2018 International Conference on Unmanned Aircraft Systems (ICUAS)
Dallas, TX, USA, June 12-15, 2018
DOI:10.1109/ICUAS.2018.8453398

Real-time Implementation of YOLO+JPDA for Small Scale UAV Multiple Object Tracking

Shuoyuan Xu[1], Al Savvaris[1], Shaoming He[1], Hyo-sang Shin[1], Antonios Tsourdos[1]

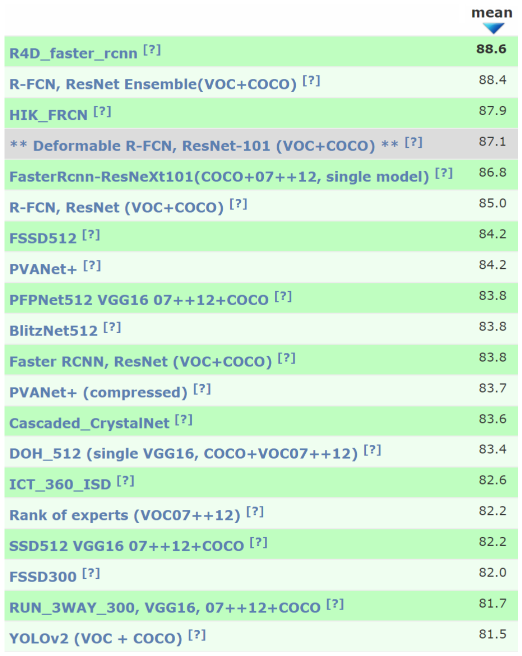
***Abstract*— This paper describes the development of a real-time multiple object detection and tracking system for a small scale UAV. The YOLO deep learning visual object detection algorithm and JPDA multiple target detection algorithm, were selected and implemented. The theory and implementation details of these algorithms are presented. The performance analysis of the system is done on both public dataset and aerial videos taken by UAV.**

I. INTRODUCTION

The need for aerial visual surveillance has been growing over the past few years along with the development of Unmanned Aerial Vehicle (UAV) related technologies. There already exists a few mature applications in UAV visual surveillance, for example, power line inspection [1], shark early warning in surfing coasts[2], poachers detection in national nature reserves[3], search and rescue[4], etc. However, most of these applications are designed with the consideration that a human operator would be processing the video data, which is inefficient since humans are economically costly and limited in working time. Fully automatic aerial surveillance will be the trending in the related fields of application. The automatic visual surveillance mission are consist of object detection, object tracking, behavior monitoring, unusual objects identification, etc.

Multiple object detection and tracking is a critical subsystem for fully automated aerial surveillance since it provides the location, ID, class, track of the observed objects to the surveillance system. These information can be applied to higher level systems such as behavior monitoring, unusual objects identification. In the recent year of computer vision research, there are many object detection and target tracking algorithms with high accuracy. The latest object detection algorithm possesses the mean average precision (mAP) of 88.6% on PASCAL VOC dataset [5]. State of the art multiple object tracking algorithms can get an up to 51.8% tracking accuracies (MOTA) on MOT Benchmark dataset [6]. However, due to the fact that small scale UAVs have limited payload, the computational capacity of UAV onboard processing is limited. The main limitation of the implementation of multiple object detection and tracking algorithms on UAVs are the real-time processing speed while maintaining a rather accurate detection and tracking performance since the advanced algorithms are too computationally expensive for UAV onboard computers. The aim of this project is to develop a real-time object detection and multiple objects tracking system with real-time processing speed, which can be implemented on an onboard computer of small scale UAVs.

There are a few categories of visual object detection approaches, among them the only kind which can get to higher than 81% mAP are deep learning based algorithms, only these methods were considered in this project. The appropriate object detection approach for this project is selected based on the PASCAL VOC benchmark in Fig. 1. PASCAL VOC is a widely used dataset for object detection, most deep learning algorithms have been tested on this dataset, which its benchmark can evaluate and rank the performance of each algorithm. The algorithm for this project is selected from those who have higher than 80% mAP from VOC benchmark. Among them, only Single Shot MultiBox Detector (SSD) [7] and You Only Look Once (YOLO) [8] based algorithms can perform real-time object detection. The processing speed of SSD based methods can get to maximum 46 FPS with an accuracy of 84.2%. On the same GPU, YOLOv2 can achieve 67FPS with an accuracy of 81.5%. Due to their similar detection mAP (only 2.7% difference), the faster algorithm YOLOv2 [9] is selected for this project. These FPS values are tested on a powerful GPU NVIDIA Titan X, which means they are still very computational heavy for a UAV onboard computer.



