial drone, yet powerful enough to smoothly run the CNN and autonomous flight algorithms.

Of the many embedded systems considered, 3 stood out as potentially viable options: Raspberry Pi 3 Model B, BeagleBoard, NVIDIA TX1 Jetson module, and the newer NVIDIA TX2 Jetson module. Table 1 below summarizes the main specifications of these 4 embedded systems.

TABLE I
COMPARISON OF POTENTIAL EMBEDDED SYSTEMS

System	Rasp. 3	B.Board	TX1	TX2
CPU	ARM A53	ARM A8	ARM A57	Dual-Denver + Quad ARM
GPU	*none*	*none*	256 CUDA	256 CUDA
MEMORY	1 GB	256 MB	4 GB	8 GB
STORAGE	Flash	Flash	16 GB	32 GB

Upon realizing that a CNN would be necessary to detect the prey drones, the Raspberry Pi 3 Model B and the BeagleBoard had to be ruled out as possible choices for the embedded system, due to low memory and lack of a dedicated GPU. The new NVIDIA TX2 Jetson module therefore seemed like a prime choice for the embedded system. However, at the time of writing this paper, the NVIDIA TX2 is a very new product and is woefully lacking in proper documentation. Even more concerning were a plethora of customer reviews online and reports from fellow engineering students who purchased the NVIDIA TX2 that the system is buggy and does not yet have properly working drivers for some of the I/O ports. The design decision was therefore made to purchase its predecessor, the NVIDIA TX1 Jetson module, which has a very well-established wealth of documentation and also still has 256 CUDA cores.

B. Flight Controller Module

While the NVIDIA TX1 Jetson module will handle all of the on-board computational processing, the motors of the drones must be controlled by a flight controller. The flight controller is able to interact with the motors via an Electronic Speed Controller (ESC), one of which is connected to each of the 4 drone motors. The desired flight controller must be capable of converting basic navigation commands sent from the remote control or the NVIDIA TX1 Jetson module into commands which can be interpreted by the ESC to spin each of the 4 drone motors appropriately to perform the desired aerial maneuver. Special consideration was given to flight controllers which have various sensors built into them, including a magnetometer, barometer, and an Inertial Measurement Unit (IMU).

Quite soon it became apparent that the Pixhawk line of flight controllers is the industry leader for consumer drone projects. There are a wide array of generic flight controllers readily available on the market, but these flight controllers were found to be far less compatible with other drone components. With Pixhawk as the industry standard, so many vital drone components have been designed to be compatible with Pixhawk, often only requiring that the component be plugged into the Pixhawk for it to function correctly. After careful consideration it was decided that the Pixhawk 1 flight controller is the best option for our project. There is a smaller version of the Pixhawk 1 called the Pixhawk Mini, but it costs $50 more than the Pixhawk 1 and the size difference is negligible in contrast to the large size of our drone. There is also a newer Pixhawk 2 flight controller, though it is as much as $100 more expensive than the Pixhawk 1.

C. First Person Video (FPV) Camera

The camera is a vital component to the autonomous drone project, serving as the means by which prey drones are detected. Such a camera would need to be able to see the small prey drones with some detail at distances up to 40 feet, the length of the competition area. Moreover, the camera would need to be able to stream FPV continuously

to the NVIDIA TX1 Jetson module, ruling out all cameras which can only store video footage to writable forms of media such as to flash memory cards.

To determine the necessary camera resolution required to effectively identify prey drones, tests were performed with various cameras of differing resolutions and image quality. There are a wide array of high-definition (HD) cameras on the market at a low cost. However, a major concern of this project was that the NVIDIA TX1 Jetson module would be not be capable of processing a continuous stream of HD video nearly as quickly as it could process lower quality video at a reduced resolution. The convolutional neural network (CNN) is particularly resource intensive to run, especially for processing real-time video.

