American University of Beirut

Faculty of Engineering & Architecture

Electrical and Computer Engineering Department

EECE 501 Final Year Project (FYP)

Group Progress Report

Fall 2018-2019

The Utilization of Drones to Monitor the Health of Pine Trees

Group 04

Ali Fehmi Shibli	Computer & Communication Engineering
Youssef Youmni Al Jrab	Electrical & Computer Engineering
Yahya Khaled Al Ali	Electrical & Computer Engineering

Advisor: Professor Naseem Daher - Department of Electrical and Computer Engineering

Co-advisor: Professor Nabil Nemer - Department of Agriculture

I. EXECUTIVE SUMMARY

Lebanon, a country distinguished by its unique ecological formation and its charming natural wealth, has been facing exponentially growing environmental problems dominated by the drawbacks of parasites. Global warming is the major concern for entomologists as climate change drives environmental disequilibrium and excites the growth of parasites worldwide. This Final Year Project targets the Pine Processionary Moth which attacks and kills - on average - 150 Pine trees in Lebanon annually, a value worth of $1 Million. The solutions which exist in the market such as biopesticides, rifles, etc., are proven by entomologists to be **cost-ineffective.** This explains why farmers and landowners do not deploy such solutions to every single tree they own or can save.

A drone, computer vision, a shooting mechanism, and blue paintball pellets are utilized in this Final Year Project to deliver an autonomous, highly accurate, and cost-effective Pine tree treatments. This paper provides a literature review about the moth's cycle, the solutions that have been used to treat the Pine trees, and the mechanisms used throughout this project. It also presents two iterations of the proposed solution methodologies along with their pivots and corresponding results.

II. ACKNOWLEDGEMENTS

The FYP team would like to acknowledge the noticeable efforts done by Prof. Naseem Daher, Prof. Nabil Nemer, Mr. Ali Kanso, Mr. Mazen Hani, and AUB staff who ensured that the team is set on the right track since initiating the project by giving prosperous advice, insights, and offering the necessary tools.

Table of Contents

III. INTRODUCTION

A. Motivation

Environments are being driven away from their equilibrium because of Global warming. Global warming, a phenomenon demonstrated by climate change, not only makes trees more receptive to viruses, bacteria, water stress, etc., but also causes the population of various harmful parasites to exponentially increase. While some parasites are environmental equilibrium stabilizers, most are equilibrium destabilizers because of their behavior that mainly disturbs the natural growth cycle of trees, animals, and other species. Pine trees are victims of such equilibrium destabilizers.

Global warming leads to the growth in volume of two detrimental parasites affiliated to pine trees: namely, (1) The pine processionary moth and (2) Leptoglossus occidentalis. The pine processionary moth destabilizes the equilibrium of Pine trees as newborns feed on the pine tree needles which deteriorates the health of the pine tree. Unfortunately, the pine processionary moth has the ability to kill pine trees whose age is less than 8 years in three months as this span represents the time needed for the moth to develop from an egg to a mature caterpillar (fifth instar caterpillar - last caterpillar stage before growing into a flying moth). On the other hand, while the pine processionary moth feeds on pine needles, Leptoglossus feeds by sucking the seeds of the pine units on trees. The negative effect that both the pine processionary moth and Leptoglossus have on the pine tree could be mainly demonstrated by the disruption caused in the worldwide pine market and the danger on human beings (i.e., hair of pine processionary moth that causes allergy and could lead to death - quality of pine units whose seeds are almost fully sucked by Leptoglossus).

The entomology problems caused by the pine processionary moths have been long lasting since 1998. Ever since, entomologists were unsuccessful in developing a radically innovative solution that utilizes advanced technologies and entomology hacks to limit their growth. However, in reference to the fact that imprecise spraying on trees was prohibited worldwide because of the green environmental regulations that unions are pushing for, the solutions present in the market utilize mainly manual precision spraying, pheromone traps, and bioinsecticides (i.e., Entomopathogenic Fungus) to kill the caterpillar using a predictive approach that forecasts the activity of the caterpillar, trap adult/caterpillar pine processionary moths, and kill the hibernate eggs at an maximum efficiency of 50%. The solutions discussed for each destabilizing parasite proved to be unsuccessful in limiting growth which explains the growth in the local penetration of the Pine Processionary Moth to peak at a rate of 60%.

To validate the importance of our primary target, Pine trees, interviews were conducted with pine importers, exporters, owners of privately-owned lands, and local researchers. The interviews justified the purpose of this project. An overview of the interviews done is under section 12 (XI), Appendix, subsection D. Moreover, utilizing drones to assist in entomologists was the outcome of a conference in Spain on June 28, 2018.

Constrained to the behavior posed by the natural processes of the parasites, the project's scope is defined to target the complete growth cycle of the Pine Processionary Moth. Accordingly, the FYP submission was segmented into two stages to demonstrate and test the two proposed solution methodologies.

B. Needs

After testing, the reliable solution is anticipated to deliver the following:

- Assist Entomologists in identifying the location of caterpillar of the moth faster for efficient treatment
- Save the environment by killing the caterpillar using the better solution which is expected to be of more than 90% efficiency
- Improve Agricultural Research through establishing an interface between the solution and current entomology hacks
- Generalize this interface to tackle more problems
- Assist in stabilizing the environmental equilibrium

The following sections will present the requirements and deliverables, constraints, a literature review, the proposed solution methodologies, results, and the timeline which was followed during the academic year.

IV. REQUIREMENTS AND DELIVERABLES

The drone's linear movements are controlled along the six degrees of freedom consisting of moving forward, backward, left, right, up, and down. Using Python code, rotational movements are sent to the drone wirelessly. Rotational movements comprise roll, pitch, yaw, and vertical thrust movements. This combination of rotational movements translates to linear commands. This section includes the requirements, specifications, and deliverables that the FYP team will adhere to and deliver by the end of the semester.

A. Requirements:

i. The drone shall search for the nest continuously while navigating around the tree according to the predefined path planning algorithm.

ii. The setup weight for the shooting mechanism shall not exceed 300 grams.

iii. The detection algorithm shall be of high accuracy (>=90%) and fast (less than 1 second max).

iv. The shooting mechanism shall be of high accuracy (>=90%) and is active only when the distance to the nest is less than 2 meters.

v. The drone shall have a high-power GPU capable of processing images efficiently (NVIDIA GTX/RTX).

vi. The diameter of the chemical pellet shall be 6mm to fit the shooting setup.

vii. The drone shall approach the nest with a reasonable speed (not exceeding 1 m/s) as a precautionary measure.

viii. The images retrieved from the drone shall be of high quality (HD) for better image processing (856 x 480 pixels).

B. Specifications:

i. The setup of the shooting mechanism shall be mounted on top of the drone with a total weight of ~ 130 grams.

ii. When the drone doesn't find a nest, it utilizes a predefined path plan so that it continues executing its detection algorithm continuously while moving around to optimize the flight efficiency.

iii. The speed of frames taken shall set to 1fps so as to retrieve good quality images and more precise results.

iv. The movement of the drone will be in increments of position (~ 0.5 meters per seconds) predefined according to the path plan.

v. The drone will center the detected nest object inside the screen to calculate the position vector to move accordingly. The nest will be in the range (in pixels): 140-340 for the x component and 328-528 for the y component).

vi. A lidar sensor is mounted on the drone to calculate the distance to the nest in the range of 2-2.5 meters.

- When the distance is below 1.5-2 meters, the shooting mechanism is activated.

C. Deliverables:

i. An intelligent drone capable of:

- Flying autonomously with the aid of computer vision
- Detecting the Pine Processionary Moth nests with a high accuracy (90-95%).
- Scanning the area around the tree continuously while flying (horizontally and vertically).
- Approaching the nest with a reasonable speed ~ 1 m/s
- Shooting the nest with the paintball pellets at a distance of 1.5m far from the nest at a high accuracy (90%)

ii. An updated neural network weights file trained specifically on this nest affordable to any party interested in this research.

V. TECHNICAL AND NON-TECHNICAL CONSTRAINTS

A set of technical and non-technical constraints had been predicted and formulated ahead of delivering the system according to the desired requirements and specifications.

A. Technical Constraints

- ➢ The vision algorithm can be trapped by rocks, space, heads, and whiteboards
- ➢ The vertical speed of the drone cannot exceed 6 m/s
- ➢ The horizontal speed of the drone cannot exceed 16 m/s
- ➢ The vision algorithm cannot realize the nest if the camera is more than 3 meters away
- ➢ The drone has a flight limit of 25 minutes
- ➢ The drone carrying ability is limited to a weight of 300 grams
- ➢ The robotic arm must be long (50 and 70 cm) to reach the nests that are inside
- ➢ The diameter of the needle should be at least 4 cm
- ➢ The paintball pellets might splash inside a shooting mechanism
- ➢ The radius of the shooting mechanism is 8 mm
  - o The radius of the pellets should be less than 8 mm
- ➢ The shooting range is limited to 5 meters
- ➢ The drone doesn't have night vision
- ➢ The sensors of the drone are not accessible for reading
- ➢ The drone cannot approach the edge of the tree

B. Non-technical Constraints

- The weather conditions (rain, wind, storms)
- The presence of birds and other insects
- The regulations put by the Lebanese Army to limit drone flights
- The cycle in which the Pine Processionary Moth is on Pine trees occurs between mid-December and early April

VI. Literature Review

Ever since the discovery of the Pine Processionary Moth, joint agreements and partnerships between various agricultural researchers and research institutes have been set to limit the growth of the parasites, preventing it from spreading and relocating to different areas. Many entomology hacks that entomologists use today to tackle the growth of the parasites, which is increasing due to global warming, go back to the discoveries of such joint efforts. Joint partnerships between Entomologists and Engineers have developed both the qualitative and the quantitative analyses in all agricultural applications.

The species of the pine processionary moth which exists in Lebanon favors wild pine over stone pine because wild pine dominates the Lebanese pine tree species. The most important characteristic associated with the Pine Processionary Moth is its growth cycle. The critical segment of its life-cycle occurs between early Fall and winter as the parasite hibernates during Spring and Summer. The pine processionary moth lays its eggs between mid-August to early September. The female moth stacks up to 300 eggs on two combined needles diversified across 6 rows. The eggs start to hatch starting mid-October until early November depending on the relative altitude of the place of interest with respect to the sea. The higher the place is, the faster is the process. After the eggs hatch, at least one nest must be formed by the first instar caterpillar. First instar caterpillar feed on the roots of the needles as the roots are the smoothest part of the needle. Also, first instar caterpillar cannot tolerate any precise spraying because of their weakness as their body still lacks natural defense systems (i.e., hair, etc.). As the caterpillars grow, they start eating a bigger portion of the needle. They may change the location of their nests or rebuild a well-established nest depending on the environmental and weather conditions that accompany their growth phase before a severe winter kicks in. The caterpillars are then expected to get into their hibernation phase when they turn into pupa in soil. The hibernation period spans on average the time-period between early March and mid-August noting that different caterpillar of the same nest might get into the hibernation phase at different instances.

Three solutions have existed to treat the pine processionary moth namely precision spraying, pheromone traps, and bioinsecticides. The only effective phase during which precision spraying could turn out to be effective with high efficiency rates is that when the caterpillars are still in their first and second instar period. The issue with this period is the complexity of using vision to detect the caterpillar. Apparently, during this period, the nests are not as established as that of the 3rd instar nests. Nests in this period can be represented by few silk lines that serve as a flag for the new caterpillar. Thus, with their green color, an ambiguity could be developed between their color and the color of the needles, the color of the branches, etc. Consequently, precision spraying was not of very high standards. Moreover, pheromone traps were proposed to eliminate the pine processionary moth. The application is directed primarily to adults (parasites that fly) while the structure could be demonstrated by the fly catcher through which a fly enters and gets stuck in a plastic bag. Those pheromone traps were able to limit the growth of the parasites; however, such a solution was reported by entomologists as an unsustainable solution for the long run. Finally, bioinsecticides are represented by biological fungus named entomopathogenic fungus and Diflubenzuron. The fungus is sprayed over the soil during the hibernation stage as it resides in an 50% efficiency rate. Diflubenzuron proved to be ineffective by entomologists its constituents used to cause harm for the caterpillars in the past before the moth developed immunity.

The solution that this FYP is developing consists of the following:

- Drone
- Hardware
  - Robotic Arm
  - Shooting mechanism
- Vision

Therefore, knowing that the drone utilized is Bebop 2.0 by Parrot, the team is set to utilize the Parrot controller to achieve the first milestone within time constraints. However, at a later stage, the team aims to develop a controller similar to that which will be presented in the next subsection.

The robotic arm is said to be static because of the limitations on the total weight the Bebop 2.0 can sustain. However, the robotic arm's subsection will present an overview about modeling using a kinematic chain. Following, two subsections will showcase a literature review for both the shooting mechanism and Computer Vision.

A. Vision and Detection

This section discusses the work done and the different algorithms previously applied for computer vision.

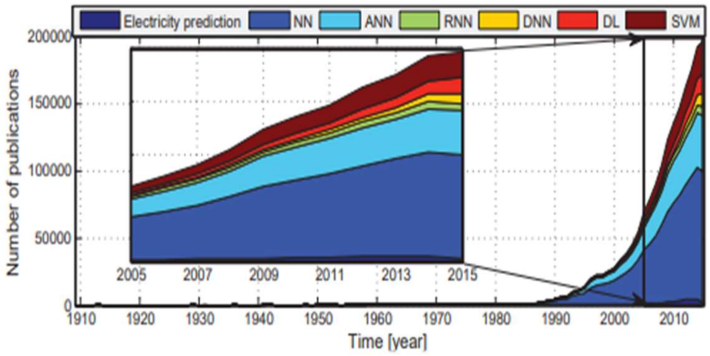


Figure 1: Number of publications versus Time in years

Object detection algorithms can be classified into two groups, deep learning and feature extraction methods. Starting by the feature extraction algorithms, these algorithms use a bunch of selected features while examining the sample for reaching a decision. For example, K. Sung [5] used a Gaussian model classifier for face detection applications. He presented an image-based feature and pattern-detection technique for finding human faces. The technique is divided into several steps including identifying canonical face manifolds, i.e., ellipse shaped regions, image pre-processing, matching features to the model, and applying normalization and optimization methods to obtain high accuracy classifier.

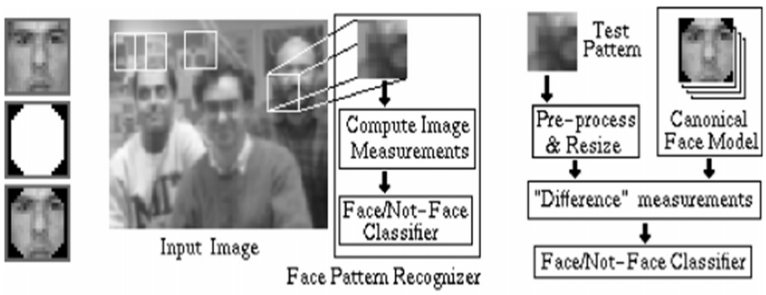


Figure 2: Face Recognition

A similar research was done by a group of researchers headed by professor Vondrick [6]. The algorithm they came up with visualized feature spaces used by object detectors. Their method works by inverting a visual feature back to multiple natural images. This way they can gain a more intuitive understanding of recognition systems.

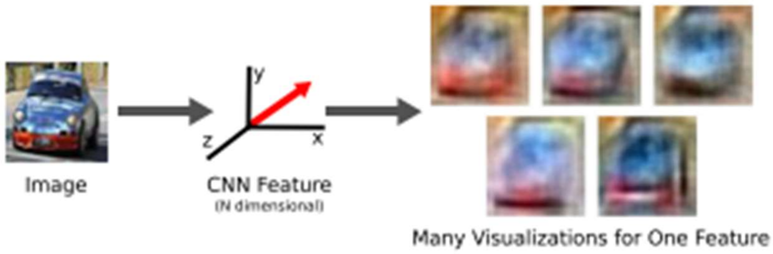


Figure 3: Visualizations of a single feature

Similarly, C. Papageorgiou et al. [7] have used the same wavelet feature in proposing a general detection framework and tested it on face recognition and on human body recognition.

However, the birth of deep learning algorithms was a breakthrough in the field of Artificial Intelligence. Deep learning algorithms use a huge number of labelled images (dataset) to training the algorithm on before going to the actual test on unlabeled images. Using deep learning methods, high levels of accuracy that were hard to achieve using conventional computer vision techniques are achieved. A study by N. Paterakis et al. [8], comparing deep learning methods such as Convolutional Neural Networks (CNN's) and Deep Boltzmann Machines (DBM's), with Support Vectors Machines (SVM's), Gaussian and Markov Processes and other traditional vision methods, showed a significant difference in the error probability density function.