	mean
R4D_faster_rcnn [?]	**88.6**
R-FCN, ResNet Ensemble(VOC+COCO) [?]	88.4
HIK_FRCN [?]	87.9
** Deformable R-FCN, ResNet-101 (VOC+COCO) ** [?]	87.1
FasterRcnn-ResNeXt101(COCO+07++12, single model) [?]	86.8
R-FCN, ResNet (VOC+COCO) [?]	85.0
FSSD512 [?]	84.2
PVANet+ [?]	84.2
PFPNet512 VGG16 07++12+COCO [?]	83.8
BlitzNet512 [?]	83.8
Faster RCNN, ResNet (VOC+COCO) [?]	83.8
PVANet+ (compressed) [?]	83.7
Cascaded_CrystalNet [?]	83.6
DOH_512 (single VGG16, COCO+VOC07++12) [?]	83.4
ICT_360_ISD [?]	82.6
Rank of experts (VOC07++12) [?]	82.2
SSD512 VGG16 07++12+COCO [?]	82.2
FSSD300 [?]	82.0
RUN_3WAY_300, VGG16, 07++12+COCO [?]	81.7
YOLOv2 (VOC + COCO) [?]	81.5

Fig. 1. VOC benchmark [5]

[1]Centre for Autonomous and Cyber-Physical Systems, School of Aerospace, Transport and Manufacturing, Cranfield University, MK430AL, UK a.savvaris@cranfield.ac.uk

As for multiple objects tracking algorithms, the top priority is not to add the computational burden to the UAV onboard computer since the selected object detection algorithms are consuming most of the available processing power. Therefore, to save the computational load, the Detection Free Trackers (DFT) and image information implemented trackers are excluded. Moreover, considering the real-time demand of this project, offline trackers cannot be applied. Among all the online trackers that do not use image information, Joint Probabilistic Data Association (JPDA) were applied due to their light computation need. With an appropriate parameter tuning, the MOTA can get to a close level to those state of art algorithms[10].

In this paper, the fundamental of YOLO is introduced first. Then JPDA related theory and mathematics are presented. Thirdly, the hardware configuration and implementation details of YOLO and JPDA algorithms are shown. Finally, the testing result of the implemented algorithms based on both public dataset and aerial video taken by UAV flight are presented and analyzed.

II. YOU ONLY LOOK ONCE

You Only Look Once (YOLO) is a convolutional neural network configuration for object detection. Different from previous deep learning object detection approaches, YOLO treats the detection task as a one-step regression problem instead of a classification then localization problem. The name YOLO comes from which the algorithm can localize the object and extract the object class within one scan. This mechanism makes the YOLO faster than most of the deep learning based object detection algorithms.

As for the logic flow of YOLO, it first divides the image into small grids, then performs five bounding boxes and class probability prediction based on each grid. Then, it analyses distributions of all box and object class based on probabilities. Finally, according to a set threshold value, detection results fulfill the threshold value are considered as the output value. The loss function is designed with the combination of bounding box position, non-object bounding box confidence, with-object bounding box confidence, and object class.

As for the network structure of YOLO, it has a fixed image input resolution (in this project 416×416), followed with 29 convolutional layers. Then the 13×13×1024 convolutional layers output are connected with 1 fully connected layer. This output is directly fed to the loss function for training and interpreter for testing.

III. JOINT PROBABILISTIC DATA ASSOCIATION

Multi-object tracking uses similar ideas as single object tracking, which can be seen as a two-step approach. Step 1 is called prediction, which propagates state probability density function to the current time. Step 2 is called update, which uses current observation to correct prediction. As for multi-object tracking, two extra steps, gating and data association are added to the prediction and update step. Gating is used to exclude the unlikely observation results. Data association are to assign the best detection to the targets.

JPDA uses probabilistic data association method to calculate the likelihood of measurement and target matching. JPDA is a modified solution of probabilistic data association (PDA) which in the case of multiple tracks, it constructs all the possible assignment cases and excludes the overlapping assignments.

JPDA is a multi-target tracker based on classic Kalman Filter and probabilistic data association algorithm. In this case, since the motions of the targets are projected into the 2D image frame, it can be assumed that all targets are moving with nearly constant velocity in the image frame.

