CNN-based Off-board Computation for Real-time Object Detection and Tracking Using a Drone

Sophan Wahyudi Nawawi[1], Ahmed Ashraf Mohamed Ahmed Abdou[1], Nur Azlina Abd Aziz[2*]

[1] Universiti Teknologi Malaysi, Johor
[2*] Multimedia Universiti, Melaka

Corresponding author* email: e-sophan@utm.my
Accepted 1 November 2021, available online December 2021

ABSTRACT

Object detection and tracking has become one of the most important applications for UAVs, especially in surveillance applications. Single object tracking using a drone is an active field for research due to its importance in surveillance. In this paper, a real-time human and cars detection and tracking approach is proposed using a CNN-based technique by combining the state-of-the-art object detection algorithm YOLOv4 with the state-of-the-art multi objects tracking algorithm DeepSORT using video streaming from a drone for the purpose of surveillance. Furthermore, an algorithm was developed to enhance the observation ability by choosing a single object to be tracked. The algorithm was implemented using Tello drone with a developed navigation algorithm based on the vision analysis.

Keywords: Object detection, Single object tracking, drone, CNN, UAV

1. Introduction

With the rapid development in AI-based robots, Unmanned Aerial Vehicles (UAVs) have been significantly developed to enhance a wide range of applications such as surveillance and security. Some of these applications require a drone equipped with vision techniques such as real-time object detection and tracking. Deep Neural Network is the state-of-the-art method for image classification. Many DNN techniques were developed to enhance the classification ability using different structures. Object detection involves classification and localization of one or more objects in a single image [1]. CNN-based object detectors can be divided into two main categories based on the structure: (1) multiple-stages detectors and (2) Single-Stage detectors. Multiple-stages detectors divide the image into various regions and classify each region then combine them [2]. These detectors are characterized by giving high accuracy detection, however, their computational process is heavy such as Faster R-CNN [3][4]. Single-Stage detectors are faster and simpler object detection techniques where they predict the object directly in an image in a single stage. YOLOv4 is an algorithm for object detection which provides high performance in terms of accuracy and speed compared to the other detection algorithms. YOLO algorithm is able to localize and classify the objects in one stage using a single CNN network. The output of this stage are bounding boxes with class prediction of these objects [5]. YOLOv4 has achieved the best results for object detection with 43.5% AP at 65 FPS on COCO dataset.

Object tracking involves the estimation of the trajectory for single or several objects in video frames over the time. Tracking based on detection has become the leading technique for object tracking [6]. DeepSORT is the state-ofthe-art tracking algorithm that combines Kalman filter, Hungarian algorithm, Mahalanobis distance and the Appearance Feature Vector, which has become the most effective and real-time tracking algorithm [7]. DeepSORT operation uses a track estimator which is the location matric provided by SORT by using intersection over union IoU, then uses the

appearance descriptor to match features for one object class using CNN, then data association using location matric and appearance matric, an ID is assigned to each existing track [8]. In this paper, we propose a novel tracking system using a drone for the purpose of surveillance, our system is using real-time offline computation and has the novelty of performing both single object tracking and multiple objects tracking. The backbone of our proposed system is using the object detection YOLOv4 with the object tracking algorithm DeepSORT, in addition to enhancing the capability of the system by developing a technique for single object tracking in order to make the drone able to follow a target

autonomously. Moreover, the proposed system is able to estimate the distance between the drone and the target object. finally, our detection and tracking algorithm has been implemented on a DJI Tello drone.

2. Methodology

The overall flowchart for our system is shown in Figure 1 and the illustration of system architecture is presented in figure 2. Our surveillance system uses off-board computation and assigns Wi-Fi to be the communication protocol between the user and the drone. The user will have the ability to control the drone from his existing location and will be able to get the streaming video from the drone through Wi-Fi. The system has four main parts as follow: first, object detection, second, multiple and single object tracking, third, object localization, and fourth, drone control.

