1D direction estimation with a YOLO network

Iu. Chyrka[1], V. Kharchenko[2]

Research department, National aviation university, Kyiv, Ukraine

[1]iurii.chyrka@gmail.com, [2] knarch@nau.edu.ua

***Abstract* — The modified You only look once (YOLO) network architecture that allows one-dimensional direction estimation along with classic object detection in real time, is considered in the task of street traffic surveillance from unmanned aerial vehicles. The key feature is a modified output fully connected layer with additional orientational parameters. It has been shown that this network can estimate the direction of vehicles on a custom testing dataset with photos.**

***Keywords* — artificial neural networks, object detection, image processing, real-time systems, remote sensing, unmanned autonomous vehicles.**

I. Introduction

In recent years, the development of unmanned aerial vehicles (UAVs) is rapidly gaining attention, especially in the context of all auxiliary systems such as automatic control and autonomous functioning, video surveillance and corresponding recognition and tracking of objects. A lot of recognition techniques, both general purpose and specifically designed for UAV applications, have been proposed to fully utilize wide opportunities proposed by usage of UAVs in many fields of human activity. One of the most important tasks is gathering as much information about the target (i.e. desired object) as possible based on the recorded video footage. The major part of the required information includes data about object class, its position, and orientation in the real world coordinate frame. This leads to the development of sophisticated target recognition algorithms in order to improve the performance of detection systems.

Conventional detection algorithms perform only detection of objects inside bounding boxes and their further classification or can additionally produce a pixel-wise segmentation of the whole image and overlaying of the mask over the detected object. Whereas usually additional information about an object's orientation is necessary in many applications. In some cases, it can be useful for additional improvement of the object recognition and classification.

In other applications such as real-time tracking, knowing the pose of a moving object (an airplane, a car, a boat etc.) also helps to estimate its three dimensional (3D) direction of motion. In robotics, estimating the orientation of surrounding objects is helpful for driving of the autonomous agent.

The problem of 3D pose estimation is difficult because the space of orientation parameters is non-linear and non-Euclidean. This obstacle leads to the appearance of many methods for their representation [1]. In the case of one dimensional (1D) orientation estimation of objects on the ground that is a typical scenario for video-footage taken from UAV, the mentioned problem remains despite significant reduction of the parameters number. At the end of the next section, we outline a few classic representations for angular parameters and explain our choice for the presented work. We do not address the problem of symmetries in the object, which cause the appearance of multiple identical orientations. In other words, in these cases, there are multiple "correct" orientations that makes it impossible to say, where is the front or the back.

One of the most crucial factors for current autonomous robots and UAV is performing of operations in real time. That requirement puts additional constraints for detection and direction estimation methods. Ideally, those two tasks should be performed simultaneously.

In this paper, we propose a new approach to 1D objects orientation estimation in the framework of the state-of-the-art YOLO ("You Only Look Once") real-time objects detection and classification neural network. Beside the classic bounding box with objects, it provides the angular estimation for each detected box. The proposed network has been tested on the task of the vehicles detection on the aerial images that is quite a typical scenario where the object direction is important.

II. Analysis of Publications

There are a few types of object detection methods that include classic detectors, two-stage detectors, and one-stage detectors. Classic object detectors operate in the sliding window, in which a classifier is applied every time over a predefined image grid. More recent approaches operate in two stages and use region proposal methods to first generate a set of potential areas that should contain all objects while filtering out the most of locations without objects and then run a classifier on these proposals for final accurate detection of desired objects. Yet another modern approach to object detection assumes a single run over the image that is closer to the natural way of how humans detect objects.

As far as we are interested in real-time processing we will focus on single-stage detectors. The three main methods can be mentioned here: Single Shot Detector (SSD) [2], YOLO [3], [4] and RetinaNet [5]. Their effectiveness roughly varies in the same order: SSD has about 10-20% less average precision, YOLO version 3 (v3) has almost the same precision as two-stage detectors, and RetinaNet so far is the best state of the art object detector. However, YOLO v3 is the fastest among them and still has acceptable precision.

Quite a comprehensive review of known orientation methods can be found in [6]. We just briefly notice key points.

One of the most popular approaches treats continuous orientation estimation as a multi-class classification problem by binning the continuous orientations. The work [7] considers up to 360 classes for the viewpoint classification problem.