It was soon realized that the CNN could not reliably identify prey drones at a distance if the camera utilized is of a lower resolution and image quality. Utilizing cameras with a low Megapixels (MP) rating of 5.0 MP or below demonstrated that any prey drones at a distance appeared to be little more than a few blurry pixels, providing very little detail for the CNN to analyze. Further testing and research also revealed that the quality of the camera sensor quality is a critical factor. Having a high MP rating merely denotes the amount of pixels utilized to represent the image, but the MP rating alone does not explicitly describe the quality of the image produced.

To find a viable balance between providing HD video and not taxing the system resources of the NVIDIA TX1 Jetson, the design decision was made to utilize a 1080p camera. The particular model chosen is the WIMIUS Q2, 12 MP Action Camera which is capable of streaming video via USB. An appealing feature the WIMIUS Q2 is the fish-eye lens, which provides a 170 degree field of view (FoV). While it is true that a fish-eye lens does have some distortion around the edges of the image, typical cameras have a smaller FoV and thus less peripheral vision. Prey drones which are within the edges of a fish-eye lens camera will appear slightly distorted, but a camera with a normal lens would not be displaying these far peripheral edges of the image at all. It would therefore be more useful to see a distorted image of the prey drone along the periphery and possibly still be able to detect the prey drone, than to not see the prey drone at all.

VI. Prey Drone Object Detection

The most mission-critical aspect of the project is the ability to detect nearby prey drones. Without this functionality, the resulting prototype would only be able to blindly fly around at best, entirely oblivious to any prey drones which may or may not be in the vicinity. This is made possible by streaming FPV via the WIMIUS Q2 Action Camera to the NVIDIA TX1 Jetson module, which runs an object detection algorithm on each frame to determine if a prey drone is currently in sight of the drone. A wide array of object detection algorithms were tested and evaluated for effectiveness. The primary object detection algorithms which were most promising are discussed below.

A. Template Matching

Perhaps the most simplistic solution is to use a template matching object detection algorithm. The template matching algorithm is setup by providing it with a sample image of the object which the algorithm is to detect. Thereafter, the algorithm can analyze images or video to see if that exact template image is present within the frame. In order for a match to occur, every pixel of the template image must match up perfectly within the image or video which the template matching algorithm is analyzing. While easy to implement, template matching is extremely inaccurate for real-time video applications.

Fig. 3. Template Matching Algorithm Using the Template Image Depicted on the Top-Left and Returning a False Positive

Figure 3 above represents the primary issue with the template matching algorithm. The algorithm is only capable of recognizing the drone if template image is found exactly the same in the image being analyzed. The drone depicted in Figure 3 is rotated differently than it is in the template image, thus the algorithm was unable to find the drone.

B. Color Histogram Analysis

An alternative method for detecting objects is to deploy a color histogram analysis algorithm. A template image is provided, which is then analyzed to determine how much of each color range is present within the template. For example, if the template image is of a blue ball in front of a yellow background, then the resulting color histogram might state that the template image is 60% blue and 40% yellow, for example. The color histogram analysis algorithm then searches an image or video for a region in which there is also approximately 60% blue and 40% yellow.

Fig. 4. Color Histogram Analysis Successfully Takes a Template Image, as Seen on the Left, and Detects the Object

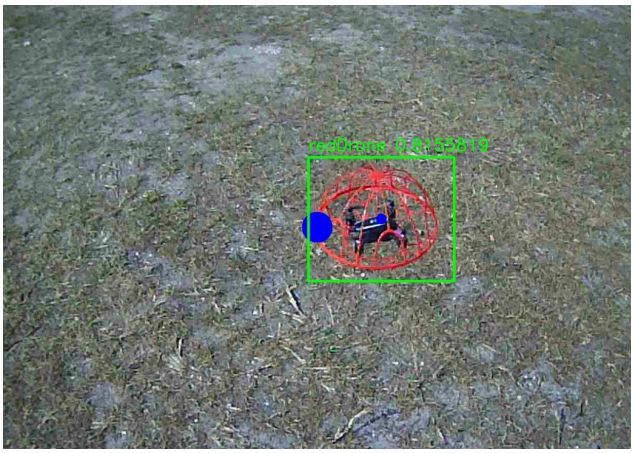
Within the Provided Image, Depicted to the Right