Moreover, according to K. Nguyen et al. [9], classifiers such as SVMs and K-Nearest neighbors' algorithms are generic and not robust to the diversity of data and are challenged with non-linear complexity of the features. Convolutional Neural Network was firstly introduced by K. Fukushima [10] in 1980. CNN is inspired from the visual cortex of the human's brain, it consists of a set of several layers, each consisting of one or more units. CNN is a deep learning technique commonly used in computer vision for different applications, such as face recognition, scene labelling, image classification, action recognition, document analysis, and human pose estimation [11]. Neural Network based algorithms have widely spread lately due to its reliability and high accuracy in solving real world problems. For example, X. Li et al. [12] have used a two eight layered convolutional neural network algorithm in a coarse-to-fine manner for detecting inshore ships. Truong et al. [13] have proposed using a light-weighted deep convolutional Neural Network instead of heavy ones for detecting tiny objects, for this they tested their network on the CIFAR-10 and CIFAR-100 datasets. Moreover, S. Zia et al. [14] used deep CNNs on RGB-D images containing small household objects for the purpose of their recognition. Last but not least, Microsoft launched their own computer vision API back in 2016. The API returns information about visual content found in an image. Yet, Microsoft's algorithm wasn't able to extract the exact features and detect the nest.

Figure 4: Recognition of the nest by a Google algorithm

However, because of the large amount of data required by deep learning algorithms, transfer learning was invented. Transfer learning algorithms are considered to be a subcategory of the deep learning. But with the ability to use previously trained neural networks on a new set of data. This will help in reducing the time required for execution, in minimizing the computational costs, and in increasing accuracy levels. Some of the most popular transfer learning networks out there are GoogleNet, AlexNet, VGG, and YOLO. These networks are trained on large datasets containing millions of images such as the ImageNet, COCO, PASCAL, VOC, and others.

Another main topic in machine learning and a proposed solution for lacking datasets is data augmentation. A lot of research has been done in this area. For instance, researchers in EPFL created a novel algorithm for this purpose in 2016 [15]. Small transformations where sought to achieve high performance. Their algorithm brought down to a linear program and an optimization problem in which they solved it using linearization techniques and approximations. Another group of researchers in the National University of Ireland [16] proposed a new algorithm in 2017 for data augmentation, called Smart Augmentation. Smart augmentation is a new algorithm that works by creating a network that learns how to generate new augmented data during the training phase. It does so in a way that reduces network loss and minimizes errors in the network. Smart Augmentation attempts to address the issue of limited training data to improve regularization and reduce overfitting. The algorithm is based on 2 main networks: one that will generate data, and the other that will perform a desired task (such as classification).

B. Hardware

a. Robotic Arm

Robotic arms are modeled in the form of a series of rigid bodies which are connected by joints and constrained to a set of links, establishing serial kinematic chains. Thus, applying an inverse kinematics algorithm computes the motion references for the actuated variables based on the desired end-effector trajectory. Therefore, applying such algorithm and establishing an interface between vision and the arm, as shown in figure 1, would upgrade the system massively and will facilitate the motion of the drone in the vicinity of the nest preventing any crash with a bush or tree branch.

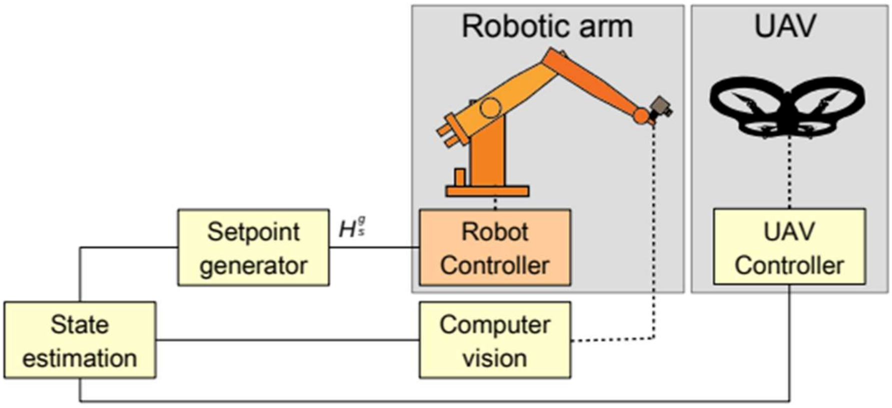


Figure 5: Connections between the UAV, Robotic arm, and Computer Vision

b. *Shooting Mechanism*

Shooting mechanisms vary from using springs to pneumatic principles and even rubber bands. Shooting mechanisms follow the concept of how conventional guns work. In the case of conventional guns, the shooting aspect is based on an explosion initiated at one end of the gun which expands to the other end of the gun, where the bullet exists. The explosion forces the bullet to traverse through the gun barrel which ends up shooting the bullet out of the gun [17]. In the case of shooting mechanisms, the concept is to create enough pressure to shoot a specific body in the required trajectory. Bullets, BB pellets, and even rocks are all considered as bodies used in shooting mechanisms. In the case of springs (or rubber bands), the spring would be held back and locked in position where elastic potential energy is stored ready to be released. When the trigger is pressed, the compression spring is released, and it would push a cylindrical part called the piston as shown in figure 1. Consequently, the piston hits the body that is being shot to exit the gun through the gun barrel. The type, size, and strength of springs or rubber bands used highly affects the trajectory and speed of the traveling bullet.

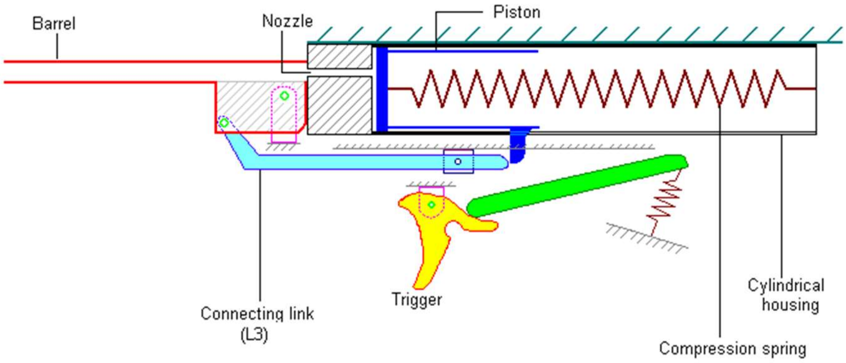


Figure 6: Schematic of the spring shooting mechanism

In the case of pneumatic based shooting mechanisms, air pressure is utilized to force the bullet out of the gun. A gas chamber is designed to contain compressed gas that is filled manually from the compressor as shown in light blue in figure 2. If the trigger is pulled, more air would enter the chamber causing the mass of the gas to increase. Due to having a constant volume in the gas chamber, increasing the mass of the gas forces the pressure to increase and causes the bullet to traverse outwards through the gun barrel [17].

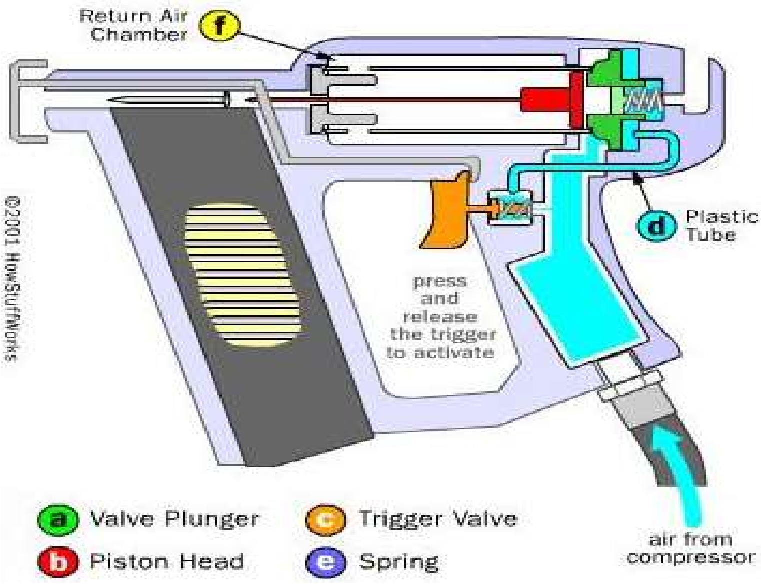


Figure 7: Schematic of the pneumatic shooting mechanism

C. Drone Control

The initial phase of the project is expected to utilize the control system embedded in the drone. However, for the second phase, the team could generate a linearized model of the system using system ID tools and thus, utilize PID controllers (Hoffman et al., 2007) or Quadratic Regulators. However, the team could apply nonlinear control through controllers such as model predictive control (Kim and Shim, 2003), backstepping and sliding mode techniques (Siegwart, 2005). Therefore, the team would utilize adaptive control. The application of the adaptive control is governed by a set of control laws proposed by (Fierro, 2011) to generate a mathematical model of the UAV system. Such mathematical model is set to establish a relation that exploits the linear dependency between position and the center of gravity. Such dependency is described by (Antonelli et. Al., 2013) as such description sized the effect of exogenous forces and their moments on the parameters of the of the center of gravity.

VII. PROPOSED SOLUTION METHODOLOGY

At the start, the idea of targeting the eggs of the Pine Processionary Moth (PPM onwards) as a way of eliminating them was promising since this would solve the problem by targeting the root cause. The aim was to target the eggs at a certain phase of their life cycle and hopefully decrease their number. It was challenging to target the eggs since it would require the drone to detect them using computer vision. The issue was that the size of the eggs were small reaching a size of 2-5 cm in width which made it very difficult for the vision to detect. In addition, their exists similarities in color between the eggs and the leaves of the tree which makes detecting them even more difficult. Another solution was introduced which focused on locating the eggs using thermal imaging. This solution was also ineffective due to the eggs being cold-blooded thus no difference in temperature can be detected between the eggs and their surroundings. The solution transformed to targeting the nest of the parasites instead of their eggs. This solution is specific to PPM in which a specific phase of their life cycle is targeted which is the nest phase.

Two solution methodologies have been proposed to eliminate the nests:

Solution A: Robotic Arm

The first hardware solution aims at eliminating the parasites by puncturing their nests. Cold temperatures is considered as a main weakness to the parasites, which forces them to build nests for protection against cold temperatures in winter time or at high altitudes. The aim is to equip the drone with a robotic arm that contains a sharp end capable of damaging the nest, decreasing the temperature inside due to the entry of cold air, and insure the death of the PPM in 2-4 hours.

Using this solution, a drone will be utilized to fly autonomously, scan Pine trees using computer vision, detect the parasitic nests, and puncture them using the robotic arm.

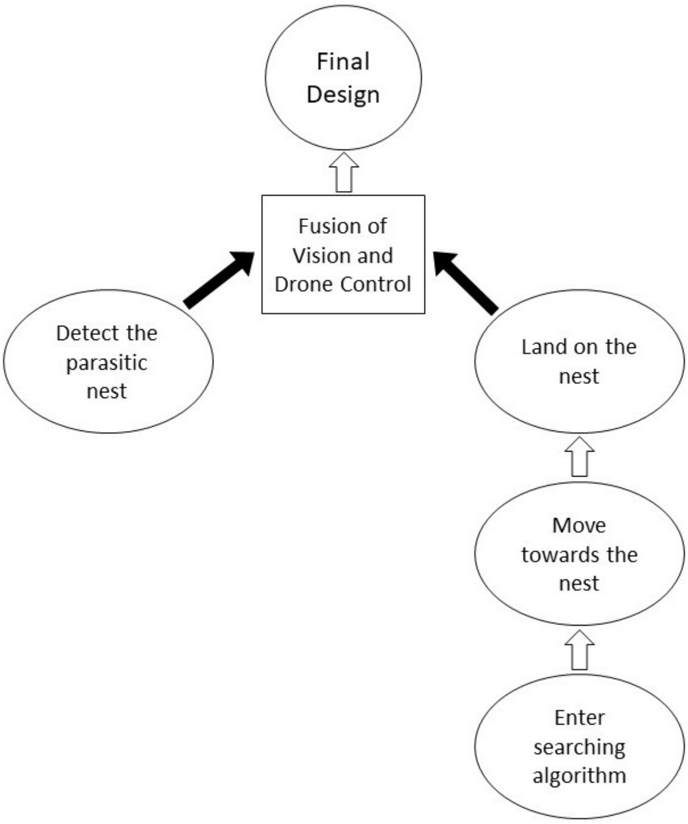


Figure 8: First flow diagram illustrating the overall view of the project

Solution B: Shooting Mechanism

The second hardware solution consists of a shooting mechanism mounted on top of the drone dedicated to shooting the nests with small chemical pellets. This solution utilizes the application of precision agriculture where each chemical pellet is filled with a pesticide called Spinosad. Only 0.01 grams of this chemical is sufficient to treat one Pine tree containing an average of 5 nests. After shooting the chemical pellet into the nest, the chemical disperses inside and kills all the parasites that contacts the chemical.

Using this solution, a drone will be utilized to fly autonomously and scan Pine trees using computer vision, detect the parasitic nests, and determine the distance away from each nest using a lidar sensor for distance measurement. After that, the drone centers itself in front of each nest and shoots the nest with chemical pellets.

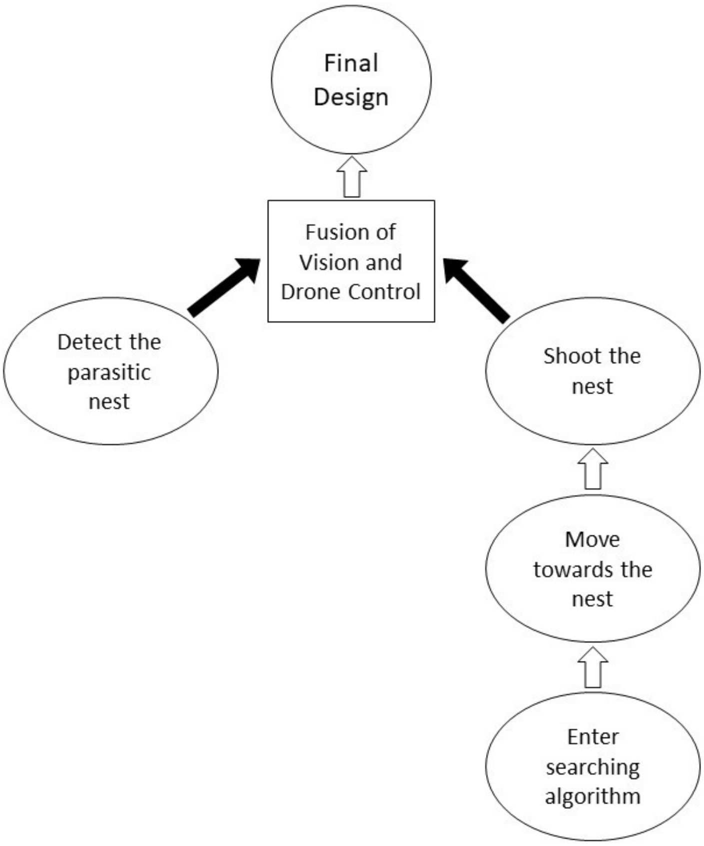


Figure 9: Second flow diagram illustrating the overall view of the project

VIII. Design

In reference to the methodology described above, the project is centered around three main components namely: Computer Vision, hardware, and the drone. The following three subsections will present the design alternatives, decisions, and iterations of each component. By the end of this section, the reader is expected to know how the following setup in the figure below was attained.

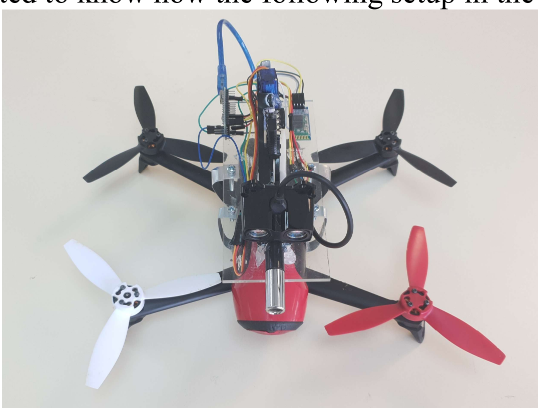


Figure 10: Image for the Setup on the Drone

A. Computer Vision

Design Alternative 1: Convolutional Neural Network

The drone will run based on deep network algorithm YOLO implemented on top on Darkflow in Python. This network is trained on millions of images from the Pascal VOC dataset over more than 20 classes (objects). Thus, given those initial weights and trained classes, the algorithm used will be extended to our newly defined object, the nest, so that we can transfer what it learned to the new object, better than making it learn from scratch. This is what is called “Transfer Learning”, in which we trained the initialized neural network on only the last 2 layers of the network.

The main issue of the detection problem is the dataset. A lot of work was done on this part, and the training had to pass through several iterations:

1. Online dataset:

A set of around 360 images was collected from different databases and forums over the internet. This set was divided into 260 images of training and 100 for testing.

The output of this dataset was able to localize the nest, but not with a high accuracy.

Figure 11: Online Images

Figure 12: Output images after training on pure online dataset

2. Artificial Nest:

Thus, to increase accuracy, we had to increase the size of our dataset. A cotton object was created similar to the shape of the nest. It was placed on trees and images of this object was taken from different perspectives, to obtain a new set of 200 images.


Figure 13: Artificial cotton nest

Nevertheless, the output of this dataset was nothing better than the previous iteration.

3. Augmented Dataset:

The next iteration was to augment our resulting dataset consisting of 560 images. Augmentation was done based on the following principles:

a. Geometric transformations:
    i. Rotation
    ii. Flipping
    iii. Changing shear range
    iv. Scaling
b. Changing lightning conditions
c. Adding noise:
    i. Gaussian noise
    ii. Poisson noise
    iii. Salt and pepper noise
    iv. Speckle noise


Figure 14: Original Image


Figure 15: Salt and Pepper Noise


Figure 16: Rotation and light conditions


Figure 17: Flipping

The final dataset was around 1500 images. Even though it seemed to be a large and good dataset, the bias and different shapes of the nest inside this dataset played against our purposes, in addition to some redundancy in images, which led us to continue trying to get better images.

