Dual Tracker

Elad Amar
Supervisor: Roee Francos

1 Introduction

Object detection and object tracking are important for many computer vision tasks. In object detection algorithms we aim to locate an object of interest and draw a bounding box around it, while in object tracking we aim to estimate trajectories of the objects. In Single Object Tracking (SOT) the target is known a-priori and our task is locate it in every frame of the sequence. In Multiple Object Tracking (MOT) our task is similar but we also need to distinguish between the objects, and sometimes a detection step in the beginning of the sequence is necessary to identify the targets in the scene. Using single object tracker for MOT has many drawbacks, like target drift and distinguish errors. Those usually caused due similar appearance of targets, occlusions and interactions between objects. Recently, there is a growing interest both remote controlled and autonomously flying Unmanned Aerial Vehicles (UAVs). With time drones are becoming more accessible, more advanced and with enhanced abilities. This make UAVs ideal for object tracking tasks such as surveillance, traffic analysis, etc. Over the last few years, there are remarkable achievements in object detection algorithms which exploit the abilities of deep learning (DL). In this work we try to create a MOT tracker based on object detection(Detector) and SOT trackers, thus the name Dual Tracker. The main goal is to combine the advantages of each of the methods to cover their flaws: speed and tracking of the SOT trackers, and the robustness of the detector for re-initializing object locations. A major problem that rises with using a detector for re-initializing object locations is that the detector does not distinguish between objects of the same type. This can be solved by matching current detections to the last locations with Intersection Over Union (IOU) or distance metrics. The code is available at Github

2 Related Work

2.1 Object Detection

Object detectors usually can be divided into two categories, one-stage and two-stage detectors [7]. The main difference between the methods is that the two-stage adds a proposal network to Regions Of Interest (ROI). The one-stage

detectors trained by optimizing classification-loss and localization-loss simultaneously. This architecture is roughly constructed with a feature extraction Convolutional Neural Network (CNN) referred as "backbone", followed by classification and localization layers. The backbone goal is to extract rich features information from input images, like the ResNet[4], Darknet[11], VGG16[19] networks. A main advantage of using deep backbone networks is the ability to train them for image classification tasks, and use fine tuning of last layers. Novel detecotrs using one-stage methodology are YOLO[14, 12, 13], SSD[9], and RetinaNet [8]. Two stage detectors use a Region Proposal Networks(RPN) to locate ROI and those pipelined into CNN for classification and localization regression. The addition of generating ROI resulting more accurate predictions but slower inference time. The RPN introduced in the RCNNs family [3, 15].

2.2 Object Tracking

There are many approaches for the object tracking problem, we will focus only on the Visual Object Tracking (VOT) problem. Most of trackers use a discriminative filter. These trackers need to get localization of the object in the first frame and then use a filter to distinguish between the target and its environment. Novel trackers using discriminative filter are KCF[6] and CSR-DCF[10]. Later these models outperformed by trackers based on siamese networks [1]. The siamese networks use a deep neural network that is fed with both target object and the frame and then use cross-correlation to locate the target in the frame. Another approach using a detector introduced in SORT [20], using solely detector and Kalman filter to estimate object trajectories and distinguish between targets. A similar work combining detector and trackers is the Hybrid Tracker [18].

3 Method

The proposed method includes combination of detector and tracker. The detector chosen is YOLOv3[13], which its inference time is above real time. For this YOLOv3 model we added a Spatial Pyramid Pooling(SPP)[5] and utilized General Intersection Over Union loss(GIOU)[16].

$$IOU = \frac{A \cap B}{A \cup B} \tag{1}$$

$$GIOU = IOU - \frac{C \setminus (A \cup B)}{|C|} \tag{2}$$

Equation 2. shows the GIOU metric, where C is smallest convex set enclosing A and B. The detector was trained on custom dataset combined VisDrone[21] and Stanford Drone Dataset(SDD) [17]. YOLOv3 is an anchor based detector, thus we ran a Kmeans with 9 clustering groups over the annotated images to get anchors sizes as seen in Figure 1. Additionally the YOLOv3 has many hyperparameters which we initialized with a genetic algorithm. The genetic algorithm

