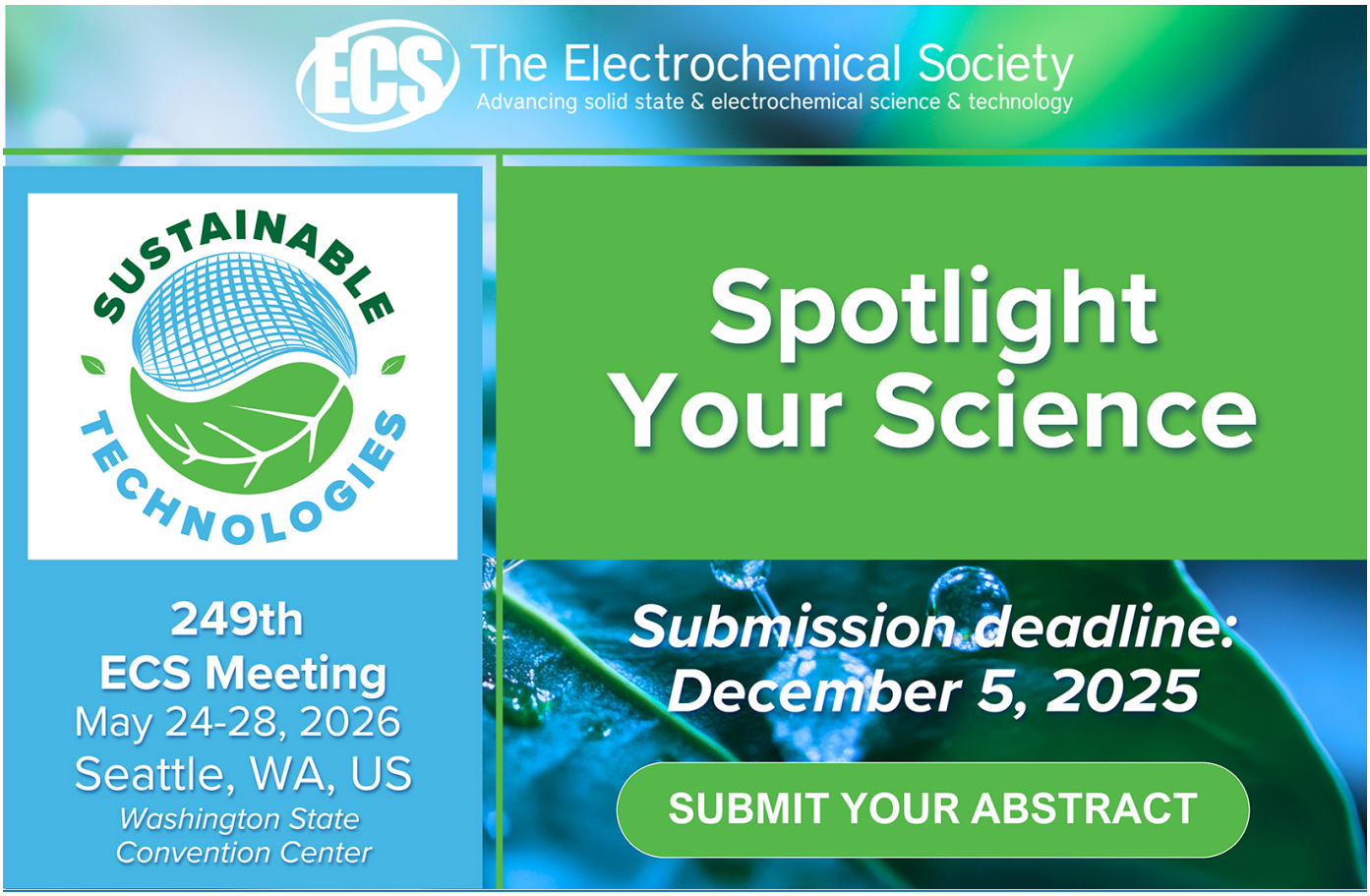
Journal of Physics:
Conference Series

PAPER • **OPEN ACCESS**

Drone to Obstacle Distance Estimation Using YOLO V3 Network and Mathematical Principles

To cite this article: N Aswini *et al* 2022 *J. Phys.: Conf. Ser.* **2161** 012022