In [8], a three-step approach is proposed. Firstly, it supposes prediction of a bounding box containing the object, then orientation is estimated based on image features inside the predicted bounding box, and finally, a classifier checks the existence of the object. As another example, the orientation estimation method for vehicles in [9] is using histogram of oriented gradients (HOG) features with support vector machines (SVM) or the AdaBoost classifier.

In [10] the problem is addressed using convolutional neural network-based features. However, generally, it can be fully implemented on the convolutional neural networks (CNN) basis as it is done in [11] and [12] where Region-CNN (R-CNN) and faster R-CNN are used to detect objects with axis-aligned bounding boxes and then another CNN-based classification model was used to classify the orientation of each bounding box. The Auto-masking Neural Network (ANN) for joint object detection and viewpoint estimation is introduced in [13]. The key component of ANN is a mask layer that synthesizes a mask for only the important parts of the image to be used for the final prediction.

Some approaches [14] address the task as a continuous prediction in order to avoid errors caused by discretization. Several works consider learning a suitable representation for the orientation estimation task. For example, in [15], an embedded representation that reflects the local features and their spatial arrangement as well as enforces supervised manifold constraints on the data is proposed. Then a regression model to estimate the orientation is learned using the proposed representation.

The essence of abovementioned methods lies in the fact these methods divide detection and orientation estimation into two steps, which make the process more complicated and time-consuming. Meanwhile, axis-aligned bounding boxes are used during the detection process. They contain not only objects but also backgrounds, increasing the difficulty of accurate detection. In [16] that problem is finally considered in the single framework of the known object detection method SSD and has better speed in comparison to stacked methods mentioned above. In the present work, we stick to the same philosophy but with YOLO network.

For the continuous orientation estimation task, the method has to predict the angular value, which belongs to a non-Euclidean space, prohibiting the direct use of a typical L2 loss function. There are a few approaches to handle this problem.

The first approach represents an angle as a point with two coordinates on a unit circle. One can interpret them as real and imaginary parts of a complex number or as quadrature components of a vector. Then, the network is trained using the common L2 loss for all predicted parameters. The predicted 2D point is not necessarily lying on a unit circle but for the final output and/or visualization it is converted back to the angular value by the arctangent function. The second approach also uses a-point-on-a-unit-circle representation. It minimizes a loss defined directly on the angular difference, not as an L2 function in 2D space. In the third approach, the discretized orientations are uniformly distributed in the output circular space where the mean-shift algorithm is carried out to find the most plausible orientation while taking into account the softmax probability for each discrete orientation [6].

Another approach lies in a similar plane and is based on the statistical representation of the angle as well. The angle can be described by a von Mises distribution, which is the circular equivalent of the Gaussian distribution. Similar in shape to the von Mises distribution is the wrapped Gaussian which wraps the ordinary normal distribution around a circle. In [17] they represent the angular hue component of the hue saturation value (HSV) model that is used for training of the color-based pedestrian detector.

III. Oriented YOLO Neural Network

As a basis for our experiments we have chosen an open-source platform: the Darknet framework written in C and Compute Unified Device Architecture (CUDA) and modified accordingly to proposed changes [18]. It has been enhanced with the functionality for estimation of object direction. It should be noted that we were focused mostly on the YOLO network of the 3rd version [4] of the architecture, however, it can be implemented for older versions of the network as well.

The YOLO network uses a single-stage approach that supposes predictions on the basis of the actual image, without regions prediction stage. Before the processing, the input image is automatically resized to a tensor of size $n \times m \times c$, where n and m represent width and height in pixels and c is the number of color channels (c=3 in most cases).

Processing of the image by the YOLO v.3 network can be described in the next way. The input image is divided onto the grid of cells of some size. For 416 × 416 input size, that grid consists of 13 × 13 cells. Inside each cell the network performs detection and prediction of a certain amount of bounding boxes. Those boxes can cover a part of this cell, if the detected object spans over a few neighbor cells, or completely lie inside that cell.

Original network supposes that each predicted bounding box contains the following geometrical information: b_x and b_y coordinates of the bounding box, width b_w, and height b_h. Additionally, we introduce two parameters q_x and q_y that we consider as quadrature components of a vector that represents the direction from the center of the bounding box. Those two parameters are not explicitly related to the position or size of the bounding box.