Color histogram analysis works quite well for identifying objects which have been rotated, since a successful object match is based upon recognizing similarities in the amount of each color. Problems arise, however, when the lighting of the template image is different from the lighting of the image being analyzed. For instance, if the template image was taken in a dark room and the image being analyzed was taken in a well-lit room, then it would be unlikely that the algorithm could successfully detect the object. This drawback makes the color histogram analysis algorithm a poor choice for our project. Moreover, it is unknown what the colors of the prey drones will be, since Lockheed Martin has indicated that some of the prey drones will be painted to alter their default colors.

C. YOLO Convolutional Neural Network (CNN)

The solution chosen for this project to detect prey drones within a live stream of video, is the You Only Look Once (YOLO) convolutional neural network (CNN). YOLO is trained by providing thousands of images with XML annotations which describe the bounding boxes around the prey drones within the images. The particular version of YOLO utilized for this project is Tiny YOLO, a streamlined version of the YOLO CNN which provides efficient object detection at a reduced cost to system resources. [4] This is particularly useful since a key design requirement of the project is that all processing be performed on-board the drone. YOLO is able to run faster than most CNN as a result of YOLO applying the CNN to the entire image once, whereas traditional neural networks often run the network across an image in many locations. Therefore, the YOLO network requires only one evaluation per image whereas traditional networks can require thousands of evaluations. [3]

Fig. 5. YOLO Detecting a Prey Drone



VII. Sensor collaboration

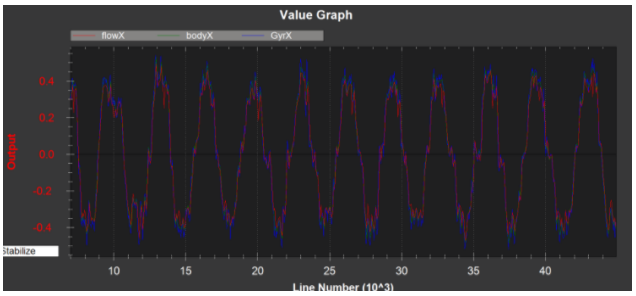
The two main sensors included in this system are the LiDAR rangefinder and the optical flow sensor. We opted not to use a GPS component to determine our position, altitude, and heading because of the nature of the competition location. GPS signal in between buildings were inaccurate and altitude measurements from the flight controller barometer were difficult to calibrate and inaccurate at best. Both the LiDAR and optical flow sensors together work to replace the functionality of a GPS and can even successfully work indoors.

A. Optical Flow

An optical flow sensor is a low resolution downward pointing camera that compares ground detail in consecutive frames. The difference in ground features from frame to frame along with the systems gyros give an accurate velocity and position value measurement to the flight controller. In autopilot flight modes such as loiter, and position hold the aircraft can remain extremely stable at a consistent location or heading. This stability is necessary for our systems object detection algorithm accuracy and for eliminating drift to stay within boundaries. Calibrating and testing the sensor required focusing the lens and running a few flight tests to ensure the flight controller IMU and optical flow are consistent in their x and y positions respectively. (Shown in figure X.) The optical flow sensor we chose was the PX4FLOW; this sensor is highly sought

after for its accuracy and compatibility with the Pixhawk flight controller but will only work with an accurate rangefinder.

Fig. 6. Optical Flow Sensor Calibration



B. Rangefinder

The purpose of the rangefinder is to give accurate altitude measurements for our flight controller in place of a barometer. It is essential for the optical flow to work accurately and is a requirement for compatibility with the flight controller. We chose the Benewake TFMini LiDAR because of its size, price, reliability, and accuracy as a rangefinder. This rangefinder communicates serially through the Pixhawk serial 4/5 or telemetry 2 port. Configuring the LiDAR was extensive and required quite a few steps including an FTDI serial to USB converter.