Drone to Obstacle Distance Estimation Using YOLO V3 Network and Mathematical Principles

N Aswini[1], S V Uma[2] and V Akhilesh[3]

1. Assistant Professor, Department of ECE, CMR Institute of Technology, Bengaluru.
2. Professor, Department of ECE, RNS Institute of Technology, Bengaluru.
3. Deep Learning Engineer, Flux Auto Inc., Bengaluru.

aswini.n@cmrit.ac.in

Abstract. Now a days, drones are very commonly used in various real time applications. Moving towards autonomy, these drones rely on obstacle detection sensors and various collision avoidance algorithms programmed into it. Development of fully autonomous drones provide the fundamental benefits of being able to operate in hazardous environments without a human pilot. Among the various sensors, monocular cameras provide a rich source of information and are one of the main sensing mechanisms in low flying drones. These drones can be used for rescue and search operations, traffic monitoring, infrastructure, and pipeline inspection, and in construction sites. In this paper, we propose an onboard obstacle detection model using deep learning techniques, combined with a mathematical approach to calculate the distance between the detected obstacle and the drone. This when implemented does not need any additional sensor or Global Positioning Systems (GPS) other than the vision sensor.

1. Introduction

Unmanned Aerial Vehicles (UAVs) or Drones are being used widely in Military and Civil applications. In Military, they are being used for Surveillance, Reconnaissance, Intruder tracking and border patrolling. In Civil and commercial sector, they are being used for Search and Rescue, Disaster management, Precision agriculture, Infrastructure inspection, Film shooting etc. The visual information acquired by the onboard video cameras are processed to perform these operations. Visual cameras provide a lot of information about the surroundings and are easier and light weighted compared to other sensors such as RADARs and LIDARs. Among the two control mechanisms of a drone – Tele operation and Autonomous, the latter one being the focus, a great amount of research work is going on to incorporate Image processing and Computer vision algorithms to detect obstacles during navigation. Obstacle detection has been a hot area of research in Computer vision from last few years. The need for real time obstacle detection has provided the way for faster deep learning techniques compared to conventional feature extraction techniques using Scale Invariant Feature Transform (SIFT) [1], Speeded Up Robust Features (SURF) [2], Object Oriented BRIEF (ORB) [3] etc. There are three different tasks in computer vision, which need to be distinguished before starting with obstacle detection algorithms. They are Object classification, Localization and Detection.

- Classification- When an image is given to us, we will tell what kind of object it is. We will assign a class label to the object.
- Localization- When an image is given to us, we not only tell the kind of object it has but also locate it in the image. We will draw a bounding box around the one or more objects in the image.
- Detection- It combines the task of classification and localization and draws bounding box around the objects of interest and will assign a label class to them.

The various Deep Learning (DL) algorithms which are used in obstacle detection can be broadly classified in to two categories- **Two stage** approach and **Single stage** approach. In two stage approach, Region Proposal Networks (RPN) are generated as a first step and then these generated RPNs are passed on to the second step for object classification and bounding box regression. The RPN generates the Region of Interest (ROI) as an initial step. The DL algorithms which work on this method are RCNN, Fast RCNN and Faster RCNN. They are highly accurate but are very slow and cannot be used in real time applications. In Single stage approach, object detection is taken as a regression problem and in one pass itself, object classification and bounding box coordinates are given. One of the DL algorithms which work using single stage approaches is You Only Look Once (YOLO) [4].

Mere detection of obstacles is not enough to avoid them during autonomous navigation. The decision whether to move ahead or divert in order avoid these obstacles need to be taken at appropriate point of time. The drones can utilize distance measurement sensors like ultrasonic sensors, laser sensors or infrared proximity sensors and use the information provided by them to any obstacle avoidance algorithms. Ultrasonic sensors have limited range and low bandwidth, and the readings may not be very accurate. Laser sensors are very expensive and are bulky. Infrared sensors can be affected by any IR light source and are good only for indoor navigation.