Bounding box parameters are calculated through predictions $t_x, t_y, t_w, t_h, t_{qx}, t_{qy}$ from cell coordinates c_x, c_y and bounding box prior width p_w and height p_h as

$$\begin{aligned} b_x &= \sigma(t_x) + c_x, \quad b_y = \sigma(t_y) + c_y, \\ b_w &= p_w e^{t_w}, \quad b_h = p_h e^{t_h}, \\ q_x &= t_{qx}, \quad q_y = t_{qy}. \end{aligned} \tag{1}$$

Besides coordinates and the object direction, the network predicts the "objectness". It is the probability that the bounding box contains the object and C conditional probabilities of belonging to each of predefined C classes. Each cell is supposed to predict up to 3 bounding boxes. Therefore, we can write down the output tensor size as 13 × 13 × [3 * (4 + 2 + 1 + C)].

The losses are described by the multicomponent L2-norm based function that can be written as

$$\begin{aligned}
&\lambda_{coord}\sum_{i=0}^{169}\sum_{j=0}^{3}\mathbb{I}_{ij}^{obj}\left[(x_{Ti}-\sigma(t_{xi}))^2+(y_{Ti}-\sigma(t_{yi}))^2\right]+\\
&+\lambda_{coord}\sum_{i=0}^{169}\sum_{j=0}^{3}\mathbb{I}_{ij}^{obj}\left[(\ln(w_{Ti}/p_w)-t_{wi})^2+(\ln(h_{Ti}/p_h)-t_{hi})^2\right]+\\
&+\sum_{i=0}^{169}\sum_{j=0}^{3}\mathbb{I}_{ij}^{obj}\left[(q_{xTi}-t_{qxi})^2+(q_{yTi}-t_{qi})^2\right]+\\
&+\sum_{i=0}^{169}\sum_{j=0}^{3}\mathbb{I}_{ij}^{obj}\left[(O_i-\hat{O}_i)^2\right]+\sum_{i=0}^{169}\mathbb{I}_{i}^{obj}\sum_{c\in classes}\left[(p_i(c)-\hat{p}_i(c))^2\right]
\end{aligned} \quad (2)$$

where $\mathbb{I}_i^{obj}$ indicates appearance of the object in the cell i and $\mathbb{I}_{ij}^{obj}$ indicates that jth predicted bounding box in the cell i is responsible for the current prediction, $\lambda_{coord}=2-b_{wN}b_{hN}$ is calculated on the basis of relative width and height b_{wN},b_{hN} of the bounding box, O denotes objectness of the current predicted box and $p(c)$ denotes probability of belonging of the detected object to the cth class, $x_T, y_T, w_T, h_T, q_{xT}, q_{yT}$ denote ground truth values of aforementioned parameters.

IV. CNN Training

We have been trained and tested the modified network in the scenario of street traffic surveillance from drones, where we measured performance of original YOLO v.3. For this purpose we used the custom dataset from the internet [19] that consists of street photos taken from above with objects of 4 main classes of interest: a car, a minibus, a truck, and a bus. There were also trams marked as buses. Images in the dataset include a variety of vehicles with many color paintings and orientations. We have cleaned that dataset from images with too small objects on them and supply it with an additional angular information, required for our network, i.e. ground truth directions of objects in each marked bounding box. Therefore, our modified network has been trained for detection of 4 object classes (C = 4), which gives us a size of the output tensor as 13 × 13 × 33.

The training set consists of 90 images containing 1480 objects instances in total. For performance evaluation on the separated testing dataset with 12 images containing 245 objects has been used.

In order to achieve a better generalization by the model, the data augmentation has been applied during network training. There were applied a few transformations: random scaling up to 60% change of the original image size, changes of hue, saturation, and exposure of the image were randomized in the range 0.1, 1.5, 1.5 respectively in the hue saturation value (HSV) model. A few hyper parameters such as batch size, learning rate, decay, iteration number, and detection thresholds have been tuned as well to get the best performance.

Bounding boxes already have been specified in the original dataset in the darknet-compliant format. We have modified YOLO mark tool [20] and have used it to specify the orientation of vehicles on all images. Training has been carried out with the next parameters, batch size 64, momentum = 0.5, decay = 0.0001, base learning rate = 0.0001, and maximum iteration number = 30000.

V. Experimental results

For testing of the trained neural network we have used different sizes of an input image from 320 × 320 to 480 × 480 with the step 32 pixels in the same way used during training for data augmentation. A small restriction for size to be divided by 32 is dictated by the feature extraction network architecture. Table 1 shows results for a few input sizes and for the confidence threshold 0.25.

