Accurate specified-pedestrian tracking from Unmanned Aerial Vehicles

Zhen Yang, Zhiyi Huang, Yanhui Yang, Fan Yang and Zhijian Yin
School of Communications and Electronics, Jiangxi Science and Technology Normal University
Nanchang, China
e-mail: {yangzhenphd, huangzhiyi72, yanhuiy, kooyang, zhijianyin}@aliyun.com

Abstract—**Recently, accurate target tracking is widely used in the field of Unmanned Aerial Vehicles (UAV). In this paper, we focus on the application of detecting and following a walking pedestrian in real time from the moving platform with many interferences. We present a scheme that uses CNN model (YOLO-V2) to detect pedestrian and matches the walking pedestrian with a postprocessing and feature queue and Locality constrained Linear Coding algorithm. After that the ground station receives and analyses the video stream from the parrot and sends back commands to control the motion of UAV. At the beginning of the tracking process, the UAV is hovering when one pedestrian will be selected as the special target. Visual information is acquired only through a front camera without assistant sensors. A parrot Bebop 2 is adopted in the experiment, which is the basis for doing experiments outdoors and experimental result verify the effectiveness of our solution.**

Keywords-Target tracking; unmanned aerial vehicle; feature queue; YOLO-V2

I. Introduction

UAV is popular in computer vision because of their flexibility and small capacity [1, 2, 3, 4]. Compared to the traditional platform such as static camera (stationary, limit to its tracking time, its tracking precision and efficiency), Dynamic platform is more complexity, useful and worthy, where traditional pedestrian tracking method based on UAV is mainly used by hand crafted features and classified, its deficiency is sensitivity to color and illumination. During object tracking, dynamic pedestrian have dramatically characteristics under occlusion and illumination changes in every direction.

In recent years, visual tracking research has made substantial progress on UAV and almost traditional tracker develop into innovative and well-known tracker for faster speed and higher precision of tracking. Currently, many deep learning trackers are applied on target tracking [5], [6], [7], e.g. MD-Net[8], however, their model require a large amount of computation (MD-Net only handle one FPS). The parrot bebop2 may get lost or follow the wrong goal by deep learning trackers. Thus these trackers are only used in traditional platform and they easy to lost the tracking object because of the interference of mixing up the target in similar dressing effectively.

To solve the deficiency of above algorithm, we present a novel strategy for accurate tracking pedestrian in the dynamic scene seeing Fig. 1. In this paper, we use YOLO-V2 deep detection model and combine postprocessing with LLC matching method, meanwhile changing the bounding-box with color feature and the queue strategy. Our approach is consist of four parts: pedestrian detection, feature extraction, reconstruction queue and feature match. Firstly, each patch of pedestrian is draw for next step; Secondly, considering the robustness and efficiency of detection algorithm, we do not use the traditional feature descriptor such as HOG [9], SIFT [10] or HOG+SVM [11], and selecting RGB+LAB feature based on stripes and blocks in the spatial structure to achieve pedestrian tracking; Next, we use postprocessing for matching queue to restrict the bounding box which are pushed into matching queue; Finally, we utilize sparse reconstruction method based on LLC which will update online in different periodic time.

The remainder of this paper is composed as follows: Section 2 describes our work in detail; Section 3 shows the tracking method; Section 4 includes experimental results and discussion; Finally, we draw the conclusions in Section 5.

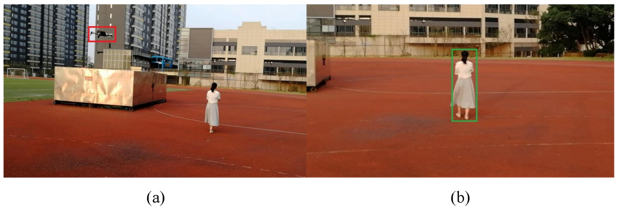


Figure 1. (a) One frame recorded from a bystander, and the UAV is marked in red. (b) One frame comes from Parrot Bebop2 and the target is marked in green.

II. Description in detail

Motion Control

In this paper, we adopt a parrot bebop2 UAV with a front automatic camera as experiment tool. In the ROS system, it achieves the connection that laptop can require the video from the camera of UAV. A lot of simple commands are inputted on laptop to control the UAV, for example, the basic motion control of UAV includes: fore-and-aft, left-to-right and horizontal rotating.

