ISBN 978-93-84422-37-0
14th International Conference on Recent Trends in Engineering and Technology (RTET--2018)
Pattaya (Thailand) April 24-25, 2018

Car Localization in Aerial Images Taken from Quadcopter

Hasan Saribas[1], Hakan Cevikalp[2] and Sinem Kahvecioglu[1]

[1]Department of Avionics, Faculty of Aeronautics and Astronautics, Anadolu University, Eskisehir, TURKEY
[2]Department of Electric and Electronics Engineering, Eskişehir Osmangazi University, Eskisehir, TURKEY

[1]hasansaribas@anadolu.edu.tr, [2]hakan.cevikalp@gmail.com, [1]skahvecioglu@anadolu.edu.tr

***Abstract*:** *Unmanned aerial vehicles (UAV) are popular research platforms that find increasing amount of applications in many areas, such as military, civil, commercial, and entertainment due to their high maneuverability, vertical take-off and landing abilities, and suitability for use in indoor and outdoor spaces. Today, small, and single board computers with very high CPU/processor capacities are developed, and by means of these processors, which will be inserted into unmanned aerial vehicle platforms, many real-time machine vision applications became possible. This study discusses the problem of car localization in aerial images taken from unmanned aerial vehicles. Within this context, a new dataset was created by using quadcopter-type unmanned aerial vehicles and various cameras. Both Polyhedral Conic Classifier and You Only Look Once (YOLO) algorithm, which is currently one of the fastest methods in literature, and uses deep learning architecture, were used to locate the cars in collected images, and the results were compared.*

Keywords: *Unmanned aerial vehicle, polyhedral conic classifier, deep learning, you only look once.*

1. Introduction

Object localization is a machine vision application, which requires detection of any example of a general object in a digital image along with its position and scale. This has recently become a popular subject with increasing number of security, robotics, military, and commercial fields of application. On the other hand, despite of numerous significant developments in the last decade, localization of the objects in digital images is very difficult. The most important reason for this is that the data samples of the same class differ in terms of appearance, color, texture, and pose/exposure. Many natural object groups, such as humans, cats, and chairs, include flexible deformations, and similar objects look quite different in images, which were taken from different viewpoints. In addition to these difficulties, differences in scale and light, complex backgrounds, overlapping, and cropped object images are some of the most significant factors that obstruct the problem of localization.

There are two main factors that affect the performance of object localization: Features that are used to describe the samples, and the learning algorithm that performs object localization process. Histogram of Oriented Gradients (HOG) [1], Scale Invariant Feature Transform (SIFT) [2], Local Binary Patterns (LBP) [3], and CNNs (Convolutional Neural Networks) are the most common features that are used for object localization. These methods can be used separately or as hybrids [4]. In order for the machine to detect the desired object, we must use a learning algorithm that separates the object class samples from the background. To this end, classifiers, such as support vector machines (SVM), artificial neural networks (ANN), nearest neighborhood, and decision trees are used.

In recent years, the most effective object localization algorithms have been deep learning-based algorithms. Krizhevsky et al. [5] obtained very successful results in image classification through large data by using convolutional neural networks. Since then, the interest in deep learning has been rapidly increasing due to the higher rates of success. Deep learning has many applications in various machine learning fields, such as image processing, sound analysis, classification, and text recognition. The difference between deep learning algorithms and other algorithms in machine learning is that it requires data in higher quantities, and devices with very high calculation capacity that will be able to process these data with its complex structure. The amount of labeled data has reached millions thanks to social media platforms that are used by too many people. Together with advancing technology, processors, which are able to run algorithms that are capable of processing these higher amounts of data, and deep learning grabbed the attention of companies with high data storage capacities (Google, Facebook, Microsoft, Nvidia, Baidu). In addition, these companies turn their libraries into open source libraries, and greased the skids for developments in deep learning. By means of graphics processing units (GPU) developed by Nvidia, it became possible to realize real-time deep learning architectures. In line with these developments, real-time algorithms in object localization have been developed in recent years. The most significant ones may be listed as follows: YOLO (You Only Look Once) [6], SSD (Single Shot MultiBox Detector) [7], Faster R-CNN (Towards Real-Time Object Detection with Region Proposal Networks) [8], YOLOv2 [9].

In this study, we worked on localization of the cars in the aerial images obtained by quadcopters. For this purpose, object localization algorithm [4] that uses deep learning-based YOLO algorithm and sliding window-based polyhedral conic classifiers were used, and the performances and real-time performances of these methods were compared.