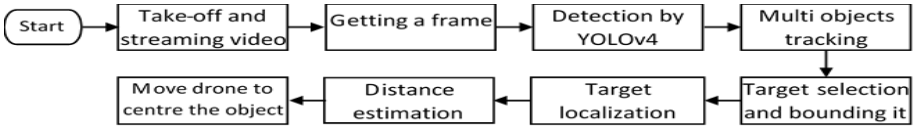

Figure 1. Overall system flow

Python will be used as main programming language in this project since it has various libraries that can be used to build the CNN algorithm and train the module in addition to designing a GUI for controlling the drone. This part illustrates details for the proposed system algorithm which includes object detection, multi-object tracking, single object tracking, object localization and drone control.

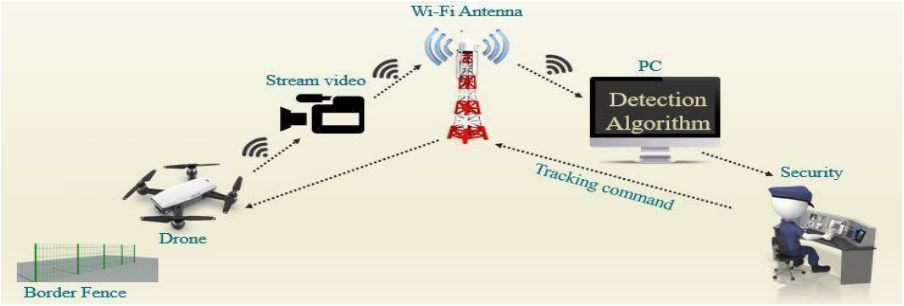

Figure 2. System Architecture

2.1 Object Detection

There are 2 main metrics need to be considered in order to choose an appropriate object detection algorithm and even modify the chosen algorithm in order to give the best result for the desired application. First, the number of frames per second (FPS) which indicates the detection speed and how much frames can be processed each second, hence, real-time applications require higher FPS to be considered as reliable application. Second, mean average precision (mAP), which indicates the average precision of detection, localization and classification of the objects exist in the scene.

YOLOv4 is the state-of-the-art object detection algorithm that was published in April 2020 in [8]. It provides high performance in terms of accuracy which indicated by average precision (AP) and speed that presented by the number or frames per second (FPS) compared to the other detection algorithms as shown in figure 3. YOLOv4 has achieved the best results for object detection with 43.5% AP at 65 FPS on COCO dataset. It has the advantage of combining some universal features such as Cross-Stage-Partial-connections (CSP), Self-adversarial-training (SAT), Cross mini-Batch Normalization (CmBN), Weighted-Residual-Connections (WRC), Mish-activation, Mosaic data augmentation, DropBlock regularization, and CIoU loss [8].

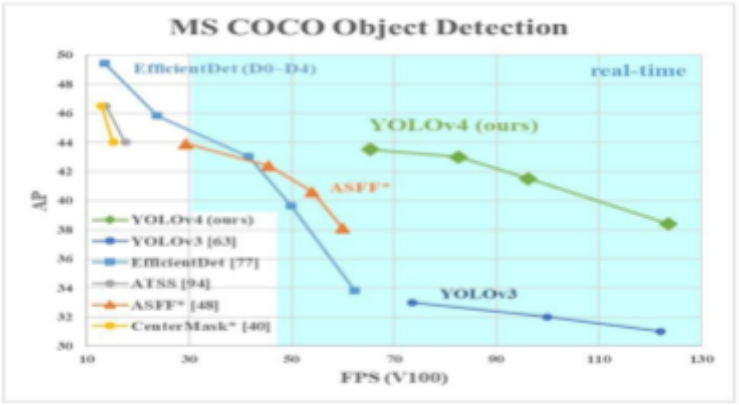

Figure 3. YOLOv4 Performance [8]

YOLOv4 architecture is divided into 4 main parts as shown in figure 4. Input image is fed to the backbone which refers to feature-extraction architecture. The backbone for YOLOv4 is CSPDarknet53 that composed of 53 convolutional layers. CSPDarknet consists of several Dense-Block which refers to multiple layers, each layer has batch normalization and ReLU, each Dense-Block is followed by convolution and pooling layers. Backbone output is fed to Neck which composed of additional layers that used for feature aggregation. Finally, the output from the Neck part is fed to the Head part that consists of prediction and drawing bounding boxes around the detected objects.

