Bulletin of Electrical Engineering and Informatics
Vol. 12, No. 4, August 2023, pp. 2236~2244
ISSN: 2302-9285, DOI: 10.11591/eei.v12i4.4990

License plate recognition in slow motion vehicles

Qudes Mb Aljelawy, Tariq M. Salman
Department of Electrical Engineering, Mustansiriyah University, Baghdad, Iraq

Article Info

Article history:

Received Oct 11, 2022
Revised Dec 21, 2022
Accepted Jan 11, 2023

Keywords:

License plate
Optical character recognition
Raspberry Pi 4
You only look once

ABSTRACT

The recognition of license plate numbers represents one of the most efficient techniques to identify any individual vehicle. The principle of the system is that the detection of the license plate will be done with two techniques first you only look once (YOLO) and cascade classifier. Then after achive correct detection, the system will send the result (the image of the license plate) to Easy optical character recognition (OCR) library to read it and transform the image into text. In this paper, an analytical study of the surveillance system which affects by parallax due to camera movement has been done, by merging the OCR technique with the attached camera using python aided Raspberry Pi. The hardware system has been designed and implemented.

Corresponding Author:

Qudes Mb Aljelawy
Department of Electrical Engineering, Mustansiriyah University
Baghdad, Iraq
Email: eema1019@uomustansiriyah.edu.iq

1. INTRODUCTION

There have recently been big break throughs in computer vision application. Real-time computer vision applications can now be used with a variety of very complicated computational hardware, cutting-edge machine learning, and image processing algorithms [1]. Every car has a license plate that is used to identify it. The purpose of a car license plate detection system is to accurately and quickly identify the existence of a license plate on a vehicle [2], [3]. Typically, the process of developing license detection and recognition involves many steps. This systems primary target is the license plate recognition that provided from image or video stream, extract its characters, and understand it. We can store these characters in text file or database. These uses include parking, tracking a certain vehicles number, counting the number of vehicles that enter a particular area [4]. The process of plate recognition is done using Open CV and easy optical character recognition (OCR) libraries in Python. The algorithms used for object detection in our study that you only look once (YOLO) v4 and cascade classifier.

Generally, the process of developing license detection and recognition involves several processes. First, some preprocessing is done to improve the suitability of machine learning algorithms for highly accurate detection and recognition. Images are resized during the pre-processing stage and noise is removed using denoising filters. Additionally, the color photographs that were initially input are changed to grayscale versions because color information is not very helpful for identifying or detecting license plates [5]–[7]. Secondly, to discover and recognize the necessary objects and boundaries in the image, a method known as image segmentation divides an image to multiple segments, or groupings of pixels, based on homogeneity criteria like color, intensity, or texture. In relation to license plates [8], [9].

YOLO is an object detection model where the algorithm detects the different objects and recognizes them in images. At real-time, object detection in YOLO is done as a regression problem and provides the

class confidence of the detected object [10]. Cascade classifiers which is a concatenation of multiple weak object classifiers that are positioned consecutively, with the output of one classifier being an input information to the next classifier, that consist of HaarCascade which classifier are an effective way [11]. The algorithm used for recognize character that OCR by EasyOCR library in python. In the system of EasyOCR library there are near 40 language (characters: letters and numbers) so the EasyOCR library will test matching between the input image and the character in memory in the library [12]. In this paper will explain the Raspberry Pi 4 as a hardware, YOLOv4, cascade classifier, OCR, detection in real time video stream mode, and the results when difrrent status of distances and angels.

2. PROPOSED SYSTEM

Figure 1 shows the flowchart describing the work of the license plate recognition system. The video frames in which come from the camera port will get in the Raspberry Pi and pre-processed frame by frame, for any individual frame, if the plate number needs some image processing such as first stage noise removal or classical segmentation, if no, then directly detects the license plate numbers sequentially by using EasyOCR algorithm, image correction using morphology and character segmentation. The system will check the end of video frames to finish the process or to continue.