The prediction of jth target can be modeled as (1),

$$\mathbf{x}_k^j = \mathbf{F}\mathbf{x}_{k-1}^j + \mathbf{w} \tag{1}$$

where $\mathbf{x}_k^j$ is the target state vector at time k, which contain its position and velocity $\mathbf{x}_k^j = (x_p^j, x_v^j, y_p^j, y_v^j)$, F is the transition matrix, where $T = 1s$ is the sampling time. w is the process noise with a zero mean Gaussian distribution, its covariance is Q, the parameters in Q is set as in (2)

$$\mathbf{F} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}; \mathbf{Q} = q_e \begin{bmatrix} T^4/4 & T^3/2 & 0 & 0 \\ T^3/2 & T^2 & 0 & 0 \\ 0 & 0 & T^4/4 & T^3/2 \\ 0 & 0 & T^3/2 & T^2 \end{bmatrix} \tag{2}$$

Where $q_e = 0.1$ is the standard deviation of the process noise, $T = 1$ is the temporal sampling rate.

The measurement of jth target is given bellow

$$\mathbf{z}_k^j = \mathbf{H}\mathbf{x}_k^j + \mathbf{v} \tag{3}$$

where $\mathbf{z}_k^j = (z_x^j, z_y^j)$ is the measurement at time k, which contains position measurement since the detector in this case can only provide position information. H is the observation model, v is the measurement noise with a zero mean Gaussian distribution, its covariance is R, the parameters in H and R is set

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}; \mathbf{R} = q_m \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4}$$

Where $q_m = 7$ is the standard deviation of the measurement noise.

From the estimation and measurement model of the system, a Kalman filter tracking system can be formulated. First, the prediction of each target state at time k is calculated by (5), where $\hat{}$ stands for prediction, $_{k-1|k-1}$ stands for measurement updated prediction from time $k-1$, and $_{k|k-1}$ is the prior estimation without the measurement. The initial state is calculated from the first two frames where targets are captured by the detector.

$$\hat{\mathbf{x}}_{k|k-1}^j = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1}^j \tag{5}$$

The prior covariance of estimation is given in (6), the initial covariance is given in (7), where $q_m = 7$ is the standard deviation of the measurement noise.

$$\mathbf{P}_{k|k-1}^j = \mathbf{F}\mathbf{P}_{k-1|k-1}^j\mathbf{F}^{\mathrm{T}} + \mathbf{Q} \tag{6}$$

$$\mathbf{P}_0 = \begin{bmatrix} q_m & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & qm & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

As for the update of the target state, basic Kalman filter cannot handle the problem since one target may have more than 1 measurement due to the multi-measurement assignment and detector inaccuracy. Joint probabilistic data association (JPDA) is introduced to handle this problem.

In JPDA, first of all, a fixed size Gating is drawn with the prediction state in (4) as the center. All measurements fell within the gating are considered as the usable measurements. In this case, the distance between the measurement and prediction are calculated using Mahalanobis distance (8). The gate size is set to be $d_G = \sqrt{30}$ based on the resolution of the image and estimation of the object moving speed.

$$\mathbf{D^j_M} = \sqrt{(\mathbf{y}^j_k)^T \mathbf{S_k}^{-1} \mathbf{y}^j_k} \tag{8}$$

Where, y^j_k is the innovation which is calculated as shown in (9), and is defined as the difference between the measurement and prior estimation. S^j_k is the innovation covariance (10)

$$\mathbf{y}^j_k = \mathbf{z}^j_k - \mathbf{H}\hat{\mathbf{x}}^j{}_{k|k-1} \tag{9}$$

$$\mathbf{S}^j_k = \mathbf{R} + \mathbf{H}\mathbf{P}^j_{k|k-1}\mathbf{H}^{\mathrm{T}} \tag{10}$$

When $D_M < d_G$, the measurements are considered to be useful for the current target. These measurements, weighted by their likelihood are summed up to update the target state. The likelihood of measurements assigning to current target can be calculated as (11):

$$p_{ij} = \begin{cases} (1-p_D)\beta, & \text{if no measurement} \\ p_D \cdot g_{ij}, & \text{otherwise} \end{cases} \tag{11}$$

Where g_{ij} is the Gaussian likelihood function of the assignment of measurement i to target j, i stands for ith measurement, j stands for jth target. M is the dimension of measurement data, in this case $M = 2$.

$$g_{ij} = \frac{e^{-\mathbf{D}_M/2}}{(2\pi)^{M/2}\sqrt{|\mathbf{S}^j_k|}} \tag{12}$$