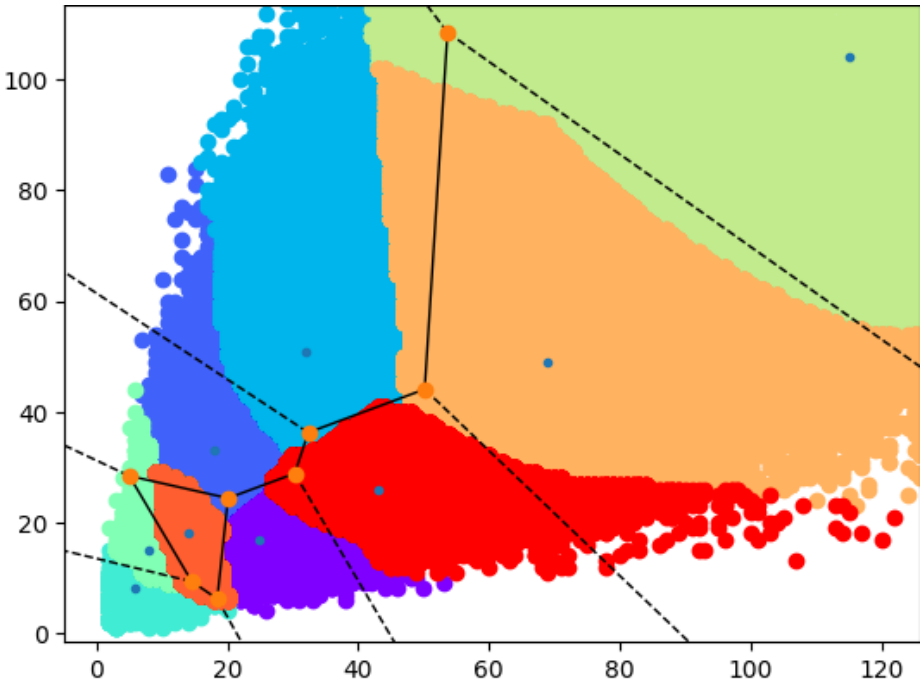


Figure 1: Kmeans over annotated images. Height and Width are in pixels on y and x axes respectively.

starts with arbitrary hyper-parameters and every five epochs of training we mutate the hyper-parameters which gave us the best score. The mutation rate taken from $N(1, 0.2)$ distribution. We are using opencv's trackers, and have the ability to assign each object different tracker. The tracker used in the experiments is CSRT[10] with psr threshold of 0.1. The Dual Tracker calls the detector upon:

1. Initialization.
2. When failures percentage is greater then failures threshold.
3. Every predefined interval of time.

For rest of time a tracker is assigned for every object and update its location on every frame. The reason of using this methodology is first because the tracker has to be initialized with the tracking target. Then when number of failures increasing and reaches the threshold, this might be because many objects left the scene or because there were significant changes in the visualization of the scene. Thus calling the detector can help us relocating the objects and get an updated visualization of the targets. The last reason is under the acknowledgment that the trackers may drift after a period of time and still return a high confidence

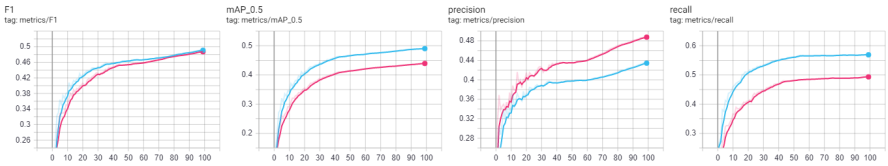


Figure 2: Results of anchor adjustments on only annotated images. Blue is the modified model.

score. When the tracker losses a target, we keep trying to locate it in the successive frames. If fails to locate it for number of frames that is greater than Frames Without Detection(FWD) threshold, we remove it. When the detector is called it returns all the objects located in the frame, we include only those which the detector confidence is high. The detector observations categorized by classes. Thus, in order to discriminate between targets we first use IOU metric. For every observation we check if target with highest IOU(Eq. 1) score from previous frames of same class, and if this IOU score greater then a given threshold. If not we try to use distance metric. Checking if the Euclidean distance(Eq. 3) of closest target of same class is lower from distance threshold. Observation that are not assigned to previous targets are added to the current targets and targets which not being re-initialized are removed from tracking.

$$d(O,T) = \sqrt{(O_x - T_x)^2 + (O_y - T_y)^2} \tag{3}$$

4 Experimental Results

The custom dataset for the detector contains approximately 20,000 annotated images with 6 classes: person, car, bicycle, motor, bus, truck. To this dataset we added approximately 30,000 background images. The first experiment was to determine the effects of changing anchor sizes as shown in Figure 1 against the anchors' sizes proposed for COCO dataset. The results show an increase of recall and decrease of precision with approximately same F1 score. We can deduce that the changes of precision and recall are harmonic. We can observe a that adjusting the anchors caused a better mAP result, which indicates better confidence with adjusted anchors, See Fig. 2. Later in order to increase the precision of the adjusted model, we have thought to add background images. The idea is to reduce the false positive detections. The models tested on a new test set with 400 images and 100 background images from VisDrone, that were not part of the train and validation set. in Fig. 4 the results show that we were able to increase precision but with almost no difference with the overall metrics.