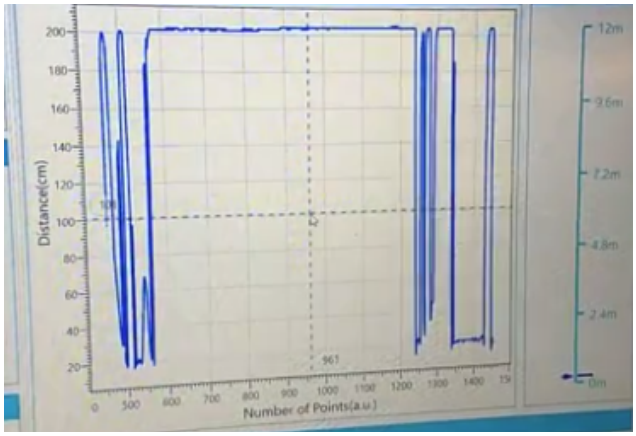


Fig. 7. LiDAR Sensor Returning Data

After soldering the LiDAR serial cable to the FTDI chip we were able to configure the LiDAR module through the Benewake GUI to output in a decimal format (the flight controllers designated input/output format for data) rather than its preprogrammed hexadecimal output format. We used Tera Term, an open sourced terminal emulator to ensure we were outputting in the correct format before we unsoldered and re-soldered the serial cable. There were quite a few parameters to configure in the mission planner software which enable this particular LiDAR and that disabled the use of a GPS module. (The flight controller will not arm or take off until you disable the GPS functionality.)

VIII. Flight Control Software

Throughout the development of this project various free software tools were implemented to speed up the design process. These tools were extremely useful and sometimes irreplaceable for configuring the hardware.

A. Mission Planner

Mission planner is the software we used to incorporate and configure sensors with our flight controller. The software provides an easy way to program flight modes, test motors, calibrate peripherals, and upload firmware. Another great feature of the mission planner software is the pre-arm check feature which ensures critical systems are functioning properly before taking off.

Calibrating the frSky Taranis radio transmitter through mission planner allows for accurate manual control of the quadcopter by setting minimum and maximum PWM signals for all switches and sticks Switches on the Controller can be set to a specific PWM frequency and within mission planner can be programed to initiate autonomous flight as a failsafe. One switch signal is set to immediately land, one signal is for hold position, and one for a preprogrammed mode

Calibrating the flight controller accelerometer is an essential part of flight only accessible through mission planner. Accelerometer calibration involves placing the drone flat on all of its sides consecutively to allow the IMU to understand where it is spatially. This allows the drones PID settings to autocorrect if the drone's stability becomes unnecessarily disrupted.

B. DroneKit

DroneKit is an open-source platform which allows developers to create flight control, obstacle detection/avoidance, and computer vision applications. The software uses MAVLink to communicate with the Pixhawk flight controller and is also compatible with our companion computer, the Nvidia Jetson TX1. The advantage of DroneKit is that the code is written in Python and it is much easier to understand than the code for the ArduPilot firmware which runs on the Pixhawk. [5]

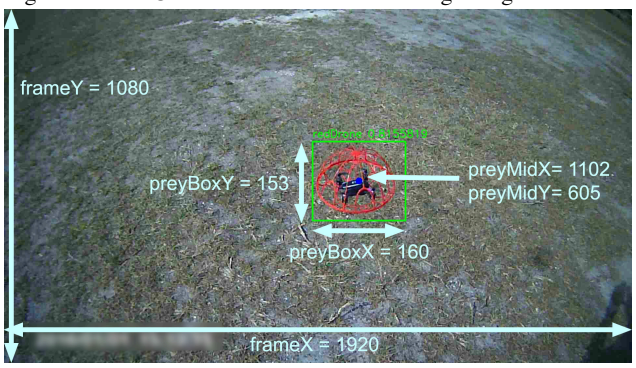
In this project, DroneKit was specifically used to integrate the computer vision algorithms with the flight control algorithms and program the search mode. For the search mode we used a yaw command in DroneKit which