2. Method

In this study, the problems about detection and localization of cars in the images taken by quadcopters were addressed. Two methods were used for this purpose. The first method is the object localization algorithm that uses extended polyhedral classifier – EPCC [4], and the second method is YOLOv2 (You only look once 2) method, which is one of the fastest methods in literature that gives the best results.

2.1. Extended Polyhedral Conic Classifier

Unlike support vector machines, extended Polyhedral Conic Classifiers– EPCC model the class of an object as a polyhedral region, and use this model for classification purposes. Polyhedral conic function was first introduced to literature by Gasimov and Öztürk [10] and extended and used for object localization and classification by Cevikalp and Triggs [4]. Extended polyhedral classifier uses the following function,

$$f_{w,\gamma,c,b}(x) = w^T(x-c) + \gamma^T|x-c| - b. \tag{1}$$

In equation (1), $x \in \boldsymbol{R^d}$ denotes d dimensional test data, $c \in \boldsymbol{R^d}$ denotes the vertex of the cone, $w \in \boldsymbol{R^d}$ *and* $\boldsymbol{\gamma} \in \boldsymbol{R^d}$ are learned weight coefficients, $|\boldsymbol{u}| = (|\boldsymbol{u_1}|, |\boldsymbol{u_2}|, \dots, |\boldsymbol{u_d}|)^{\boldsymbol{T}}$ is the component-wise modulus, and b is the bias parameter. While all samples that satisfy the condition, $\boldsymbol{f(x) < 0}$, are classified as positives in polyhedral conic classifiers, all samples that satisfy $\boldsymbol{f(x) \geq 0}$ are classified as negatives. In order to make sure that the classifier is not subject to over-fitting, and that it can be used for very large databases, the problem is formulated as a quadratic programming similar to the one used in support vector machines. Within this context, if the cone vertex is taken as the average of positive samples, and if we show any data sample with

$$\widetilde{x} \equiv \begin{pmatrix} x-c \\ |x-c| \end{pmatrix} \tag{2}$$

and show the weight vector to be learned with the following equation

$$\widetilde{w} \equiv \begin{pmatrix} -w \\ -\gamma \end{pmatrix} \tag{3}$$

the classification problem turns into a quadratic optimization problem as in SVMs [4]. After obtaining the training parameters of the classifier, first, the test sample is augmented as,

$$\tilde{x}_{test} \equiv \begin{pmatrix} x_{test-c} \\ |x_{test-c}| \end{pmatrix} \quad (4)$$

Then the following decision function

$$f(\tilde{x}) = \tilde{w}^T \tilde{x}_{test} + b \quad (5)$$

can be used to assigned the test sample to the object class or back-ground based on the sign of the decision function.

The object localization algorithm using this classifier consists of root detectors based on sliding windows method as in [4]. First, the number of root detectors to be used for different car poses was determined, and then the dimensions of sliding windows of these root detectors were determined by using the annotated data in the training set. In order to eliminate the problems that may arise at labeling process of the data set during training of the localization, the latent training [11], [12] method was used. In this method, the positions of the boxes that show the actual positions of the object samples on the image were addressed as latent variables. First, the classifiers are trained by using the ground truth of objects. Then the trained system searches for the sample of the object in different positions and scales around each object sample. The position with the highest score is regarded as the actual position of the object, and these positions are used to re-train the system. Then the samples that do not include the samples of the object (difficult background objects and false positive objects) are collected and these samples are used as negatives to train the system. This process is repeated a few times, and the localization algorithm is finalized.

2.2. You Only Look Once v2 (YOLOv2)

YOLO, which is one of the deep learning algorithms that give the most correct and fast results in object detection and location, uses the open source-coded darknet library [9]. YOLO combines the different components of object detection in a single neural network. This neural network first divides the image into $S \times S$ cells and uses all features of the image in order to extrapolate all bounding boxes, and synchronously extrapolates all bounding boxes in that image. While creating its architecture shown in Figure 1, YOLO was inspired by the model created by GoogleNet for classification. This network consists of 23 convolutional layers, 5 maxpooling, and 2 fully connected layers. In addition, YOLO Tiny architecture was also used, which is a faster version of YOLOv2 model with a smaller architecture, and slightly lower performance. YOLO Tiny version consists of 9 convolutional layers and 6 maxpooling layers [9].