Before taking off, the UAV keep a fixed distance from object for safety. When the interesting object occurs, we input the command named m to enforce UAV into autonomic module, it will draw a bounding-box to flag object and track the object seeing Fig. 2. If possible, it will calculate the bounding-box of object in tracking module later. In case, it finds no object in this bounding-box, the UAV will receive a hover command, soon after, it will use the bounding-box of prior frame.

Drone is a flying robot running freely, thus we need to control both in horizontal and vertical directions. On each image, we assume the object is placed at the center of each frame, even if some shift in the vertical direction is allowed on the occasions that we want the UAV to fly high. When target move to left or right, the UAV will rotate in horizontally.

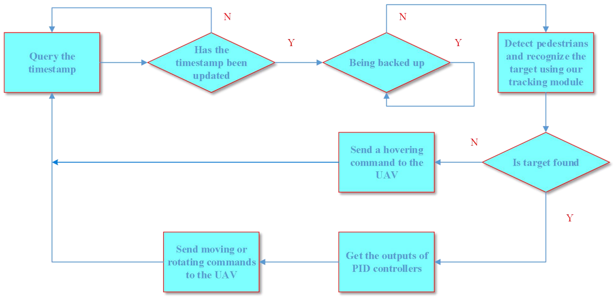

Query the timestamp
Has the timestamp been updated
N
Y
Being backed up
N
Y
Detect pedestrians and recognize the target using our tracking module
Is target found
N
Send a hovering command to the UAV
Y
Get the outputs of PID controllers
Send moving or rotating commands to the UAV

Figure 2. The framework of the auto-control thread, which is the main working thread after the auto-control mode is started.

Adopting the front camera, drone tries to keep a adaptive distance from the target. Assume the height of image is H and the height of bounding-box is h , the distance is estimated by the value of h/H . Because, using the width of image and bounding-box to calculate distance may cause many challenges when target make a turn. The median filtering method is used to reduce the influence when any motion abruptly.

Proportional-derivative controllers (PD controllers) are used to control the motion of UAV. Among them, one PD controller is mainly used to control the fore-and-aft motion, while other is responsible for rotating motion.

III. PROPOSED APPROACH

A. Pedestrian detection

In our approach, the specified-pedestrian tracking method is based on UAV. Therefore, we use deep neural network (YOLO-V2) to detect the target from camera of parrot. Considering YOLO-V2 model have a great performance on detection, we adopt YOLO-V2 model to detect pedestrian, and the weight is trained by the data set (total of images is 41719) that includes the VOC2007 training set, VOC2012 training set and other data (shoot by drone). In the training period, we set the confidence within 0.3 and 0.5 because of good confidence can improve the precision of detection in object tracking. In our experimental process, when the target is over the edge of the picture, one frame is built unsuccessfully in matching process. Thus, we add a restrictive condition that the left-top point or the right-bottom point of bounding is less than the left-top point or the right-bottom point of image when the target is over the edge of the picture.

B. Feature extraction

We combine RGB and LAB color feature in this paper. Due to the affection of lightness, we mainly use five color features that includes three channels (R, G, B) of RGB and two channels (A, B) of LAB for extracting object feature.

For reducing the affection of requiring bounding-box of pedestrian by YOLO-V2, Lisanti [12] pointed out a method that flagging target with inscribed elliptical area rather than rectangle area where most effective information is contained. In the experiment, we calculate the information within the inscribed elliptical area and regard others as background seeing Fig. 3. Given a patch of pedestrian, we suppose the height of object is h and the width is w, then, setting $a = w/2$ and $b = h/2$. We choose the center point as origin point. And $f(x, y)$ denotes the pixel value of the point (x, y) ,the formula as follows:

$$f(x,y) = \begin{cases} f(x,y), \dfrac{x^2}{a^3} + \dfrac{y^2}{b^2} \le 1 \\ 0, others \end{cases} \tag{1}$$

$$g(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}} \tag{2}$$

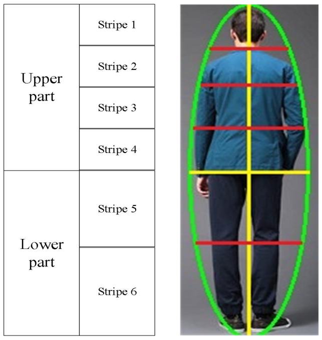


Figure 3. Color feature extraction. The bounding-box is change from rectangle to ellipse.