In the case of Low flying Drones, the obstacles can be static like buildings, trees, walls, pillars or dynamic like birds, human beings, and vehicles. A real time detection of these obstacles and estimation of their distance from drone is an extremely difficult task when we rely only on the monocular images obtained from the camera. To address this difficulty, the approach we have used is to detect obstacles using Deep Learning algorithms and then by applying Camera-lens principle, the distance between the detected obstacle and drone camera can be estimated. There is no need of any additional sensor to calculate distance. For demonstration purpose, we have used self-generated videos taken using a low flying quadcopter drone equipped with a monocular front facing camera. Three scenarios are considered here: - (1) The quadcopter is made to fly with a slow speed of 1-2 m/s towards a car parked at 30 m from the drone. (2) The quadcopter and car are moving towards each other at a slow speed. (3) The quadcopter is made to fly towards Multiple static frontal obstacles (Car, Motorcycle, and a Person).

This paper is divided into 5 sections. In section 2, a detailed survey on obstacle detection using deep learning algorithms is performed. Section 3 gives the proposed methodology and the mathematical equations used in calculating distance between the detected obstacle and drone camera. Section 4 consolidates the results obtained and Section 5 gives the conclusion and future enhancements followed by references.

2. Literature Review

The research on object detection using computer vision started from the very old face detection using Viola Johns Algorithm (Viola et.al 2001) [5]. Object detection is a generic term, it can be face detection, pedestrian detection, vehicle detection etc. When it comes to autonomous navigation of

vehicles, the object detection can be called as "obstacle detection". There are a lot of research publications on obstacle detection using traditional feature extraction algorithms in the last 20 years. Post Yr. 2015, due to the advancement in Machine learning and Deep learning, researchers started switching over to Convolutional Neural Networks for feature extraction and classification to accurately locate the obstacles. An optical flow measurement-based navigation system for an unmanned vehicle through sub urban areas has been proposed by Pooja et.al (2017) [6]. A frontal camera captures images continuously and optical flow is measured for the sequence of images. A guidance command is given to the vehicle depending upon the optical flow measured between the left and right-side sub images. The turn rate of the vehicle follows an inverse relationship with the optical flow measured. They have considered different navigation areas such as a straight path, T shape, L shape and a combination of L and T. The entire simulation is performed by creating a virtual 3D environment in MATLAB.

A different approach from the conventional methods of optical flow estimation and feature extraction has been proposed using CNN by Chakravarty et. al (2017) [7]. They have collected a set of RGB images from online and by their own to train a Deep CNN for predicting the depth maps. Supervised training of the network has been performed using the collected dataset. By accepting the depth maps, the quadcopter is steered away from the frontal obstacles by predicting the angular velocity. Their main contribution is in training the network for generating the depth maps from the input images. An indoor simulation has been performed using a Parrot Bebop Drone controlled using a Laptop with NVIDIA GPU. The entire process of depth map and control command generation is performed in the laptop. By incorporating a Rein-forced learning module to a detection module, an obstacle avoidance mechanism is proposed by Zhaowei Ma et.al (2017) [8]. The obstacle detection module extracts monocular visual cues using a Deep CNN. They basically imitate a human visual system to estimate the location of obstacles in the Field of View of the UAV. The moving trajectory of flying obstacles is predicted using a Kalman Filter. The research work is validated by performing a software simulation using Xplane by opting the forward view pattern for the aircraft. The data communication between the software simulation tool and flight control module is done in a ground control station. The implementation is done in MATLAB with an additional GPU for acceleration.

Valencia et.al (2018) [9] describes how to locate a possible frontal obstacle by generating a salient map using histogram back propagation algorithm. The videos taken by a monocular camera mounted on a quadcopter sends the video signals to a ground station via wireless connection. The map generated by the algorithm is divided into squares of fixed size and the intensity within the square is calculated for determining whether an obstacle is present or not. Experimental analysis is performed by simulating a parrot quadcopter using Gazebo and Robot Operating software. The proposed idea covers various indoor and outdoor scenarios. Another software simulation for drone obstacle detection using Gazebo has been performed by Bumsoo park et.al (2018) [10]. They have collected various image and control data from human flight demonstrations. A simulated drone is steered by a human expert is utilized for this task. This accumulated data is used to train a Neural Network for imitating human expert decisions. Accordingly, the drone is made to move Left, Right or Straight. For the 3D CNN architecture, four convolutional layers and two max pooling layers are used. At the end, a softmax pooling layer is added. The CNN model has learnt a general rule for obstacle detection and avoidance using a monocular camera.