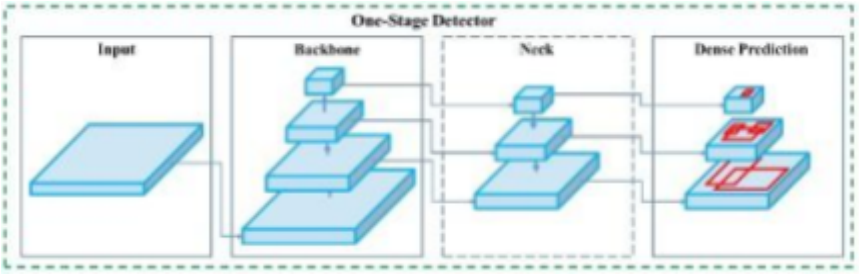

Figure 4. YOLOv4 Architecture [9]

Overall, YOLOv4 algorithm operation starts by localizing the objects in a frame then classify the detected objects in one stage using a single CNN network. The output of this stage are bounding boxes with class prediction of these objects in the form of (x, y, w, h, confidence) where x and y are the box coordinates, w and h are the width and the height of the box respectively and confidence is the class probability [10].

2.2 Object Tracking

Object tracking has become one of the most emerged fields in computer vision that has uncounted application nowadays and in the future. The term object tracking refers to three basic processes[11], first, getting set of bounding boxes coordinates which is the initial step in order to build a tracker where these bounding boxes can be determined through several ways such as drawing a bounding box manually around an object to be tracked, however, this method considered to be an old and inefficient technique for choosing the object to be tracked because the object size in the scene is not fixed, since the object size in the frame depends on its far from the camera whereas the distance between them increases, the object size in the frame decreases. Hence, the state-of-the-art technique for setting bounding boxes to be tracked is based on object detection result where bounding boxes are drawn around the detected objects which overcomes the insufficiency of the first method where as long and the object is detected even if its size is varying, a bounding box will be drown around it which is also vary to fit the object.

Object tracking involves the estimation of the trajectory for single or several objects in video frames over the time. Tracking based on detection has become the leading technique for object tracking where it consists of two main operations: first is applying object detection algorithm which outputs boundary boxes around the detected objects then applying data association algorithm in order to compare the new detected objects with the previous detection to determine the association between them [12]. Object tracking algorithms uses several filters such as Kalman filter which is the most efficient filter that has been used for tracking. Kalman filter is expressed as ($u, v, \gamma, h, x, y, \gamma, h$) where the first four parameters present the bounding box coordinates and the last four parameters represent the velocity for each coordinates, hence the Kalman filter can estimate the new position based on the previous location. After Kalman filter, an appearance descriptor is used in order to extract features from the new frame and compare them with the previous frame so that features that are belong to the same identity are near from each other. Then Hungarian algorithm is used to estimate tracks from the new detection outputs. By combining these filters, a real-time tracking can be achieved which is called Simple Online Real-time Tracking (SORT).

DeepSORT is the state-of-the-art tracking algorithm that combines Kalman filter, Hungarian algorithm, Mahalanobis distance and the Appearance Feature Vector, which has become the most effective real-time tracking algorithm [11]. DeepSORT uses the output of YOLOv4 which returns the coordinates of the detected objects in the scene and their width and height then draw bounding boxes around the objects.

The second process for tracking is creating a unique ID for each detected object which ideally should be fixed
for an object as long as it is exist in the scene and even if it lost and detected again, the same ID will be assigned for it, however, practically the functionality of this process not always accurate due to various reasons such as low accuracy detection algorithm, low intensity or contrast and small size or far objects.

By referring Figure 5. objects IDs allow us to enhance many applications such as objects counting, speed detection or even for single object tracking. In this system DeepSORT is used as multiple object tracking and an algorithm is developed in order to track a single object just by entering its ID so that another bounding box will be drawn around it.