For the inscribed elliptical area, the patch is quickly divided into the upper half and the lower half in a ratio of 0.55. In the upper half part, we divided into four equal stripes and regard the head as the first stripes. Because of the enormous difference between foreground and background, we abandon the first stripe. We use Gaussian distribution function to change the weight of each point according to the x value of stripes except the first stripe.

We set $\mu = 0$, $\sigma = 1$, and in the lower half part, we cut this part into two stripes in the vertical direction. Note that, the pedestrian's calves part varies greatly on the front and sides and it contains a lot of background information. Therefore, we only adopt the first stripe of the lower part. Furthermore, we do not use the Gaussian distribution function and chain the color feature of RGB and AB channels directly.

In addition, to avoid the influence of overlap part with other pedestrian, each stripe is divided into two blocks in horizontal direction and extract color feature individually. We use a 6×6×6 cube to store the RGB feature and use a

15×15 cube to store the AB feature, after extracting feature, each block is resized into a vector. Then, color feature descriptors can then be obtained by linking feature vectors.

C. *Postprocessing for matching queue*

After getting some bounding-boxes of each image from the output of YOLO-V2, some bounding-boxes are useful and near to target pedestrian, but the other bounding-boxes are far from the target and they are also pushed to count distance between the bounding-box of current frame and the bounding-box of last frame and it will impact the result of matching queue and raise the calculation load. Thus, we set a postprocessing for matching queue to eliminate some bounding-boxes which are far away from target.

We assume the height of image is H and the height of bounding-box is h_i meanwhile the width of image is W and the width of bounding-box is w_i . We set the left-top point of current bounding-box (current frame) $A(x_i, y_i)$ and the left-top point of last bounding-box (last frame) is $B(x_l, y_l)$, then, we do a judge and the formula as fellows:

$$(x_i - x_2)^2 + (y_i - y_2)^2 \le (h_i \times \frac{h_i}{H})^2 + (w_i \times \frac{w_i}{H}) \tag{3}$$

$$0 \le i \le n-1 \tag{4}$$

D. *Queue-based Matching*

Finding out object pedestrian and matching it are the vital process of Queue-matching. We introduce a queue to store the highest accuracy of match of feature descriptor. According the reconstruction error, we decide to implement upon the queue in real time and whether to trust the current patch and whether to update the queue.

Figure 4. The feature queue. Measuring the Euclidean distances between a pedestrian and the queue feature ,which is updated by highly matched feature descriptors online.

After flagging a object pedestrian by a bounding-box, the tracking module starts. In an image, all of the features within the bounding-box will be added into a reserve queue and calculate the Euclidean distance between the feature descriptor and the all feature descriptor of queue in short period of time. Then we choose the most similar feature descriptor and record the matched times. If the number of matched time exceed the current size of queue at 50%, it will stop at current frame and add its feature descriptor into queue.

Meanwhile, once the number of matched times is not smaller than a higher ratio of the current size of the queue, 80% for example, the feature descriptor of the patch will be pushed into the queue. Before the size of current queue get to the max size of queue, 15 for example seeing Fig. 4, no feature descriptor will be pushed out of the queue. This is the procedure that the queue is constructed within several seconds.

Once it gets to the max size of queue, rule is changed. Supposing y as the feature descriptor of current image, and matrix C is consist of the feature descriptor of current queue. Using LLC criteria [13] to acquire the coefficient of Matrix C , and place $\mathrm{L}_1 - norm$ constraint with $\mathrm{L}_2 - norm$ constraint which is defined as:

$$\min_{\alpha} \left\| y - C \cdot \alpha \right\|_2^2 + \lambda \sum_i \left(\alpha_i \cdot \exp\left(\frac{\left\| y - C_i \right\|_2}{\sigma} \right) \right)^2 \tag{5}$$

$$s.t. 1^T \alpha = 1 \tag{6}$$

Where λ controls the sparsity degree and σ is used to adjust the decay speed of the locality adaptor. Then we get the response coefficient vector α , and calculate the reconstruction error e :

$$e = \frac{\left\| y - C \cdot \alpha \right\|_2}{\left\| y \right\|_2} \tag{7}$$

