

# VEHICLE DETECTION AND COUNTING UNDER MIXED TRAFFIC CONDITIONS IN VIETNAM USING YOLOV4

IAEME Publication


*IAEME Publications*

## Cite this paper

Downloaded from [Academia.edu](#) 

[Get the citation in MLA, APA, or Chicago styles](#)

## Related papers

[Download a PDF Pack](#) of the best related papers 



[AN ANALYSIS OF URBAN T RAFFIC INCIDENT S UNDER MIXED T RAFFIC CONDIT IONS BASED ON ...](#)  
IAEME Publicat ion

[REFINING YOLOV4 FOR VEHICLE DET ECTION](#)

IAEME Publicat ion

[T RAFFIC SIGNAL VIOLAT ION DET ECTION USING ART IFICIAL INT ELLIGENCE AND DEEP LEARNING](#)

IAEME Publicat ion



# VEHICLE DETECTION AND COUNTING UNDER MIXED TRAFFIC CONDITIONS IN VIETNAM USING YOLOV4

**Vuong Xuan Can\***

School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, China  
University of Transport and Communications, Hanoi, Vietnam

**Phan Xuan Vu**

Hanoi University of Science and Technology, Hanoi, Vietnam

**Mou Rui-fang**

School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, China

**Vu Trong Thuat**

University of Transport and Communications, Hanoi, Vietnam

**Vu Van Duy**

University of Transport and Communications, Hanoi, Vietnam

**Nguyen Duy Noi**

University of Transport and Communications, Hanoi, Vietnam

\*Corresponding Author

## ABSTRACT

*Recent video-based vehicle detection and counting algorithm are one of the main components to determine the traffic state. With improvements in computer vision and machine learning approaches for object detection, advanced algorithms based on artificial neural networks, such as YOLO (You Only Look Once) with high precision, are commonly used to replace classical approaches. In this paper, we provide a method using YOLOv4 for the vehicle detection and counting of mixed traffic flow in the context of Vietnam's transport. We have tested the network for five vehicle types, including motorcycles, bicycles, cars, trucks, and buses. The test results show that our algorithm achieves the vehicles' detection accuracy and counting on the test set than others (e.g., Background Subtraction and Haar Cascade), indicating that the proposed method has higher detection performance.*

**Key words:** Vehicle detection, Vehicle counting, YOLOv4, Mixed traffic conditions.

**Cite this Article:** Vuong Xuan Can, Phan Xuan Vu, Mou Rui-fang, Vu Trong Thuat, Vu Van Duy and Nguyen Duy Noi, Vehicle Detection and Counting under Mixed Traffic Conditions in Vietnam using YOLOv4, *International Journal of Advanced Research in Engineering and Technology (IJARET)*, 12(2), 2021, pp. 722-730.  
<http://iaeme.com/Home/issue/IJARET?Volume=12&Issue=2>