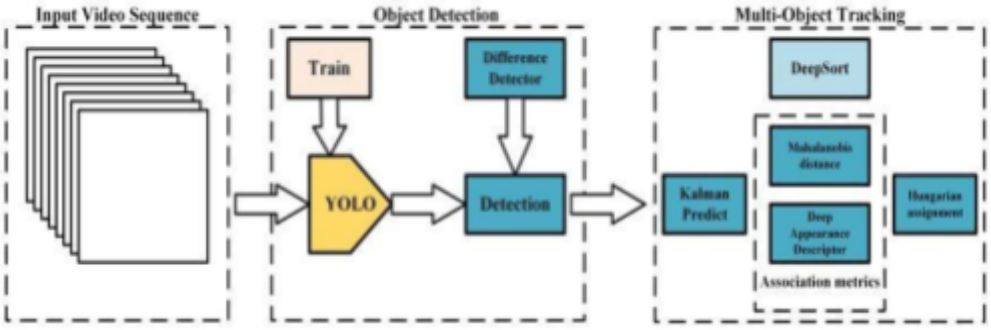

Figure 5. DeepSORT Architecture

2.2 Object Localization

One of the essential tasks for object detection is to localize all detected objects from different classes and return coordinates of bounding boxes that refer to the objects location in the scene. On the other hand, object localization refers to finding the position of a single object of specific class in the image. As for surveillance drone, object localization is emerged in order to get the current target location relative to the drone position so that the system will decide how to control the drone to keep the target object at the centre of the scene. Tracking algorithm output gives the coordinates of the target object as a list that includes 4 values which indicate the starting and ending points in the x and y axis as follow bbox=[x_start, y_start, x_end, y_end] .Therefor to get the target object centroid, the following equations are used:

$$cx = x\_start + (x\_end - x\_start)/2 \quad (1)$$
$$cy = y\_start + (y\_end - y\_start)/2 \quad (2)$$

Where (cx,cy) is the centre point of the object. Frame centre point can be identified simply as we know the frame dimension or even we can set it on any desired size. The frame size is set to be 640x480 pixels, hence the frame centroid is located at point (320,240).

2.3 Drone Control

There are three basic actions that control the drone's movement based on its rotation around one of the 3 axis x, y, and z [13]. In this experiment we use only Pitch and Yaw to control the drone where Yaw adjusts the direction and Pitch moves the drone forward and backward. In order to make the drone move smoothly with high accuracy, a movement technique is developed to vary the Pitch and Yaw speed based on the target centroid location even in the same zone. Tello drone's speed limits for the four directions (Up_Down, Forward_Backword, Right_Left, Yaw) varies from -100 cm/s to 100 cm/s in addition to the minimum speed is 10 cm/s. Figure 5 illustrates the frame division into zones where once we select the target object and get its centroid. The drone is assigned to take different actions based on which zone the target's centroid is located. The drone task is to keep the centroid in the Dead Zone which is zone 5.

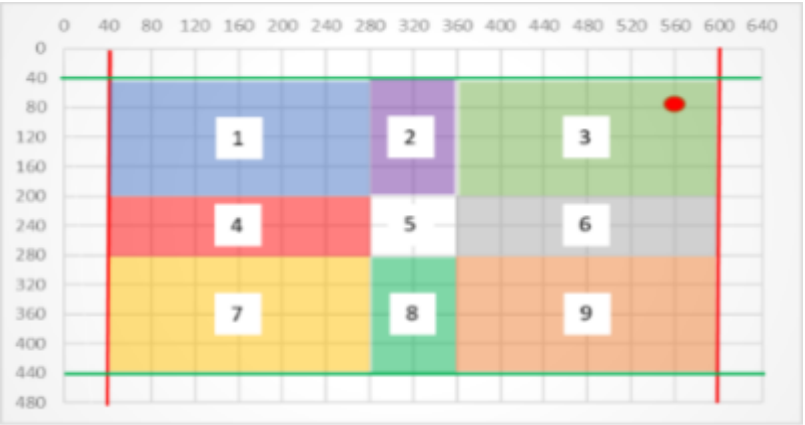

Figure 6. Frame's zone

Each frame is divided to 9 zones, if the object's centroid exceeded the green lines, the drone will move with its maximum Pitch speed (forward or backword) in order to catch the object to keep it in the scene. Same as for the red

lines, it will rotate clockwise or counter clockwise with its maximum Yaw speed. For zones 1, 2, 7 and 9 the drone yaws and pitches in the same time while in zones 2 and 8 it pitches only and in zones 4 and 6 yaws only. Therefore 4 linear equations were developed in order to vary the speed for Pitch and Yaw according to the target's centroid coordinates since we want the pitch and yaw speed be minimum as long as the centroid in near from the Dead zone and the speed increases as the centroid getting far.