Since there exists multiple target in this tracking mission, JPDA are introduced to prevent measurement repetitive assignment. JPDA calculate the likelihood function on a global scale. The probability of each non-repeat assignment of all in Gating measurement to all targets p_a are calculated first.

$$p_a = \prod_{i\,to\,j} g_{ij} p_D \prod_{null\,to\,j} (1-p_D) \prod_{i\,to\,null} \beta \tag{13}$$

Where each global assignment probability is the multiplication of individual assignment probability (i to j), no assignment target (null to j), and assignment to none target measurements (i to null). After all the assignment probability is calculated, the likelihood of measurements assigning to current target can be calculated by summing up all the assignment probability that contains the current assignment

$$p_{ij} = \sum_{a=1}^{A} \{p_a \mid if\ p_a\ contains\ i\ to\ j\} \tag{14}$$

The weighted innovation $\mathbf{y^j_k}$ of target j at time k can be calculated as (15):

$$\mathbf{y^j_k} = \sum_{i=1}^{I} p_{ij}\mathbf{y_{ij}} \tag{15}$$

After likelihood calculated, the update of measurement is similar to PDA combined Kalman filter, Kalman gain for jth target is calculated

$$\mathbf{K}^j_k = \mathbf{P}^j_{k|k-1}\mathbf{H}^{\mathrm{T}}(\mathbf{S}^j_k)^{-1} \tag{16}$$

The update of jth target state is given in (16), where $\hat{\mathbf{x}}^j_{k|k}$ is the measurement updated estimation of the jth target state.

$$\hat{\mathbf{x}}^j_{k|k} = \hat{\mathbf{x}}^j_{k|k-1} + \mathbf{K}^j_k\mathbf{y}^j_k \tag{17}$$

As for the update of estimation covariance

$$\mathbf{P}^j_{k|k} = \mathbf{P}^j_{k|k-1} - (1-p_{0j})\mathbf{K}^j_k\mathbf{H}\mathbf{P}^j_{k|k-1} + \mathbf{P}^j_k \tag{18}$$

Where p_{0j} stands for no measurement is available for jth target. Therefore the second term stands for only assigned measurement can cause the covariance to be updated. The third term $\mathbf{P}^j_k$ is a correction term which is used to model the assignment uncertainties.

$$\mathbf{P}^j_k = \mathbf{K}^j_k \left[\sum_{j=1}^{J} p_i j \mathbf{y_{ij}}\mathbf{y_{ij}}^{-1} - y^j_k(y^j_k)^{-1}\right](\mathbf{K}^j_k)^{-1} \tag{19}$$

IV. IMPLEMENTATION

A. Hardware Details

The YOLO detection and JPDA tracking algorithm are implemented on an NVIDIA jetson TX2 microcomputer due to its light weight and good GPU based computational capability. Gopro hero 5 is selected as the image streaming sensor. As for the UAV, considering the platform payload capacity and implementation simplicity, a DJI F550 hexacopter frame with a Pixhawk autopilot is selected. The camera is mounted to the UAV directly. The system configuration is presented in as Fig. 2.

When performing the detection and tracking mission, the UAV is set to position-hold flight mode, where it uses the onboard GPS, barometer, and IMU to maintain its position and altitude at a set of given value. In this project, the altitude of the UAV is set to be $5m$; this value is set based on the image resolution and camera field of view (FOV).

B. YOLO Implementation

The Implementation of YOLO is done via Tensorflow + Python. YOLO real-time object detection algorithms have three different scaled models available for Tensorflow implementation, Tiny YOLO, YOLOv2, and YOLOv2 608×608, which is a compromise between processing speed and detection accuracy. The overall performance of each model is

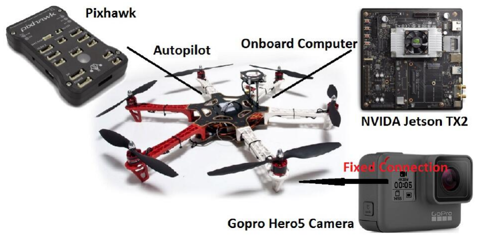


Fig. 2. Hardware Configuration

given in TABLE I, which shows the mean average precision (mAP), the frames per second (FPS) processing speed on a NVIDIA Titan X GPU and Nvidia Jetson TX2.