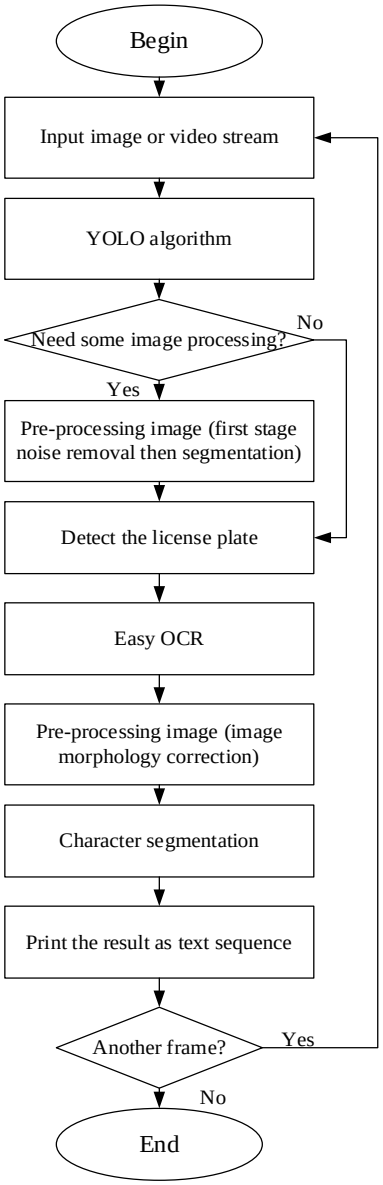

Figure 1. Flowchart video frame license number detection

3. METHOD

For license plate recognition, various techniques are proposed. In this part, a compact summary of the components is provided and explain each components what responsible for (hardware description, YOLO algorithms, cascade classifier and OCR) as follows. Each one of these components of the license plate recognition system will be clear to understand after this part.

3.1. Hardware description

The extraction of a vehicle's license plate number from an initial image will be carried out in this phase utilizing a Raspberry Pi 4 and a camera with a spatial resolution of 2,592×1,944 pixels. Figure 2 shows the hardware utilized in this study [13], [14]. Any ordinary image contains a lot of unnecessary details, such as noise and background texture. The following steps are an algorithm that has been developed to determine the area of interest inside the license plate in order to address the difficulty of reading a vehicles number plate.

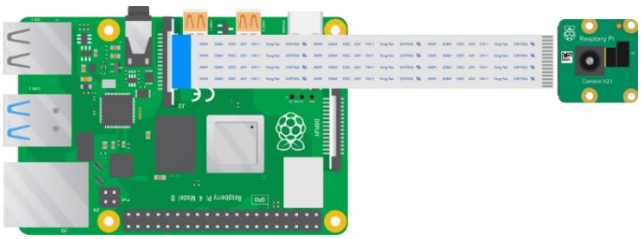

Figure 2. RaspberryPi 4 with a camera

3.2. YOLO algorithm

A deep learning model called YOLOv4 has been used to identify objects in input video frame or image. The YOLO model is employed to take images that include the license plate area, which serves as the region of interest [15], [16]. One neural network can forecast bounding boxes for YOLO, which only needs to process an image once to accomplish detection, and class probabilities can be improved end-to-end in direct relation to detection performance.

The YOLOv4 algorithm, which can detect any object required, is based on the detection of objects in an image or video stream. Only the license plate had to be detected for this application; other things, such the car's body, were unimportant. Consequently, once YOLO detects the license plate, it can be used to recognize the characters with EasyOCR using OCR algorithm. The YOLO algorithm receives the image, processes it to extract features, gathers all the objects, and determines which objects the program will use and which ones it will ignore.