4. Synthesized Nest:

Prior to our final step in this sequence of iterations, synthesizing images was a key idea to try to upgrade our dataset. An online software was used to create and synthesize new artificial images. A total of around 400 images was gained from this procedure.


Figure 18: Synthesized artificial nests

5. Real Nests:

Finally, after all collecting all this dataset, real nests had started to grow on the trees. Thus, our next step was to play on those real nests. A big survey was done down in Aalay and Choueifet to collect as much real and "useful" images as possible.

A total of around 300 good quality images was taken for this real nest.

Figure 19: Collection of real images

6. EVERYTHING PUT IN PERSPECTIVE:

After collecting this entire dataset, it was formed of about 2000 images. This dataset was dividing into 2 parts:

a. 1500 images for training
b. 500 images for testing

Nevertheless, this dataset had some problems. There was a big amount of redundancy inside it, as well as images that deteriorate rather than help in training on the nest. Thus, the dataset had to be adjusted and compressed down to ~ 1500 images, again, 1000 images for training (2/3 of the total) and 500 for testing (1/3).

Thus, we are left with 1000 images to train on, collected from different places and resources, in addition to 500 images to test on.

Labelling the images was done via the software Labelimg.

Figure 20: Examples of labels done of the resultant images

The labels of the images are outputted as XML files.
Those XML files are sent to the neural networks to be read accordingly.

```xml
<annotation>
  <folder>images</folder>
  <filename>36</filename>
  <path>C:/ProgramData/Anaconda3/darkflow/labelsPrime/images/36.jpg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>520</width>
    <height>390</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>nest</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>162</xmin>
      <ymin>73</ymin>
      <xmax>370</xmax>
      <ymax>277</ymax>
    </bndbox>
  </object>
</annotation>
```


Figure 21: Example of the output xml file to be inputted to the neural network training phase

Design Alternative 2: Feature Extraction

The drone will run based on a conventional computer vision algorithm, object detection, corner and edge detections. The algorithm will be based on extracting the features of the object, generating augmented features that are used for training, and then validating and testing on the input space images.

Canny detector example:

```
import cv2
import numpy as np
import os

imgOriginal = cv2.imread("X.png")

if imgOriginal is None:
        print ("error: image not read from file \n\n")
        os.system("pause")
        return
end if

imgGrayscale = cv2.cvtColor(imgOriginal, cv2.COLOR_BGR2GRAY)

imgBlurred = cv2.GaussianBlur(imgGrayscale, (5, 5), 0)

imgCanny = cv2.Canny(imgBlurred, 100, 200)

cv2.namedWindow("imgOriginal", cv2.WINDOW_AUTOSIZE)
cv2.namedWindow("imgCanny", cv2.WINDOW_AUTOSIZE)

cv2.imshow("imgOriginal", imgOriginal)
cv2.imshow("imgCanny", imgCanny)

cv2.waitKey()

cv2.destroyAllWindows()

return
```

Figure 22: Canny detection code

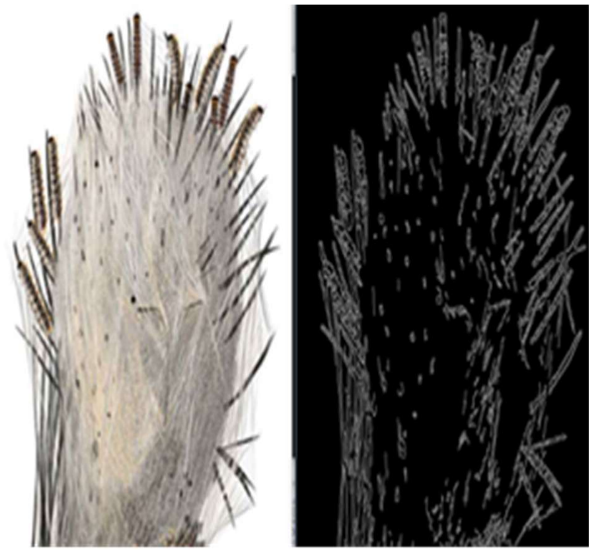


Figure 23: Canny detection result on the nest

SURF detector:

```
clear;
nest = rgb2gray(imread('nest3.jpg'));
figure; imshow(nest);
title('Image of a nest');

nestPoints = detectSURFFeatures(nest);
figure; imshow(nest); hold on;
plot(selectStrongest(nestPoints, 100));
title('100 Strongest Feature Points from Nest Image');
```

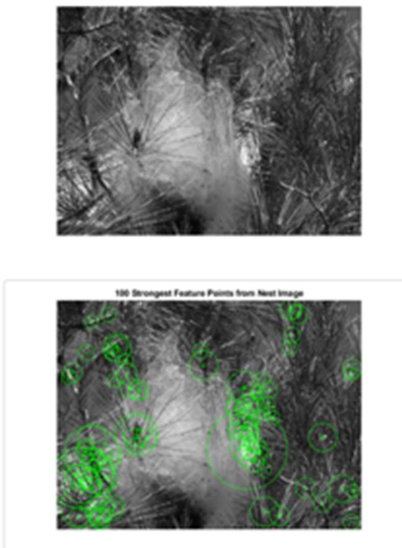


Figure 24: Detection code using SURF features

B. Hardware

Design Alternative 1: Robotic Arm

a. Preliminary Design Alternatives

The function of the robotic arm is to ensure a punctured hole in the nest with a width of at least 2 cm. Two design alternatives have been discussed, where one design is claimed to have a higher efficiency than the other. The most important aspect in the robotic arm is its weight. The drone must not carry weights more than 300 grams, so the design of the arm should respect this technical constraint.

The total system of the robotic arm consists of three main parts: joint, connector, and arm.

The joint is designed to be connected to the bottom of the drone using two locks. It is designed to have two holes with exact dimensions as the holes present at the bottom side of the Bebop (0.4cm x 1cm). Each lock enters one hole of the joint and one hole of the drone, then the two locks are rotated to lock everything in place.

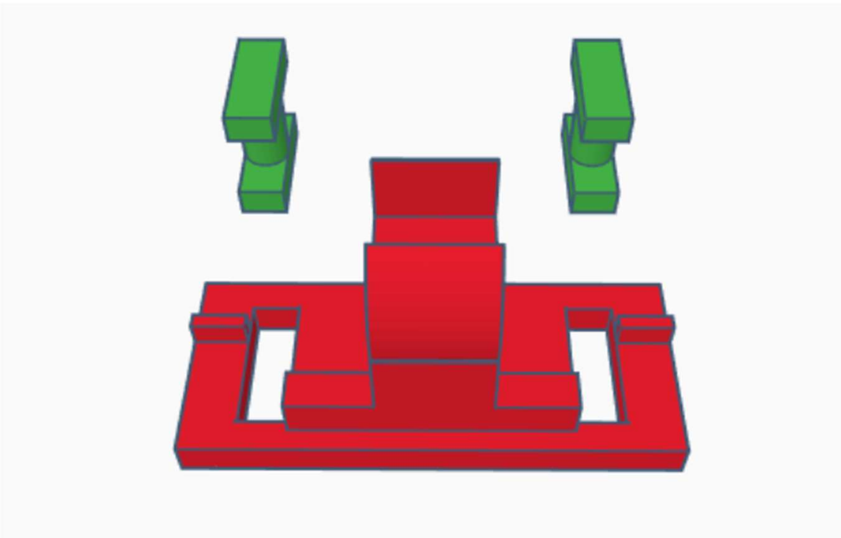


Figure 25: Joint part and locks

In addition, the joint contains a hole of width 2mm so that it is connects to the connector part using a pin that handles high shear forces. The joint also contains two edges at its top to create space for the connector lock to fill. The joint part has dimensions of 3.5cm x 1.6cm x 19.57cm and has a mass of 4g.

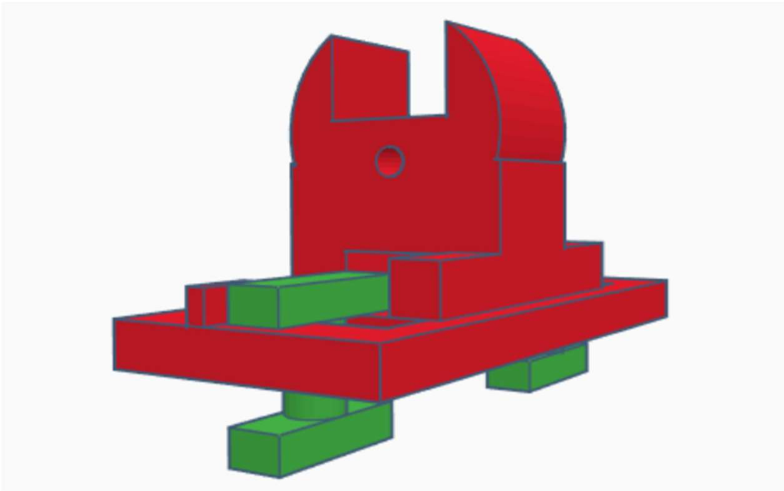


Figure 26: Joint part side view

The connector part contains two main features in its design. It has two symmetric parts at the top with the same 2mm hole in each. A pin will be used to connect the joint part with the connector part using the holes; this gives the ability for the connector to swing around the y-axis of the joint part.

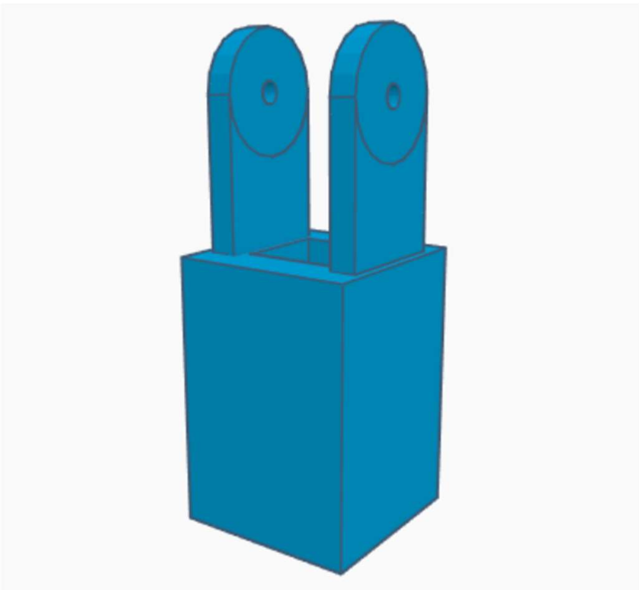


Figure 27: Connector part

Moreover, the second feature is the box shaped hole in the body of the connector part. A spring is placed inside the hole and a lock is place on top of the spring. When the spring is pressed the lock will go downward into the hole and vice versa.

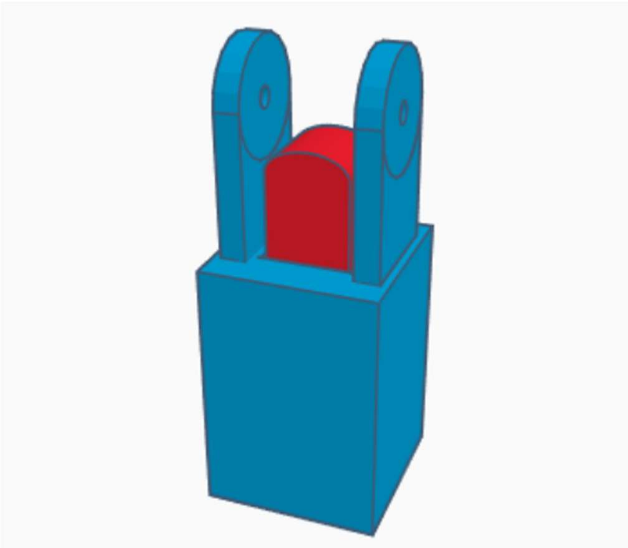


Figure 28: Connector part with lock inside

Once the connector part and the joint part are connected using the pin, the spring will be loaded as well as the lock. When the connector swings forward and the lock is placed directly under the space created on top of the joint part, the spring will unload itself and push the lock into the space created on the joint. This creates a locking mechanism that will hold everything in its place. The connector part has dimensions of 1.5cm x 1.5cm x 3.96 cm and a mass of 4g.

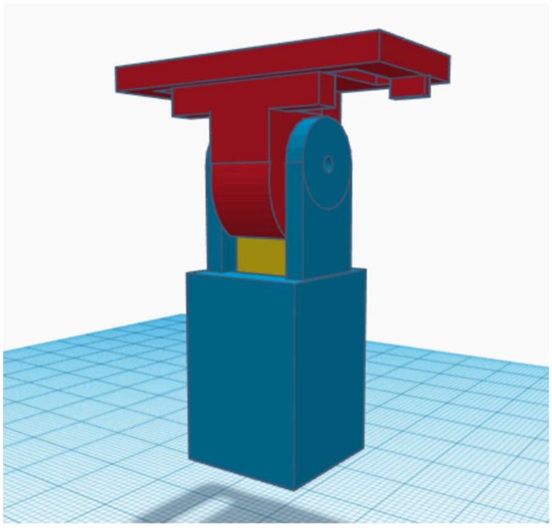


Figure 29: Locking mechanism final position

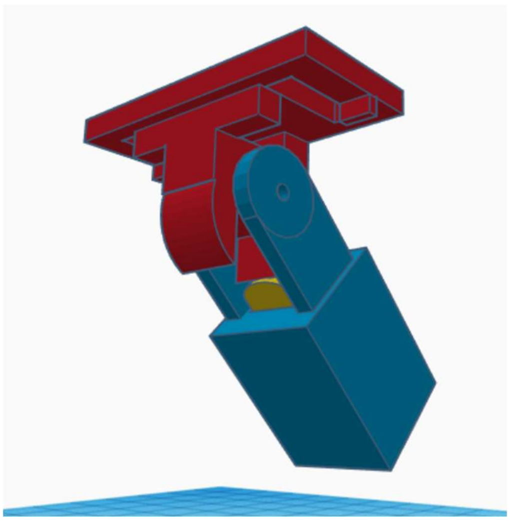


Figure 30: Locking mechanism midway position

The arm will be connected to the bottom of the connector part and form one piece. Whenever the connecter part swings from the joint part, the whole arm swings as well. The locking mechanism will ensure a 180 degrees angle for the arm with respect to the drone, as well as ensuring a robust design capable of puncturing the nest without shaking.

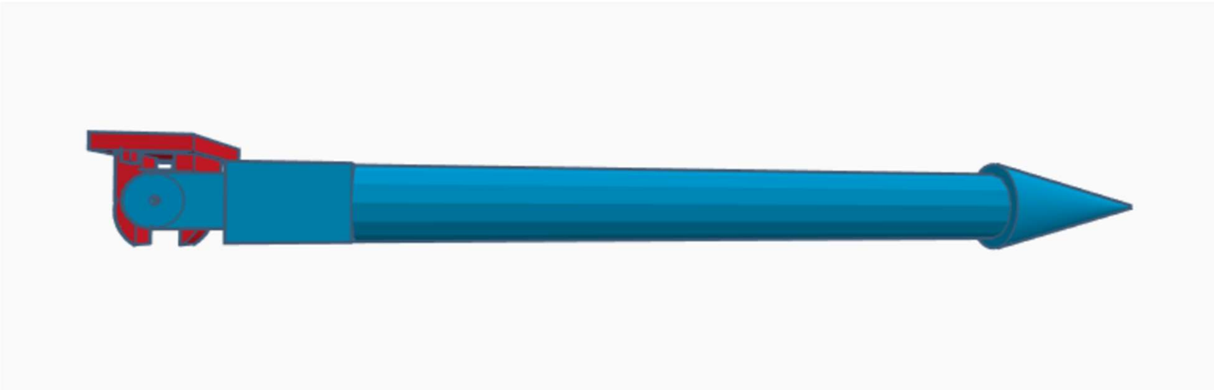


Figure 31: Total robotic arm system 90 degrees

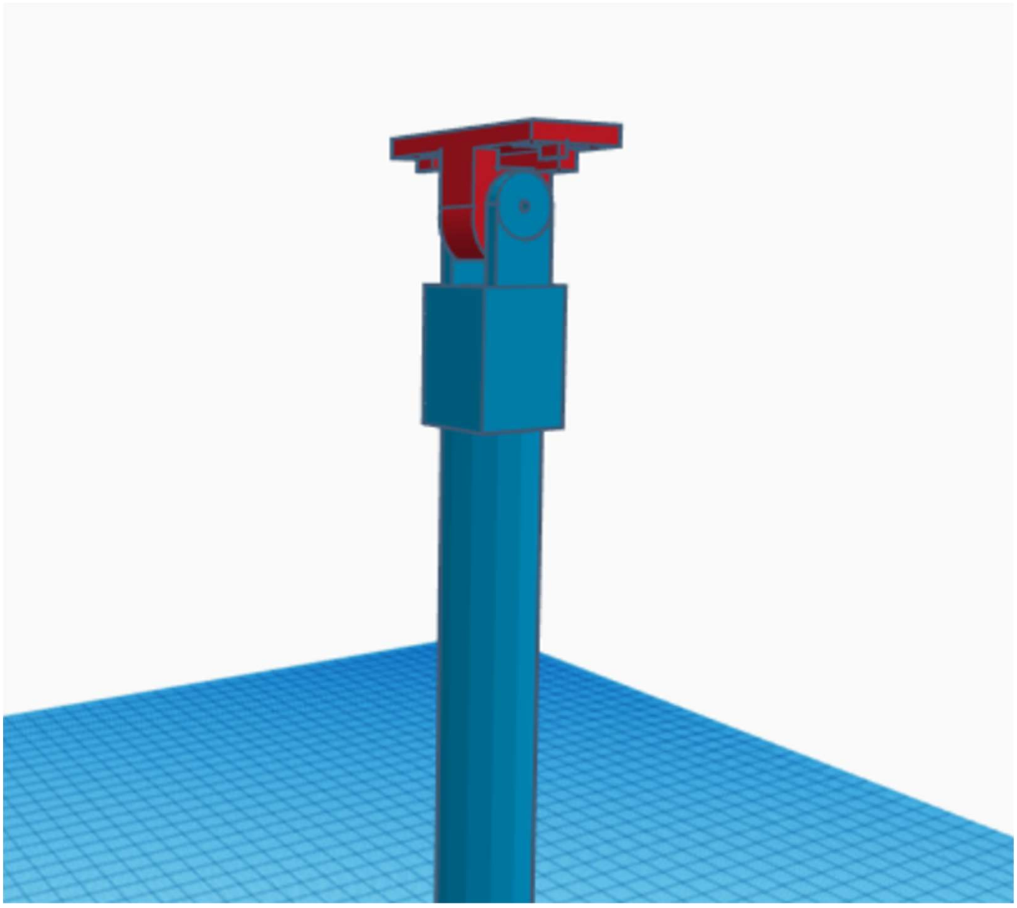


Figure 32: Total robotic arm system 180 degrees (locking position)