TABLE I
PERFORMANCE OF DIFFERENT YOLO MODELS

Model name	mAP	FPS (Titan X)	FPS (TX2)
Tiny YOLO	57.1	200	5
YOLOv2	76.8	67	3
YOLOv2 608×608	81.5	40	2

From the performance above, it can be concluded that Tiny YOLO has the best processing speed but worst accuracy, YOLOv2 is the most accurate but is the slowest in terms of processing time. From real image testing, Tiny YOLO is not accurate enough for this project. The YOLOv2 model is selected in the end since the detection mAP is close to YOLOv2 544×544 but have a faster processing speed.

C. JPDA Implementation

As for the implementation of JPDA, the original JPDA algorithms doesn't have the functionalities of processing new targets initiation, nor undetected target termination. Two functionalities are added to solve this problem.

The termination performs when the targets are no longer in view. In this case, it is possible that detection can have temporal fails even when the target is still within the camera FOV, since the detector can only perform 76.8% mAP, and can not deal with partially occlusion objects. Therefore, target termination should not be performed immediately once the detection is lost. The threshold value for termination is set to $N_T = 8$. The detection lose condition of a target j can be described as

$$max\{pij\} = (1 - p_D)\beta \tag{20}$$

That means the maximum assignment likelihood within the jth target Gating is when no measurement is assigned to the target.

The initiation of the target starts when a new target appears within the FOV. Measurement assignment is needed to be performed before target initiation since the algorithm needs to know which measurement is a new target. However, as described in (14), JPDA uses the weighted sum of all measurements within the Gating to update the state estimation, this assignment strategy cannot be applied directly to target initiation. A nearest neighbor assignment strategy is introduced for target initiation (19)

$$\mathbf{z}_k^j = \{\mathbf{z}_{ij} \mid \mathbf{D}_M^{ij} = min\left(\sqrt{(\mathbf{y}_k^j)^T \mathbf{S_k^{-1}} \mathbf{y}_k^j}\right)\} \tag{21}$$

After Mahalanobis distance is calculated as (8), the single measurement with the smallest Mahalanobis distance to jth prior estimation is assigned to the jth target. After this assignment, unassigned measurements are considered as new targets and initiated.

Algorithm 1 Joint Probabilistic Data Association Tracker

```
Input: $\mathbf{z}_k$, $\mathbf{H}$, $\mathbf{R}$, $\mathbf{F}$, $\mathbf{Q}$, $\mathbf{P}_0$, $\mathbf{X}_0$, $p_D$, $\beta$, $N_T$
Output: $\hat{\mathbf{x}}_{k|k}$, $\mathbf{P}_k^j$
 1: for $j$ in J do
 2:     $\hat{\mathbf{x}}_{k|k-1}^j = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1}^j$
 3:     $\mathbf{P}_{k|k-1}^j = \mathbf{F}\mathbf{P}_{k-1|k-1}^j \mathbf{F}^{\mathrm{T}} + \mathbf{Q}$
 4:     $\mathbf{S}_k^j = \mathbf{R} + \mathbf{H}\mathbf{P}_{k|k-1}^j \mathbf{H}^{\mathrm{T}}$
 5:     $\mathbf{S}_k^j = \mathbf{R} + \mathbf{H}\mathbf{P}_{k|k-1}^j \mathbf{H}^{\mathrm{T}}$
 6:     $\mathbf{K}_k^j = \mathbf{P}_{k|k-1}^j \mathbf{H}^{\mathrm{T}} (\mathbf{S}_k^j)^{-1}$
 7:     $\mathbf{D_M^j} = \sqrt{(\mathbf{y}_k^j)^T \mathbf{S_k}^{-1} \mathbf{y}_k^j}$
 8:     $\mathbf{y}_k^j = \mathbf{z}_k^j - \mathbf{H}\hat{\mathbf{x}}^j_{k|k-1}$
 9:     if if $p_a$ contains $i$ to $j$ then
10:         $p_{ij} = \sum_{a=1}^{A} p_a$
11:     end if
12:     $\mathbf{y_k^j} = \sum_{i=1}^{I} p_{ij}\mathbf{y_{ij}}$
13:     $\hat{\mathbf{x}}_{k|k}^j = \hat{\mathbf{x}}_{k|k-1}^j + \mathbf{K}_k^j \mathbf{y}_k^j$
14:     $\mathbf{P}_{k|k}^j = \mathbf{P}_{k|k-1}^j - (1 - p_{0j})\mathbf{K}_k^j \mathbf{H}\mathbf{P}_{k|k-1}^j + \mathbf{P}_k^j$
15:     if if $max\{pij\} == (1 - p_D)\beta$ then
16:         Terminate $j$th target
17:     end if
18: end for
19: for $i$ in I do
20:     $\mathbf{z}_k^j = \{\mathbf{z}_{ij} \mid \mathbf{D}_M^{ij} = min\left(\sqrt{(\mathbf{y}_k^j)^T \mathbf{S_k^{-1}} \mathbf{y}_k^j}\right)\}$
21:     if if $z_k^i \; ! \in z_k^j$ then
22:         Initiate $z_k^i$ as a new target
23:     end if
24: end for
25: return $\hat{\mathbf{x}}_{k|k}$, $\mathbf{P}_k^j$
```