Our preliminary training runs have shown that the model reaches its best performance on the testing set at about 30000 iterations and therefore we specified it as the maximum iterations number in order to avoid unnecessary iterations and additional overfitting of the model on the training set.

Table 1. Modified YOLOv3 performance on a testing set.

Metrics	**Input Size**		
	320 × 320	416 × 416	480 × 480
IoU	86.22%	89.33%	89.97
mAP	99%	99%	99%
F1 score	0.99	0.99	0.87
Ang. RMSE	0.1174	0.1079	0.1170

The table contains the next information: a set of sizes of the input tensor; mAP is a mean average precision that means an averaged value on precision/recall curve calculated over 11 points [0, 0.1, …, 1]; IoU is an intersection over union that is the ratio of an intersection area of a ground true bounding box and a predicted bounding box to area of union of these boxes, Ang. RMSE is the angular root mean square error, F1 score is the harmonic mean of precision and recall.

One can see that the network has a quite high precision level. It has a slight dependence on the input image size in terms of localization and correct classification.

It should be noted that unlike the original network [18] we have used a modified, bidirectional metric for calculation of the angular error. It means that instead of a single true value of a direction angle ϕ rad. we accepted both values ϕ rad. and $\phi+\pi$ rad. and calculated the error to the closest of them. The logic behind this decision is the next: most of vehicles are quite hardly identified from the top view in terms correct front-rear orientation. For example the bus can look as a pure rectangle without any signs for a rear or a front.

Fig. 1 shows several examples with detected vehicles surrounded by bounding boxes and supplied with estimated orientation angles represented by lines drawn from the center of each bounding box.

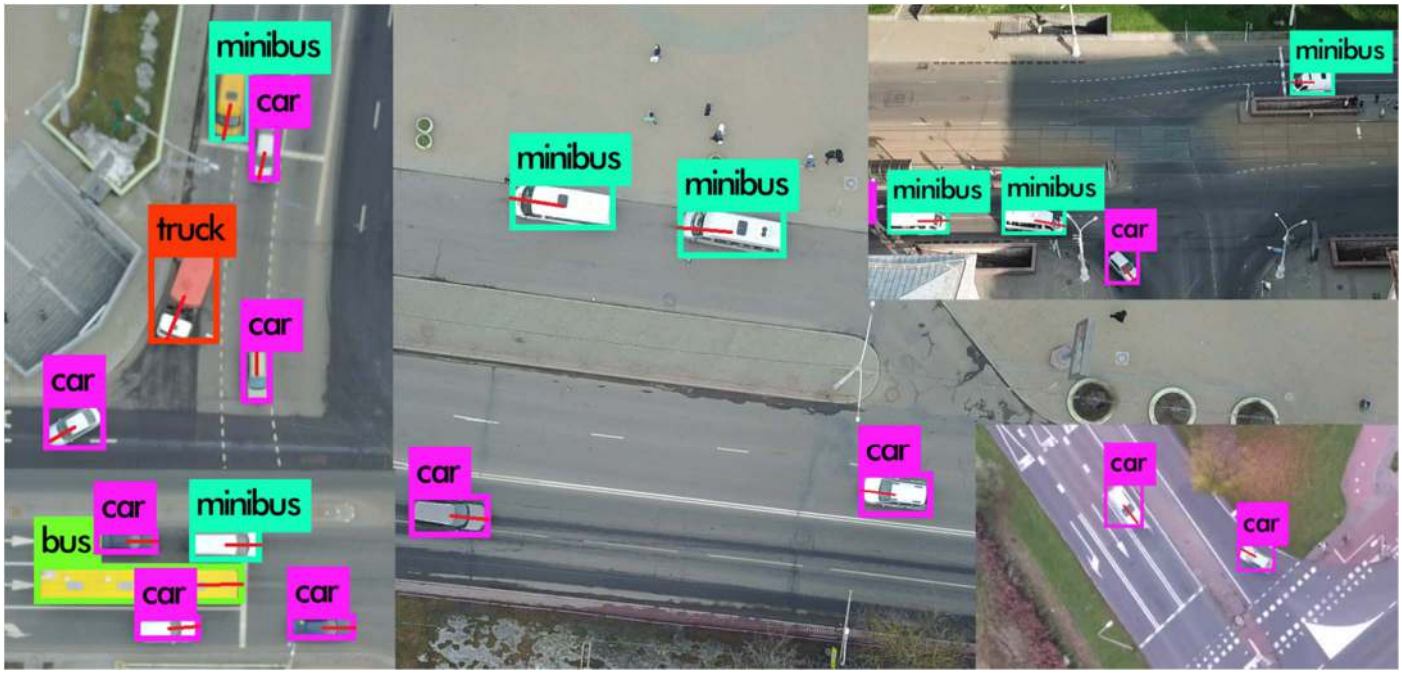

Fig. 1. A sample photo with detected objects and estimated orientations.

