DRDO SASE's
UAV Fleet Challenge
Mid-Evaluation Report

॥ सा विद्या या विमुक्तये ॥
भारतीय प्रौद्योगिकी संस्थान धारवाड
Indian Institute of Technology Dharwad

The Problem Statement

The problem statement defines a task to identify and locate 4 out of 5 green cuboids of dimensions 15cm x 15cm x 15cm (among other random objects) in a field of 40m x 40m using 3 or 4 UAVs.

We are planning to tackle the problem with 3 quadcopter drones flying at a height between 1 to 2 meters. Following are the key points which we aim to achieve working on the given problem statement:

1. Three quadcopter drones capable of independent flight and avoiding collisions.
2. An image processing system mounted onto each drone to recognize the targets.
3. A communication system between every pair of drones in the system sharing the spatial location of each other.

The Quadcopter Drones

It has been decided upon by the team to design and build 3 identical quadcopter drones. Each drone would be capable of an independent planned flight, processing the images received from the camera mounted on it, and communicating with each other while avoiding collision. Here we elaborate on the primary components of a drone:

1. **Chassis:** An aluminum and 3D printed chassis. (We may also use a readymade chassis)
2. **Propellers:** Four 10 inch propellers
3. **Flight Controller:** APM (Ardupilot Mega) 2.8 flight controller
4. **Motors:** A2212 13T Brushless motors
5. **Motor controllers:** Emax BLHeli 30A ESC
6. **Distance sensors:** Ultrasonic sensors and/or laser range finder
7. **Onboard Processor:** Raspberry Pi 4 (For image processing and communication)
8. **Camera:** Raspberry Pi Camera Module V2
9. **Battery:** 5000mAh LiPo battery
10. **GPS Module:** Ublox Neo 7M GPS module
11. **Wifi Module:** ESP8266 for sending and receiving data.

Approach used for Object Detection:-

Method and framework used:- YOLO and DarkNet frameworks.

How does the above method works :- The system models detection as a regression problem. It divides the image into an **S × S grid** and for each grid cell predicts **B bounding boxes** (defined in the following paragraph), **confidence** for those boxes, and **C class probabilities**. These predictions are encoded as an **S × S × (B ∗ 5 + C) tensor**.

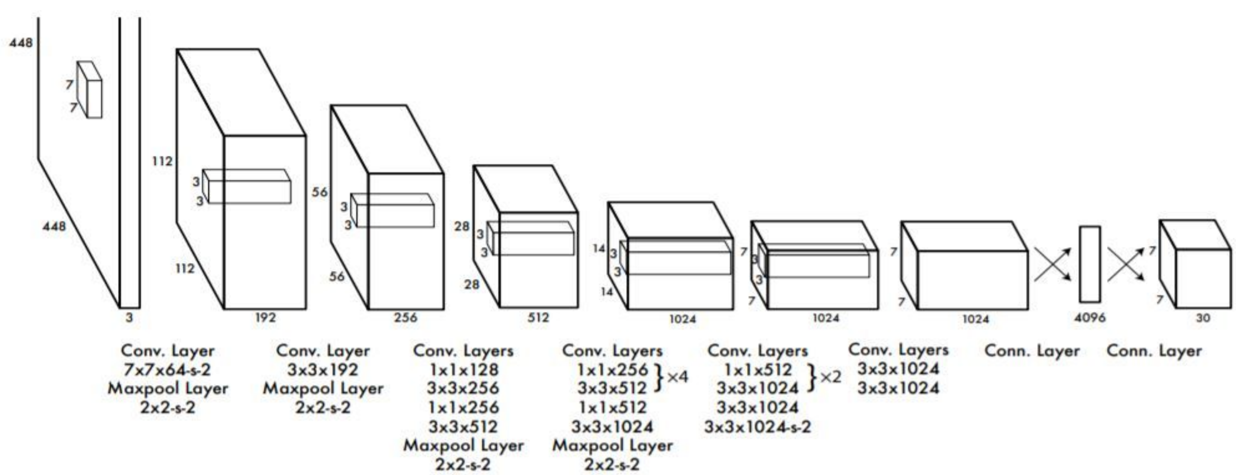
The bounding box consists of 5 predictions: **x, y, w, h, and confidence**. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. The width and height are predicted relative to the whole image.