3.3. Cascade classifier

An ensemble of various weak object classifiers arranged in a cascade is referred to as a cascade classifier. The output of one classifier is passed along to the next classifier as additional data. The ensemble's successive operation enhances the classifiers accuracy. In reality, it comprises of the HaarCascade, a reliable method for facial recognition. Viola and Jones [17] proposed this approach. The workflow of HaarCascade classifier is shown in Figure 3.

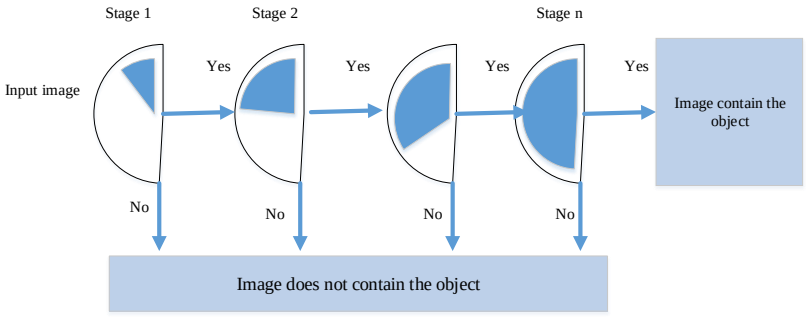

Figure 3. HaarCascade classifier

A big amount of both positive and negative images is used to train the classifier in this machine learning-based method. The negative images are the ones that contain everything else, which we typically aren't interested in, while the positive images are the ones that contain the images that we want our classifier to recognize. This is another disadvantage of this approach because it necessitates samples of the object that we are not interested in for reading license plates [18].

3.4. Optical character recognition

Using OCR, we can take textual data from images. It is quite helpful since it can assist in both the processing of textual data that can be utilized for a variety of applications as well as the acquisition of semantics from scanned images. Different OCR engines exist. Some are proprietary, while others are open source. Two well-known OCR engines that are free source are TesseractOCR and EasyOCR. The EasyOCR approach has been applied in this work to implement the plate number recognition. Although this application uses OCR for license plate recognition, its primary usage is to handle document images captured by desktop scanners. To the fact that the resolutions of scanned images give sufficient resolution, which can be used in car plate recognition efficiently [19].

EasyOCR is an OCR library built on Python that extracts text from images. With over 40 languages, it is an OCR that is ready to use. Before extracting the text, the library does certain pre-processing steps (noise removal, gray scaling, and segmentation) within its library. The text is also recognized using the Craft algorithm. In order to correctly discover text areas, the scene text detection system Craft investigates each character and converges among characters. Figure 4 shows the EasyOCR structure.