---

## 1. INTRODUCTION

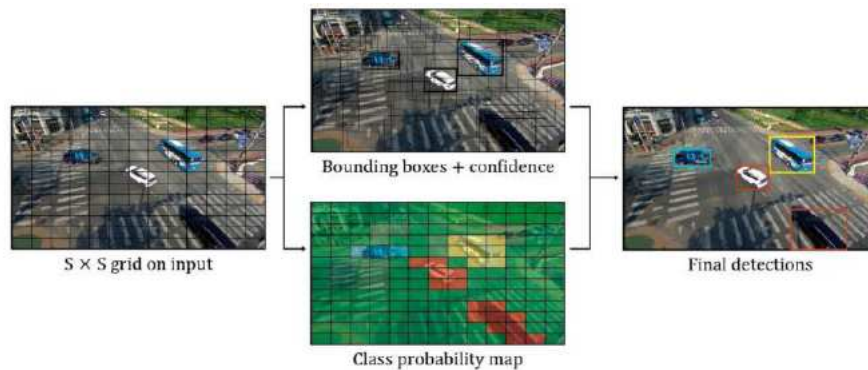
Vehicle detection and counting are important data used in computing traffic capacity, establishing structural design criteria, computing expected roadway user revenue, and so on. Vehicle detection systems using video-based offer several advantages in comparison with earlier methods (e.g., magnetic loop detectors). Except for vehicle counting, those systems can measure a larger set of traffic parameters such as vehicle classifications, vehicle speed, lane-changing, parking areas, etc. Therefore, they are widely used and continuously improved and developed in parallel with the development of information and communication technology. Video-based vehicle detection systems using deep learning such as deep convolutional neural networks (CNNs) are better than others (e.g., Background Subtraction, Haar Cascade [1]), and can divide into two types, including region-based algorithms (Two-Stage Approaches) and regression-based algorithms (One-Stage Approaches) [2][3]. The typical region-based algorithms consist of R-CNN (Region-based CNN) [4], Fast R-CNN [5], Faster R-CNN [6], R-FCN (region-based fully convolutional network) [7]. The representatives of one-stage object detection algorithms include YOLO (You Only Look Once) series algorithms and SSD (Single Shot MultiBox Detector) algorithm [8]. YOLO firstly proposed by Redmon, et al. [9] is a typical regression-based algorithm. After that, many versions were improved and upgraded such as YOLOv2 [10], YOLOv3 [11], YOLOv4 [12]. Previous researches showed that YOLOv3 and YOLOv4 have the same accuracy as the region-based algorithms at a higher speed [13]. YOLO has been successfully used for vehicle detection in homogeneous traffic with 4-wheelers vehicles used predominantly [14][15][16][17]. However, most of those researches do not focus on vehicle detection in the mixed traffic flow like Vietnam where on-road motorcycles are used frequently. For example in Hanoi, the capital of Vietnam, motorcycle traffic accounts for 86% of traveling in the urban area [18]. Because of non-lane-based movements of vehicles on roads, traffic flow is very complex in comparison with homogeneous traffic like developed countries. Hence, accurate vehicle detection and counting for mixed traffic flow like Vietnam remain a major challenge. In this paper, we follow the latest development technology in the field of the deep neural network and present a method for vehicle detection under mixed traffic conditions on an urban road segment in Hanoi (Vietnam) based on YOLOv4 [12].

## 2. MATERIALS AND METHODS

YOLO is a state-of-the-art and real-time object detection system that uses a CNN architecture which is called Darknet [19] to detect objects in images. YOLO architecture has 24 convolutional layers, followed by 2 fully connected layers. It alternates 1x1 convolutional layers to reduce the features space from the preceding layers. Initial convolutional layers extract features from the image and the fully-connected layers predict the output probabilities and coordinates of the object.

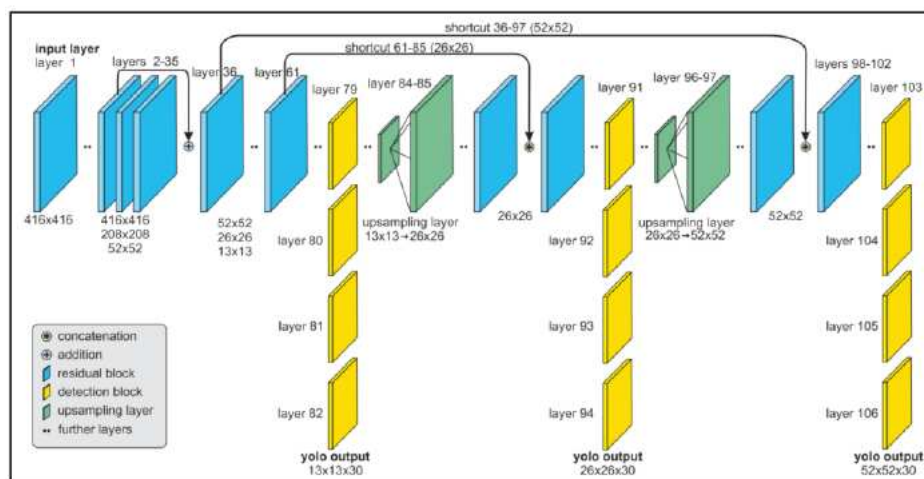
YOLO divides the input image into a grid  $S \times S$ , and each grid cell predicts  $B$  bounding boxes with confidence scores and  $C$  class scores simultaneously. The confidence scores reflect the confidence level of the model as the box contains an object. The confident score of the cell should be zero if no object exists in that cell. During the training, each grid cell assigns a candidate bounding box for an object, and then during the inference, YOLO predicts  $S \times S \times B$