A Deep Reinforcement Learning (DRL) model is proposed by Cetin et.al (2019) [11] for the navigation of drones. With the help of Airsim simulator and Unreal Engine, several obstacles such as trees, parked cars, other moving drones, and houses are created for experimentation. A drone with frontal camera captures depth images continuously. The DRL takes the actions performed by the drone and give those inputs to the environment created. The interaction between the drone and the environment takes place at each step which is the basis of reinforcement learning. A virtual barrier is

created in the geographical area which will keep the drone at a specified area. If this area is violated, suitable warning signals are generated to avoid probable collisions with the obstacles. In Sang-Yun Shin et.al (2019) [12], diverse Reinforcement Learning (RL) is utilized for drone obstacle avoidance. Two control spaces, discrete and continuous are experimented using a Deep Q network. RGB and Depth map images of various indoor and outdoor scenarios are collected from a simulated environment created using Airsim. A combination of U-net based segmentation and actor-critic method of learning is adopted by them. The drones are trained using several RL algorithms in three different environments such as wood land, block land and arena world. Optical flow between sequential images is calculated for training the U net-based segmentation network. Training and testing is performed with Intel i7 CPU and NVIDIA Titan X GPU.

A multi hidden Neural network, named as DisNet has been trained using a supervised learning method by Haseeb et.al (2018) [13] for predicting distance between drone camera and obstacle. The output parameters of a You Only Look Once (YOLO) classifier trained on COCO dataset has been used to detect the obstacles. The training of DisNet is performed using a set of RGB images collected in a railway scene for possible static obstacles on the railway track. In a very work, Huang et.al (2020) [14] has also chosen YOLO object detector to sense and avoid obstacles for a small, unmanned vehicle. Without using any other sensing mechanism, they have tried to calculate the distance of the obstacle from the drone camera. The frontal obstacles are detected by the YOLO detector which is pre trained with COCO dataset and a set of custom images. The detected images are cropped to 100 x 100 size and passed on to a Deep Neural Network regression model for estimating the distance.

All the aforesaid research works are carried out in the last three years. They either rely on optical flow estimation and feature extraction techniques or supervised and RL models by training a deep CNN. Most of them have evaluated the performance in a simulated environment. The work presented in this paper is a step forward in obstacle detection and distance estimation using real time video/images collected and fed into a YOLO V3 network. Once detected, distance between camera and obstacle is calculated. There is no need of any additional sensor other than a monocular camera. The approach can work in real time for various environmental scenarios and can detect static and dynamic obstacles.

3. Proposed Methodology

Generally, Deep Neural Networks can be used to detect various types of object classes from an input image. The earlier existing RCNN, Fast RCNN and Faster RCNN could also detect, but not suitable for real time applications. We have used a single stage deep neural network – YOLO V3, pre trained with COCO Dataset [15]. YOLO V3 is much faster than its predecessors YOLO V2 and V1. In V3, the feature extractor used is a deep neural network called DarkNet-53. The Darknet-53 architecture uses 53 Fully Connected layers each of them followed by batch normalization layer and Leaky ReLU activation layer. There are no pooling layers, and the feature map is down sampled with a stride 2 in convolutional layer. This prevents the loss of low-level features often attributed to pooling. It is invariant to input image size. The Darknet model with different layer distribution is shown in figure 1.

The algorithm will apply a single Convolutional layer for the complete image and then divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The steps used in YOLO V3 is depicted in figure 2.