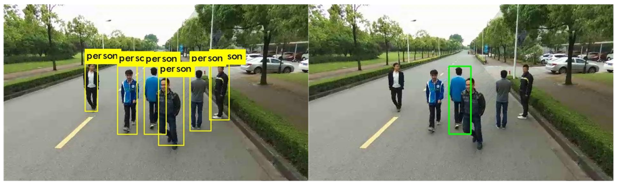
We calculate the reconstruction error of pedestrian of each frame and choose one which has minimum reconstruction error as special object. Furthermore, if this error is small adequately, within 3% to 50%, for example, this feature descriptor is similar to previous feature descriptor. In order to obtain the best construction object, we calculate many errors and choose the minimum among them that is pretty useful when the target and the neighboring pedestrians are similar.

Because the adjustable criterion of queue that makes queue matching more precise and updated online. Tracking module is able to receive image from control module and the location of previous reconstruction object is stored in tracking module for the reason that tracking module can send the location of previous reconstruction object when losing object.

IV. Experiments

In this experiment, the hardware includes parrot bebop 2 and a laptop with an Intel i7-7700HQ CPU @2.80 GHZ. Tracking algorithm is written in C++ + opencv + DarkNet, with four test datasets: dataset A seeing Fig. 5 (2351 frames), dataset B seeing Fig. 6 (1105 frames), dataset C seeing Fig. 7 (2025 frame) and dataset D seeing Fig. 8 (public datasets).

In the dataset A: target walking along road with the interference of someone who dresses similarly. In the dataset B, the UAV tracks the target who walking around continuously. And in the dataset C: when the target pedestrian walks whose appearance is similar to background, other pedestrian shades the target frequently. Dataset D is downloaded from https://motchallenge.net/vis/MOT16-11. In the tracking procedure, the UAV keeps away from the target about four meters. Four performance indexes and Eight tracking methods include MIL [14], BOOSTING [15], MEDIANFLOW [16], TLD [17], KCF [18], CHAO [19] are used to compare the tracking methods. In the four comparison table, the bold number is the best indicator. The first one is average error of measuring the distance between object pedestrian and the real one. The second one is maximum error. Then, the third one is the percentage of lost frame and the subsequent index is time loss of tracking. The last one is FPS (frames per second).

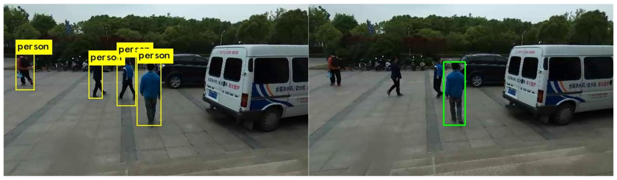


(a) (b)

Figure 5. (a) The view of the detection at one frame of the dataset A. (b) The view of the tracking of UAV at the frame which is the same as (a).

TABLE I. PERFORMANCE ON DATASET A

Tracking Methods	Maximum Error	Average Error	Lost Frames	Cost Time (ms)
MIL [14]	322.00	51.25	13.66%	197.23
BOOSTING [15]	261.44	30.24	13.89%	34.62
MEDIANFLOW [16]	110.49	59.95	77.75%	7.593
TLD [17]	110.49	45.95	31.03%	114.28
KCF [18]	364.93	11.33	**0%**	**6.578**
CHAO [19]	98.59	**3.22**	0.36%	94.4776
OURS(yolo-v2)	**19.00**	4.29	**0%**	109.17
OURS(yolo2-post processing)	**19.00**	4.21	**0%**	116.27

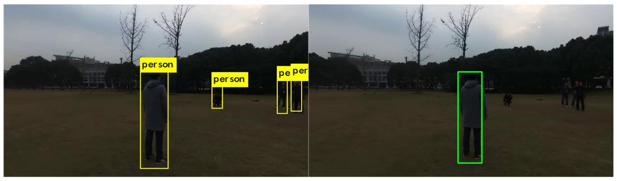


(a) (b)

Figure 6. (a) The view of the detection at one frame of the dataset B. (b) The target pedestrian of tracking is marked in green. (a) and (b) happened at the same time.