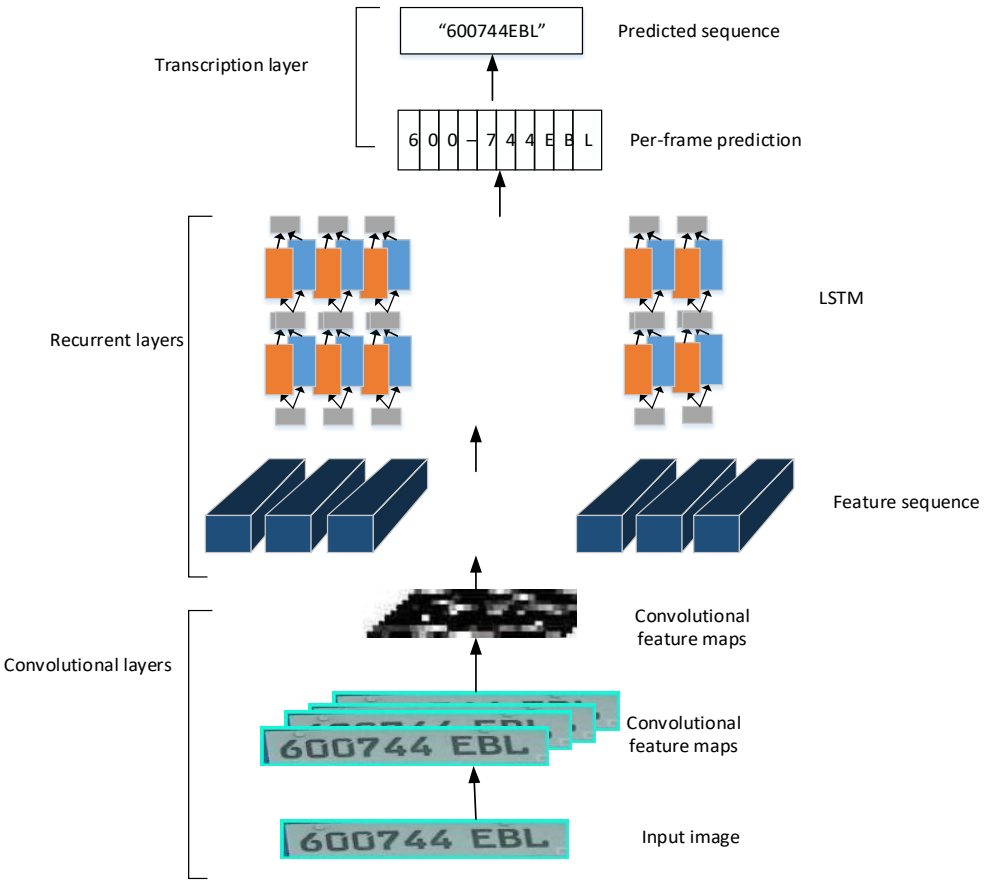

Figure 4. EasyOCR structure

Localizing individual character regions and connecting the newly discovered characters to a text instance are the primary goals of Craft [20]. In the recognition model, convolutional recurrent neural network (CRNN) is utilized. In charge of sequencing labeling are connectionist temporal classification (CTC) [21] and long short-term memory (LSTM) [22]. In this scenario, unsegmented sequence data is labeled with RNN

using the CTC. Since there are more than 40 languages (letters and digits) in the EasyOCR library system, the library will assess how well the input image matches the character stored in the library's memory.

4. REAL TIME DETECTION

Real-time software applications are popular nowadays because they offer faster processing of computer tasks, activities, and operations. A realtime application usually allows the user to perform several tasks and activities at the same time while the application is running. Alternatively, real-time systems in a computer allow multiple programs to execute concurrently, even if the user is only engaged with one application or program. In a computer system, some of these applications are considered as system tasks, developed specifically to do certain tasks, responding to the computer's time clock and producing results even if they are not opened by the user. Real-time systems are those in which the system's precision is determined by the time at which the result is produced including the ultimate result of computation [23].

Single stage methods they are employed for real-time object detection since, in general, they sacrifice some accuracy for significant speed increases. In a single evaluation, a neural network can immediately predict bounding boxes and class probabilities from entire images. The algorithms from the family serve as the best examples of this category [24], [25]. The process of real-time detection shown in Figure 5. The system will take the input first frame as image and detect the area of interest and localize it then show the outout of the first frame and take the second frame doing same steps untill the video ends or close the webcam. Using the system with real time video, as seen in Figure 6. with car in front of camera the license plate detected correctly and when another car came front camera the result of recognition of the second car is added to results as shown in Figure 7 and so on if any numbers of car moving front camera.