bounding boxes, but most of them are eliminated by keeping the ones with a high confidence score and using Non-Max Suppression (NMS) algorithm [20] to eliminate duplicate detections, and to obtain C class scores (see Figure 1). Thus, YOLO is very fast and easy to generalize to different scenes.



**Figure 1** Illustration of YOLO framework [17]

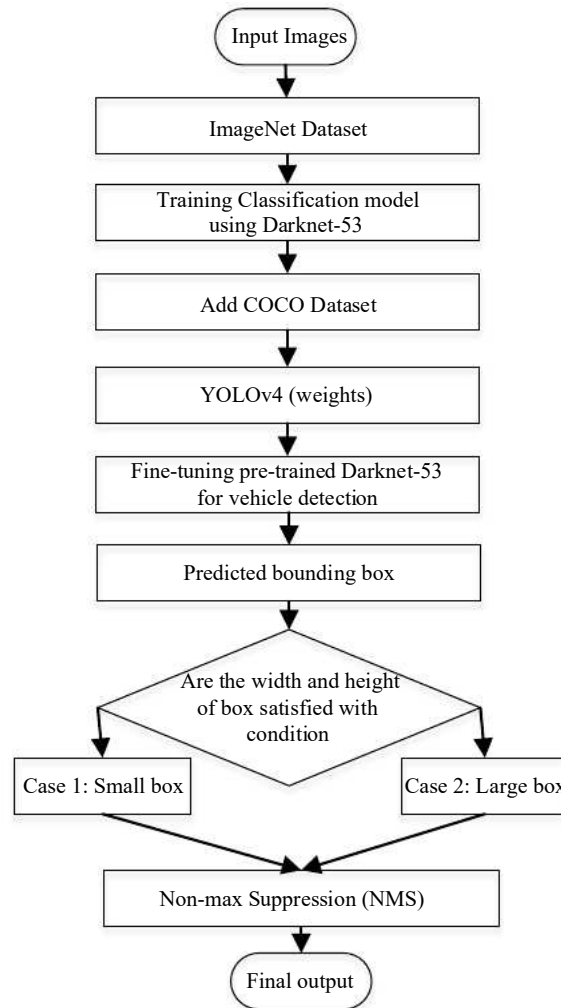
YOLOv4 proposed by Alexey et.al. [12] is the improvement of YOLOv3 architecture (see Figure 2) to optimize parallel computing and improve the speed of object detection. YOLOv4 divides the model into three parts with different functions, including backbones, neck, and heads. The backbone using CSPDarknet-53 mainly extracts the features. The CSPDarknet-53 is a CSPNet (Cross Stage Partial Network)[21] added on the basis of Darknet-53 [11], which draws on the idea of ResNet (Residual Network) [22] to ensure that the network has depth while also alleviate the vanishing gradient problem. CSPNet can enhance the learning ability of CNNs while reducing the amount of calculation and memory cost. The neck based on SPP (Spatial Pyramid Pooling) module [23] and PANet (Path-Aggregation Network) [24] is used to fuse the features extracted from the main part. The SPP module applied can effectively increase the receptive field and help separate contextual features. The PANet plays a role in shortening the path connecting low-level information and high-level information, and converging parameters at different levels. YOLOv4 head inherits the head structure of YOLOv3 (anchor-based) to predict, including predicting the bounding boxes and the object classification. YOLOv4 has not included the sparse prediction that is used in two-stage algorithms such as Faster-R-CNN, etc.



**Figure 2** YOLOv3 architecture [16]

In this work, the detected vehicle will be classified as cars, motorcycles, buses, trucks, and bicycles. Also, it is necessary to have the width and length of each vehicle's bounding boxes in pixels to diagnose that the passing vehicle belongs to which of the mentioned vehicle classes.