The arm will be either 3D printed from PLA material, or it will be a PVC tube with the required dimensions. The first choice leads to a mass of 50g while the latter leads to a mass of 100g. Both choices are within the limit of 300g. The design of the needle was with respect to 3D printing material and has a mass of 12g with dimensions of 4cm x 4cm x 6cm. Two designs for the arm were discussed which are:

i. Design Alternative 1

The first design of the robotic arm consists of two parts connected to each other. The first part is a cylindrical rod of radius 0.75 cm and height 60 cm. The radius is chosen to be small to decrease the weight of the arm. The height is set to be quite large so that the drone does not hit the branches of the tree while puncturing the nest with the arm. The second part consists of a cone shaped needle at the end of the arm. It has a height of 5 cm and base diameter of 4 cm thus ensuring a puncture of width 4 cm.

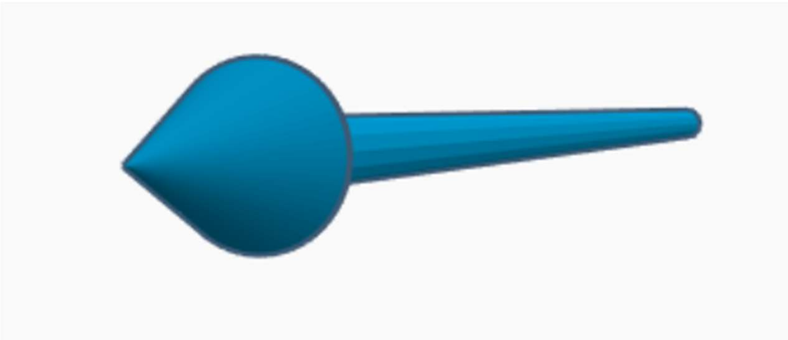


Figure 33: Robotic Arm Design 1

ii. Design Alternative 2

The second design alternative also consists of two parts. The first part is the same of that of design 1; cylindrical rod with radius 0.75 cm and height 60 cm. However, the second part consists of 4 cone shaped needles instead of 1. All needles have the same height of 5 cm, where two needles have a base diameter of 2 cm each and two other needles have a base diameter of 1.25 cm each.

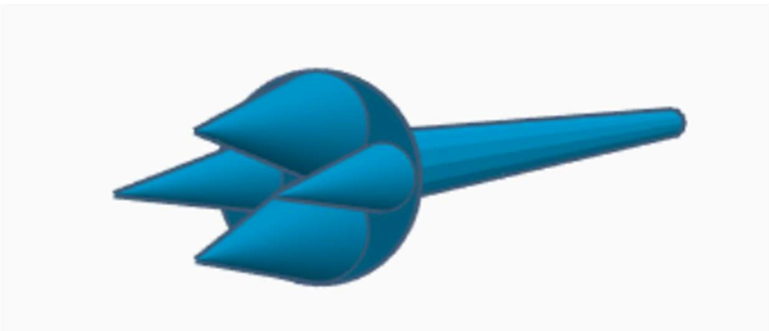


Figure 34: Robotic Arm Design 2

The first 2 needles will ensure 2 punctures of width 2 cm each, while the other 2 needles will ensure 2 punctures of width 1.25 cm each. Due to having the needles close to each other, the overall puncture will have a width greater than 4 cm, around 6 cm.

Design Alternative 2: Shooting Mechanism

The shooting mechanism was one of the proposed solutions to target the nests other than the robotic arm. The idea is to have a spring shooting mechanism to be mounted on top of the drone. The spring will be loaded with a chemical pellet before takeoff, where the command for shooting is automated and performed by the drone when needed.

a. Design

A BB gun was disassembled, and the spring shooting mechanism was extracted to be utilized.

Figure 35: gun

Figure 36: gun internals

A servo motor, capable of rotating 180 degrees, was added to the system to make the shooting automated and on command. The servo motor was glued to the lower part of the system as shown in the figure below. In order to make the shooting command controlled by the drone, an HC-05 Bluetooth module was added to the system so that the shooting command would be sent from the drone to the shooting mechanism via Bluetooth anytime needed.

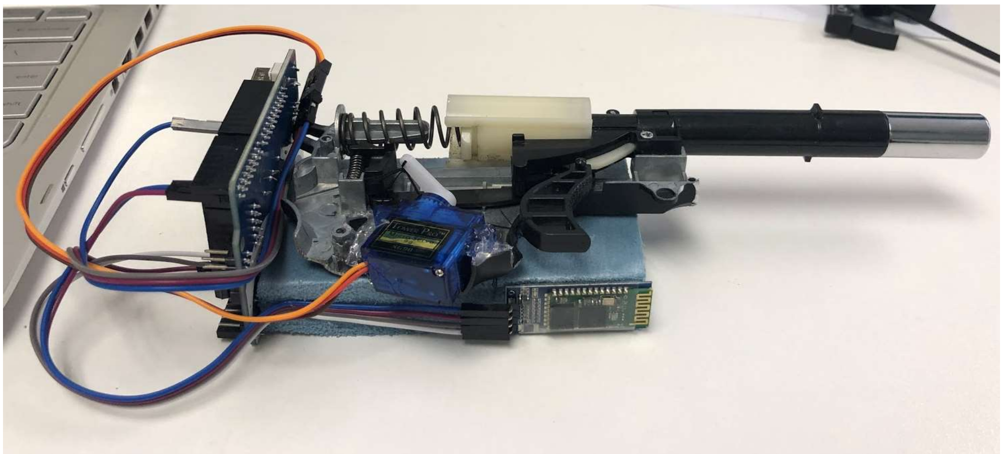


Figure 37: First Setup

Moreover, an Arduino microcontroller was added to the system to power both the servo motor and the Bluetooth module, and to establish the communication between the two components as shown in the figure below.

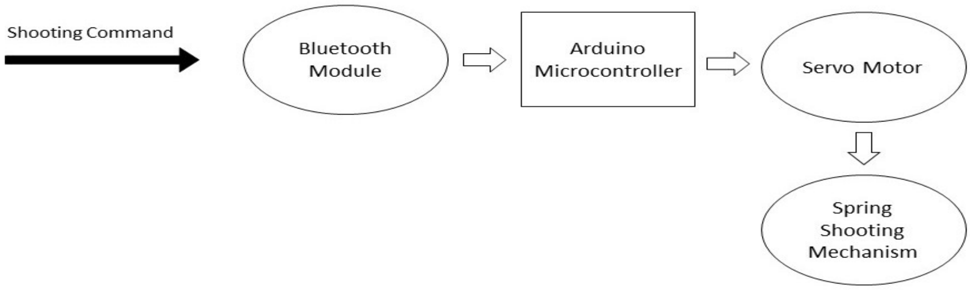


Figure 38: Flow diagram of the project

The servo motor is linked to the trigger using a string. Whenever the spring is loaded, the motor rotates from the 0 degrees position to the 90 degrees position. This creates enough force to pull the trigger and release the spring. Once the mechanism finishes shooting, the motor rotates back to its initial position at 0 degrees.

b. Chemical Pellets

As described in the solution methodology, chemical pellets will be used along with the shooting mechanism to deliver the required treatment to the trees. Each pellet should not exceed a diameter of 6 mm to fit inside the shooting mechanism. Every time the drone needs to start its regular procedure of treating the trees, the spring in the shooting mechanism is held back and locked and the chemical pellet is loaded inside. Once the spring is released, the pellet exits the mechanism through the gun barrel and towards the nest.

For now, blue paintball pellets are used instead of actual chemical pellets to visualize the blue color on the nests.

Figure 39: Paintball pellets

c. Mounting

The entire set up for the shooting mechanism is attached to a plexi platform and mounted on top of the drone using four pieces of aluminum as shown in the figure below. In the figure, it is shown that a lidar sensor is added to the setup on the front and hooked up to the Arduino for distance measurement.

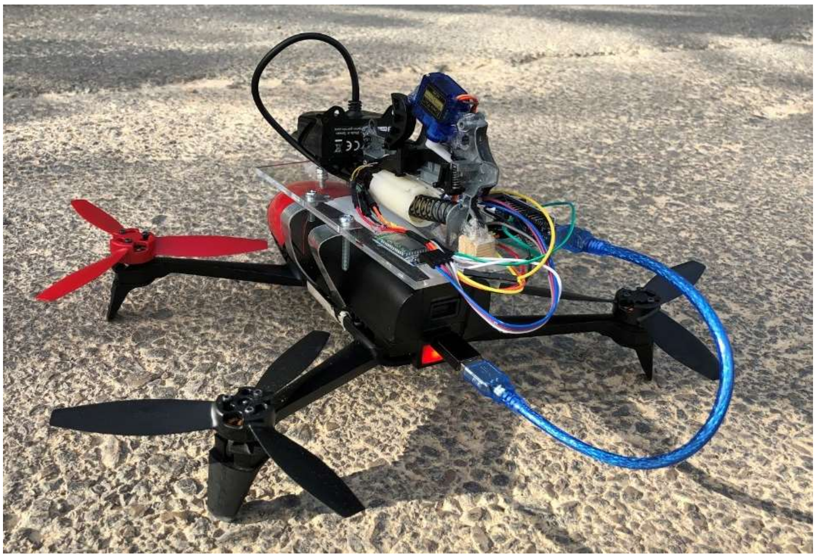


Figure 40: Tree-D

C. The Drone

A Parrot drone was outsourced during the span of the project. The main application of the drone was to navigate between trees, look for target parasite formations, and put precision agriculture into action through a new approach - precision chemical spraying. By the end of this section, the reader is expected to have an answer to the following questions:

- What is the type of the drone used? How powerful is it?
- What does the drone use to navigate in a forest?
- How is the distance from the target nest determined?
- What is the searching algorithm used?
- Is the searching algorithm hardware specific?
- How to cater the needs of each hardware design upon landing?

The following 4 main subsections will present the main alternatives and decisions encountered during the span of the project:

The Drone: Platform Alternatives and Decision

Ms. Sara Khaddaj, AUB MSFEA Lab Manager, allowed the team to choose between two different platforms that AUB own. The two platforms are:

i. AR 2.0 Parrot Drone

   The AR 2.0 Parrot Drone includes a 720p, 30 frames per second HD and wide-angle lens camera, and is capable of a 12 minutes flight duration because of its 1000mAh small battery. The Drone is distinguished by an ARM Cortex A8 1 GHz 32-bit processor with DSP, a linux 2.6.32 operating system, and a set of Gyroscope, Accelerometer, Magnetometer, Pressure, and Ultrasound sensors.

ii. Bebop 2.0 Parrot Drone

   The Bebop 2.0 Parrot drone includes a 14 Mega-pixels, fish-eye lens, camera which records 3-axis full HD 1080p videos. The Drone has a 2700 mAh battery which allows for a 25 minutes flight duration. The Drone’s dual core processor with quad-core GPU and 8 GB flash storage system ensure a reliable interface with computer vision. The Drone is capable of carrying any object weighing under 300 grams. The drone has also a set of Gyroscope, Accelerometer, Magnetometer, Pressure, Global Positioning, and Ultrasound sensors.

For the purposes of our application, Drones in Agriculture, long flight duration along with a high resolution camera are two of our main Key Performance Indicators. Therefore, the latter platform, Bebop 2.0 Parrot Drone was chosen.

The Drone: Navigation Alternatives and Decision

To navigate in the forest, the team studied the feasibility of deploying any of the following three alternatives:

i. Computer Vision Navigation

As mentioned earlier, the Bebop 2.0 Parrot Drone has an embedded camera. Thus, to navigate in a forest, the team would need to train a Neural Network that could realize objects. With this model, coding an object avoidance algorithm would become possible noting that the Bebop 2.0 Drone does not have an embedded object avoidance algorithm. Object avoidance could alternatively be embedded using 2D Lidars. However, in a forest, this might lead to a disaster if the Lidar happened to target the space and not a close object.

ii. Path Planning: GPS

As mentioned earlier, the Bebop 2.0 Parrot Drone has Global Positioning Sensors - GPS. Those sensors are capable of localizing the drone's geographical position with an error margin that could reach up to 5 meters. Therefore, surveying forests with 95% accuracy would be put at risk.

iii. 3D Lidar

Restricted to both, (1) the Drone's capability to carry weight and (2) the fact that hardware needs to be mounted on the Drone with an average weight of 200 grams, mounting a 3D Lidar on the setup was not possible.

Given the constraints of the project, Computer Vision Navigation was deployed to navigate in the forest. However, the model was not trained on every single object it might face, given the variety of items that the drones can see in a given forest. The 3D Lidar would have been an optimal choice if AUB Labs had a more powerful Drone.

The Drone: Distance Calculation Alternatives and Decision

Measuring the distance from the nest is at the core of the delivering the treatment. The following two solutions would present the two ways that the team could utilize to measure the distance from the nest.

i. Computer Vision Calculation

To measure the distance from an object via computer vision, a neural network algorithm would be needed to map the detected object to a specific distance. What does that mean? This means that given the detection box, as in figure 58, the percentage of the object's pixel span to the total pixels over x and y implies a specific distance.

Nevertheless, the triangle similarity method can be used utilizing vision. This method estimates the distance based on three key parameters namely: Focal Length, Average width of the object, and the Perceived width measured by calculating the relative horizontal separation between the right and the left edge of the marker (i.e., rectangle), which implies D = (Focal length * Perceived Width)/Width. This alternative calculates the focal length given a known distance. The focal length is assumed to be uniform for all other trials. Therefore, the estimated value diverges from the true value because given a constant Focal Length and object width, the change in Distance is not necessarily equal to that of the perceived width.

ii. 2D Lidar

A light based 2D Lidar could be mounted on the Drone. If the Lidar has a proper orientation (i.e., directed towards the nest when the drone is between 1.5 - 2 meters away from the nest). The lidar measurements then guide the drone when approaching the nest with the arm or when shooting.

To measure the distance using vision, a stereo camera is needed. Therefore, A fusion between Computer Vision and the 2D Lidar would ensure the optimal distance calculation given the constraint possessed by the type of the Drone's Camera.

The Drone: Search Algorithm Alternatives and Decision

The search algorithm is broken down into two segments. The first is shared among all alternatives as it consists of the following:

- Taking Off
- Incrementing the height to ~2 meters
- Starting from a height of 2 meters, 0.5 meter vertical increments are scanned with a 120 Degrees ARC
- Vertical scanning is pursued until the Drone reaches the top of the tree
- Landing

The second segment differs according to the approach as would be detailed in each of the following:

i. The Arm

~85% of the nests which are present on a Pine tree are available at the outer-edge, in the south-facing side. Therefore, we identified two schemes to approach the nests.

1. Approaching the Nest Vertically

Approaching the nests vertically implies that the ARM needs to be placed vertically downward (i.e., in the y-plane), issued from the Drone's bottom surface as is be shown in the following figure:

Figure 41: Vertical approach

This approach is distinguished by allowing the drone to operate at a distance without needing to fly between the branches or trunks of a tree.

The second segment which is specific to vertical approaching consists of the following:

- Tilting the camera until the detection box shows up in the upper half (0, 240) pixels on the vertical - y - axis of the image
- Centering the nest in the middle of the frame's x axis within a confidence bound (428 +- 50) pixels
- Approaching the nest
- The process will repeat until the drone is above the nest (i.e., camera is tilted greater than 85 degrees)
- Accelerate downward on the nest with bounds.
  - Tested in AUV Room, RGB, on a virtual nest by fully landing on the nest.
- Regain the lost height and return back

2. Approaching the Nest Horizontally

Approaching the Nest horizontally implies that the ARM needs to be placed horizontally (i.e., in the x-plane), placed parallel to the orientation of the body of the Drone as is shown in the following figure:

Figure 42: Horizontal approach

This approach increases the efficiency of treating pine-nests. However, it poses a threat given the fact that the drone may hit a branch because of an involuntary movement which could be caused by Wind, among other factors.