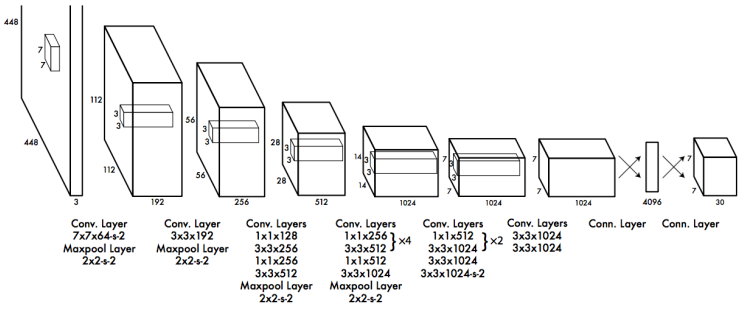


Fig. 1: YOLO architecture [9].

3. Experiments

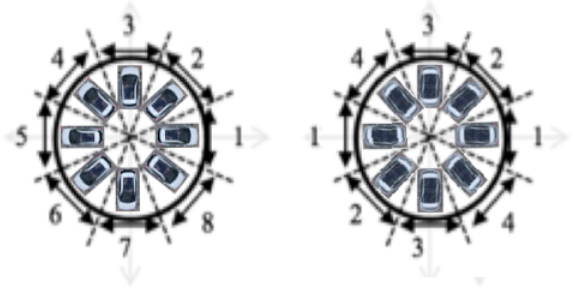
Although there are numerous sites where we can find data sets, one of the biggest problems is that these data must be annotated specifically for our problem. For a car detection and localization problem, we created our own data set consisting of colored digital images and the images that we obtained in different weather conditions and scales by using DJI matrice 600 Pro and DJI Inspire 1 unmanned aerial vehicles. The data set consists of approximately 10.000 colored digital images, and it contains approximately 30.000 aerial view car images. We annotated the cars by using the bounding boxes and created the data belonging to the positive class. The data set

of the negative class was created by using 700 digital images that did not contain any car views taken in various conditions. Figure 2 shows examples from the data set that contains positive images.

The most common and most realistic metric for success criterion in object localization algorithms is PASCAL VOC criterion. According to this metric, the position of the object is classified as wrong or right in accordance with the overlapping ratio of the detected coordinates and the ground truth positions. This overlapping was calculated by using $\frac{\boldsymbol{area\ |Q \cap R|}}{\boldsymbol{area\ |Q \cup R|}}$ formula. In this formula, Q shows the ground-truth location of object and R shows the location returned by the algorithm. If this is over 50%, the detected position is considered as true positive – TP, if not, it is considered as false positive – FP. Then the mean average precision-mAP was determined by using precision-recall curves.

Fig. 2: Samples of the positive dataset

For training with polyhedral conic classifier, the features of each car image among 30.000 car images within the data set were represented by the histogram of oriented gradient. In order for these features to make better sense, 8 different root detectors were designed by dividing the 360° angle into 45-degree angles based on aspect ratios of the front parts of the cars, and models were created by training the system. Then the images, of which 8 different orientations are located on the symmetries of each other according to x-y axis, are re-trained to be aligned in the same direction to accelerate the system, and models were created for 4 different root detectors. These orientations are designed as shown in Figure 3 (a) and (b). HOG coefficients of EPCC classifiers, which were trained for different orientations, were drawn in Figure 4 and Figure 5. As can be seen in Fig. 5, the trained classifiers learned each car orientation successfully.



(a) (b)

Fig. 3: (a) 8 Different orientations that the root detector was trained, (b) 4 Different orientations that the root detector was trained

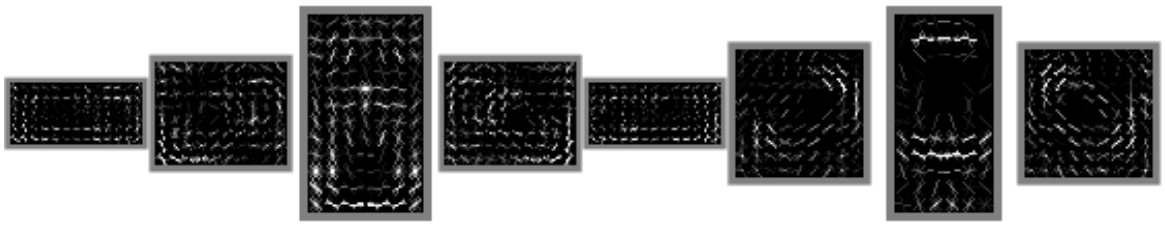


Fig. 4: HOG coefficients of classifiers trained for 8 different root detectors.