The confidence or confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. If no object exists in that cell, the confidence scores should be zero.

The detection network has **24 convolutional layers** followed by **2 fully connected layers**. Alternating 1 × 1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution and then double the resolution for detection.

Finally the confidence prediction represents the intersection over union(IOU) between the predicted box and any ground truth box. The final output of the network is the S × S × (B ∗ 5 + C) tensor of predictions. The Darknet framework is used for all training and inference about the image.

A sample figure illustrating the procedure is shown:-



Why this method is better than others/ Why we chose this method : -

1) It is more accurate and involves real-time approach to object detection based on convolutional neural networks.
2) The network can simultaneously perform classification so that in a single step it recognizes the object and finds a good grasp rectangle/bounding box.
3) Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance.
4) Even the base version of YOLO model processes images in real-time at 45 frames per second, which is quite good.
5) YOLO makes more localization errors but is less likely to predict false positives on background.
6) It outperforms other detection methods, including DPM and R-CNN by a large margin in computer vision problems such as classification and detection.
7) It avoids the computational costs of running a small classifier numerous times on small patches of an image. We harness the extensive capacity of a large convolutional network to make global grasp predictions on the full image of an object.

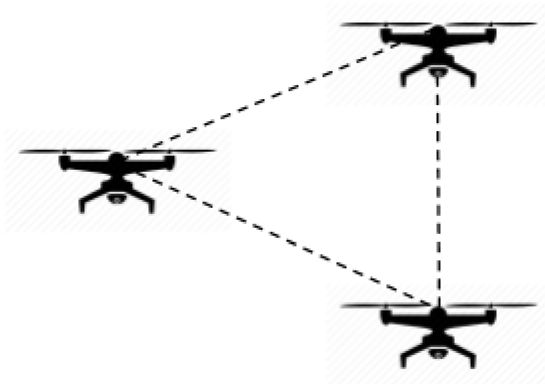
Communication and Collision avoidance

Communication between nodes (here drones) is a key feature of swarm technology. We're planning to share the spatial location and a positive/negative signal to signify the identification of one of the green cuboids among the nodes. Current drone swarm communication algorithms are mainly of two types:-

- **Flooding:** Data is shared with all nodes. It requires more energy.
- **Routing**: Data is shared along a path by hopping from node to node. It requires more computing power.

We are using the flooding method as it can run faster on an embedded device like Raspberry Pi. Raspberry Pi is connected to NodeMCU which has the ESP8266 WiFi module onboard which can be used for transmitting and receiving data over the network. Raspberry Pi interprets the data and sends commands to the flight controller to control the drone movement accordingly. NodeMCU has been selected as it has good range, bandwidth.

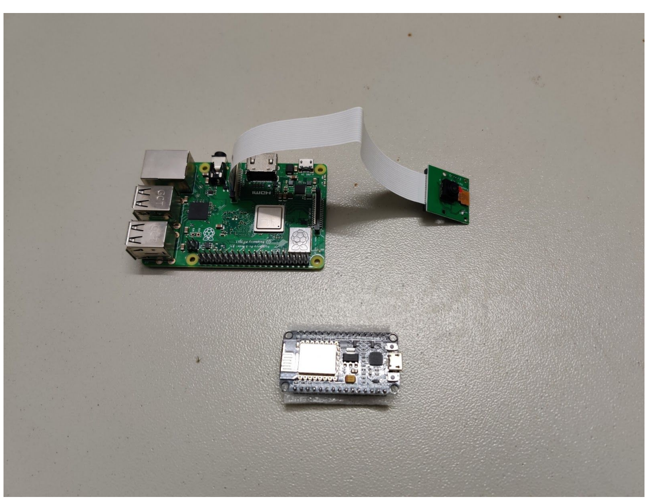
We'll be using "De-centralized drone swarm architecture" for the network. An illustration is depicted below:-



- Each drone updates its observed parameters to the other two drone members regularly after a certain interval of time and this data is being stored in those two drones also.
- Also, total observed area by all three drones is mapped as a region which will be displayed on a computer.
- The objects to be found over the region will be updated over the common database shared between the drones.

Current Progress

Currently we’ve built a drone which is capable of independent flight. Also, the camera and Raspberry Pi and camera setup is being worked upon. Some pictures are shown below.



Raspberry Pi and camera with NodeMCU (left) and the quadcopter drone (right).