The second segment which is specific to horizontal approaching consists of the following:

- Centering the detected object with respect to the y'oy in the frame through vertical increments and decrements within a confidence bound (240+-30)
- Centering the nest in the middle of the frame's x-axis within a confidence bound (428 +- 50) pixels
- Approaching the nest via vision navigation
- The process will repeat until the drone is 1.5 meter away from the nest as detected by the Lidar
- Accelerate the Drone to do the puncture and stop via lidar measurement
    - Tested in AUV Room, RGB, on a virtual nest by fully approaching the nest horizontally
    - Return back

ii. Shooting Mechanism

The Shooting Mechanism establishes an interface between precision agriculture and the intended revolutionary treatment. Given the size of the shooting mechanism as shown in the figure below, the only viable option would be to mount the shooting mechanism on the top of the Bebop, similar to the Horizontal Arm positioning as is in the previous case.

Figure 43: Gun with servomotor

This approach acts from a distance and is expected to treat all detected pine-nests which drives efficiency and effectiveness massively. This approach has a tolerance margin to Wind because the drone need not approach the nest.

The second segment which is specific to the shooting mechanism consists of the following:

- Centering the detected object with respect to the y'oy in the frame through vertical increments and decrements within a confidence bound (240+-30)
- Centering the nest in the middle of the frame's x-axis within a confidence bound (428 +- 50) pixels
- Approaching the nest via vision navigation
- The process will repeat until the drone is 1.5 meter away from the nest as detected by the Lidar
- The shooting mechanism launches the chemical pellet bringing the paintball experience to trees
  - Tested at a Forest in Chuweifat
- Return back

After testing the Arm in the Forest and identifying that it is impossible to do the puncture because the following constraints were not considered:

- Branch moves when puncturing the nest
- Most nests are built on cones
  - The arm's needle may not get fully into the nest to do its desired work

The idea of carrying an arm was dismissed and the focus narrowed down to the shooting mechanism

i. The Drone: Landing

The Landing strategy differs from one setup to another according to the placement of the hardware. The following two landing mechanisms tackle the three different hardware schemes.

1. Landing: Vertical approaching

The second design alternative was the development of a base station for the drone. The base station is expected to be covered by a smooth and penetrable table cover which would allow the arm to establish its path through a puncture. In estimating the dimensions of the station, the radius of divergence from the takeoff position of the chosen drone upon landing and the length of the arm need to be taken into consideration.

2. Landing: Horizontal approaching and Shooting Mechanism

Normal landing without any constraint except for the initial position.

Design Iterations

Note: In what follows, a cycle is a complete treatment scheme which consists of takeoff, treatment delivery, and landing.

Each of the three designs presented above was subjected to pivoting. Pivoting allowed the Drone's configuration to substitute a conservative behavior for an aggressive one.

Starting with the first alternative: "Approaching the Nest Vertically", the following design iterations were encountered:

- First Iteration (Big error)
    - Distance measurement scheme: The Triangular Similarity Method
    - Vertical Scanning: 1 meter above takeoff (i.e., 2 meters high)
    - Horizontal Scanning:
        - Approaching
    - Vertical decrement in space: set to 0.5 m
    - Homing and Landing:
        - Command Drone to move backward what dead-reckoning position measurements claimed was moved forward

Figure 44: Testing drone

- Second Iteration
    - Distance measurement scheme: Computer Vision Navigation
    - Vertical Scanning: 1 meter above takeoff
    - Horizontal Scanning:
        - Camera Adjustment: Shifting down by 4 degrees
        - Approaching:
            - Centering Image Scenario: Conservative
                - Require the center to be: x-axis: 428 +- 20 pixels | y-axis: 240 +- 10 pixels
            - Centering Image Scenario: Aggressive
                - Require the center to be: x-axis: 428 +- 60 pixels | y-axis: less than 240
                - At the expense of increasing camera increments: 8 degrees
    - Lands on the virtual nest (i.e., Vertical arm not mounted)
- Third Iteration
    - Distance measurement scheme: Fusion between Computer Vision Navigation and Lidar
    - Vertical Scanning: 1 meter above takeoff
    - Horizontal Scanning:
        - Camera Adjustment: 8 degrees
        - Approaching:
            - Centering Image Scenario: Aggressive
                - Require the center to be: x-axis: 428 +- 60 pixels | y-axis: less than 240
    - Attacking the nest:
        - Vertical decrement for altitude using Lidar's input
    - Reversing the move:
        - Vertical increment for altitude using Lidar's change
        - Horizontal movement decrements using vision navigation
    - Landing on the designed table

On the other hand, the second alternative, "Approaching the Nest Horizontally", went through the following design iterations:

- First Iteration
    - Distance measurement scheme: Computer Vision Navigation and Lidar
    - Vertical Increment: 1 meter above takeoff
    - Horizontal Scanning:
        - Approaching:
            - Centering Image Scenario: Aggressive
                - Require the center to be: x-axis: 428 +- 60 pixels | y-axis: less than 240
            - Until the drone is ~1.5 to 2 meters away from the nest as per below:
                - A neural network algorithm trained on a dataset of more than 150 trials to map the 1.5 meters to a specific

            - percentage span of the detection box in pixels to the total pixel span
            - Model output was 90% accurate
    - Vertical Adjustments:
        - Increment by 0.2 meters if the y-axis condition is not satisfied
    - Attacking the nest:
        - Reading Lidar and approaching till distance is 0.5 meters in 0.3 meters increments
    - Reversing the distance covered
        - Move backward 2 meters
    - Landing
- Second Iteration
    - Distance measurement scheme: Computer Vision Navigation
    - Vertical Increment: 1 meter above takeoff
    - Horizontal Scanning:
        - Approaching:
            - Centering Image Scenario: Aggressive
                - Require the center to be: x-axis: 428 +- 80 pixels | y-axis: greater than 180 less than 300
            - Until the drone is ~1.5 to 2 meters away from the nest as per below:
                - A neural network algorithm trained on a dataset of more than 150 trials to map the 1.5 meters to a specific percentage span of the detection box in pixels to the total pixel span
                - Model output was 90% accurate
    - Vertical Adjustments:
        - Increment by 0.2 meters if the y-axis condition is not satisfied
    - Attacking the nest:
        - Reading Lidar and approaching till distance is 0.5 meters in 0.3 meters increments
    - Reversing the distance covered
        - Move backward 2 meters
    - Landing

Finally, the third alternative, "The Shooting Mechanism", went through the following design iterations:

- First Iteration
    - Distance measurement scheme: Computer Vision Navigation
    - Vertical Scanning:
        - Increment of 1 meter: (2 meters altitude)
        - Initiate the scanning process
            - For each 0.5 meters increment, check for the nest in a 120 degrees arc
            - 120 degrees arc is represented by 6 stops, each has 20 degrees shift
                - Search time at each stop is ~5 seconds

- Horizontal Scanning:
  - Approaching:
    - Centering Image Scenario: Aggressive
      - Require the center to be: x-axis: 428 +- 80 pixels | y-axis: greater than 180 less than 300
    - Until the drone is ~1.5 to 2 meters away from the nest as per below:
      - A neural network algorithm trained on a dataset of more than 150 trials to map the 1.5 meters to a specific percentage span of the detection box in pixels to the total pixel span
      - Model output was 90% accurate
- Vertical Adjustments:
  - Increment by 0.2 meters if the y-axis condition is not satisfied
- Attacking the nest:
  - Shooting Chemical Pellets demonstrated by Paintball 6 mm pellets imported from Amazon
- Reversing the distance covered
  - Move backward 2 meters
- Landing

- Second Iteration
  - Distance measurement scheme: Fusion between Computer Vision Navigation and Lidar
  - Distance measurement scheme: Computer Vision Navigation
  - Vertical Scanning:
    - Increment of 1 meter: (2 meters altitude)
    - Initiate the scanning process
      - For each 0.5 meters increment, check for the nest in a 120 degrees arc
      - 120 degrees arc is represented by 6 stops, each has 20 degrees shift
        - Search time at each stop is ~5 seconds
  - Horizontal Scanning:
    - Approaching:
      - Centering Image Scenario: Aggressive
        - Require the center to be: x-axis: 428 +- 80 pixels | y-axis: greater than 180 less than 300
      - Until the drone is ~1.5 to 2 meters away from the nest as per below:
        - A neural network algorithm trained on a dataset of more than 150 trials to map the 1.5 meters to a specific percentage span of the detection box in pixels to the total pixel span
        - Model output was 90% accurate
  - Vertical Adjustments:
    - Increment by 0.2 meters if the y-axis condition is not satisfied
  - Attacking the nest:

        - Reading Lidar and approaching till distance is 1 - 1.5 meters in 0.3 meters increments
        - Shooting Chemical Pellets demonstrated by Paintball 6 mm pellets imported from Amazon
    - Reversing the distance covered
        - Move backward 2 meters
    - Landing

IX. Preliminary implementations and Testing

A. Computer vision and nest detection:

Preliminary Implementation:

Iteration 1: Local Hardware:

First iteration was done on local computers, having decent GPU's of NVIDIA GTX 950. However, as the dataset increased, personal laptops were unable to handle this amount of RAM passing at the same time.

"Allocator (GPU_0_bfc) ran out of memory trying to allocate 3.54GiB. The caller indicates that this is not a failure but may mean that there could be performance gains if more memory were available."

Therefore, another solution should have been found.

Iteration 2: Cloud Computing:

Cloud computing is the state of art practice of using cloud functions and the internet, to rent GPU's offered by different parties across the world, thus creating a network between providers and users of those powerful hardware.

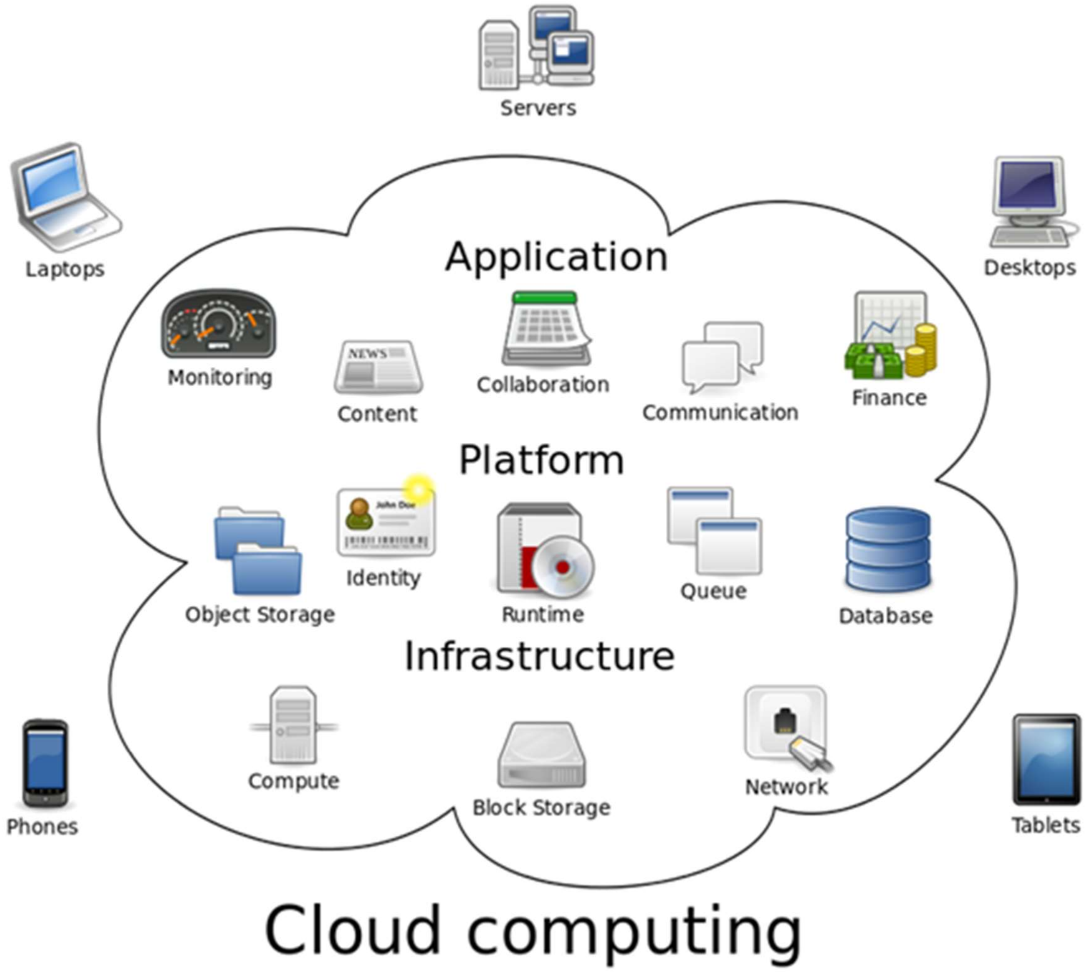


Figure 45: Diagram illustrating the working mechanism of cloud computing

Amazon Web Services (AWS) is one the leading figures in the domain of cloud computing, artificial intelligence, and IoT.
The dataset that we collected was upload as an s3 bucket to the amazon database.

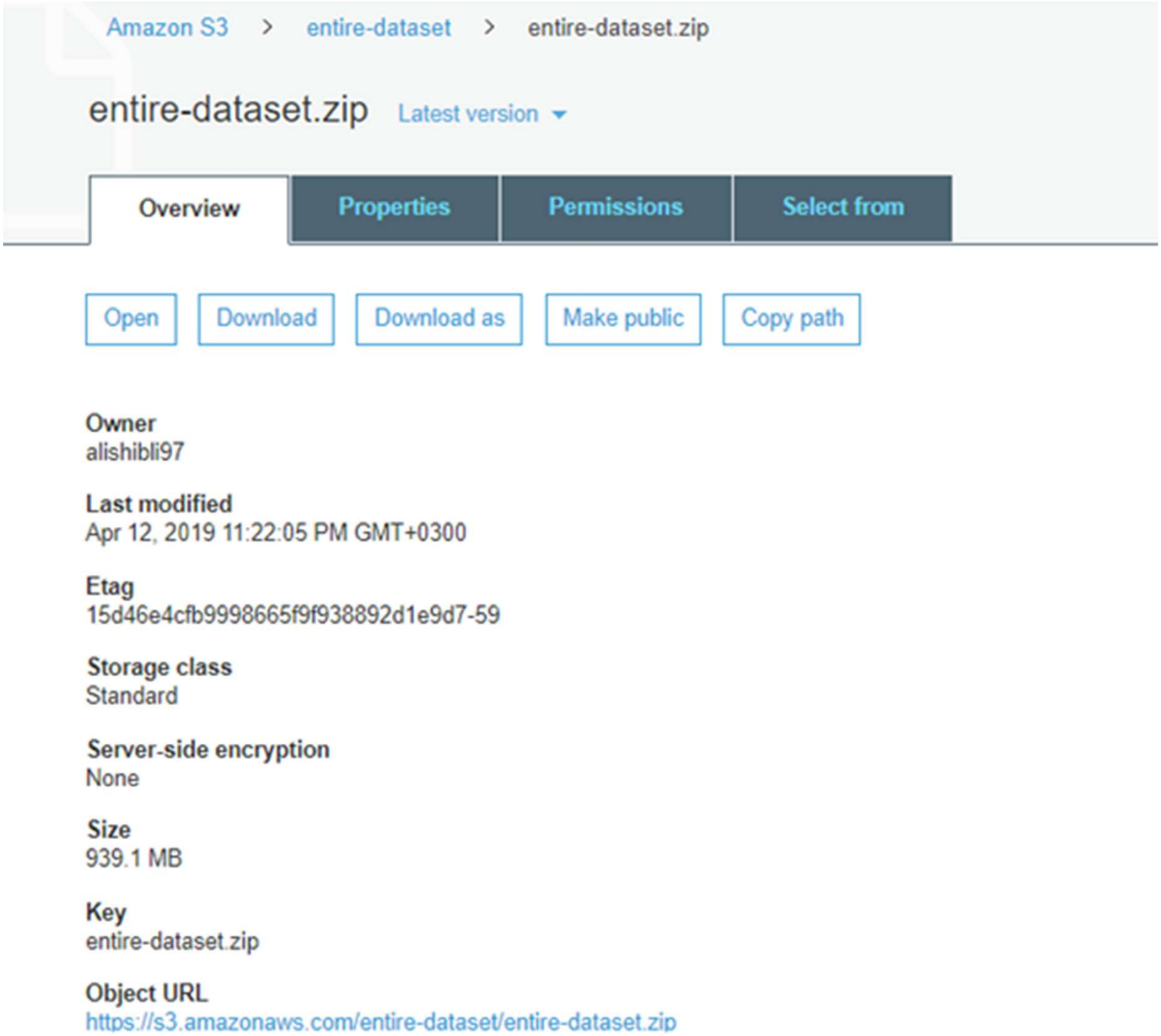


Figure 46: s3 bucket for the dataset

Having shared the dataset on the cloud, a GPU now had to be rented. There were several options, from different types and speeds of GPU's. The final GPU that was of use was the 2080Ti NVIDIA RTX. This is the second-best GPU in the market today, leading in terms of speed and performance.