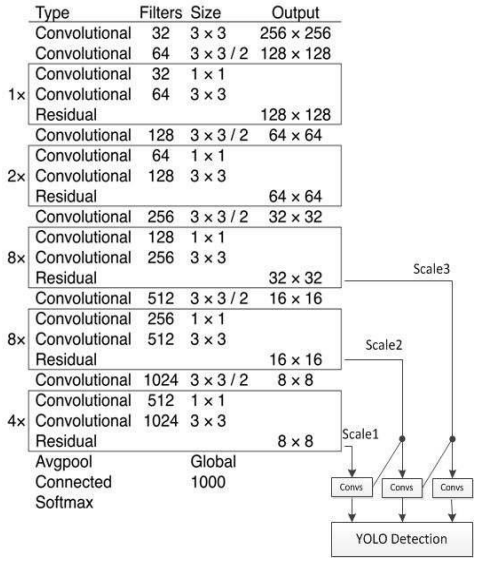


Figure 1: Darknet – 53 model

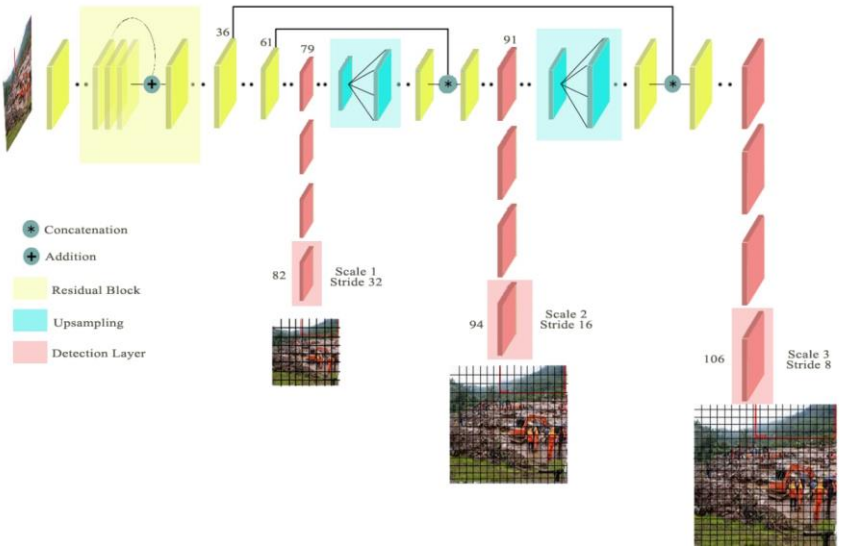


Figure 2: YOLO V3 Architecture

The input image can be of any size. There is a pre-processing module which will resize the image into 1024x720 using Nearest Neighbour (NN) interpolation. NN selects the value of the nearest pixel rather than finding average or other weighting functions. In YOLO V3 architecture, there are 106 convolutional layers. Among these 53 layers are pretrained and the rest 53 can be used for training. The first detection is made by the 82nd layer having stride 32 which results in a feature map of size 13 x 13. The second detection is made by the 94th layer having stride 16 which results in a feature map of size 26 x 26. The third detection is made by the 106th layer having stride 8 which results in a feature map of size 52 x 52. The residual layer is used as skip connections to jump between any layers. The 13 x 13 map obtained after the first prediction helps to detect larger objects present in the image while the 52 x 52 feature map helps to detect smaller objects whereas medium objects are detected by a 26 x 26 feature map. For an input image of size 416 x 416, YOLO V3 predicts (52 x 52 x 3) +(26 x 26 x 3) +(13 x 13 x 3) =10647 Anchor boxes. Three anchor boxes are used for the three scales in YOLO V3. So, in total there are 9 anchor boxes. The anchor box contains 5 vectors which compose of <x, y, w, h,

c>, where 'c' represents the presence or absence of an object in the grid defined as the confidence. The bounding box coordinates are t_x, t_y, t_w and t_h. If the cell is offset from the top left corner of the image by (c_x, c_y) and the bounding box prior has width and height p_w, p_h, then the predictions correspond to the predicted box (depicted in blue in figure 3) with new coordinates as b_x, b_y, b_w and b_h.

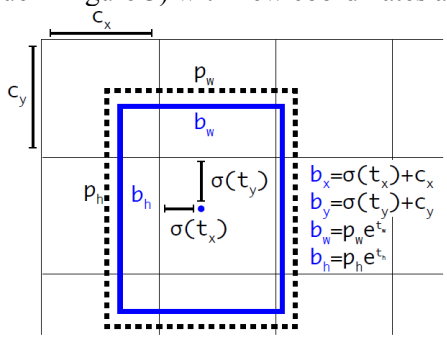


Figure 3: Bounding Box

It predicts an objectness score for each bounding box using logistic regression. The anchor boxes are arranged in the descending order. The Intersection Over Union (IOU) of the class scores of the adjacent anchor boxes are calculated and compared with threshold value and if it is less than that value then it will be replaced with 0. The class score values are calculated for all the other classes. So, each anchor box will contain class scores equivalent to the number of classes. After drawing bounding box around the object of interest, next step is to calculate the distance between drone and obstacle. As explained in section 1, two scenarios are considered here: -

1. The drone is made to move towards a parked car.
2. Both the drone and the car approaching each other.

To measure the distance between camera and obstacle, a mathematical relationship between the Real Height of the Object, Image height, and focal length of the camera has been used. The image height can be obtained from the bounding box drawn around it. The flow chart given in figure 4 shows the various steps involved in detecting static and dynamic objects and distance calculation.