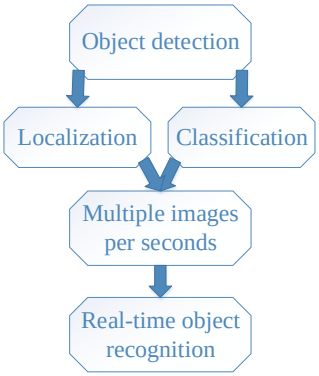

Figure 5. Steps pf real time recognition

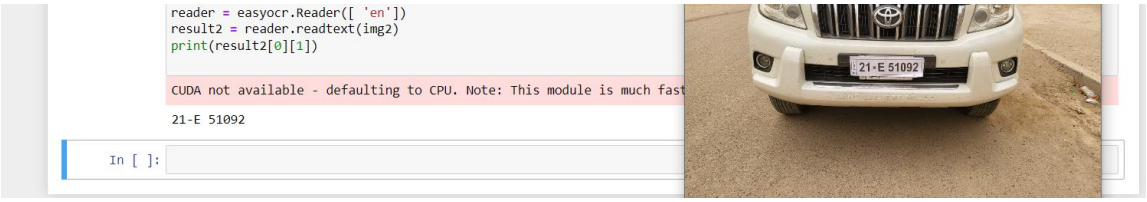

Figure 6. Example of working the system with real time

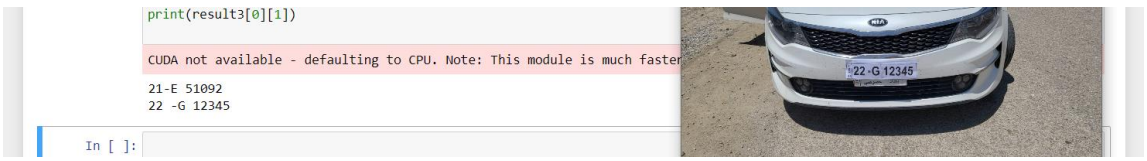

Figure 7. Another example of working the system with real time

5. RESULTS AND DISCUSSION

In this paper will explain the effects of angels and distances on system by try the system which two diffrenet techniques (YOLO and cascade classifier) with diffrenet distances and angels and measure the accuracy in each case and discuss the results. So, the results will change according to the movements of the location of camera. The difference of angels and distances may affect the accuracy of system so have to study this effects as will show in this section.

5.1. Direct location

In the setup shown in Figure 8, the camera will be in front of the car as shown in the figure and will measure the accuracy when the distance between camera and car changes from 1 meter to 10 meters. The distance (a-b) shown in Figure 8 will changes by 1 meter till reaches 10 meters. The input of system will be the image of car shown in Figure 9 and the system will extract the license plate then read it as shown in Figure 9. In Figure 9, are some examples of how the system will read the images.

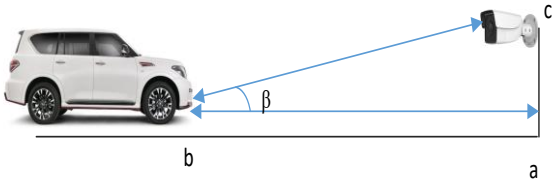

Figure 8. Raspberry Pi camera setup

Figure 9. Some exampels of results

5.1.1. Using YOLO with direct location

When using YOLO with the front license plate of car, image at each meter distance between camera and car. The results of 10 images are clear at Figure 10 with x-axis represent the distance and the y-axis represents accuracy noticed that efficiency is high and near to 100% until 6 meters after that the system becomes can't detect the number correctly. In this paper will change the input images and apply the system to see what will affect the results. as shown in Figure 10, the results will be affected with changes the distances. In near distances (from 1 meter to 6 meters) the results are excellent but at far distances (6 meters to 10 meters) the results will be not good.

5.1.2. Using cascade classifier with direct location

When using cascade classifier that depends on HaarCascade to detect the license plate the results will be as shown in Figure 11. It's clear to say that the result less accuracy compared with YOLO but it still detects correctly even over 6 meters. The advantage of this algorithm that can detect the license plate correctly even at far distances. The accuracy near 75% so there is a clear difference between this case and the previous case.

5.2. Results with 20 angle difference

To analysis, the effect of parallax the camera-positing angle will be changed. Measure the accuracies again in the case of angle 20 between the camera and car, also will change the distances in constant angle. After fixing the angle of the camera, the camera will move away from the car by one meter, until it reaches 10 meters, as the distance between the camera and the car and apply the system with two algorithms and compare the results.