The complete JPDA algorithm can be formulated as **Algorithm 1**, where JPDA first takes the previous targets states and covariance and current measurement. Then use Kalman filter to calculate the prior prediction of these targets states. Thirdly, based on the joint probabilistic data association likelihood, states predictions and their covariance are updated by the weighted sum of in Gating measurements. After all targets from the previous frame are processed, the Termination of non-observable targets and initiation of new observed targets are performed. Finally, the updated states predictions and covariances are returned as the previous targets states and covariance for the next frame.

V. Results

This algorithm is tested using both public dataset and UAV aerial videos.

A. *Testing on 2D MOT 2015*

Since the detection algorithm and the model applied in this project is identical to the one evaluated in VOC benchmark, only the performance of JPDA is evaluated using the public dataset. The performance evaluation of the JPDA tracking algorithm is done on the popular PETS09 and TUDS video sequences of the MOT15 dataset[11].

The performance evaluation of JPDA can be seen as TABLE II. Where MOTA stands for the Multiple Object Tracking Accuracy, which combines three error sources: false positives, missed targets and identity switches. The performance of JPDA is also compared with the some state-of-art algorithms KCF[11], Appearance Model with R-CNN[12], SiameseCNN[13] based tracker, and Dynamic Programming[14] tracker on this dataset. These algorithms are called as KCF, APRCNN, SiameseCNN, DP_NMS for simplification.

TABLE II
PERFORMANCE COMPARISON OF JPDA AND OTHER METHODS

Dataset	Algorithm	MOTA	FPS	Hardware Spec
TUD	JPDA	56.3%	30	TX2, 2GHz
	KCF[11]	76.0%	0.3	i7 3.6GHz, 4 Core
	APRCNN[12]	61.3%	6.7	GTX1080, 2.3GHz
	SiameseCNN[13]	73.7%	52.8	3Ghz 24 Cores
	DP_NMS[14]	48.6%	448	2.6GHz, 16 Cores
PETS	JPDA	32.7%	60	TX2, 2GHz
	KCF[11]	51.8%	0.3	i7 3.6GHz, 4 Core
	APRCNN[12]	38.9%	6.7	GTX1080, 2.3GHz
	SiameseCNN[13]	34.5%	52.8	3Ghz 24 Cores
	DP_NMS[14]	33.8%	448	2.6GHz, 16 Cores

From TABLE II, it can be seen that the performance of JPDA is approximately 20% worse than the most accurate algorithm on this dataset, KCF. However, the processing speed of KCF is only 0.3 FPS on an i7 3.6GHz, 4 Core CPU. As for JPDA, the processing speed is highly dependent on the number of targets in the FOV. For PETS and TUDS, the processing speed of JPDA can get to 30 FPS and 60 FPS on NVIDIA Jetson TX2, which is much faster than KCF. As for the DP, one of the fastest algorithm on this dataset, JPDA has an approximately 8 accuracy advantage. From the comparison to other popular algorithms, it can be seen that JPDA is a good compromise between tracking accuracy and processing speed. Moreover, the 2D MOT 2015 dataset uses a publicly available detection algorithm for the tracking algorithm evaluation. It can be seen from the VOC benchmark, YOLO has much better performance than the discriminatively trained part-based model [11]. The detection mAP of YOLOv2 exceeds 80% on all object classed whereas discriminatively trained part-based model can only achieve less than 60% mAP on all object classes. Therefore, it can be predicted that the change of detector to YOLOv2 can improve the performance of JPDA on this dataset.