TABLE II. PERFORMANCE ON DATASET B

Tracking Methods	Maximum Error	Average Error	Lost Frames	Cost Time (ms)
MIL [14]	359.52	71.34	87.32%	144.30
BOOSTING [15]	357.17	102.13	74.95%	29.98
MEDIANFLOW [16]	283.92	88.29	97.45%	7.30
TLD [17]	352.41	88.29	24.88%	127.22
KCF [18]	511.03	88.38	52.32%	**5.57**
CHAO [19]	76.78	**4.12**	**0.04%**	97.75
OURS(yolo-v2)	68.06	21.30	3.00%	106.38
OURS(yolo2-post processing)	**61.5873**	20.43	1.77%	112.01

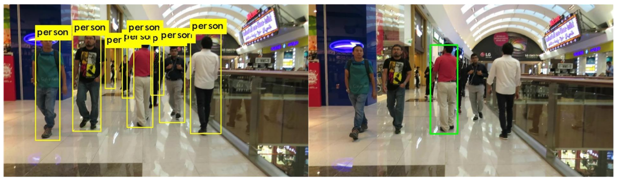


(a) (b)

Figure 7. (a) The view of the detection at one frame of the dataset C. (b) The target is marked with a green rectangle who the appearance is similar to background.

TABLE III. PERFORMANCE ON DATASET C

Tracking Methods	Maximum Error	Average Error	Lost Frames	Cost Time (ms)
MIL [14]	349.92	181.81	68.50%	182.15
BOOSTING [15]	341.40	204.69	84.40%	23.61
MEDIANFLOW [16]	164.83	33.67	32.20%	**7.15**
TLD [17]	424.31	70.39	35.30%	228.31
KCF [18]	260.60	90.11	61.70%	7.37
CHAO [19]	280.06	46.11	32.77%	105.26
OURS(yolo-v2)	**138.00**	33.04	31.31%	103.95
OURS(yolo2-post processing)	**138.00**	**32.11**	**29.72%**	110.02



(a) (b)

Figure 8. (a) The view of the detection at one frame of the dataset D. (b) The target is marked with a green rectangle and the test of tracking is happening in public.

TABLE IV. PERFORMANCE ON DATASET D

Tracking Methods	Maximum Error	Average Error	Lost Frames	Cost Time (ms)
MIL [14]	295.91	71.34	40.89%	182.15
BOOSTING [15]	290.90	102.13	92.22%	25.16
MEDIANFLOW [16]	172.25	88.29	90.11%	**7.80**
TLD [17]	366.96	88.29	66.22%	204.08
KCF [18]	580.10	88.38	21.00%	10.36
CHAO [19]	109.48	23.22	12.80%	105.26
OURS(yolo-v2)	82.55	30.46	31.22%	114.94
OURS(yolo2-post processing)	**68.06**	**16.91**	**10.56%**	110.01

V. CONCLUSIONS

Compared to the traditional method, this method-one that use YOLO-V2 module to detect has obvious promotion on maximum seeing Table I error and lost frame seeing Table II meanwhile has small amplitude enhancement on average error of individual data sets. The cost time is only about 110ms (9.09fps) seeing Table III because of the CNN, which needs huge amount of calculation. Compared to method-one, the method-two combines YOLO-V2 and postprocessing have significant promotion seeing Table IV on maximum error, average error, lost frame and cost time in dataset D, because the target pedestrian of image of data set D walk in a dense public place. Based on UAV, this work adopt CNN to detect object while achieve special pedestrian tracking through add the postprocessing and feature extraction and matching queue. Adopting LLC (linear-coding method) can eliminate the time of tracking meanwhile achieve a high precision of tracking. This approach is tested in real circumstance through the UAV seeing and the influence of communication overhead between the UAV and the laptop can be ignored. When the lightness changes and continuous pedestrian interference, this approach have a robust performance. This is a stable method that used in special pedestrian tracking on platform of UAV.