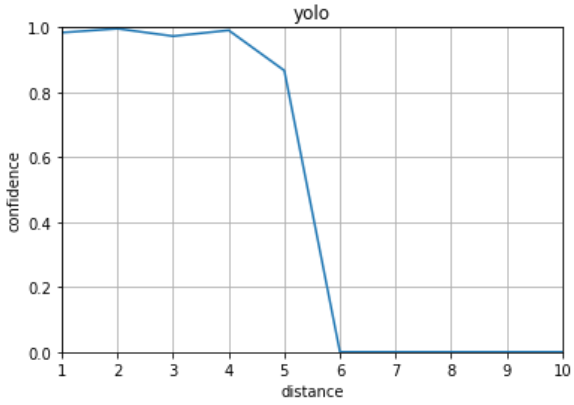

Figure 10. Curve of results with direct location using YOLO

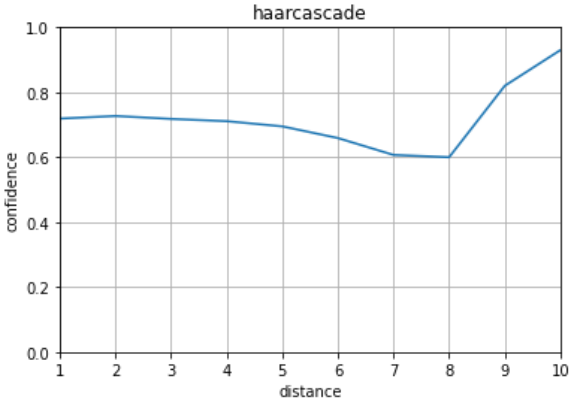

Figure 11. Curve of results with direct location using cascade classifier

5.2.1. YOLO with angle 20

When using YOLO with an angle 20 between the camera and car and different distances from 1 to 10 meters the result be as shown in Figure 12 the results has been too similar to result at direct location, the system can detect the license plate till 6 meters. After fixing the angle of the camera at 20, the camera will move away from the car by one meter, until it reaches 10 meters, as the distance between the camera and the car. In near distances (from 1 meter to 6 meters) the results are excellent but at far distances (6 meters to 10 meters) the results will be not good.

5.2.2. HaarCascade with angle 20

When using cascade classifier with angle 20 different between car and camera. The results be as shown in Figure 13 the system cannot detect correctly but in the far distances the system will be able to detect, at far distances, the camera can see the all license plate clearly the effect of slope will decrease. After fixing the angle of the camera, the camera will move away from the car by one meter, until it reaches 10 meters, as the distance between the camera and the car. The system in this case according to parallax effect will detect the license plate only at far distances.