Moreover, the YOLO+JPDA detection and tracking algorithm is also tested on a traffic surveillance video [15]. In order to test the feasibility of this approach in real UAV missions, the JPDA tracker is not only tested on the original video, but also tested on 5 FPS video with the consideration of the slow processing speed of YOLO. Results are shown in Fig. 3. Since the original resolution is not compatible with the YOLO detection algorithm, quantitative analysis is not done on this dataset. However, from the testing video, it can be seen that the pedestrian and road vehicles can be detected and tracked both with original and lower FPS videos. It can also be seen from the detection and tracking result that, JPDA not only gives the ID of each target but also captured the location of temporary undetected objects. Since there are less than 10 targets in the FOV of the video, JPDA can maintain a very high processing speed from 100 FPS to 200 FPS on NVIDIA Jetson TX2.

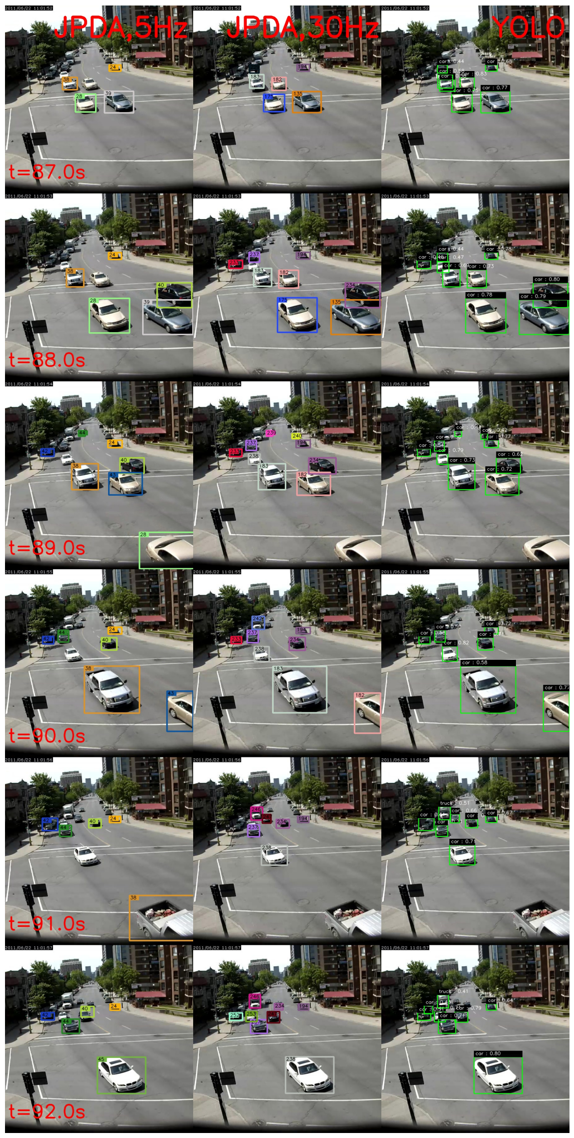


Fig. 3. Detection and tracking in a traffic surveillance video

B. Testing on UAV aerial videos

The YOLO+JPDA algorithm is also tested on UAV aerial videos [16] (also with original and 5 FPS video) to assess its robustness for possible disturbances during UAV flight like vibration, image blurry, FOV temporary lose, too small target due to high altitude, etc. The video is obtained from UCF-Lockheed-Martin UAV Dataset. From the result in Fig. 4, it can be seen that during UAV flight disturbances, the detection and tracking algorithm is capable of capturing the objects. The JPDA can compensate for the detection lose for a short periods. However, because of the unmodeled motion of the UAV, the constant-velocity dynamic model in JPDA can cause miss tracks.