Another experiment was testing how changing the hyper-parameters with genetic algorithm can affect our result. The default parameters achieved good result on COCO dataset, but here we are facing different problem, with different annotations and representation. The parameters include the YOLOv3 loss

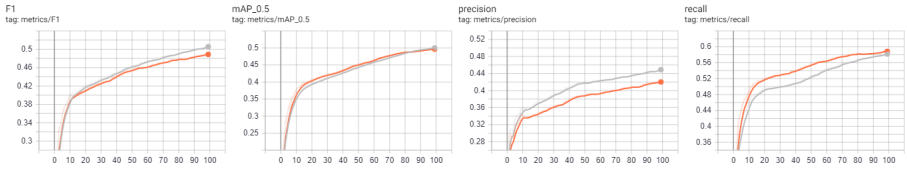


Figure 3: Results of model with hyper-parameters mutation and without. Orange is the mutated model.

Model	parameters	Precision	recall	mAP	F1
With background images	IOU- 0.5 Confidence- 0.4	0.8184	0.1835	0.1761	0.2627
	IOU- 0.5 Confidence- 0.6	0.8182	0.1836	0.1761	0.2627
	IOU- 0.7 Confidence- 0.4	0.6416	0.1084	0.1074	0.1696
Without background images	IOU- 0.5 Confidence- 0.4	0.7706	0.1911	0.1794	0.2751
	IOU- 0.5 Confidence- 0.6	0.767	0.1912	0.1795	0.2748
	IOU- 0.7 Confidence- 0.4	0.7238	0.1107	0.1096	0.1717

Figure 4: Detector Results

scalars, optimizer parameters such as learning rate and weight decay, etc. The genetic algorithm tested against the average between the F1 score and the mAP. In Fig. 3 we can see that as expected the model achieve better performance in the beginning, but that rapidly saturated and outperformed by the default model.

$$\phi_N = \frac{1}{N}\sum_{i=1}^{N}\phi_i \tag{4}$$

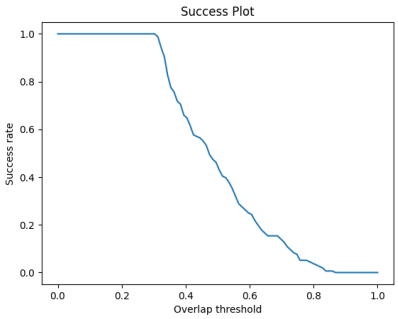
$$EAO = \langle \phi_N \rangle \tag{5}$$

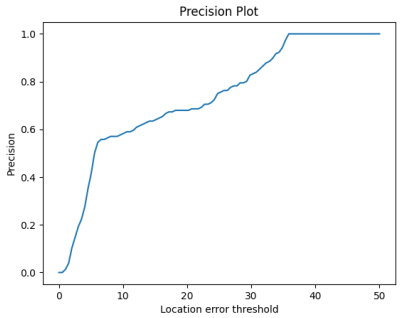
The Dual Tracker has been tested for SOT and MOT tasks with arbitrary hyper-parameters(do not confuse with detector's hyper-parameters), such as IOU threshold for Non Maximum Suppression(NMS), detector confidence score: where objects with less confidence are omitted. Also the re-initializing parameters: period of time the detector is called, number of failures needed to call the detector, and the IOU and distance threshold to bind between detections. Demo can be seen here and here, respectively for the tasks, with the hyper-parameters in the description.

The metrics used are precision, robustness and Expected Average Overlap(EAO). The precision is the IOU between the predicted bounding box and the ground truth bounding box over the sequence. The robustness is the ratio between the number of frames the object localized correctly with the length of sequence. The EAO combines both of those metrics, if the average overlap of a sequence is calculated as shown at Eq. 5, where ϕ_i is the IOU in the i_{th} frame. Then Eq. 5 shows the EAO [2].