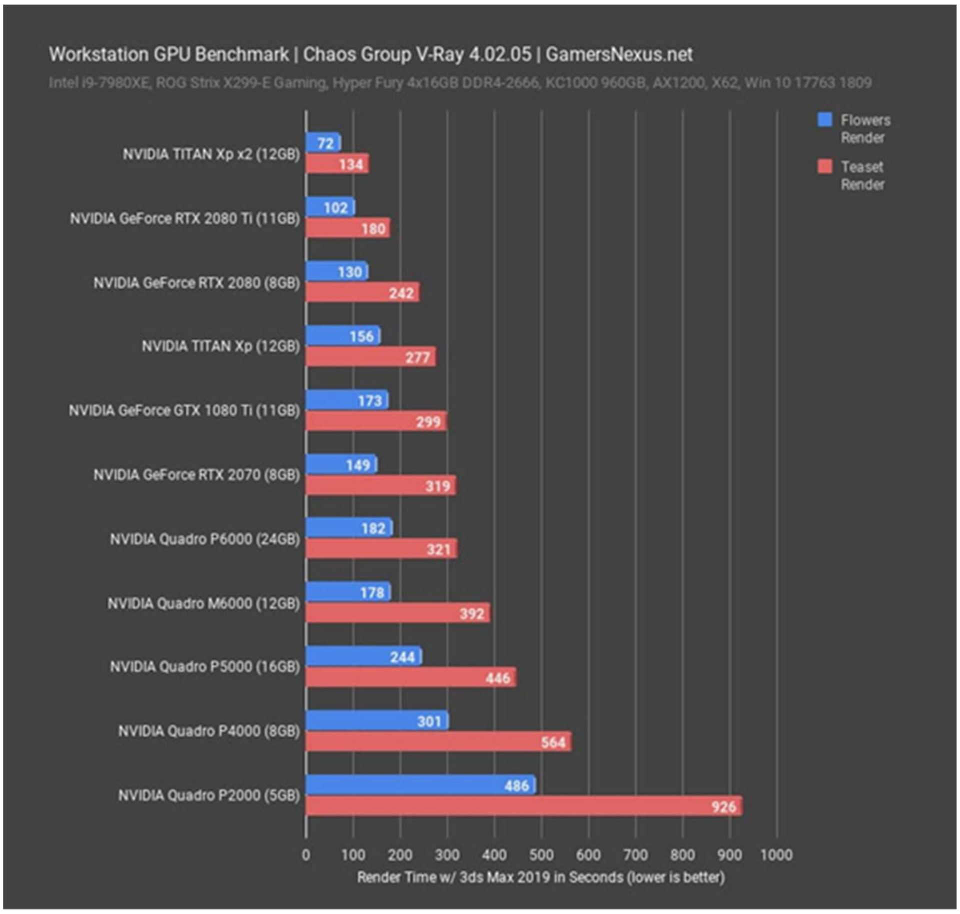


Figure 47: Figure illustrating the performances of the different GPU's w.r.t. time

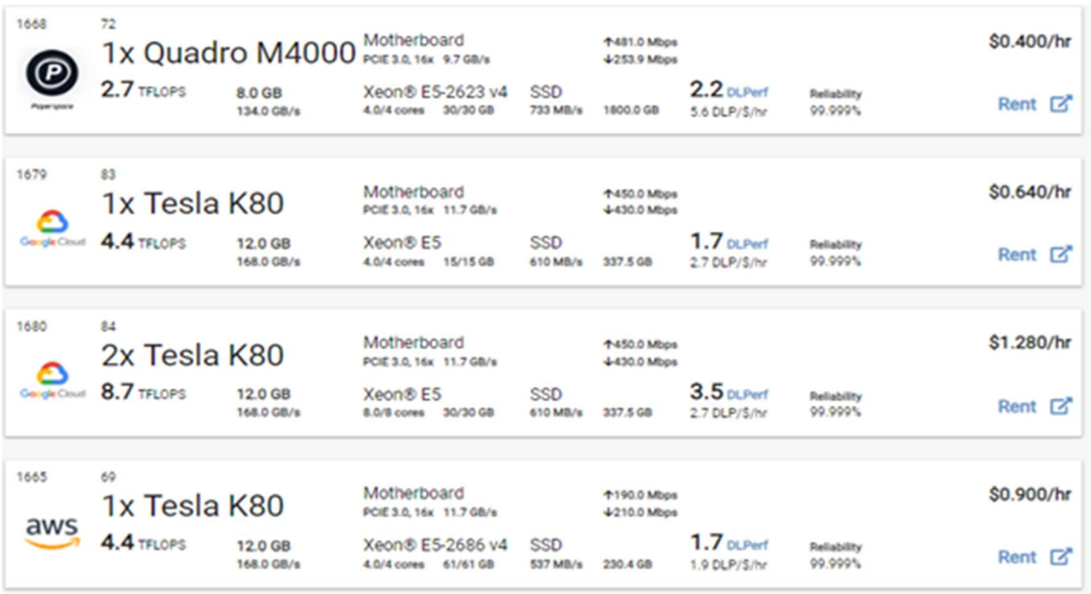


Figure 48: Different GPU's available with their corresponding prices

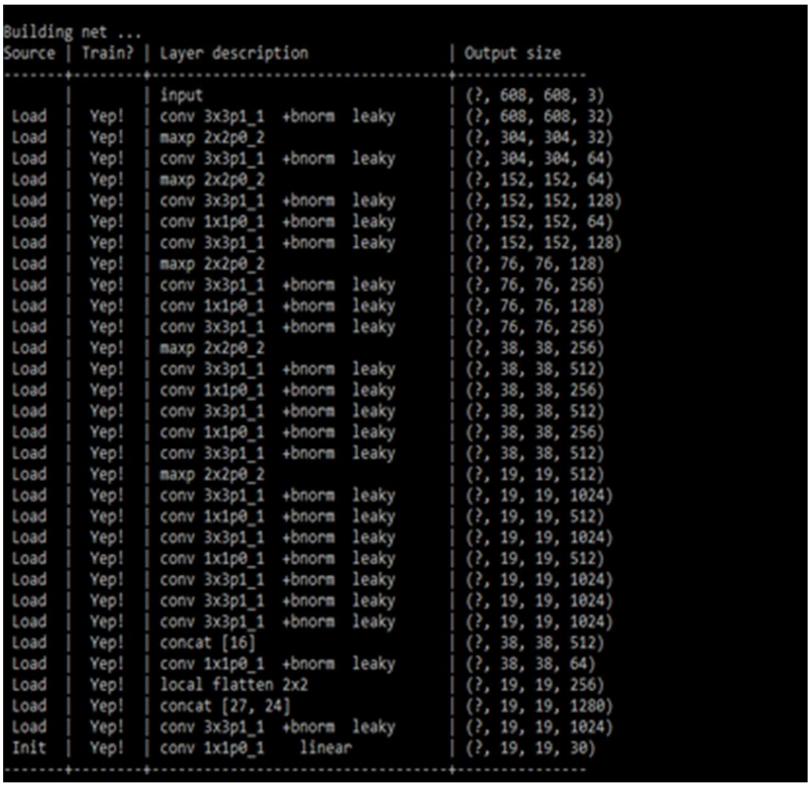
Finally, for training, the set of training images was inputted to the neural network.

The only changed layers in the neural network were the last two layers, as the algorithm is to be trained as per "transfer learning".

The activation functions for all the convolutional layers were set to Leaky, except the last one was set to Linear.
The learning rate was set to 0.001 and the threshold for detection was set 0.1.

```
pad=1
filters=1024
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=30
activation=linear


[region]
anchors =  0.57273, 0.677385, 1.87446, 2.06253, 3.33843, 5.47434, 7.88282, 3.52778, 9.77052, 9.16828
bias_match=1
classes=1
coords=4
num=5
softmax=1
jitter=.3
rescore=1

object_scale=5
noobject_scale=1
class_scale=1
coord_scale=1

absolute=1
thresh = .1
random=1
```

Figure 49: Setup file for the training with all required parameters



```
Building net ...
Source | Train? | Layer description                | Output size
-------+--------+----------------------------------+---------------
       |        | input                            | (?, 608, 608, 3)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 608, 608, 32)
 Load  |  Yep!  | maxp 2x2p0_2                     | (?, 304, 304, 32)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 304, 304, 64)
 Load  |  Yep!  | maxp 2x2p0_2                     | (?, 152, 152, 64)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 152, 152, 128)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 152, 152, 64)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 152, 152, 128)
 Load  |  Yep!  | maxp 2x2p0_2                     | (?, 76, 76, 128)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 76, 76, 256)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 76, 76, 128)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 76, 76, 256)
 Load  |  Yep!  | maxp 2x2p0_2                     | (?, 38, 38, 256)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 38, 38, 512)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 38, 38, 256)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 38, 38, 512)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 38, 38, 256)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 38, 38, 512)
 Load  |  Yep!  | maxp 2x2p0_2                     | (?, 19, 19, 512)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 19, 19, 512)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 19, 19, 512)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Load  |  Yep!  | concat [16]                      | (?, 38, 38, 512)
 Load  |  Yep!  | conv 1x1p0_1  +bnorm  leaky      | (?, 38, 38, 64)
 Load  |  Yep!  | local flatten 2x2                | (?, 19, 19, 256)
 Load  |  Yep!  | concat [27, 24]                  | (?, 19, 19, 1280)
 Load  |  Yep!  | conv 3x3p1_1  +bnorm  leaky      | (?, 19, 19, 1024)
 Init  |  Yep!  | conv 1x1p0_1    linear           | (?, 19, 19, 30)
-------+--------+----------------------------------+---------------
```

Figure 50: Initializing the neural network for training


```
step 10980 - loss 1.434236764907837 - moving
 ave loss 1.4758624821855957
step 10981 - loss 1.3759632110595703 - movin
g ave loss 1.4658725550729932
Invalid SOS parameters for sequential JPEG
step 10982 - loss 1.6778664588928223 - movin
g ave loss 1.4870719454549761
step 10983 - loss 1.8869504928588867 - movin
g ave loss 1.5270598001953672
step 10984 - loss 0.8757531642913818 - movin
g ave loss 1.4619291366049687
step 10985 - loss 1.1473373174667358 - movin
g ave loss 1.4304699546911455
step 10986 - loss 1.3121285438537598 - movin
g ave loss 1.4186358136074069
step 10987 - loss 1.8750689029693604 - movin
g ave loss 1.4642791225436023
step 10988 - loss 1.0967432260513306 - movin
g ave loss 1.4275255328943752
step 10989 - loss 1.0132954120635986 - movin
g ave loss 1.3861025208112976
step 10990 - loss 1.0494569540023804 - movin
g ave loss 1.352437964130406
```

Figure 51: Training phase and updating of the loss function

Parameters:

a. Total time spent training = 14 hours.
b. Number of epochs = 100.
c. Batch size = 8.

Finally, the output of the neural network was a weights file of size 0.6 GB.
The code was then implemented with the code for controlling the drone:

```
img = self.vision.get_latest_valid_picture()

if img is not None:
    result = tfnet.return_predict(img)
    print('The results found are ' + str(result))
    print("Detection?")
    if result != []:
        print("Detected")
        self.countindex3 += 1
        print("Self.countindex3 = ", self.countindex3)
        if self.countindex3 >= 3:
            self.inspect = 6

        tl = (result[0]['topleft']['x'], result[0]['topleft']['y'])
        br = (result[0]['bottomright']['x'], result[0]['bottomright']['y'])

        print("Center Calculation")
        icx = tl[0] + (br[0] - tl[0]) / 2
        icy = tl[1] + (br[1] - tl[1]) / 2

        print("Area Calculation")
        lgth = br[0] - tl[0]
        wdth = -1 * (tl[1] - br[1])
        cod1 = lgth / 856
        cod2 = wdth / 480

        img = cv2.rectangle(img, tl, br, (0, 255, 0), 7)
        img = cv2.putText(img, "Nest", tl, cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 0), 2)
        cv2.imwrite('imagesFromDrone/Image' + str(self.imageNum) + '.jpg', img)
        self.imageNum += 1
```

Figure 52: Snapshot of the control-vision code

Testing:

After training the neural network, the output updated weights file of size ~ 0.6 GB was tested twice:

LOCAL IMAGES OF THE DRONE

From the total images collected, as discussed earlier, around 500 images were left to test on.
On those local images, the algorithm showed perfect results. The algorithm was able to detect the nests with high accuracy of 95-99%, not leaving any nest without detection, as the following images show:

Figure 53: Tested Images from Local Folder

Nevertheless, testing on local images is harder than actual live testing. Thus, the next step was to combine everything together, the algorithm of detection and flight so that we can test in a live demo.

LIVE CAMERA OF THE DRONE

After the combination of control and vision codes, the entire code was uploaded to the drone. The camera of the drone read images live at a rate of 1fps to 3fps to ensure than the images taken were of good quality for processing.

Figure 54: Tested Images from Live Camera of the Drone

Nevertheless, after scanning and detecting the nest, the drone will fly per the control algorithm to the nest directly to ensure the treatment (shooting the nest with the pesticide).
Even with the presence of all those noises in the images and the nests and variations of the shapes, the algorithm was able to detect those nests as well.

Figure 55: Examples of bad quality images and nests after treatment

Nevertheless, after scanning and detecting the nest, the drone will fly per the control algorithm to the nest directly to ensure the treatment (shooting the nest with the pesticide).
Also, the algorithm was able to detect at a distance, for instance 4-5 meters away.

Figure 56: Examples of detected nests at a distance of 4-5 meters away

B. Shooting/arm mechanism

Preliminary Implementation

Design alternative 1 of the arm was considered and 3D printed first using PLA material as seen in the figure below.

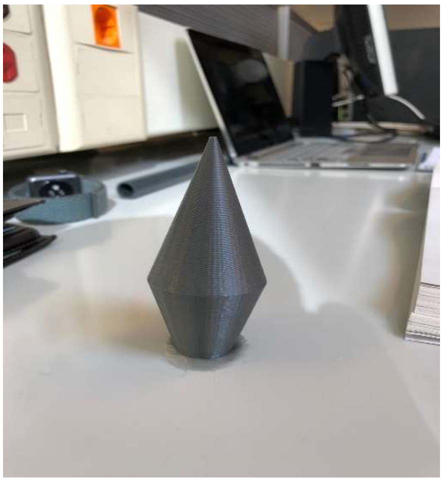


Figure 57: 3D printed needle

In addition, the joint component, connector component, three locks, and needle were also 3D printed using the Ultimaker 3 3D printer. However, the arm itself was not 3D printed, but a PVC pipe of length 20 cm was used for the prototype. The connector and the PVC arm were glued together, and the needle was glued to the bottom of the arm to form one component as seen in the figure below.

Figure 58: The arm

The joint part was attached to the bottom of the drone and locked using the two locks. The third lock along with the spring were placed in the hole of the connector. Finally, the connector part was attached to the joint using a 2mm pin.

The 3D printed needle wasn't sharp enough to puncture the body of the nest. To compensate for that, an aluminum bar was inserted into the 3D printed needle, where its end was sharpened to become needlelike as seen in the figure below.

Figure 59: Needle's head

Finally, several holes were made in the body of the PVC pipe to make it lighter for the drone to carry as seen in the figure below. The complete system for design alternative 1 is shown in the figure below that includes the needle, the arm, and the connector part.

Figure 60: Arm

Design alternative 2 was also implemented using the same materials as in design alternative 1. The second design of the needle is also made out of PLA material, but it consists of several aluminum needle bars instead of one as shown in the figure below.

Figure 61: 4-needles head

The original design consisted of having only four needles. In the implementation, five sharp needles were added instead of four to increase the piercing power of the total needle. The middle aluminum needle was constructed to be larger in diameter than the other four surrounding needles, whereas the other needles were larger in length. This slight change in design was made to have the four needles contact the nest at first followed by the larger middle needle. By doing so, it would be easier for the four surrounding needles to puncture the nest at first due to their smaller diameter. After having the surrounding needles pierce through the nest, the middle needle will follow to make a larger puncture than the other needles, and by that, all the surrounding punctures will meet with the middle puncture to make one large hole of 4 cm in diameter.

Testing

1. ROBOTIC ARM

The robotic arm system was connected to the drone successfully. A takeoff command was sent to the drone to test the robustness of the arm design as well as the success of the locking mechanism. At first, the one headed needle was tested to evaluate its puncturing power with the nylon sheet that is mimicking the nest's surface.

Figure 62: Drone on station with robotic arm attached at initial position

Figure 63: Drone taking off with robotic arm midway position

The locking mechanism was successful upon the drone's take off. The station was designed with nylon material so that the arm would puncture it and the drone would land safely on the nylon paper, while keeping the arm from braking.

Unfortunately, the arm broke after landing due to the ineffectiveness of the needle's puncturing power.

After that, the multi headed needle was tested to also evaluate its puncturing power against the nylon sheet.

Figure 64: Needle through the nylon surface

The second design alternative for the needle also failed to puncture the nylon sheet. As seen the figure above, all of the needles pierced through the sheet, but that was not enough to ensure a unified hole through the sheet.