VI. ACKNOWLEDGEMENTS

This work was supported by the Foundations of Open Project under no. KFKT2018003, and supported in part by the National Natural Science Foundation of China (61603161), and the Science Foundation of Artificial Intelligence and Bioinformatics Cognitive Research Base Fund of Jiangxi Science and Technology Normal University of China (2017ZDPYJD005).

REFERENCES

[1] Rahimi, A.M., Ruschel, R., and Manjunath B.S., "UAV Sensor Fusion With Latent-Dynamic Conditional Random Fields in Coronal Plane Estimation," Computer Vision and Pattern Recognition, Vol.1, pp. 4527-4534, 2016.

[2] H. Zhou, H. Kong, L. Wei, D. Creighton, and S. Nahavandi, "Efficient road detection and tracking for unmanned aerial vehicle," IEEE Transactions on Intelligent Transportation Systems, vol. 16, pp. 297-309, 2015.

[3] R. Montanari, D. C. Tozadore, E. S. Fraccaroli, and R. A. Romero, "Ground vehicle detection and classification by an unmanned aerial vehicle," in 2015 12th Latin American Robotics Symposium and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR), 2015, pp. 253-258.

[4] Zhou H, Kong H, Wei L, Creighton D, and Nahavandi S, "Efficient road detection and tracking for unmanned aerial vehicle," IEEE Transactions on Intelligent Transportation Systems, 16.1, 297-309, 2015.

[5] Li H , Li Y , Porikli F . DeepTrack: Learning discriminative feature representations online for robust visual tracking[J]. IEEE Transactions on Image Processing, 2016, 25(4): 1834.

[6] Chen Y , Yang X , Zhong B , et al. CNNTracker: online discriminative object tracking v deep convolutional neural network[J]. Applied Soft Computing, 2016, 38: 1088- 1098.

[7] Qiao Liu, Xiaohuan Lu, Zhenyu He*, Chunkai Zhang, Wen-Sheng Chen, Deep Convolutional Neural Networks for Thermal Infrared Object Tracking, 2017, Knowledge-Based Systems, vol.134, pp.189-198, 2017. (SCI, JCR 1, Impact factor: 4.529)

[8] MDNet: Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. Hyeonseob Nam, and Bohyung Han. CVPR 2016.

[9] Dalal N, and Triggs B. "Histograms of oriented gradients for human detection," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 886-893, 2005.

[10] Ng PC, and Steven Henikoff. "SIFT: Predicting amino acid changes that affect protein function," Nucleic acids research, 31(13), 3812-3814, 2003.

[11] C. Cortes and V. Vapnik, "Support-vector networks," Machine learning, vol. 20, pp. 273-297, 1995.Ng PC, and Steven Henikoff. "SIFT: Predicting amino acid changes that affect protein function," Nucleic acids research, 31(13), 3812-3814, 2003.

[12] G. Lisanti, I. Masi, A. D. Bagdanov, and A. Del Bimbo, "Person re-identification by iterative re-weighted sparse ranking," IEEE transactions on pattern analysis and machine intelligence, vol. 37, pp. 1629-1642, 2015.

[13] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality constrained linear coding for image classification," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 3360-3367.

[14] Babenko, B., Yang, M.-H., Belongie, S.: Robust object tracking with online multiple instance learning. IEEE Trans. Pattern Anal. Mach. Intell. 33(8), 1619 – 1632 (2011)

[15] Helmut Grabner, Michael Grabner, and Horst Bischof. “Real-time tracking via on-line boosting.” in BMVC, volume 1, page 6, 2006.

[16] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. “Forwardbackward error: Automatic detection of tracking failures.” in Pattern Recognition (ICPR), 2010 20th International Conference on, pages 2756–2759. IEEE, 2010.

[17] Shi MY., Zhan DC. (2013) Multi Gesture Recognition: A Tracking Learning Detection Approach. In: Sun C., Fang F., Zhou ZH., Yang W., Liu ZY. (eds) Intelligence Science and Big Data Engineering. IScIDE 2013. Lecture Notes in Computer Science, vol 8261. Springer, Berlin, Heidelberg.

[18] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, pp. 583-596, 2015.

[19] Bian, C., Yang, Z., Zhang, T., & Xiong, H. (2017). Pedestrian tracking from an unmanned aerial vehicle. IEEE, International Conference on Signal Processing (pp.1067-1071). IEEE.