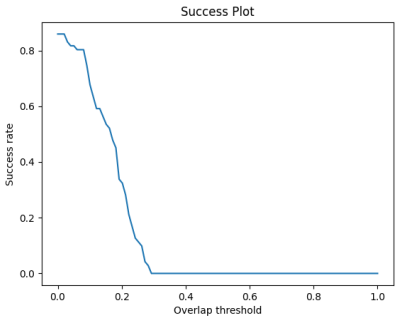
Metrics success plot precision plot

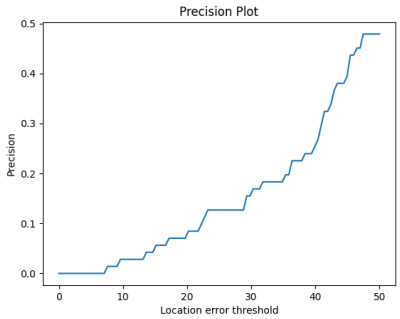
Video 1



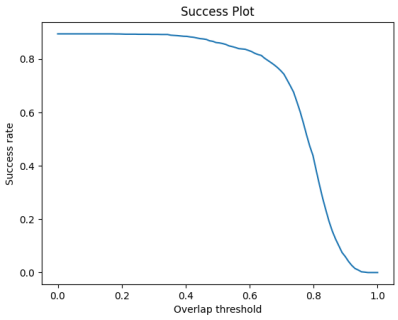


Video 2





Video 3



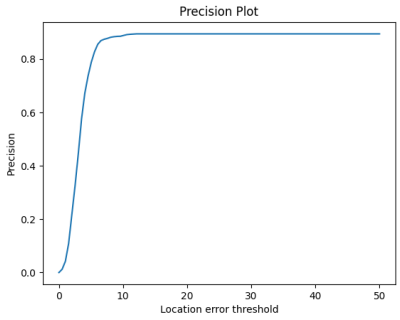


Figure 5: SOT results based on videos of demo.

Another evaluations methods are the success score and precision score. The success score is the ratio of the number of frames with IOU greater than a threshold, to the total number of frames. We create a plot of IOUs threshold varied from 0 to 1 with intervals of 0.01. The precision score holds similar idea, here we calculate the distance between the center of the ground truth bounding box and the predicted bounding box. The ratio is calculated between number of frames that the distance calculated is less then a given threshold. We create a plot of distance threshold varied from 0 to 50 with intervals of 0.5. In Fig. 5 we can see the results for the SOT tracking. As expected the tracker struggles more as the IOU threshold increases or distance threshold decreases. The important thing to observe in those graphs are the thresholds that changes curve's slop significantly. The difference in performance between videos 1 & 3 to video 2 is due targets types which are car and a person respectively. The MOT result are in Fig 6. The blue line represent the average over detected objects, whereas red for all target. From the robustness results we can infer that the Dual Tracker doesn't lose targets easily, but many times failed to discover those. The Frames Per Second(FPS) for that task is about 20 FPS depends on video's resolution. As mentioned the trackers are not parallel thus adding target effectively decreases FPS rate, even to 0 for large number of targets.

5 Conclusions

The Dual Tracker consists both a detector and a tracker, in the experiment we used YOLOv3 and CSRT tracker. The Dual Tracker symbiosis aims to cover detector's flaws with tracker advantages and vice versa. In the experiments we noticed that the detector performs badly on some of the targets, probably due unbalanced dataset. A solution for that issue can be by adding class weights to the loss function. Another main issue is the fact that for multiple targets the Dual Tracker perform lower than real time. This could be solved by changing the trackers to parallel tracker based on GPU such as the siamese networks. Switching to better performance trackers can also provide us the flexibility to use a slower detector which performs better. Yet, the Dual Tracker holds good results for long term tracking of detected targets. It shows that combining detector and tracker together has potential to achieve competitive results for tracking from a UAV.