2. Shooting Mechanism

The shooting mechanism was tested on real Pine trees in Chweyfet. The mechanism was mounted on the drone and loaded with chemical pellets ready to shoot. After takeoff, the drone scanned the tree and located the nests using the vision algorithm. Once the drone approached the nest, it shot with the blue paintball pellets loaded in the mechanism.

Figure 65: Drone between trees

The drone was successful at locating the nest and shooting it with the pellet. As shown in figure below, a dot of blue paint at the center of the nest is seen, which confirms the success of the shooting part of the device.

Figure 66: Shooting nest

C. The Drone

Preliminary Implementation

The Bebop 2.0 was programmed using Python via Pycharm which established a layer of interface between the drone and the code. Parrot libraries such as pyparrot and bebop, and other relevant libraries such as serial and cv2 to interface with the Arduino and were installed to control the drone autonomously.

The code implemented on the drone which was utilized during the in forest testing where the shooting mechanism was deployed is copied to Appendix C. The difference between any two designs is a difference in the indices as was illustrated previously in the Design iterations section.

The code works as follows: If the model detects the nest then, as per the design iterations' indices the code regulates on the x and y axes. Then approaches and does its function.

Testing

i. Vertical Nest Approaching

a. Trial 1: Fall's results

Figure 67: The results of the fall semester

b. Trial 2: Successfully landing on the nest

Figure 68: Successful landing on the nest

ii. Horizontal Approaching: Early after takeoff before height adjustment

Figure 69: Horizontal approaching adjustments

iii. Shooting Mechanism

a. Trial 1: Shooting Mechanism while looking for the nest

Figure 70: Shooting mechanism in lab

b. Trial 2: Forest testing – in the process of shooting the nest

Figure 71: Shooting mechanism in forest

X. List of resources and engineering tools used

The following mutually exclusive, collectively exhaustive list presents a summary of the fundamental instruments and software used to utilize and execute the project. Those tools are mainly divided into three categories:

a. Control: which mainly constitutes the flight algorithm of the drone, under which lies the entire path algorithm which the drone will follow autonomously.
b. Vision: which is the main platform used to scan the area on the tree and check for the presence of the nests on the tree.
c. Hardware/robotic setup: the killing mechanism, which holds the strategy used to kill the parasites by shooting chemicals filled inside beads of a plastic pistol.

The detailed structure of each category is summarized in the following:

Vision:

- YOLOv2: the main algorithm used to train on the images of the nest. It is a neural network formed of 24 layers and was trained on around 1000 images of the nest. YOLOv2 is built on top on TensorFlow and is the translation of Darknet from C to Python.
- OpenCV: used to utilize the real time processing of the images and output the specific boxes around and coordinates around the nest.
- Cloud Computing: training of the neural network was done on the cloud on Amazon Web Services (AWS) on an NVIDIA 2080-Ti GPU (the best in the market today).

- Camera Full-HD (the default camera of the drone): used to return the images seen by the drone for processing.
- NVIDIA 950 GPU: used to run the algorithm for real time processing of the drone.

Hardware/shooting mechanism:

- Plastic pistol (gun): a pistol used to shoot the nest from a distance, with beads of diameters about 7mm.
- Pesticides: inside each pistol bead, there are pesticides saturated that on contact with the nest, the beads explode in the nest thus killing all the parasites.
- Lidar: used to measure the distance from the drone to the nest. At a distance of 1.5-2 meters, the shooting mechanism should activate.
- Arduino Nano: used to control the gun and send it the command for shooting at the exact timing.
- Bluetooth module: used as a connection tool between the Arduino and the gun. The Arduino send its commands as Bluetooth signals to the gun.
- Servo motor: a small thin wire is connected to the gun so that as the wire pulls (by means of the servo motor), it induces the gun to shoot.
- Robotic Arm: used only in the first iteration of the project.

Control:

- Bebop 2.0 by Parrot Drone will be utilized to fly around the tree and scan the entire area to detect the nests.
- PyCharm: main platform to program the drone using python through a wireless connection that the drone offers.

XI. Detailed Project Schedule

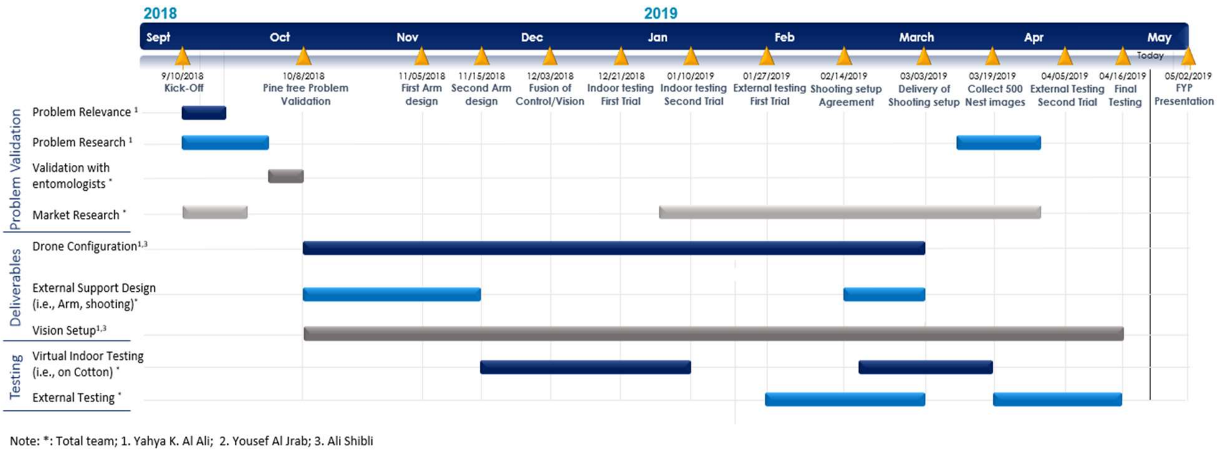


Figure 42. Detailed Project Schedule

XII. References

[1] R. Hawk, "Ten Most Common Trees," Swain County Center, 13 June 2018. [Online]. Available: https://swain.ces.ncsu.edu/2009/08/ten-most-common-trees/. [Accessed 19 October 2018].

[2] S. Cade and R. Hedden, "Growth Impact of Pine Tip Moth on Loblolly Pine Plantations in the Ouachita Mountains of Arkansas," Southern Journal of Applied Forestry, vol. 11, no. 3, pp. 128-133, 1987.

[3] M. Wingfield, A. Hammerbacher, R. Ganley, E. Steenkamp, T. Gordon, B. Wingfield and T. Coutinho, "Pitch canker caused by Fusarium circinatum -- a growing threat to pine plantations and forests worldwide," Australasian Plant Pathology, vol. 37, pp. 319-334, 2008.

[4] ImageNet, [Online]. Available: http://www.image-net.org/. [Accessed 19 October 2018].

[5] K. Sung and T. Poggio, "Example-based Learning for View-based Human Face Detection," Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 39-51, 1994.

[6] C. Vondrick, A. Khosla, H. Pirsiavash, T. Malisiewicz3, A. Torralba, "Visualizing Object Detection Features," International Journal of Computer Vision, 09/2016, Volume 119, Issue 2

[7] C. Papageorgiou, M. Oren and T. Poggio, "A General Framework for Object Detection," in 6th International Conference on Computer Vision, 1998.

[8] N. Paterakis, E. Mocanu, M. Gibescu, B. Stappers and W. Alst, "Deep Learning Versus Traditional Machine Learning Methods for Aggregated Energy Demand Prediction," in Innovative Smart Grid Technologies Conference, Torino, Italy, 2017.

[9] K. Nguyen, C. Fookes and S. Sridharan, "Improving Deep Convolutional Neural Networks with Unsupervised Feature Learning," in International Conference on Image Processing, Quebec, Canada, 2015.

[10] L. Fukushima, "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position," Biological Cybernetics, vol. 36, no. 4, pp. 193-202, 1980.

[11] A. Bhandare, M. Bhide, P. Gokhale and R. Chandavarkar, "Applications of Convolutional Neural Networks," International Journal of Computer Science and Information Technologies, vol. 7, no. 5, pp. 2206-2215, 2016.

[12] X. Li and S. Wang, "Object Detection Using Convolutional Neural Networks in a Coarse-to-Fine Manner," IEEE Geoscience and Remote Sensing Letters, vol. 14, no. 11, pp. 2037-2041, 2017.

[13] T.-D. Truong, V.-T. Nguyen and M.-T. Tran, "LightweightDeepConvolutionalNetworkforTinyObjectRecognition," in International Conference on Pattern Recognition Applications and Methods, 2018.

[14] S. Zia, B. Yuksel, D. Yuret and Y. Yemez, "RGB-DObjectRecognitionUsingDeepConvolutionalNeuralNetworks," in IEEE International Conference on Computer Vision Workshops, 2017.

[15] Alhussein Fawzi, Horst Samulowitz, Deepak Turaga, and Pascal Frossard, "ADAPTIVE DATA AUGMENTATION FOR IMAGE CLASSIFICATION," in IEEE International Conference on Image Processing (ICIP), 2016.

[16] Joseph Lemley, Shabab Bazrafkan, Peter Corcoran, "Smart Augmentation Learning an Optimal Data Augmentation Strategy", in IEEE International Conference on Image Processing (ICIP), 2016.

[17] "How do BB guns work?", 1 June 2001. HowStuffWorks.com. [Online]. Available: https://entertainment.howstuffworks.com/question649.htm/. [Accessed 21 April 2019].

XIII. Appendix

A. Updated Project Description and Agreement Form

Faculty Supervisor	**Professor Naseem Daher**
Co-Supervisor [optional]	Professor Nabil Nemer
Sponsor [optional] Is there industry support or funding the project?	Not Applicable

Project Title	The Utilization of Drones to Monitor the Health of Pine Trees

Descriptive title not necessarily the final title that will be adopted by the team	
Project Description and Design Aspects What is the main motivation for the project? Specify the desired needs that the final product is expected to meet.	Recently, parasites have been growing in volume massively in all regions across the world and their negative impact on nature has been increasingly recognized especially in Apiculture and among trees. To preserve Lebanese monuments and fortunes, this project was realized to tackle the latest issue governing the Pines. Recently, at Lebanese Natural Reserves, Pine trees were diseased by the Pine Processionary Moths – Parasites. The project is to be generalized to all monumental trees present around the world to lessen the barriers to entering such healthcare industry and to ensure a sales volume for this project. The main needs such FYP is anticipated to tackle, primarily, are: - Detecting/Killing the parasites that are not easily detected by humans - Keeping records about the health of a tree by monitoring key performance indicators - Utilizing Machine Learning and Data Analytics to analyze KPIs and interpret health
Expected Deliverables Required deliverable(s) from the team at the conclusion of the design project	iii. An intelligent drone capable of: ▪ Flying autonomously with the aid of computer vision ▪ Detecting the Pine Processionary Moth nests with a high accuracy (90-95%). ▪ Scanning the area around the tree continuously while flying (horizontally and vertically). ▪ Approaching the nest with a reasonable speed ~ 1 m/s ▪ Shooting the nest with the paintball pellets at a distance of 1.5m far from the nest at a high accuracy (90%) iv. An updated neural network weights file trained specifically on this nest affordable to any party interested in this research.
Technical Constraints	➢ The vision algorithm can be trapped by rocks, space, heads, and whiteboards

A preliminary list of multiple realistic technical constraints, e.g. power, accuracy, real-time operation , … The technical constraints included should be detailed and specific to the design project not generic.	➢ The vertical speed of the drone cannot exceed 6 m/s ➢ The horizontal speed of the drone cannot exceed 16 m/s ➢ The vision algorithm cannot realize the nest if the camera is more than 3 meters away ➢ The drone has a flight limit of 25 minutes ➢ The drone carrying ability is limited to a weight of 300 grams ➢ The robotic arm must be long (50 and 70 cm) to reach the nests that are inside ➢ The diameter of the needle should be at least 4 cm ➢ The paintball pellets might splash inside a shooting mechanism ➢ The radius of the shooting mechanism is 8 mm o The radius of the pellets should be less than 8 mm ➢ The shooting range is limited to 5 meters ➢ The drone doesn’t have night vision ➢ The sensors of the drone are not accessible for reading ➢ The drone cannot approach the edge of the tree
Non-Technical Constraints A preliminary list of multiple realistic non-technical constraints, e.g. cost, environmental friendliness, social acceptance, political, ethical, health and safety, etc… The non-technical constraints included should be detailed and specific to the project not generic.	➢ The weather conditions (rain, wind, storms) ➢ The presence of birds and other insects ➢ The regulations put by the Lebanese Army to limit drone flights ➢ The cycle in which the Pine Processionary Moth is on Pine trees occurs between mid December and early April
Contemporary Issues Cite one or more recent articles pertaining to the project or project area: news articles, blog discussions, academic articles, conference topics, etc.	Motivation to use Drones in Agriculture: (Credits to Prof. Naseem Daher) - https://www.futuristspeaker.com/business-trends/agriculture-the-new-game-of-drones/ The Motivation of such project at Natural Reserves: - https://newsroomnomad.com/fatal-disease-threatens-lebanons-historic-parasol-pines/

	The Impact and Definition of Pine Processionary Moths (PPMs) - https://www.forestry.gov.uk/pineprocessionarymoth Insights about the critical effect of PPMs: - http://www.ciudadverde.net/en/the-processionary-pine-caterpillar.html Vision based recognition systems for flying insects: - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5982143/ Image processing techniques for insect shape detection in field crops: - https://ieeexplore-ieee-org.ezproxy.aub.edu.lb/document/8365226/ Drones utilize Chemicals to spray land and take out the Mosquito threat: - http://www.govtech.com/dc/articles/drones-hired-to-take-out-mosquito-threat.html
Resources and Engineering Tools Identify resources and engineering tools needed and whether they are available or need to be acquired (if known), e.g. software licenses, instruments, facilities, components, …	1. Control: a. Bebop 2.0 by Parrot Drone will be utilized to fly around the tree and scan the entire area to detect the nests. b. Pycharm: main platform to program the drone using python through a wireless connection that the drone offers. 2. Vision: a. YOLOv2: the main algorithm used to train on the images of the nest. It is a neural network formed of 24 layers and was trained on around 1000 images of the nest. YOLOv2 is built on top on tensorflow and is the translation of DarkNet from C to Python. b. OpenCV: used to utilize the real time processing of the images and output the specific boxes around the nest. c. Cloud Computing: training of the neural network was done on the cloud on Amazon Web Services (AWS) on an NVIDIA 2080-Ti GPU (the best in the market today). d. Camera Full-HD (the default camera of the drone): used to return the images seen by the drone for processing. e. NVIDIA 950 GPU: used to run the algorithm for real time processing of the drone. 3. Hardware/Shooting Mechanism: a. Plastic pistol (gun): a pistol used to shoot the nest from a distance, with beads of diameters about 7mm.

	b. Pesticides: inside each pistol bead, there are pesticides saturated that on contact with the nest, the beads explode in the nest thus killing all the parasites. c. Lidar: used to measure the distance from the drone to the nest. At a distance of 1.5-2 meters, the shooting mechanism should activate. d. Arduino nano: used to control the gun and send it the command for shooting at the exact timing. e. Bluetooth module: used as a connection tool between the arduino and the gun. The arduino send its commands as bluetooth signals to the gun. f. Servo motor: a small thin wire is connected to the gun so that as the wire pulls (by means of the servo motor), it induces the gun to shoot. g. Robotic Arm: used only in the first iteration of the project.
List of Disciplines Identify at least THREE engineering disciplines (within or outside ECE)	◯ Biomedical Systems ◯ Circuits and Electronics ◯ Communications and Networking ◯ Control, Robotics, and Instrumentation ◯ Electromagnetics and RF ◯ Hardware, Computer Architecture, and Digital Systems ◯ HMI, Graphics, and Visualization ◯ Intelligent Systems ◯ Machines and Power Systems ◯ Signal Processing ◯ Software Engineering ◯ Engineering discipline outside ECE (Specify): Additional Non-Engineering discipline(s) (Specify): ML, Data Analytics, CV
Number of Students Please consider the number of disciplines checked above	◯ 3 students OR ◯ 4 students (if 4 please provide a justification)
Required Courses [Optional]	EECE 310, EECE 311, EECE 460, EECE 464, EECE 603, EECE 660, EECE 692

List the courses that are essential for the successful execution of the project (especially advanced courses)	
Date Last Updated	April 25, 2019

B. Minutes of All Meetings Up-to-Date

Meeting #: 1-26			
Date: Mondays	Time: 3 – 3:30 pm	Location: Bechtel 435/536; Masri 508	
Meeting called by	Prof. Naseem Daher		
Attendees	Prof Naseem Daher, Ali Shibli, Youssef Al Jrab, Yahya Khaled Al Ali		
Minutes taker	Team		
Agenda Item: Validating the Problem, Market Study, and Directing the research efforts			
Discussion			
As a start, we chose to target pine, for proof of concept. The next main thing is choosing nests rather than eggs as eggs seemed to be not feasible. Next, we divided the tasks between control and vision. Finally, we conducted a way fuse vision with control.			
Conclusions			
Dividing the tasks between team members. Deciding on the algorithms to use. Deciding on the control algorithm of the drone. Deciding on the timeline.			