Figure 7 shows the graph for Pitch and Yaw speed according to y_cordinate and x_coordinate respectively. The following equations have been derived to set the speed for each pixel, however these equations are not fixed and might vary from system to another where there are four parameters identify the constants, frame width, frame length, maximum speed and minimum speed. Basically, we determined that the speed at the green and red lines should be the maximum speed which is ±100 and we determined that the speed just after the Dead zone should be minimum which is ±10, then we defined the frame size to be 640x480. Hence, by creating two equations using the desired maximum and minimum speeds and their coordinated and solve them together, we can come out with 2 equations for Yaw and Pitch each. For example, let`s consider the red dot in Figure 6 refers to the target centroid where its coordinates (560, 80). By substituting cx and cy in equation 4 and 5 respectively, we can get Yaw speed = 91.8 and Pitch speed = 83.125cm/s.

(Zones: 1,4,7): Yaw_speed = 0.409*cx - 124.545 (3)
(Zones: 3,6,9): Yaw_speed = 0.409*cx - 137.273 (4)
(Zones: 1,2,3): Pitch_speed = -0.5625*cy + 128.125 (5)
(Zones: 7,8,9): Pitch_speed = -0.5625*cy + 141.875 (6)

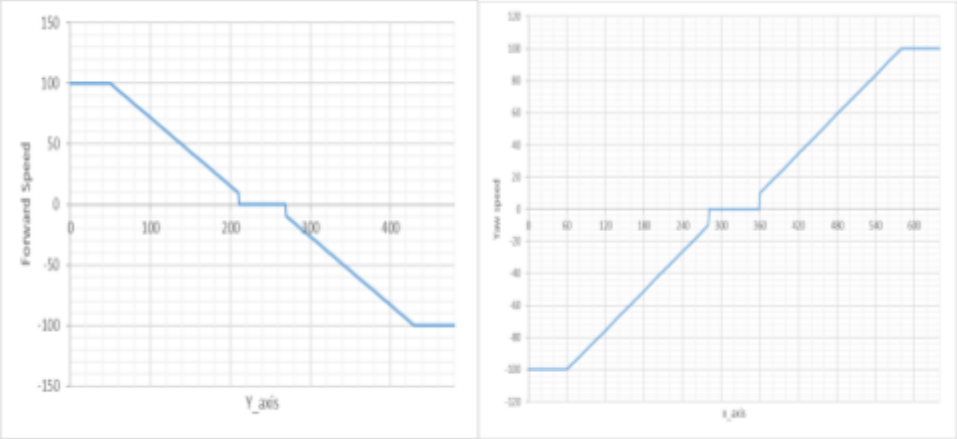

Figure 7. The discrete cell of each detection system

Finally, once the drone allocates the target's centroid in the Dead zone, the algorithm will apply a distance estimation equation. The purpose of this process is to ensure that the drone is not too far from the object. Hence, the following equation is used

Distance = focal length * width_real/width_pixels (7)

where width_real is the object's width in reality where we have made an assumption that the average of cars width is 2m and the average of human width is 0.5m since we are not targeting 100% distance accuracy. Width_pixels is the width of the bounding box around the object. focal_length is fixed number that indicated the focal length of the camera which in our case it is equal to 480. Therefore, as the focal length and the real width considered to be constants in our case, a relationship can be drawn between distance and target bounding box's width. We set the drone to keep the target in distance less than 7 meters, however, this value can be changed according to the requirements.

3. Results and analysis

3.1 Graphical User Interface (GUI)

In order to facilitate the system usage, a friendly graphical user interface (GUI) has been developed so that any user can use the system easily and fast without any programming knowledge required. In Figure 8 shows the interface to control the drone by using compatible devices. The GUI was designed using QT Designer tool and PYQT5 library using python programming language. It gives user the ability to manually control the drone to any place in the drone's range and once the user suspect of any object, he can enter its ID in the field of ID selection in the GUI then the drone will automatically move to make it in frame's centre. The GUI shows the drone's battery percentage, number of frames per second (FPS), number of detected objects in the scene and the distance between the drone and the target. The user can change the target by entering a new ID or stop the auto tracking by inserting 0 in the ID selection.