The area of each bounding box shows that which class should be allocated for the vehicle. Each vehicle class can be shown by a special rectangle color. Vehicle counting is according to the number of passing vehicles in the detection zone (cross-section) in a specific direction and classified in one of the mentioned classes. We create a log of a text file giving the following details (1) number of counted cars, (2) number of counted motorcycles, (3) number of counted buses, (4) number of counted trucks, (5) number of counted bicycles, (6) total number of counted vehicles, and (7) time and date of recording. The flowchart of vehicle detection and counting is shown in Figure 3.



**Figure 3** Flowchart of vehicle detection and counting system using YOLO [25]

To evaluate the accuracy of vehicle detection and counting, we use quality index, which is calculated as the following.

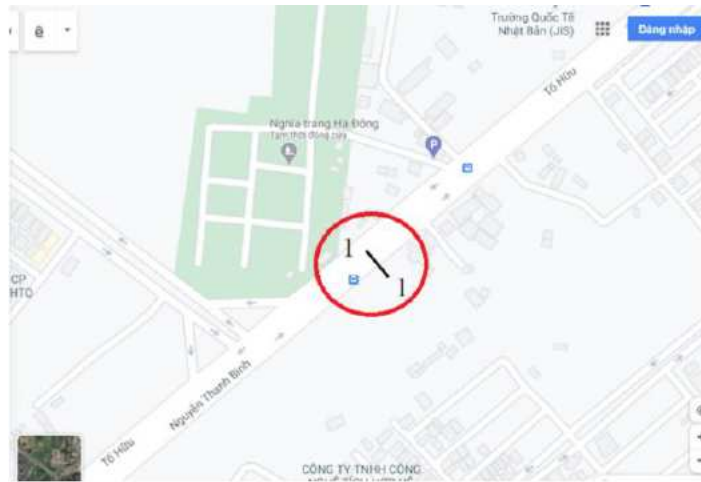
$$Quality = \frac{TP}{TP + FP + FN} \quad (1)$$

Where,  $TP$ ,  $FN$ , and  $FP$  are the numbers of true positives, false negatives, and false positives, respectively.

### 3. EXPERIMENTAL RESULTS

The video source is used in this study from a single camera mounted on a pole, looking down on the traffic scene of an urban road segment in Hanoi City, the capital of Vietnam. Video files are extracted from the camera at one-minute on 27th November 2020, resulting in a total of 10

video files during the study period. The first 5 files are used to identify vehicles during the peak hour, the rest is used to identify vehicles during off-peak hours. The location of the installed camera from Google Map (Cross-section 1-1) and traffic conditions are shown in Figure 4-6.



**Figure 4** The location of the installed camera



**Figure 5** Traffic conditions during peak hour



**Figure 6** Traffic conditions during non-peak hour

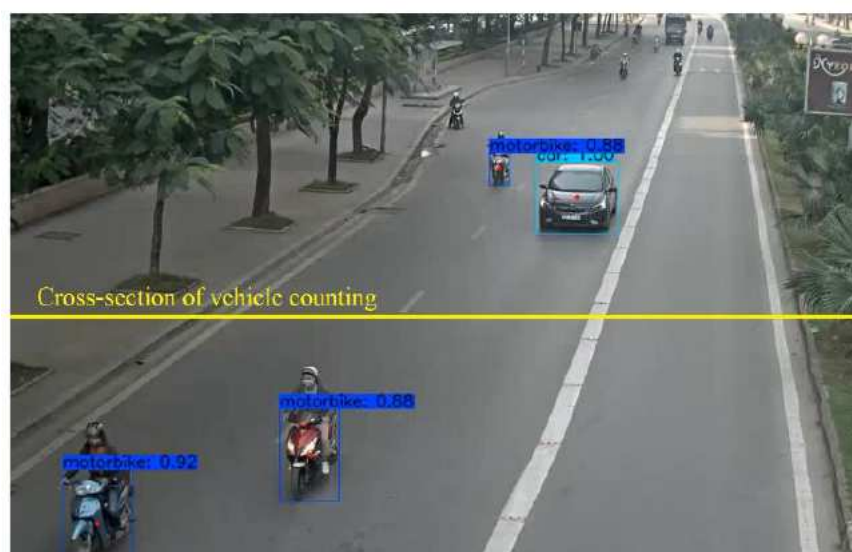
For our implementation of YOLOv4, we use Tensorflow [26][27] which is an open-source software developed by Google and Keras [28], written in Python language programming [29].