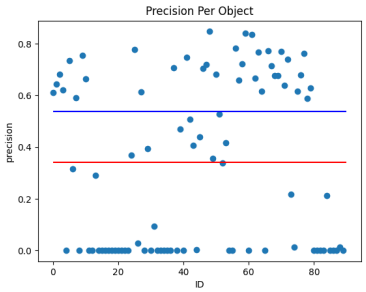
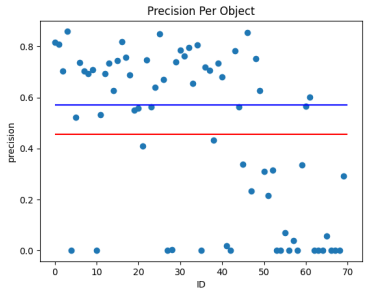
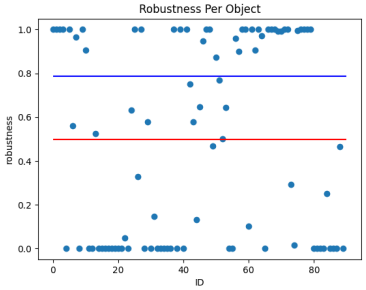
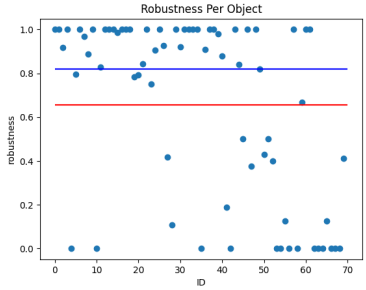
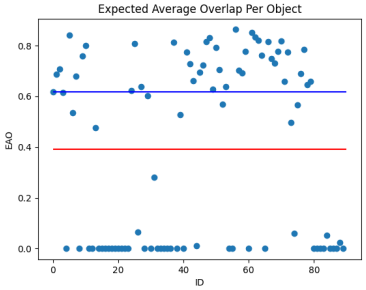
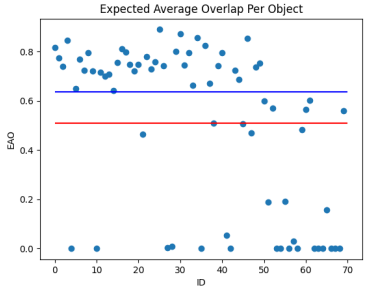
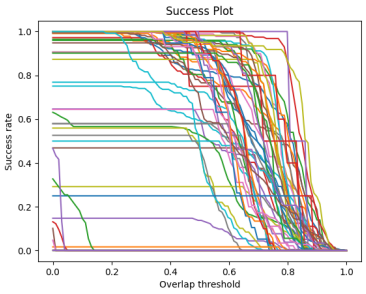
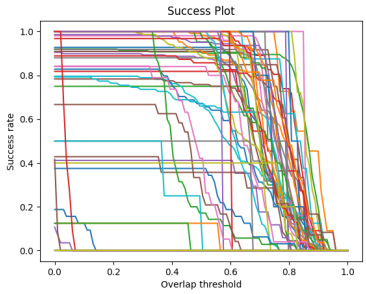
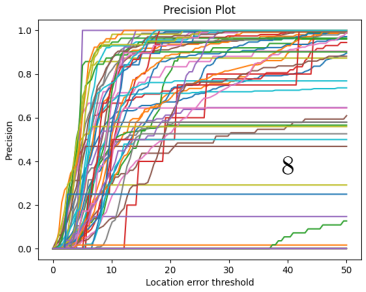
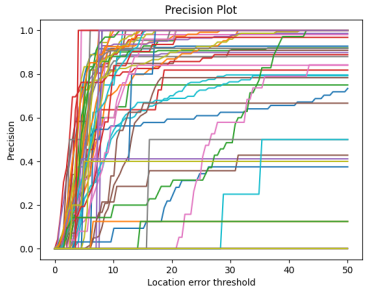
Videos	Video 1	Video 2
precision		
robustness		
EAO		
success plot		
precision plot		

Figure 6: MOT results based on videos of demo.

References

[1] Luca Bertinetto et al. "Fully-convolutional siamese networks for object tracking". In: *European conference on computer vision*. Springer. 2016, pp. 850–865.

[2] Michael George, Babita Roslind Jose, and Jimson Mathew. "Performance Evaluation of KCF based Trackers using VOT Dataset". In: *Procedia Computer Science* 125 (2018), pp. 560–567.

[3] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[4] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[5] Kaiming He et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015), pp. 1904–1916.

[6] João F Henriques et al. "High-speed tracking with kernelized correlation filters". In: *IEEE transactions on pattern analysis and machine intelligence* 37.3 (2014), pp. 583–596.

[7] Licheng Jiao et al. "A survey of deep learning-based object detection". In: *IEEE Access* 7 (2019), pp. 128837–128868.

[8] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[9] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[10] Alan Lukezic et al. "Discriminative correlation filter with channel and spatial reliability". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6309–6318.

[11] Joseph Redmon. *Darknet: Open Source Neural Networks in C*. http://pjreddie.com/darknet/. 2013–2016.

[12] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *arXiv preprint arXiv:1612.08242* (2016).

[13] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018).

[14] Joseph Redmon et al. "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.

[15] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks". In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[16] Hamid Rezatofighi et al. “Generalized Intersection over Union”. In: (June 2019).

[17] A. Robicquet et al. “Stanford Drone Dataset”. In: (). URL: http://cvgl.stanford.edu/projects/uav_data/.

[18] Hasan Saribas et al. “A Hybrid Method for Tracking of Objects by UAVs”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0.

[19] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[20] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. “Simple online and realtime tracking with a deep association metric”. In: *2017 IEEE international conference on image processing (ICIP)*. IEEE. 2017, pp. 3645–3649.

[21] Pengfei Zhu et al. “Vision meets drones: A challenge”. In: *arXiv preprint arXiv:1804.07437* (2018).