Figure 8. Graphical user interface for drone control

3.2 Results of Detection

Once the drone take-off, the detection algorithm will start and show a streaming video and detection result as shown in figure 8. Since the detection algorithms that provides high accuracy such as YOLOv4 require high computation speed in order to perform real-time detection, thus, the detection accuracy and speed depends on the power of the processing unit. Despite the advancement in computer vision and deep learning techniques, current

models that show real-time and high accuracy require powerful GPUs ether for training or detection purposes. Since GPUs have very fast computation process as compared with CPUs, therefore, as the GPU power increases, the detection speed and precision increase as well. Image resolution is a curtail parameter that affects the processing speed and accuracy, where, as the image sizes increases, the detection precision increases, however, on the other hand, the FPS decreases. Thus, the frame size should be chosen carefully specially for real-time applications in order to get the two features, optimal speed and accuracy, and this depends on the GPU's power used in the model. Although YOLOv4 comes with lighter models that doesn't require high computation power and gives high speed detection such as YOLOv4-tiny and YOLOv4-tflite, however, these model shows low detection accuracy which is not reliable to be used in such surveillance application that requires high detection accuracy to perform a reliable and accurate tracking system. As for this proposed system, we use NVIDIA GeForce GTX 1050 GPU and tensorflow 2.3 alongside with YOLOv4 pre-trained weights and the frame size that fed to the detection algorithm is set to 416x416. The model isolates each frame into regions and apply probabilities to the confined-edge boxes for each region, simultaneously, it expects several confined-edge boxes for these classes.

The developed algorithm shows high accuracy detection when high resolution video is fed to it. It is set to detect cars and people only, however, the system can be trained to detect any other objects if required. It has been noticed that the detection accuracy varies significantly based on video quality and light intensity.

Figure 8. Detection Design

3.3 Results of Tracking

By referring to Figure 9, once the streaming video starts, object detection and multiple objects tracking result will be shown in the GUI where each detected object will be given a unique ID. The output from YOLO and

DeepSORT is a list of detected objects that have object class, object ID, and bounding box coordinates. User can insert target's ID in the GUI so that the drone will track it. The code loops in the detections and once it found an object that its ID is same as the target object ID, it will draw another bounding box around it so that this target has special tracking. practically, sometimes the tracking algorithm switches the ID of an object with another ID repeatedly, this behavior is observed when the image quality is low or the object is too far or small. Therefore, to solve this issue, the target is presented as list of IDs where initially one ID exist in the target list, however, once this issue appeared, the user can add another ID to the target list so that even if the algorithm assigned any of these IDs for that object it will be tracked smoothly.

The algorithm uses seven (7) equations as was explained in the previous section to control the drone movement
and speed. Thus the drone will keep tracking the target until the target stops or the user cancel tracking order

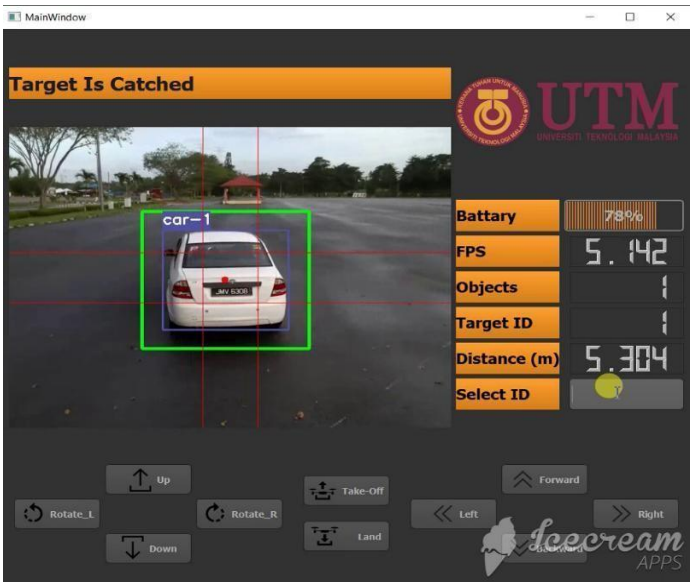

Figure 9. Single Tracking Result

The experiment showed accurate detection results and the drone was able to autonomously follow the car and keep its centroid in the Dead zone as shown in Figure 10. However, it is important to mention that this implementation is considered as a proof of concept since Tello drone is an educational drone which showed several limitations such as:

- Tello drone camera doesn't capture high quality images which reduced the accuracy, however, the algorithm showed high accuracy for high quality video from YouTube.
- Tello drone doesn't have on-board computation and uses Wi-Fi for communication which reduces the flight range to around 50m only.
- Tello drone maximum speed is 1m/s which is not reliable in real applications.

Figure 10. The discrete cell of each detection system

4. Conclusion

In this study, a single object tracking algorithm was built on top of the state-of-the-arts object detection algorithm YOLOv4 and the state-of-the-arts multiple objects tracking algorithm DeepSORT to be applied to an educational Tello drone for the purpose of surveillance. Furthermore, developing a single-object tracking algorithm for observing cars and humans. The algorithm is able to detect cars and humans with high accuracy in addition to controlling the drone in order to autonomously track any target.

Acknowledgment

We would like to thank to School of Electrical Engineering, Faculty of Engineering UTM for valuable contribution in providing workspace and equipments at Applied Control Lab to make this project successful.

5. References

[1] Rohan, A., Rabah, M., & Kim, S. H. (2019). "Convolutional neural network-based real-time object detection and tracking for parrot AR drone 2". *IEEE Access, 7*, 69575-69584.

[2] Pulma Sharma (2018). "A step-by-step introduction to the basic object detection algorithms". https://www.analyticsvidhya.com/blog/2018/10/a-step-by-step-introduction-to-the-basic-object-detection-algorithms-par

[3] De' Bruin, A & Booysen T. (2015). "Drone-based traffic flow estimation and tracking using computer vision: transportation engineering". *Civil Engineering= Siviele Ingenieurswese,* 2015(v23i8), 48-50.

[4] Ren, S., He, K., Girshick, R., & Sun, J. (2016). "Faster r-cnn: Towards real-time object detection with region proposal networks". *IEEE transactions on pattern analysis and machine intelligence, 39(6)*, 1137-1149.

[5] Dang, T. L., Nguyen, G. T., & Cao, T. (2020)"Object tracking using improved deep SORT YOLOv3 architecture". *ICIC Express Letters 10.24507/icicel.14.10.961*

[6] Hou, X., Wang, Y., & Chau, L. P. (2019, September). "Vehicle tracking using deep SORT with low confidence track filtering". In 2019 *16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)* (pp. 1-6). IEEE.

[7] Han, S., Shen, W., & Liu, Z. (2016). "Deep drone: Object detection and tracking for smart drones on embedded system". *URL* https://web. stanford. edu/class/cs231a/prev_projects_2016/deepdrone-object__2_. pdf.

[8] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: "Optimal speed and accuracy of object detection". *arXiv preprint arXiv:2004.10934.*

[9] YOLOv4 Architecture. Adapted from: *YOLO v4: Optimal Speed & Accuracy for object detection*, (2020), Retrieved from:https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection79896ed47b50

[10] Dang, T. L., Nguyen, G. T., & Cao, T.(2020) "Object Tracking Using Improved Deep Sort YOLOV3 Architecture", *ICIC Express letter*, v14(10), 961-969.

[11] Zhang, X., Hao, X., Liu, S., Wang, J., Xu, J., & Hu, J. (2019). "Multi-target tracking of surveillance video with differential YOLO and DeepSort". *In 11th International Conference on Digital Image Processing (ICDIP 2019),* Vol. 11179, 111792L.

[12] Hou, X., Wang, Y., & Chau, L. P. (2019). Vehicle Tracking Using Deep SORT with Low Confidence Track Filtering., *16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS),* 1-6..

[13] S.W.Nawawi & S. Sabikan, (2016) "Open-source Project(OSPs) Platform for Outdoor Quadcopter. *Journal of Advanced Research Design*, 24(1), 13-27.