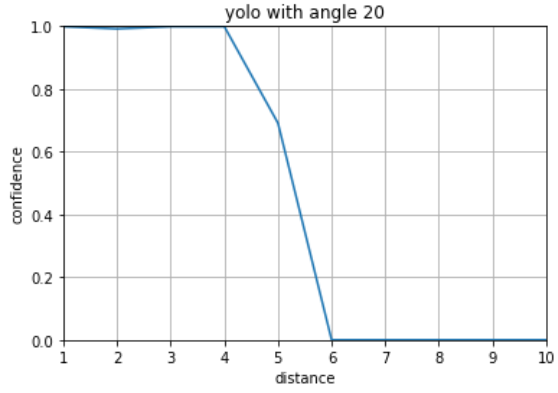

Figure 12. curve of results with angle 20 using YOLO

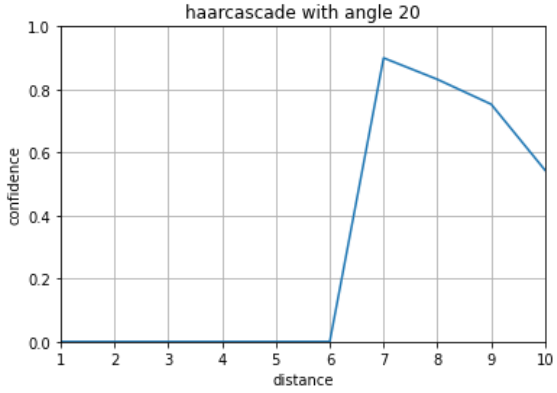

Figure 13. Curve of results with angle 20 using cascade classifier

6. CONCLUSION

In this paper, we created a system for automatically detecting and recognizing license plates. Designing such systems may benefit from incorporating Raspberry Pi, image processing, and machine learning. The license plate was sent to the EasyOCR engine for recognition after being discovered by the YOLO method. Because EasyOCR is less complicated, using it makes detection simpler. In this paper, will study the influence of parallax visual effect in the license plate number recognition system has been analyzed by changes the position of camera movement will cause a dramatic change on the result of detection accuracy. Changes the camera position with distances and angles. When compared to earlier strategies, which were insufficient under the same circumstances, the new method generated encouraging results with 99% accuracy for close ranges.

ACKNOWLEDGEMENTS

The authors wish to thank Al-Mustansiryah University for supporting this research.

