***Abstract*—**Fall quarter was focused on getting the base of the drone up and running. We secured all the necessary hardware to build the drone and image processing, and assembled each separately. We tested flight with the drone using different code and algorithms. Then we tested image processing on a raspberry pi to make a basic version of the scoring algorithm. Both parts of the project were assembled separately so that they would be able to be combined next quarter.

1. Introduction

OUR purpose in this project is to make an automated target scoring drone that would be able to score mortar tests. It was originally proposed to us as being partnered with the Navy, which we later talked to them to secure the project. The purpose of the project is to get the current target scoring more efficient. The system that is currently in place is a manual drone that can only process one image at a time, so it can cover only one test at a time. Our drone will be automated and able to go to several different locations. It will be able to process far more than the current system can.

1. Challenges

In terms of actually making it, the most difficult part was figuring out how to automate it and how to make it compatible with the raspberry pi. Although we could make it fly fairly simple, we had to be able to control it by sending signals through a raspberry pi, hence we had to research multiple examples [1.], to get an idea of how other autonomous drones had been programmed.

In terms of the image processing, getting the program to work for a digital painted picture was easy. However once you factor in real life situations, it becomes much more difficult. The main problem is that there are too many colors in real life and it gets mixed in with what the program is looking for. This is especially true for shadows as they mix in when looking for the black shot to compare to the target. The next step will most likely be machine learning and machine vision. The only real way to have it recognize the target and shot every time is to have it recognize what a circle is. And then the next step, have it recognize the difference between a target circle and a shot circle.

1. Hardware and Software Used

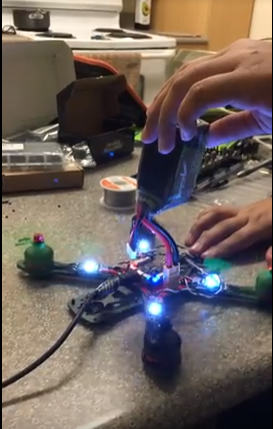
In order to fabricate the drone we purchased:

* Drone frame
* 4 Electronic speed controllers
* 4 brushless motors/propellers
* 2 Raspberry Pi’s
* Raspberry Pi gps module
* Raspberry Pi camera
* LiPo battery
* 1 Flight controller
* Portable battery for the Pi

The image processing and the GUI module were made using Python.

1. *Methods for Fabrication of Drone:*

After considering the autonomous system that Rabah implemented with the Raspberry Pi, and how cheap of an option it was, we decided to have the Raspberry Pi act as the microcontroller [1.] Fabrication was done by utilizing the materials listed above and connecting each component. The first step was to assemble to frame. We used M3 screws and standoffs to connect the base plate onto other components such as the PCB. As soon as as good foundation was established the Motors were soldered to the electronic speed controllers, which were then soldered onto the PCB board. Next, the flight controller was attached above the PCB, and the ESC’s were soldered to the motor pins on the flight controller. Next, the top plate was connected to the standoffs attached to the base plate and the LiPo battery was velcroed on the top plate and connected to the PCB, in order to deliver power to the motors. The GUI is done using Python and Pycharm IDE.



*Fig 1: Shows the drone with the top plate detached*

1. *Methods of GUI:*

GUI is used in base station for this project. GUI consists of simple window where user can enter how many locations they can go. Then the GUI will prompt user on entering the longitude and latitude for the locations they can go. For example they can enter: Longitude: +40.689060 , Latitude: -74.044636 or Latitude: 40 degrees, 42 minutes, 51 seconds N ,Longitude: 74 degrees, 0 minutes, 21 seconds W. After entering the longitude and latitude user can enter to go to those locations. On the bottom of the screen, the user can see the update of every single movement. Example: Update: Drone is on the base, Enter longitude and latitude to go to (number of) locations. Going to first locations. Also on the GUI there is a current status on the screen where every time a drone is landed on the certain location it will show the checkmark.

Then, there is a live update of a drone going from one location to another location. When the drone gets to certain location, image processing takes place. The image that is captured by drone is then shown on the GUI screen.