Besides, OpenCV library of Python is equipped with functions that allow us to manipulate videos and images [30]. All the OpenCV array structures are converted to and from Numpy library [31], which is a library for numerical operations with a MATLAB-style syntax. This also makes it easier to integrate with other libraries that use Numpy such as Matplotlib library [32]. The pre-trained YOLOv4 using the COCO dataset [33] obtained from [34] is used to detect and classify objects in the study. We test our system on a laptop powered by an Intel Core i7 (1.80 GHZ) CPU and 8GB RAM. Table 1-3 and Figure 6-7 show some results of our algorithm in comparison with other algorithms, such as Haar Cascade and Background subtraction using mixture of Gaussian (MOG) [35].



**Figure 6** Vehicle detection by YOLOv4 during peak hour



**Figure 7** Vehicle detection by YOLOv4 during non-peak hour

As seen from the Fig.6-7, YOLOv4 algorithm is successfully detected multiple vehicle classes in a video by detecting the vehicles in each frame by frame until the whole frame of video closes, the class was denoted in bounding box, the detected classes are cars, and motorcycles.

Accuracy of vehicle detection and counting depends on many factors such as training datasets, image quality, climatic conditions, etc. In the same traffic and weather conditions, we apply a number of different algorithms for vehicle detection and also for comparison. As seen from the Table 1-3 shows that YOLOv4 has the highest stability and accuracy.

**Table 1** Detection results during peak hour

Algorithms\Objects		Cars	Motorcycles	Buses	Trucks	Bicycles	Total
Haar Cascade	TP	40	378	2	1	1	422
	FP	2	0	0	2	3	7
	FN	12	67	1	0	0	80
MOG	TP	50	386	3	0	0	439
	FP	16	26	2	2	0	46
	FN	2	59	0	1	1	63
YOLOv4	TP	52	441	3	1	1	498
	FP	0	0	0	0	0	0
	FN	0	4	0	0	0	4

**Table 2** Detection results during non-peak hour

Algorithms\Objects		Cars	Motorcycles	Buses	Trucks	Bicycles	Total
Haar Cascade	TP	55	176	1	9	1	242
	FP	6	0	0	7	8	21
	FN	25	49	0	0	1	75
MOG	TP	50	220	1	1	0	272
	FP	2	39	11	5	0	57
	FN	30	5	0	8	2	45
YOLOv4	TP	79	225	1	9	2	316
	FP	0	0	0	1	0	1
	FN	1	0	0	0	0	1

**Table 3** Summary of the performance of algorithms

Measures	Quality (%)	
	Peak hour	Non-peak hour
Haar Cascade	82.9	71.6
MOG	80.1	72.3
YOLOv4	99.2	99.4

## 4. CONCLUSION

Traffic flow is basic data for transportation sector and its accuracy and processing within limited time frame is challenging task for traffic and transportation engineer. In this paper a methodology based on YOLOv4, which is implemented by Python language programming has been proposed to count, classify the traffic under mixed traffic conditions, at the same time to save the time of engineers. Hence, this algorithm will be very useful for urban road design, traffic planning, and so on in transport contexts like Vietnam. In the future work, the accuracy of the algorithm could be improved with training on the bigger and more diverse datasets that cover different conditions

## ACKNOWLEDGMENTS

This work was supported by the Education and Training of Vietnam, under grant number CT2019.05.05.