REFERENCES

[1] K. Pulli, A. Baksheev, K. Kornyakov, and V. Eruhimov, "Real-time computer vision with OpenCV," *Communications of the ACM*, vol. 55, no. 6, pp. 61–69, 2012, doi: 10.1145/2184319.2184337.
[2] Q. M. Aljelawy and T. M. Salman, "Detecting license plate number using OCR technique and Raspberry Pi 4 with camera," in *2022 2nd International Conference on Computing and Machine Intelligence (ICMI)*, 2022, pp. 1–5, doi: 10.1109/ICMI55296.2022.9873776.
[3] Y. Zou *et al.*, "License plate detection and recognition based on YOLOv3 and ILPRNET," *Signal, Image and Video Processing*, vol. 16, no. 2, pp. 473–480, 2022, doi: 10.1007/s11760-021-01981-8.
[4] Y. Yuan, W. Zou, Y. Zhao, X. Wang, X. Hu, and N. Komodakis, "A robust and efficient approach to license plate detection," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1102–1114, 2017, doi: 10.1109/TIP.2016.2631901.
[5] S. Gollapudi, "OpenCV with Python," in *Learn Computer Vision Using OpenCV*, Berkeley, CA: Apress, 2019, pp. 31–50, doi: 10.1007/978-1-4842-4261-2_2.
[6] S. van der Walt *et al*., "Scikit-image: image processing in Python," *PeerJ*, vol. 2, pp. 1–18, 2014, doi: 10.7717/peerj.453.
[7] R. C. Gonzalez and R. E. Woods, *Digital image processing, 4th edition*. Chennai, India: Pearson Education, 2018.
[8] H. D. Cheng, X. H. Jiang, Y. Sun, and J. Wang, "Color image segmentation: advances and prospects," *Pattern Recognition*, vol. 34, no. 12, pp. 2259–2281, 2001, doi: 10.1016/S0031-3203(00)00149-7.
[9] D. L. Pham, C. Xu, and J. L. Prince, "Current methods in medical image segmentation," *Annual Review of Biomedical Engineering*, vol. 2, no. 1, pp. 315–337, 2000, doi: 10.1146/annurev.bioeng.2.1.315.
[10] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," *Multimedia Tools and Applications*, pp. 1–33, 2022, doi: 10.1007/s11042-022-13644-y.
[11] L. Arreola, G. Gudino, and G. Flores, "Object recognition and tracking using Haar-like features cascade classifiers: application to a quad-rotor UAV," in *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2022, pp. 45–50, doi: 10.1109/CoDIT55151.2022.9803981.
[12] S. S. Omran and J. A. Jarallah, "Iraqi car license plate recognition using OCR," in *2017 Annual Conference on New Trends in Information & Communications Technology Applications (NTICT)*, 2017, pp. 298–303, doi: 10.1109/NTICT.2017.7976127.
[13] H. Jasim Habil, Q. A. Al-Jarwany, M. Nema Hawas, and M. Jabbar Mnati, "Raspberry Pi 4 and Python based on speed and direction of DC motor," in *2022 4th Global Power, Energy and Communication Conference (GPECOM)*, 2022, pp. 541–545, doi: 10.1109/GPECOM55404.2022.9815716.
[14] Z. Isherwood and E. L. Secco, "A Raspberry Pi computer vision system for self-driving cars," in *Intelligent Computing*, Cham: Springer, 2022, pp. 910–924, doi: 10.1007/978-3-031-10464-0_63.
[15] C. Liu, Y. Tao, J. Liang, K. Li, and Y. Chen, "Object detection based on YOLO network," in *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 2018, pp. 799–803, doi: 10.1109/ITOEC.2018.8740604.
[16] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of YOLO algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022, doi: 10.1016/j.procs.2022.01.135.
[17] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 2004, vol. 1, pp. 1–9, doi: 10.1109/CVPR.2001.990517.
[18] R. Yustiawati *et al.*, "Analyzing of different features using HaarCascade classifier," in *2018 International Conference on Electrical Engineering and Computer Science (ICECOS)*, 2018, pp. 129–134, doi: 10.1109/ICECOS.2018.8605266.
[19] K. Hamad and M. Kaya, "A detailed analysis of optical character recognition technology," *International Journal of Applied Mathematics, Electronics and Computers*, vol. 4, no. 1, pp. 244–244, 2016, doi: 10.18100/ijamec.270374.
[20] M. R. Hasan *et al.*, "Smart OCR for recognizing bangla characters with Craft and deep learning models," in *2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2022, pp. 573–577, doi: 10.1109/UEMCON54665.2022.9965668.
[21] A. Graves, "Connectionist temporal classification," in *Supervised Sequence Labelling with Recurrent Neural Networks*, Berlin, Heidelberg: Springer, 2012, pp. 61–93, doi: 10.1007/978-3-642-24797-2_7.
[22] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, pp. 1–28, 2020, doi: 10.1016/j.physd.2019.132306.
[23] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.
[24] S. Lu, B. Wang, H. Wang, L. Chen, M. Linjian, and X. Zhang, "A real-time object detection algorithm for video," *Computers & Electrical Engineering*, vol. 77, pp. 398–408, 2019, doi: 10.1016/j.compeleceng.2019.05.009.
[25] R. Huang, J. Pedoeem, and C. Chen, "YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 2503–2510, doi: 10.1109/BigData.2018.8621865.

BIOGRAPHIES OF AUTHORS

Qudes Mb Aljelawy borned in Baghdad, Iraq in 1996. She holds a B.Sc in Department of Electrical Engineering at 2017 from al Mustansiriyah University. She worked at some non-governmental organization (NGO). Currently, she is a master's student in Electronic and Communication at Mustansiriyah University. She can be contacted at email: eema1019@uomustansiriyah.edu.iq.

Tariq M. Salman was born in Baghdad, Iraq in 1972. He obtained his B.Sc. in Department of Electrical Engineering in 1995, M.Sc. in Communication Engineering in 2003 at University of Technology, Iraq and Ph.D. in Telecommunication and Network devices in 2012 at Belarussian State University of Informatics and Radio Electronics, Belarus. From 2006 to 2012, he worked as a lecturer in the Faculty of Electrical Engineering, at Al-Mustansiriyah University, Iraq. Since the beginning of 2018, he works as an assistant professor in the same faculty. He is a consultant member of the Iraqi Engineering union since 2013. He is interested in the subject of wireless and network devices, video, and image processing systems. He can be contacted at email: tariq.salman@uomustansiriyah.edu.iq.