rotated the drone in place for 10 degrees. It continued to rotate the drone 10 degrees until a prey drone was detected in the frame. After the drone was detected using Tensorflow and Tiny Yolo, we performed some calculations to figure out the exact angle and the x,y,z velocities that the drone needed to travel to pursue the drone. Using these calculations, we fed the angle into the yaw function and the velocities to the velocity function.

IX. Autonomous Flight Control Strategies

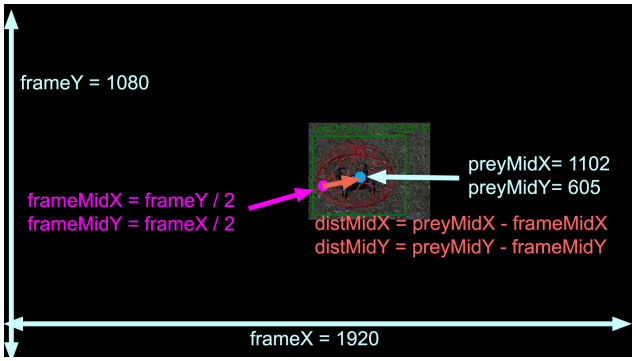
The flight mode describing how to autonomous control the drone when a prey drone has not yet been found, is referred to as "search mode". If a prey drone is identified, then the autonomous flight control algorithm switches to a mode referred to as "pursuit mode".

Fig. 8. Data Collected for Autonomous Flight Algorithms



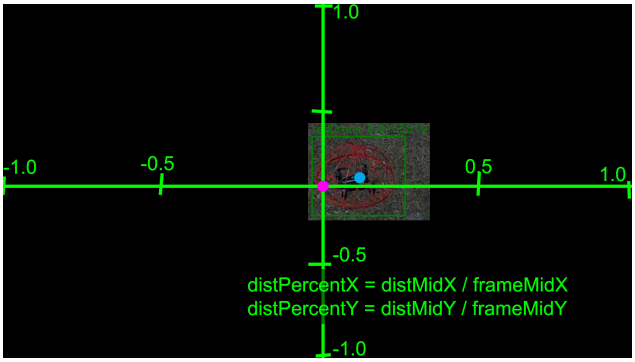
Once YOLO has processed an image and detected a prey drone, the autonomous flight algorithm collects useful data from the image, including the bounding box dimensions in pixel-space and the mid-point of the detected prey drone. A displacement vector in pixel-space is then calculated.

Fig. 9. Displacement Vector in Pixel-Space is Calculated



This pixel to degree ratio was then normalized so we could use any resolution camera with our algorithm. Based on this idea, we created a similar algorithm to match the altitude of the prey drones within the z-axis. We measured the approximate height of prey drones and we knew the total pixel height of our camera, 1080p. From there, we created a pixel-to- distance ratio that should theoretically allow us to move our drone the correct distance (up or down) based on where the prey drone located.

Fig. 10. Normalized Displacement Vectors Which Are Adaptable to any Camera Resolution



There were many pitfalls with this approach, like the blurry edges of our fish-eye camera, along with a difference between our drone and the simulation drone. However, given the time constraints and budget constraints, we believe this is the most effective approach to move our drone around the field to pursue the prey drones. Given the size difference between our drone and the prey drones (3 feet versus 5 inches), we may even be able to make contact even if the algorithm ratios are not perfect. TensorFlow allowed us to run the 23,000 training data images through a neural network on the TX1 chip, and then use the detection results to navigate our drone around the field. To direct the drone autonomously, flight control algorithms were developed. There are 3 primary flights: search mode, pursuit mode, and manual mode.