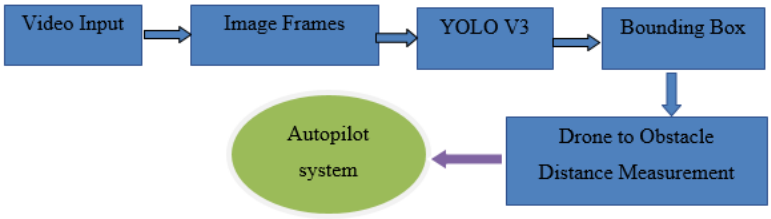


Figure 4: Obstacle Detection and distance measurement using YOLO V3

A camera – lens diagram is given in figure 5 where D represents the distance from object to camera, f the focal length (all dimensions are calculated in milli meter). From the Lens diagram, Δ AOB and Δ COD are similar right-angled triangles. So, their Opposite side/Adjacent side ratios are equal. i.e.,

$$\frac{Sensor\ Height\ (mm)}{Focal\ Length\ ,f\ (mm)} = \frac{Object\ Dimension\ (mm)}{Distance\ to\ Object\ ,D\ (mm)} \quad (1)$$

From mathematical principles, the Object dimension is given by,

$$Object\ Dimension\ (mm) = \frac{Real\ Height\ of\ Object(mm) * Camera\ frame\ height\ (pixels)}{Image\ Height\ (pixels)} \quad (2)$$

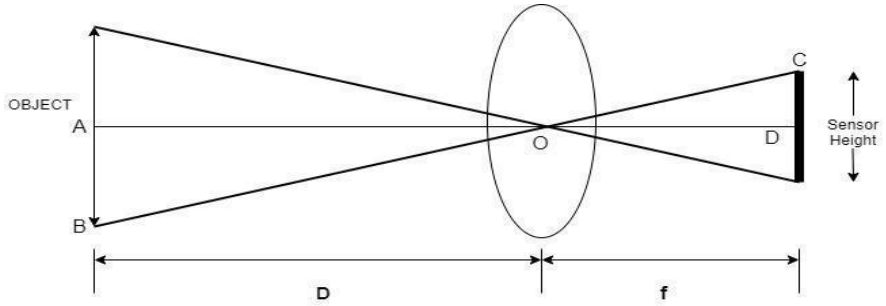


Figure 5: Distance from camera to object calculation

So, combining these two equations, the Distance to Object, D is given by: -

$$Distance\ to\ Object = \frac{Real\ height\ of\ Object(mm) * Camera\ Frame\ height(pixels) * f(mm)}{Image\ Height(pixels) * Sensor\ Height(mm)} \quad (3)$$

Using this basic lens principle, the distance of the obstacles from the drone camera can be calculated. We have used a car with a height of 1666mm. The parameters for the drone camera used have been tabulated in table 1.

Table 1: Parameters used for Obstacle Detection

Parameter	Value (Units)
Focal Length of Drone Camera	24 mm
Camera Frame height	1080 Pixels
Real height of the Obstacle (Car)	1666 mm
Sensor height	13.3 mm

The two videos taken are converted into frames and is given to YOLO V3 algorithm for object detection. The maximum distance maintained between the parked car and drone is 30 m. Using the YOLO V3 algorithm, bounding box has been drawn around the obstacle detected. The real height of the car being known **to** us, an accurate measurement of distance to camera is possible by this proposed method.

4. Simulation Results and Discussions

The entire simulation is performed in Google Colab. Every time, the real height of the obstacle is given as input to the program. More than one obstacle can also be detected at the same time. The video taken is converted into frames and is given to YOLO V3 algorithm for object detection. In the first experiment, only a static car is considered as the frontal obstacle. As the drone moves towards the car, the program is calculating the distance between the car and the drone camera. The run time snapshot for the same is given in figure 6. The distance measured for image frames from 1209 to 1222 is shown. In table 2 sample image frames with estimated distance and actual distance measured is compared. Figure 7 shows the distance displayed in millimetre when the obstacle is 9.943m away from drone.