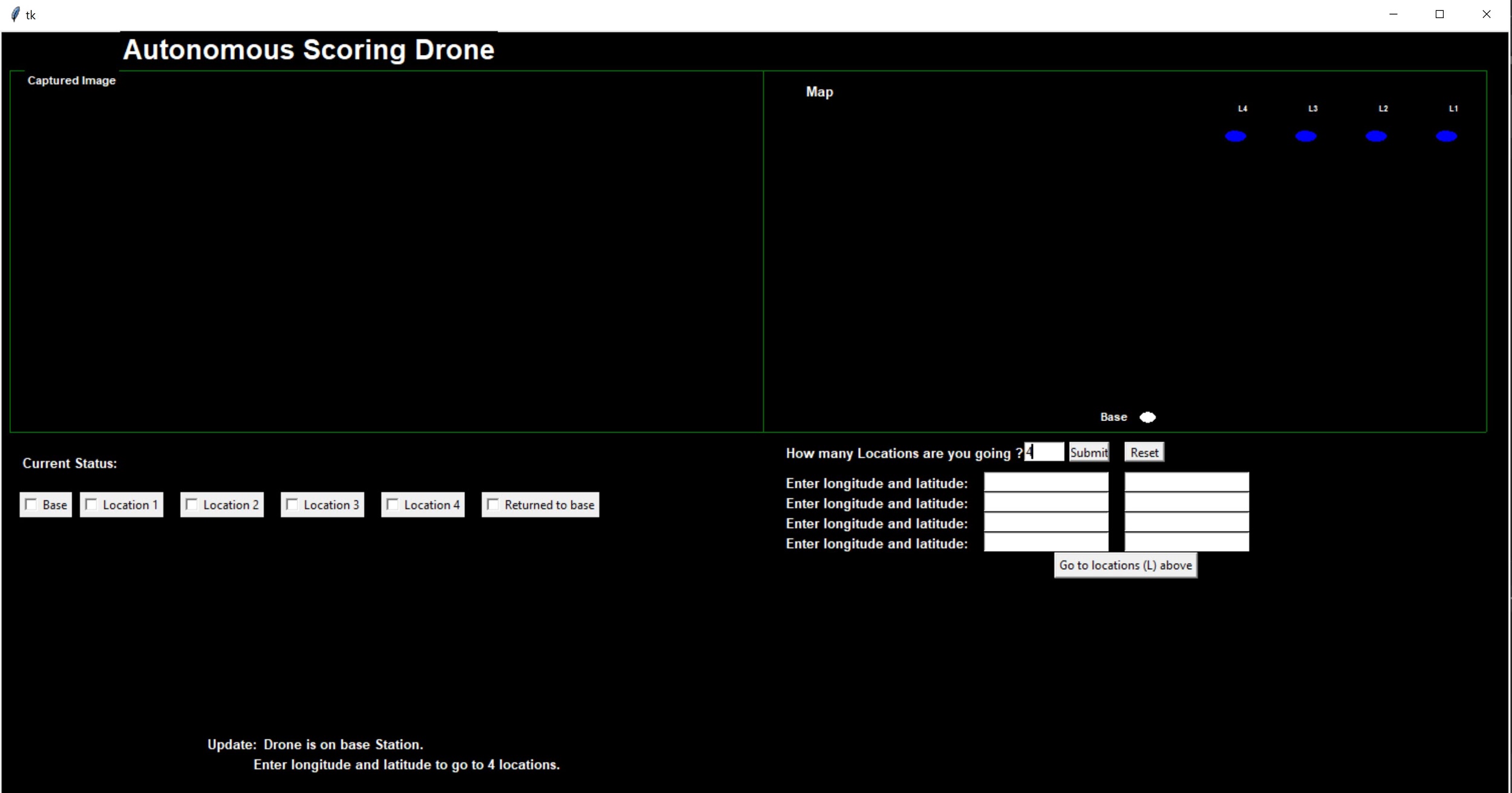
Fig 2: Simple GUI window



Fig 3: Expected output

1. *Methods of Image Processing*

For image processing the program recognizes two things: the color of the target and the color of the shot. The color of the target is variable and can be chosen by the user, while the color of the shot is assumed to be black. After the colors are chosen, the program adds up the X and Y values of each pixel for each color. Then it will divide by the total number of pixels counted and that will produce the average, in other words the center of the circle. Based off of the distances between the two centers, it will give a score from 0-10.

After going over [4] I realized how is best to expand on the program. Even though it recognizes circles well on a painted picture, it is almost impossible to recognize it on the camera. The next stage will be using deep learning in order to have the program recognize circles, and then hopefully recognize specifically colored circles. This is so that it can always recognize the target and the shot regardless of the clarity and of the surroundings.

1. *Results:*

After assembling all necessary components, the motors were tested and the drone was able to fly straight up without any problem. However flight to certain endpoints are marked as the next goal. Additionally, landing gear and a piece to hold the raspberry pi and its battery still need to be fabricated and attached.

The GUI allowed for the user to input multiple coordinates(longitude and latitude). These coordinates are saved and sent over bluetooth to the raspberry pi acting as the flight controller. The bluetooth between the pi’s has yet to be tested.

*The image processing*

1. *Performance:*

The overall performance of the drone has yet to be tested. However the individual components have performed to our expectations.

The motor tests with the drone determined that it flys perfectly in an environment without any environmental constraints. Autonomous flight performance will be tested next quarter.

The GUI executes as expected, saving user inputs. Outputting a map and the picture the drone took will be tested next quarter.

IV. Conclusion

This quarter was primarily focused on getting the drone built, the construction of a graphical user interface and the completion of an image processing algorithm. In order to meet the objectives of the project, a drone built from scratch was believed to be the best option, rather than buying a prebuilt one. Building a drone meant we could just start having it function the way we intended it to without having to “hack” the drone in order to have it meet the objectives. Additionally, a drone built from scratch would allow for choosing the most optimal components to meet our objectives and the constraints of the project. This quarter, the drone was put together and could perform basic flight features, the code to fly autonomously will be implemented by early next quarter.