## REFERENCES

- [1] A. Dubey and S. Rane, "Implementation of an intelligent traffic control system and real time traffic statistics broadcasting," in 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), 2017, vol. 2, pp. 33–37.
- [2] M. Algabri, H. Mathkour, M. A. Bencherif, M. Alsulaiman, and M. A. Mekhtiche, "Towards deep object detection techniques for phoneme recognition," *IEEE Access*, vol. 8, pp. 54663–54680, 2020.
- [3] Y. Zhang, C. Song, and D. Zhang, "Deep learning-based object detection improvement for tomato disease," *IEEE Access*, vol. 8, pp. 56607–56614, 2020.
- [4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.
- [5] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
- [6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv Prepr. arXiv1506.01497*, 2015.
- [7] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," *arXiv Prepr. arXiv1605.06409*, 2016.
- [8] W. Liu et al., "Ssd: Single shot multibox detector," in *European conference on computer vision*, 2016, pp. 21–37.
- [9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [10] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.
- [11] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv Prepr. arXiv1804.02767*, 2018.
- [12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv Prepr. arXiv2004.10934*, 2020.
- [13] C. E. Kim, M. M. D. Oghaz, J. Fajtl, V. Argyriou, and P. Remagnino, "A comparison of embedded deep learning methods for person detection," *arXiv Prepr. arXiv1812.03451*, 2018.
- [14] X. Li, Y. Liu, Z. Zhao, Y. Zhang, and L. He, "A deep learning approach of vehicle multitarget detection from traffic video," *J. Adv. Transp.*, vol. 2018, 2018.
- [15] J. Sang et al., "An improved YOLOv2 for vehicle detection," *Sensors*, vol. 18, no. 12, p. 4272, 2018.
- [16] A. Ćorović, V. Ilić, S. Đurić, M. Marijan, and B. Pavković, "The real-time detection of traffic participants using YOLO algorithm," in *2018 26th Telecommunications Forum (TELFOR)*, 2018, pp. 1–4.
- [17] S. Seong, J. Song, D. Yoon, J. Kim, and J. Choi, "Determination of vehicle trajectory through optimization of vehicle bounding boxes using a convolutional neural network," *Sensors*, vol. 19, no. 19, p. 4263, 2019.
- [18] N. H. Van, V. X. Can, and V. T. Thuat, "A Fuzzy Traffic Signal Control Method for a Single Intersection under Mixed Traffic Conditions," *IJARET*, vol. 11, no. 10, pp. 1715–1723, 2020.
- [19] J. Redmon, "Darknet: Open source neural networks in c." 2013.

- [20] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in 18th International Conference on Pattern Recognition (ICPR'06), 2006, vol. 3, pp. 850–855.
- [21] C.-Y. Wang, H.-Y. Mark Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of cnn," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 390–391.
- [22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 9, pp. 1904–1916, 2015.
- [24] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8759–8768.
- [25] C. Kumar and R. Punitha, "YOLOv3 and YOLOv4: Multiple Object Detection for Surveillance Applications," in 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020, pp. 1316–1321.
- [26] M. Abadi et al., "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," arXiv Prepr. arXiv1603.04467, 2016.
- [27] "TensorFlow." <https://www.tensorflow.org/> (accessed Nov. 19, 2020).
- [28] F. Chollet and et.al., "Keras: Deep Learning for Python," GitHub, 2015. <https://github.com/keras-team/keras> (accessed Jan. 28, 2021).
- [29] "Python." <https://www.python.org/> (accessed Oct. 28, 2020).
- [30] "OpenCV." <https://opencv.org> (accessed Jan. 27, 2021).
- [31] "Numpy." <https://numpy.org/> (accessed Jan. 31, 2021).
- [32] "Matplotlib." <https://matplotlib.org/> (accessed Jan. 31, 2021).
- [33] T.-Y. Lin et al., "Microsoft coco: Common objects in context," in European conference on computer vision, 2014, pp. 740–755.
- [34] B. Alexey, "AlexeyAB/darknet," GitHub. [https://github.com/AlexeyAB/darknet/releases/download/darknet\\_yolo\\_v3\\_optimal/yolov4.weights](https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights) (accessed Jan. 31, 2021).
- [35] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., 2004, vol. 2, pp. 28–31.