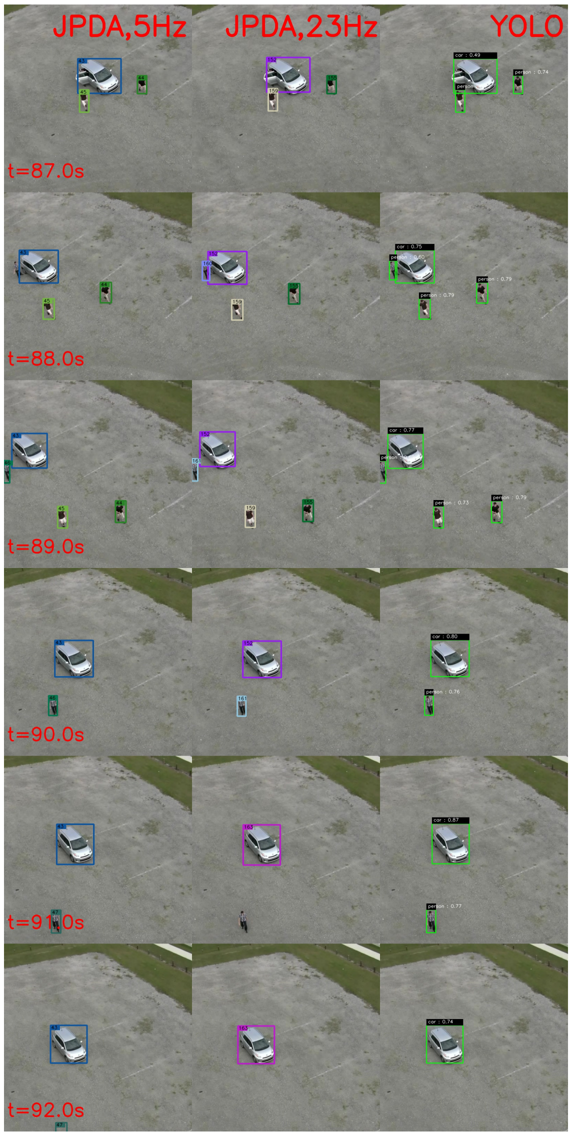


Fig. 4. Detection and tracking in an aerial video

VI. CONCLUSIONS

This paper presented the development and implementation of a computer vision based multiple object detection and tracking algorithm. The YOLOv2 deep learning object detection and JPDA multiple object tracking algorithm are developed and implemented. The performance is evaluated quantitatively and qualitatively on both public dataset of a traffic surveillance video and UAV flight video.

The performance evaluation of JPDA on public dataset had an approximately 20% performance downgrade when comparing to a state-of-art multiple object tracking algorithm. However, the processing speed was significantly faster, improving from 6.7 FPS to 38 FPS respectively. The aerial video test shows the pedestrian and road vehicles can be detected and tracked, tracking algorithm can compensate the detection lose for a short period. The limitation of this algorithm in the application of UAV visual object detection and tracking is that the unmodeled motion of the UAV could lead to object false tracking. Future work will be dedicated to solving this problem by applying more complex dynamic models in JPDA.

REFERENCES

[1] Li, Zhengrong, et al. "Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved Hough transform." Machine Vision and Applications 21.5 (2010): 677-686.

[2] Kiszka, Jeremy J., et al. "Using unmanned aerial vehicles (UAVs) to investigate shark and ray densities in a shallow coral lagoon." Marine Ecology Progress Series 560 (2016): 237-242.

[3] Mulero-Pzmny, Margarita, et al. "Remotely piloted aircraft systems as a rhinoceros anti-poaching tool in Africa." PloS one 9.1 (2014): e83873.

[4] Doherty, Patrick, and Piotr Rudol. "A UAV search and rescue scenario with human body detection and geolocalization." Australasian Joint Conference on Artificial Intelligence. Springer, Berlin, Heidelberg, 2007.

[5] Everingham, M. and Van Gool, L. and Williams, C. K. I. and Winn, J. and Zisserman, A. "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[6] "Multiple Object Tracking Benchmark." https://motchallenge.net/results/MOT17/.

[7] Li, Zuoxin, and Fuqiang Zhou. "FSSD: Feature Fusion Single Shot Multibox Detector." arXiv preprint arXiv:1712.00960 (2017).

[8] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[9] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." arXiv preprint 1612 (2016).

[10] Rezatofighi, Seyed Hamid, et al. "Joint Probabilistic Data Association Revisited." ICCV. 2015

[11] Ristani, Ergys, et al. "Performance measures and a data set for multi-target, multi-camera tracking." European Conference on Computer Vision. Springer, Cham, 2016.

[12] Chen, Long, et al. "Online multi-object tracking with convolutional neural networks." Image Processing (ICIP), 2017 IEEE International Conference on. IEEE, 2017.

[13] Leal-Taix, Laura, Cristian Canton-Ferrer, and Konrad Schindler. "Learning by tracking: Siamese CNN for robust target association." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2016.

[14] Pirsiavash, Hamed, Deva Ramanan, and Charless C. Fowlkes. "Globally-optimal greedy algorithms for tracking a variable number of objects." Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. IEEE, 2011.

[15] Saunier, Nicolas, et al. "A public video dataset for road transportation applications." Transportation Research Board Annual Meeting Compendium of Papers. 2014.

[16] "UCF Aerial Action Data Set". http://crcv.ucf.edu/data/UCF_Aerial_Action.php