In the second experiment, the distance is estimated when both drone and obstacle are moving towards each other is shown in figure 8. As the drone is approaching the moving obstacle, the distance between them is reducing. There is minimal error between measured distance and actual distance. It is of the order of 0.01m when it is at a very close distance of 9.94 meter (in the case of static obstacle) and zero error at a safe distance of 6 meter (in the case of dynamic obstacle). When the obstacle is 6 meters away from the drone, a warning signal need to be given to the autopilot system of the drone to

deviate from the straight-line path. The results obtained for an input video where both the obstacle and drone moving towards each other is given in table 3.

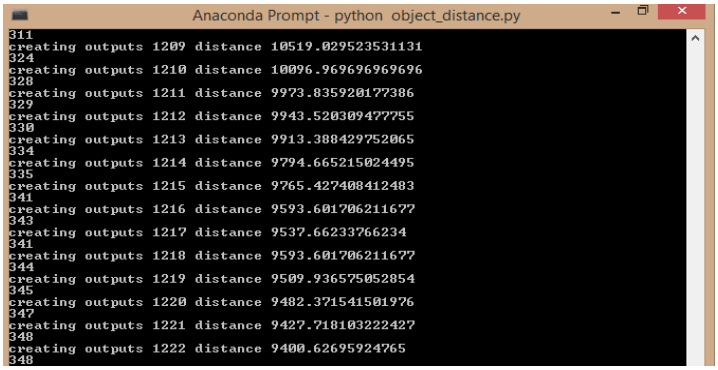


Figure 6: Run time snapshot of Distance measured in millimetre for image frames #1209 to #1222.

Figure 7: Runtime snapshot for frame#1212, distance to camera is 9.943m

Table 2: Estimated Distance versus Actual Distance in meters for static obstacle

Frame #	**980**	**1010**
Image Frame		
Estimated Distance (m)	19.70	18.69
Actual Distance (m)	20.00	19.00
Frame #	**1140**	**1212**
Image Frame		
Estimated Distance (m)	12.82	9.94
Actual Distance (m)	13	10.00

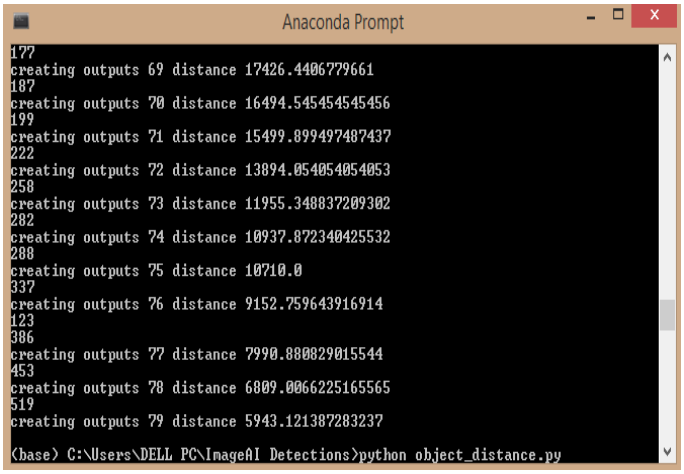


Figure 8: Simulation results for both Obstacle and Drone moving towards each other.

Table 3: Estimated Distance versus Actual Distance in meters for dynamic obstacle

Frame #	**71**	**75**
Image Frame		
Estimated Distance (m)	15.49	10.71
Actual Distance (m)	15.0	11.00
Frame #	**78**	**79**
Image Frame		
Estimated Distance (m)	6.80	5.94
Actual Distance (m)	7.00	6.00

Next, Object Detection and Distance measurement has been performed for multiple Objects. Here a Car, a Motorcycle and a Person are the obstacles to be detected and distance measured. Two image frames are shown in Figure 9. The distance measurement algorithm works well for the three obstacles (provided the real height is known to us). A comparison between the estimated distance and measured is given in table 4.

Figure 9: Two Image frames considered for multiple obstacles.