Since the drone’s flight would ultimately be dictated by a Raspberry Pi flight controller, the GUI would also be coded in python, since we would be using Python to communicate between the Raspberry Pi at the base station, and the Pi which would act as a flight controller. Using Pycharm, the user would be allowed to enter the coordinates of multiple waypoints which would be communicated via bluetooth to the pi on the drone. Additionally, the interface is set up so that in the future, one can track the location of the drone relative to the waypoints and the takeoff point.

Once the Pi flight controller detects that a waypoint has been reached, the drone will snap a picture using the Raspberry Pi Camera and then it will send it back to the other Pi for processing. Like the other modules, the image processing algorithm was written in python and returns a score which would rate how accurate the mortar shot was from the target. This will be achieved by utilizing machine learning in order for the module to recognize an accurate shot from an inaccurate one. Overall, most of the components for the drone are done, and all that is left is to combine what we have.

V. Acknowledgements

This section serves as a brief acknowledgment to all the members of Team Yark: Bert Yu and Sukhmanjit Kaur who worked on the fabrication of the drone and achieving basic flight, Eric Rodriguez, who developed the image processing algorithm, and Bibek Adhikari who created a graphical user interface. Also thanks to our advisors, Mahdi Maaref and Zakher M. Kassas for providing advice and setting weekly goals for us to achieve. Also thanks to the EECS 159A course and UROP, for reimbursing and providing us with the necessary equipment in order to work on this project.

VI. Appendix

*A. Communication*

For communication we chose to use Bluetooth communication between Raspberry PIs. [2.][3.] For our current position in the project, long range flying is out of the question. So until we hit that point, Bluetooth communication is enough. It allows for us to have a private connection between the drone and our homebase, but allows us to have a good range compared to other things such as infrared. Bluetooth also does not need a network in order to connect, only a sending signal and a receiving signal. The ideal environment for our drone will be in the middle of nowhere at places where the military is going to test. These may be places in the middle east with no network connectivity whatsoever, so it will need another way for data to transmit.

*B. Constraints*

One of the main constraints in building a drone is always going to be power. You have to provide enough power in order to supply the Raspberry Pi, controllers, and motors and other electronics. The battery has to provide enough power to the motors to take the drone on a long autonomous flight. In addition, the battery must have correct voltages that meet the specifications of the other components such as the motors and ESCs. Because of this, we had to consider multiple power sources: one for the drone and one for the raspberry Pi. These problems are easily solvable by researching the specifications of the hardware.

Another constraint for us was the price and budget. Building a drone is one of the more expensive projects, and when there are a lot of different parts to buy, one has to do their research. We had to find the correct electrical components and the correct sizes for physical parts to make the drone we did. The parts have to be up to a certain spec in order to actually fly the distances necessary for the project. The base also has to be large enough so that it can hold all of the electronics. For our testing, it will not fly high enough in order to warrant any regulations, but if necessary we would simply have to file for a permit.

Another constraint we were faced with was the weight of the drone. With the motors we have, we cannot surpass the thrust of the motors we purchased. Considering this, we had to purchase a LiPo battery and a raspberry pi power supply that did not go over the threshold set by the motors. Having a weight that is not too high, or too low, allows for the drone to fly to multiple waypoints much more efficiently. Additionally, less power would be supplied to the motors, extending the runtime of the drone by a significant amount. To solve this issue, basic calculations can be performed to determine just how much weight it can carry.

*C. Hardware and Software Insecurities*

As for hardware insecurities, there are so many small components connected on a small surface that it is subject to a large failure if even one part messes up. For instance, if a motor shorts, then the entire drone can fall and ruin all of the equipment. Our GUI only allowed the user to access the GPS coordinates of the drone and controlling it manually. There is no way to access the drones code without actually plugging in to the drone physically. However we currently have no protections in place to prevent a way for the software to be accessed when they have the drone. There is currently no password protection on it, but it should be implemented once it is connected to a Raspberry Pi. Once the Pi is the main controller, then a password will be necessary to login.

*References*

[1.] Rabah, M., Rohan, A., Talha, M. et al. Int. J. Control Autom. Syst. (2018) 16: 3013. <https://doi.org/10.1007/s12555-018-0017-x>

[2.] Kan M, Okamoto S, Lee JH. Development of drone capable of autonomous flight using GPS. In: Proceedings of the international multi conference of engineers and computer scientists, vol. 2, 2018.

[3.] Patrik, A., Utama, G., Gunawan, A.A.S. *et al.* GNSS-based navigation systems of autonomous drone for delivering items. *J Big Data* 6, 53 (2019) doi:10.1186/s40537-019-0214-3

[4.] T. S. Huang, W. F. Schreiber and O. J. Tretiak, "Image processing," in *Proceedings of the IEEE*, vol. 59, no. 11, pp. 1586-1609, Nov. 1971.

doi: 10.1109/PROC.1971.8491

keywords: {Image processing;Image enhancement;Optical sensors;Pattern recognition;Image coding;Optical computing;Image quality;Optical imaging;Degradation;Holographic optical components},

<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1450421&isnumber=31150>