VI. Conclusion

Presented modified YOLOv3 network has shown the high level of orientation estimation and the detection accuracy in the particular application for vehicles detection, even with relatively small training dataset. It has shown mean average precision almost 90% and angular root mean square error 0.11 rad on the considered custom dataset.

References

[1] N. Fisher, *Statistical Analysis of Circular Data*. Cambridge University Press, 1993.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, "SSD: Single shot multibox detector. European Conference on Computer vision and Pattern Recognition," in *Proc. ECCV'16*, 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR'16*, 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91

[4] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint, arXiv:1804.02767, 2018.

[5] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Doll´ar., "Focal Loss for Dense Object Detection," in *Proc. ICCV'17*, 2017, pp. 2999–3007. doi: 10.1109/ICCV.2017.324

[6] K. Hara, R. Vemulapalli, and R. Chellappa, "Designing Deep Convolutional Neural Networks for Continuous Object Orientation Estimation," arXiv preprint arXiv:1702.01499, 2017.

[7] H. Su , C. R. Qi, Y. Li, and L. J. Guibas, "Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views," in *Proc. ICCV'15*, 2015, pp. 2686–2694, 2015. doi: 10.1109/ICCV.2015.308

[8] M. Ozuysal, V. Lepetit, and P. Fua, "Pose Estimation for Category Specific Multiview Object Localization," in *Proc. CVPR'09*, 2009. doi: 10.1109/CVPR.2009.5206633

[9] K. Liu and G. Mattyus, "Fast multiclass vehicle detection on aerial images," *IEEE Geoscience and Remote Sensing Lett.*, Vol. 12, pp.1938–1942, 2015. doi: 10.1109/LGRS.2015.2439517

[10] A. Ghodrati, M. Pedersoli, and T. Tuytelaars, "Is 2D In-formation Enough For Viewpoint Estimation? ," M. Valstar, A. French, T. Pridmore, Ed. *Proceedings of the British Machine Vision Conference*, BMVA Press, 2014. doi: 10.5244/C.28.19

[11] Y. Taigman, M. A. Ranzato, L. Wolf, and M. Yang, "DeepFace: Closing the Gap to Human-Level Perfor-mance in Face Verification," in *Proc. CVPR'14*, 2014. doi: 10.1109/CVPR.2014.220

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR'14*, 2014. doi: 10.1109/CVPR.2014.81

[13] L. Yang, J. Liu, and X. Tang, "Object Detection and Viewpoint Estimation with Auto-masking Neural Network," in Proc. ECCV'14, 2014, pp. 441–455. doi: 10.1007/978-3-319-10578-9_29

[14] K. Hara and R. Chellappa, "Growing Regression Forests by Classification: Applications to Object Pose Estimation," in *Proc. ECCV'14*, 2014, pp. 441–455. doi: 10.1007/978-3-319-10605-2_36

[15] M. Torki and A. Elgammal, "Regression from Local Features for Viewpoint and Pose Estimation," in *Proc. ICCV'11*, 2011. doi: 10.1109/ICCV.2011.6126549

[16] T. Tang, S. Zhou, Z. Deng, L. Lei, and H. Zou, "Arbitrary-Oriented Vehicle Detection in Aerial Imagery with Single Convolutional Neural Networks," *Remote Sensing*, vol. 9 (11), p. 1170, 2017. doi: 10.3390/rs9111170

[17] F. Seitner and B. Lovell, "Pedestrian tracking based on colour and spatial information," in *Proc. DICTA'05*, 2005, pp. 36–43. doi: 10.1109/DICTA.2005.64

[18] Darknet: Open source neural networks in C. [Online]. Available: https://github.com/Yurasyk/darknet (Accessed 05.09.18)

[19] Aerial cars dataset. [Online]. Available: https://github.com/jekhor/aerial-cars-dataset (Accessed 15.01.19)

[20] YoloMark (with angular modification.) [Online]. Available: https://github.com/Yurasyk/Yolo_mark (Accessed 25.09.18)