Action Items	**Person Responsible**	**Deadline**
Augmenting the dataset of images. Training the algorithm on the resultant images. Implementing the algorithm for detection. Autonomous landing using vision. Autonomous navigation around trees. Setup for puncturing the nest. Setup for shooting the nest.	Whole team	End of semester

Agenda Item: Pivoting Fall Semester’s design		
Discussion		
Identify new metrics for: - Flight duration - Scanning algorithm - Image Collection - Detection Algorithm - Shooting Mechanism design		
Conclusions		
▪ Should get aggressive ▪ Should develop a path planning algorithm ▪ Capture new set of images for new training ▪ Training the algorithm ▪ Build the shooting setup		
Action Items	**Person Responsible**	**Deadline**
Get Aggressive Speed by making conditions less conservative	Yahya	February 17
Path planning around the tree	Yahya	February 17
Capture the new set of images for training the algorithm	Ali	February 17
Finalize the algorithm and training	Ali	February 17
Build the shooting mechanism using BBGun	Youssef	February 17

Agenda Item: Finalizing Design for Testing at a Pine Forest in Chuweifat		
Discussion		
- Scanning Algorithm - Data Set fixing and Adding new real images - Lidar Integration - Live Demo		
Conclusions		
▪ Scanning Algorithm should take 2 mins per tree ▪ Need to reduce augmented images from images ▪ Lidar integration and testing ▪ Live Demo by April 16 latest		
Action Items	**Person Responsible**	**Deadline**
Update Scanning algorithm	Yahya	April 13
Fix the dataset	Ali	April 13

Lidar Integration	Yousef	April 13
Demo	Team	April 16

C. Preliminary Testing on Drone: code

```
from pyparrot.Bebop import Bebop
from pyparrot.DroneVisionGUI import DroneVisionGUI
import threading
import cv2
import matplotlib.pyplot as plt
from darkflow.net.build import TFNet
import time
import serial

options = {
    # set the location of your cfg file
    'model': 'darkflow/cfg/yolo-nest.cfg',
    'load': -1,
    'threshold': 0.3,
    'gpu': 1.0
}

tfnet = TFNet(options)

isAlive = False

x = False
icx = 0
icy = 0


class UserVision:
    def __init__(self, vision):
        self.index = 0
        self.index2 = 0
        self.index3 = 0
        self.countindex3 = 1
        self.inspect = 1
        self.inspect1 = 0
        self.inspect2 = 0
        self.inspect3 = 0
        self.d1cond = True
        self.vision = vision
        self.angle = 0
        self.imageNum = 0
        self.v = 1
        self.ykaindex = 0
        self.flag = 0

    def save_pictures(self, args):
        print("Entered Save Pictures")
        if self.index == 0:
            print("Will Take Off now")
            bebop.safe_takeoff(5)
            self.inspect = 1
```

```
            self.index = 2
        if self.index == 1:
            print("Will land now")
            bebop.safe_land(5)
            print("disconnecting")
            bebop.disconnect()

        ser = serial.Serial('com7', 9600)
        distance = int(ser.readline())

        img = self.vision.get_latest_valid_picture()

        if img is not None:
            result = tfnet.return_predict(img)
            print('The results found are ' + str(result))
            print("Detection?")
            if result != []:
                print("Detected")
                self.countindex3 += 1
                print("Self.countindex3 = ", self.countindex3)
                if self.countindex3 >= 3:
                    self.inspect = 6

                tl = (result[0]['topleft']['x'], result[0]['topleft']['y'])
                br = (result[0]['bottomright']['x'], result[0]['bottomright']['y'])

                print("Center Calculation")
                icx = tl[0] + (br[0] - tl[0]) / 2
                icy = tl[1] + (br[1] - tl[1]) / 2

                print("Area Calculation")
                lgth = br[0] - tl[0]
                wdth = -1 * (tl[1] - br[1])
                cod1 = lgth / 856
                cod2 = wdth / 480

                img = cv2.rectangle(img, tl, br, (0, 255, 0), 7)
                img = cv2.putText(img, "Nest", tl, cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0,
0), 2)
                cv2.imwrite('AYY1/Image' + str(self.imageNum) + '.jpg', img)
                self.imageNum += 1

                print("Center of Image - X'oX: ", icx)
                print("Center of Image - Y'oY: ", icy)
                print("Length is: ", lgth)
                print("Width is: ", wdth)
                print("% Length condition is: ", cod1)
                print("% Width condition is: ", cod2)

                if 100 <= icy <= 380:
                    bebop.fly_direct(0, 10, 0, 0, 1)
                    if 300 <= icx <= 556 and not (0.310569 < cod2 < 0.361653) and not
(0.175 < cod1 < 0.207553):
                        bebop.fly_direct(0, 15, 0, 0, 1)
                    elif 300 <= icx <= 556 and 0.175 < cod1 < 0.207553 and 0.310569 <
cod2 < 0.361653:
                        print("Shoot and Land")
                        ser.write(750)
                        bebop.fly_direct(0, 5, 0, 0, 1)
                    elif icx < 300:
                        print("GOING LEFT")
                        bebop.fly_direct(0, 0, -14, 0, 1)
```

```
                elif icx > 556:
                    print("GOING RIGHT")
                    bebop.fly_direct(0, 0, 12, 0, 1)
            elif icy < 100:
                bebop.fly_direct(0, 0, 0, 15, 1)
                print("Adjusting height, UP")
            elif icy > 380:
                bebop.fly_direct(0, 0, 0, -12, 1)
                print("Adjusting height, Down")

        else:
            print("Looking for the Nest")
            if self.countindex3 < 3 and self.inspect == 1:
                if self.ykaindex == 0:
                    self.v += 1
                    print("Height", self.v)
                    if 2 <= self.v <= 3:
                        bebop.fly_direct(0, 0, 0, 100, 1)
                        print("2 - 3, Increasing Height")
                    elif 4 <= self.v <= 8:
                        bebop.fly_direct(0, 0, 0, 80, 1)
                        self.ykaindex = 1
                        print("4 - 8, Increasing Height")
                    elif self.v == 9:
                        self.index = 1
                        print("Going Down")
                elif self.ykaindex == 1:
                    if self.inspect1 == 0 and self.inspect2 < 3:
                        self.inspect2 += 1
                        if self.inspect2 == 3:
                            self.inspect2 = 0
                            self.ykaindex = 0
            if self.inspect == 6:
                print("Inspection Algorithm")
                if self.inspect1 == 0 and self.inspect2 < 10:
                    print("Looking for position at the center")
                    self.inspect2 += 1
                    if self.inspect2 == 10:
                        self.inspect2 = 0

def demo_user_code_after_vision_opened(bebopVision, args):
    bebop = args[0]
    res = args[1]
    print("takeoff")
    # bebop.safe_takeoff(5)
    # bebop.smart_sleep(2)

    if bebopVision.vision_running:
        bebop.set_video_stabilization('roll_pitch')
        # print("Moving the camera using velocity")
        # bebop.pan_tilt_camera_velocity(pan_velocity=0, tilt_velocity=-20,
duration=4)
        print("Entering Res")
        print("I am", res)
        print("Leaving Res")
        print("land")
        # bebop.safe_land(5)

        print("Finishing demo and stopping vision")
        bebopVision.close_video()

    # disconnect nicely so we don't need a reboot
```

```
    print("disconnecting")
    bebop.disconnect()


if __name__ == "__main__":
    # make my bebop object
    bebop = Bebop()

    # connect to the bebop
    success = bebop.connect(10)

    if success:
        # start up the video

        print("Setting Parameters")
        bebop.set_max_altitude(5)  # Assume height above tree
        bebop.set_max_distance(10)  # for later, so that it doesn't go beyond
        bebop.set_max_vertical_speed(0.5)
        bebop.set_max_tilt(10)
        bebop.set_max_rotation_speed(90)

        res = 'YAHYA'
        bebop.set_video_recording('quality')
        bebopVision = DroneVisionGUI(bebop, is_bebop=True,
user_code_to_run=demo_user_code_after_vision_opened,
                                     user_args=(bebop, res))
        userVision = UserVision(bebopVision)
        bebopVision.set_user_callback_function(userVision.save_pictures,
user_callback_args=None)
        bebopVision.open_video()
    else:
        print("Error connecting to bebop.  Retry")
```

D. Interviews

Pine trees have always been one of the critical components of the Lebanese nature. Lebanon is characterized first by its Cedars, and second by its Pine trees. If any of those two were in danger, then the entire Lebanese natural habitat is in danger. The Pine trees are being defoliated by an insect called the Pine Processionary Moth. It is a dangerous insect that is capable of eating the tree in only three months.

Our FYP aims to help cure the problem that is facing the Pine trees. We aim to implement computer vision on drones and with the help of AI to detect the insect that is killing the pine trees and kill it using a specific strategy.

We interviewed several people including entomologists, a professional in humanitarian issues, farmers, tourists, and Lebanese citizens to understand the problem fully and its impact on the trees and other issues.

a. Supply chain

Throughout its life cycle, the pine processionary moth proved its ability to affect the environmental equilibrium. Worldwide, the pine processionary moth attacks both stone and wild pine trees. However, it prioritizes wild pine trees in Lebanon knowing the dominance of such tree species locally. The pine processionary moth can kill a wild pine tree in 3 months. Thus, knowing that most Lebanese farmers vaccinate wild trees to utilize their pine yield, this insect decreases the total yield which causes an increase in the pine's prices and thus, decreases the total demand as low/mid income people decrease their utilization of pine. This section will present the outcome of interviewing farmers representing the supply side and low income people who represent the demand side.

A. Supply side, consolidated outcome of two interviews:

1. How much do you rely on wild pine in the production of pine products?

The yield of wild pine trees presents a significant percentage of the total production. This is because, wild pine trees require less attention and less labor costs than stone trees especially with the global warming issues presented worldwide. As global warming increases, we believe that the dependence on wild pine trees will increase.

2. Knowing this dependence, how is the pine processionary moths effecting your trade activities?

Along with leptoglossus occidentalis, a parasite that sucks the seeds of pine, the pine processionary moth decreases our total supply capacity, a problem which decreases the potential income from this trading activity.

3. Can you say that the pine moth affected your living standards negatively?

Yes. And the issues is that year after year, the insect is expanding to other trees which embarks a red flag for us.

4. But, did you follow any regimen to treat this moth?

Yes, we asked for the help of entomologists and parasites specialists. Their intervention was not significant as their solutions are not that effective to limit the growth of such insect. The moth grows exponentially this year.

5. Did you report any injury or infection on human beings or animals because of this moth?

Yes, but not that frequent on human beings because upon the detection of the moth's nests, we prefer to stay far away.

6. Is it easy to detect the presence of the moth?

No, unfortunately, we cannot detect the moth that easily. What happens is that we wait until the nest shows by January where the larvae reach the final stages of their growth period.

<u>Those interviews present a good argument that defend the scope of our project and solution.</u>

B. **Demand side, low income**:

1. How often do you rely on pine in food preparation?

Usually, pine is something that my family enjoys in food. I usually put pine products in Kebbeh, Rez Ala Djej, etc. However, the current increase in the pine's prices made it hard for us to afford it.

2. What are some substitutes that you are using and are they causing the same degree of leisure?

It is either pine or nothing. I tried using nuts before. However, my family didn't like food as much as they did with pine.

3. Do you wish that this problem resolves rapidly?
Definitely!!!

Such interview was a critical factor in helping us decide on pine against Cedars as a preliminary proof of concept. We plan to establish a solution for this global issue that is primarily led by Global warming by the end of February.

b. Green Trips and Tourism

Because we still aim to target the issue facing Cedars after finishing that of the pine, we believed that understanding the effect of such parasites on the visitor activity in nature parks and natural reserves. As we were unable to collect such data given the time constraints on board, we ought to look for the other side of the coin – tourists and local people who would like to spend a day in nature for a picnic, for fun, etc. – for the time being. This section interviews one Lebanese resident and another tourist (network of one of the team's members from his internship) who value the Lebanese nature a lot.

A. The resident and the tourist consolidated due to the massive similarity in arguments

1. Why do you prefer the Lebanese nature over others?
Resident: Usually, it is because of Cedars. Cedars represent our national flag.

Tourist: Cedars are so significant for you, Lebanese, as they are demonstrated in your flag. Also, knowing that Lebanon is known for its beaches and mountains, visiting the mountains in both seasons presents some leisure as it is a way to escape from consulting problems, listen to birds, and chillax.

2. When was your last visit?
Resident: My last visit was almost 2 years ago before entomologists announced an 80% penetration rate of a parasite in Cedar forests. However, just for clarification, my total visits to nature decreased because of the increasing number of parasites that I am afraid of getting infected from.

Tourist: Never been there before however, I would have liked to do so before reading an article about an insect that infected your forests. Enjoying the beach is a better option for the time being!

3. What if you visited a pine forest, would you enjoy?
Yes, However, the long-lasting problem of the nests and the moth that is infecting pine trees makes us hesitant to that activity.

This interview is a clear call of action as they provide us with a foreseeable insight that shows the tourist activity to natural reserves and resources will decrease with time especially that we they represent a source of income for both public and private entities.

c. Datasets and Computer Vision

For the detection of the insect, we'll use machine learning (ML) and computer vision (CV). One of the key components of ML and CV are datasets. Without datasets, we won't be able to do any detection or identification of the insect. Therefore, we set interviews with an entomologist and professors working in ML and CV areas.

1. Do you have any resources for the pine processionary moth?

Entomologist: Yes. I have pictures and a couple of videos about the moth. I will send my images that I have. I've been working in the field in a while and I'll send my own datasets that I collected.

Professors: No.

2. Are there any other resources for the datasets?

Entomologist: Yes, you can search in http://ecocrop.fao.org/ecocrop/srv/en/cropFindForm and http://ecoport.org/ep.

Professor: Yes, I'll provide you with the best websites that can help in collecting datasets and may contain datasets for the PPM. Some of the websites are https://www.data.gov/, https://data.worldbank.org/, https://cloud.google.com/bigquery/public-data/, https://www.quandl.com/, and others.

3. Do you think that with those resources we can find enough data for training, testing, and validating our vision algorithm?

Entomologist: Those are the main databases that I know that contains lots of images for the actual PPM. It should be sufficient. Else, you should go and get your own images.

Professor: Of course! Those are the biggest datasets in the world and will have a lot of resources in addition to metadata and analytics that maybe useful for you. You must find datasets about the PPM.

4. How many images are required to ensure the validity of our images? Is something between 500 and 1000 images good?

Both: There is no specific range, but the bigger the dataset is the better and more efficient the algorithm is.

After conducting the interviews and searching for sources, we actually found a lot of datasets for the moth itself and its nest. Yet, they weren't organized, and we realized that we need more data and thus need to gather our own data as the entomologist suggested. So, we're going to suffice ourselves and collect as much images as we can from the net, and then try to take our own images from reserves and trees infected by this moth.

d. Behavior of the Moth and the Ecosystem

In the past, experts in the agriculture field in Lebanon were heavily dependent on spraying pesticides all over the trees, preventing the growth of the insects that are harming and feeding off the trees. In 2006, spraying pesticides on trees have become prohibited worldwide due to its negative effects on the environment. Spraying would affect beneficial insects and other harmless animals in nature thus causing an unbalanced ecosystem. Due to this issue, experts are seeking new solutions for killing this insect.

As a first step, it is crucial for us to understand the behavior of the moth itself, its lifecycle and its relation with the ecosystem. An interview was conducted with an entomologist and the following questions were answered:

1) Why isn't spraying pesticides on trees efficient in killing the moth?

The issue isn't concerned with efficiency, it is quite larger than that. The problem is that spraying pesticides on trees became prohibited worldwide and we stopped applying this method in Lebanon. This ban started from 2006, and since then we only were allowed to spray biological pesticide in the soils and place sticky paper traps on trees to kill the moth which is not an efficient method to kill them.

2) And why is spraying on trees became prohibited?

Spraying on trees indeed targets the harmful insects, but not to forget that it targets the harmless ones as well. Additionally, the spray might reach animals due to drift, and thus creating an unbalanced ecosystem.

3) Why is the moth harmful to the Pine trees?

First of all you should now that the PPM only targets Wild Pine and not Stone Pine. Second, when the PPM is in the larvae stage of its lifecycle, it feeds on the bottom parts of the needles causing complete defoliation of the tree in only 3 months.

4) How do you think is the best way for targeting these insects if spraying is now prohibited?

You have 3 options. You can target the eggs of the Moth early in its lifecycle and eliminate the root cause. Another method is applying precision spraying on the larvae which instantly kills them. The last option is targeting the nests of the larva to kill them. You should know that the larva build nests in the winter to protect itself from the cold air; by targeting the nests and exposing them to the cold air you are basically killing them.

5) Is it enough to puncture the nest with a stick in order to kill them?

Yes, a small hole in the nest is enough for the cold air to enter and kill them all. But the nest is quite strong, stronger than a spider's web for example. The stick should be pointy enough to puncture the nest.

6) Are the insects harmful to the trees only?

Absolutely not! Other than harming Pine trees, these insects are harmful to humans and animals in a direct manner. These insects carry diseases on its hairs which can transfer to humans and animals. Serious skin rashes are some of the symptoms that are found on people, and it might lead to death in severe cases. Most people as well as their pets catch these diseases mostly when in parks due to the high number of Pine trees.

This interview gave us information about the lifecycle and behavior of the moth. The entomologist also gave us some options for targeting the moth to kill it.