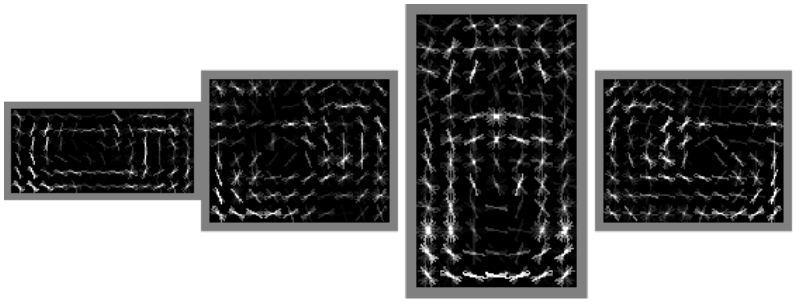


Fig. 5: HOG coefficients of classifiers trained for 4 different root detectors

Two different architectures were used when using Yolo method. YOLOv2 and YOLO Tiny. 2 Quadro K5000 GPUs were used when training these methods. Experimental results are shown in Table 1. The highest performance was obtained through EPCC method with 8 root detectors. This method was followed by YOLOv2. The lowest performance was obtained through YOLO Tiny method. In terms of speed, the fastest method was YOLO Tiny method, which has a smaller architecture. The slowest method was EPCC method with 8 root detectors, which showed the highest performance. However, it should be noted that while parallel programming was adopted in YOLO methods, parallelization was not adopted in EPCC methods. If the EPCC methods are run in parallel, the models will operate faster. Figure 6 shows the detector outputs of tested methods on some test images.

Fig. 6: Performances of classifiers on test images are represented as: YOLOv2 – Red, YOLO Tiny – Yellow, EPCC with 8 orientations – Black, EPCC with 4 orientations – Green rectangles.

TABLE I: Comparison of Methods for the Dataset

Method	Average Precision Score	Speed
EPCC with 8 Root Detectors	%84,49	2,2 sec. – 0,45 FPS
EPCC with 4 Root Detectors	%81,19	1.7 sec. – 0,58 FPS
YOLOv2	%83,35	0,136 sec. – 7,7 FPS
YOLO - Tiny	%80,19	0,0175 sec. – 57,1 FPS

4. Conclusion

In this study, we have compared the EPCCs with 8 and 4 root detectors and YOLOv2 and YOLO Tiny algorithms, which can detect the cars and return their positions in the images taken from UAVs in real time. The EPCC classifier using 8 root detectors achieves the highest score, but it is slower than the model that uses 4 root detectors. In YOLO models, which use deep-learning architecture, the average precision score of YOLOv2 model was higher, but slower than YOLO Tiny model. If the localization algorithms that use EPCC method are run in parallel, their speed will be similar to deep learning methods. As it is seen in experimental results, all tested methods achieved very high accuracies since the car poses in aerial videos can easily be grouped under 8 different categories.

5. References

[1] N. Dalal and B. Triggs, “Histogram of oriented gradients for human detection,” *IEEE Conference on Computer Vision and Pattern Recognition,* 2005.

[2] D. G. Lowe, “Object recognition from local scaleüinvariant features,” Computer vision, *International Conference on Computer Vision*, 1999.

[3] T. Ahonen, A. Hadid, M. Pietikainen, “Face description with local binary patterns,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[4] H. Cevikalp, B. Triggs, “Polyhedral conic classifiers for visual object detection and classification,” *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[5] A. Krizhevsky, I. Sutskever, G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Advances in Neural Information Processing Systems (NIPS)*, 25, 2012.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” *IEEE Conference on Computer* Vision and Pattern Recognition, pp. 779-788, 2016.

[7] Liu, Wei, et al. "Ssd: Single shot multibox detector," *European Conference on Computer Vision*, pp. 21-37, Springer, Cham, 2016.

[8] S. Ren, K. He, RGirshick, J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *Advances in Neural* Information Processing Systems, pp. 91-99, 2015.

[9] J. Redmon, A. Farhadi, “YOLO9000: better, faster, stronger,” *arXiv preprint* arXiv:1612.08242, 25, Dec., 2016.

[10] R. Gasimov, G. Ozturk, "Separation via polyhedral conic functions," *Optimization Methods and Software*, 2006.

[11] P. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, “Object detection with discriminatively trained part based models,” *IEEE Transactions on PAMI*, vol. 32(9), pp. 1627-1645, 2010.

[12] H. Cevikalp, B. Triggs, “Visual object detection using cascades of binary and one-class classifiers,” *Int*. *Journal of Computer Vision*, 123 (3), pp. 334-349, 2017.