VII. Power Distribution

Onboard the drone, we have several DC voltage sources -28 Volt 10000mah battery used to power the motors and ESCS. We also used a 5500mah battery as a backup battery and two 4200mah batteries for the prototype quadcopter. Since each motor draws an approximate current of 30 amps, we have a total of 120 amps running through our subsystem. This large current amount unfortunately was not compatible with many of our connection joints, and we resoldered new connections to use a better current-rated connector.

We used the PCB to power the Pixhawk, because our main power distribution board fried last minute due to high

current levels. We were able to remedy this by providing Pixhawk with a steady 5 volts. Additionally, we decided to mount and power the NVIDIA Orbitty Carrier board, we required an additional DC voltage source and purchased a smaller, 9 Volt DC Battery. All of our power management research enabled us to solve last minute problems like our battery fusing together from too much current flowing through each connector, and the power distribution board frying. As a result, we were able to directly modify our hardware design to prevent any battery failures, fire risks, and other safety issues.

VIII. Conclusion

The Lockheed Martin Robocopter project has proven to be quite challenging, yet rewarding. Much has been learned about concepts we previously knew very little to nothing about. Creating both a large prototype drone for the actual competition, as well as a smaller drone for prototyping and testing, was an incredibly useful strategy which enabled our team to discover many solutions to common issues before experimenting with the large, prototype drone.

Our team communication between each member was remarkably effective. When one member needed help in a particular area, 3 or 4 people would offer to jump in and help with any issues in any area – regardless of particular majors. Every single member of our team put in effort to ensure our success within this project, which is quite rare in larger projects like these.

As a result, we were able to put our efforts together to achieve far more, and ensure that the bridge between hardware and software would allow us to fly effectively. Our team consists of Mechanical, Aerospace, Electrical, and Computer engineers, along with the Computer Science Team as well. This multi-disciplinary project gave us all a unique experience to learn about areas that integrate within other engineering fields, and also enabled us to learn the most effective ways to collaborate as a team. We were very pleased when our final flight tests reflected that immense effort. By learning every possible failure and how to solve it – we learned how to succeed.

Acknowledgement

The authors wish to thank Lockheed Martin for providing funding and mentorship for this project.

About the Authors

Kelsey Cameron is a computer engineering student at UCF. Upon graduating in May of 2018, Kelsey will be moving to Texas to begin working at Texas Instruments (TI). She enjoys embedded systems, everything about engineering, and composing piano music.

Jacob Hazelbaker will be attending Carnegie Mellon University in the fall to study a Masters of Science in Information Security. Despite being hit by a car this semester from walking to campus, he still enjoys long walks by the beach.

Justin is an Electrical Engineering student at UCF. He will be working as an electrical engineer at DRS Technologies in Melbourne Florida. He's interested in computer hardware and music as a hobby.

Trisha Singh is a computer engineering student at the University of Central Florida. She is considering pursing a Masters in Computer Science. She enjoys high-level software development, singing, and biking.

References

[1] J. R. Miller. (October 16, 2016). Drone hits passenger plane for first time in North America. *New York Post*. Retrieved from https://nypost.com/2017/10/16/drone-hits-passenger-plane-for-first-time-in-north-america/

[2] D. Reid. (January 11, 2018). A swarm of armed drones attacked a Russian military base in Syria. *CNBC*. Retrieved from https://www.cnbc.com/2018/01/11/swarm-of-armed-diy-drones-attacks-russian-military-base-in-syria.html

[3] Redmon, Joseph."YOLOv3: An Incremental Improvement." *YOLO: Real-Time Object Detection*, 2018. [Online]. Available: pjreddie.com/darknet/yolo/.

[4] Hollemans,Matthjis. "Real-time Object Detection with YOLO" 20 May 2017. [Online] Available: machinethink.net/blog/object-detection-with-yolo/.

[5] "About Dronekit" *About Dronekit.* [Online]. Available: python.dronekit.io/about/overview.html/.