Table 4: Detection of multiple obstacles

Image Frame Number	Image Frame	Measured Distance (m)	Actual Distance (m)
1		Motor Cycle 13.0 m Person 13.8 m Car 14.2 m	Motor Cycle 12.5 m Person 13.0 m Car 15 m

2		Motor Cycle 2.8 m Person 3.2 m Car 5.3 m	Motor cycle 2.5 m Person 3.0 m Car 5.0 m

5. Conclusion and Future Enhancements

In this paper, Obstacle detection is performed using a single stage algorithm, YOLO V3 combined with a distance estimation method. The pre trained network has accurately detected the obstacle and estimated distance in three different scenarios.

- From a very safe distance of 30 m.
- When both the drone and the obstacle are moving towards each other.
- When multiple objects are there in front of the camera.

There is no need of any additional sensor or GPS to calculate the distance between the two. The incorporation of distance measurement module into a pre trained deep learning network is the main outcome of this research work. In future, we would like to perform custom based obstacle detection rather than using a pre trained network. The deep learning network should be trained using the real height of the various obstacles considered. A hardware implementation of the same can also be performed in the future.

References

[1] Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". International Journal of Computer Vision. 60 (2): 91–110.
[2] Herbert Bay, Andreas Ess , Tinne Tuytelaars , Luc Van Gool , “Speeded-Up Robust Features (SURF)”, Computer Vision And Image Understanding 110 (2008),346–359.
[3] E. Rubleeet al., “ORB: An efficient alternative to SIFT or SURF,” in IEEE International Conference on Computer Vision, Barcelona, ICCV, 2011, pp. 2564-2571.
[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.
[5] P. Viola and M. J. Jones, “Robust real-time face detection,” International journal of computer vision, vol. 57, no. 2, pp. 137–154, 2004.
[6] Pooja Agrawal et al., Inverse optical flow-based guidance for UAV navigation through urban canyons, 385 Aerosp. Sci. Technol. (2017), http://dx.doi.org/10.1016/j.ast.2017.05.012.
[7] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars and L. Van Eycken, CNN-based 387 single image obstacle avoidance on a quadrotor, *IEEE International*

Conference on Robotics and Automation 388 *(ICRA)*, Singapore, **2017**, pp. 6369-6374, doi: 10.1109/ICRA.2017.7989752.

[8] Z. Ma, C. Wang, Y. Niu, X. Wang, L. Shen, A saliency-based reinforcement learning approach for a UAV to avoid flying obstacles, Robotics and Autonomous Systems (2017),doi.org/10.1016/j.robot.2017.10.009.

[9] D. Valencia and D. Kim, Quadrotor Obstacle Detection and Avoidance System Using a Monocular Camera, 2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Singapore, 2018, pp. 78-81, doi: 10.1109/ACIRS.2018.8467248.

[10] Bumsoo Park and Hyondong Oh, Obstacle Avoidance for UAVs via Imitation Learning from Human Data with 3D-Convolutional Neural Networks, 10th International Micro-Air Vehicles Conference 22nd-23rd November 2018. Melbourne, Australia, 2018.

[11] E. Çetin, C. Barrado, G. Muñoz, M. Macias and E. Pastor, Drone Navigation and Avoidance of Obstacles Through Deep Reinforcement Learning, 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC), San Diego, CA, USA, 2019, pp. 1-7, doi: 10.1109/DASC43569.2019.9081749.

[12] Sang-Yun Shin, Yong-Won Kang and Yong-Guk Kim, Obstacle Avoidance Drone by Deep Reinforcement Learning and Its Racing with Human Pilot, Applied. Sciences. 2019, 9, 5571; doi:10.3390/app9245571.

[13] Muhammad Abdul Haseeb, Jianyu Guan, Danijela Ristić-Durrant, Axel Gräser, DisNet: A novel method for distance estimation from monocular camera, 2018. https://project.inria.fr/ppniv18/files/2018/10/paper22.pdf

[14] Z. -Y. Huang and Y. -C. Lai, Image-Based Sense and Avoid of Small-Scale UAV Using Deep Learning Approach, 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 2020, pp.545-550, doi: 10.1109/ICUAS48674.2020.9213884

[15] https://cocodataset.org/#homeMore references.