IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 6, June 2021

DOI: 10.17148/IARJSET.2021.86146

AUTOMATIC GATE CONTROL SYSTEM USING NUMBER PLATE RECOGNITION WITH OCR

Chethan G [1], Rahul R Nadig [2], Supriya V [3], Praveen A [4]

Final year B.E, Department of ECE, K.S Institute of Technology, Bangalore, India[1, 2, 3]

Assistant professor, Department of ECE, K.S Institute of Technology, Bangalore, India[4]

Abstract: In this paper we have conducted a survey of most authentic techniques of license plate detection of a vehicle and an automatic gate control system that will increase convenience and security at entrance of all the important places that require protection and Security. Here the gate will work automatically without the need of human beings and also the system will be able to recognize license plates from vehicles at the entrance gate and decide whether to let vehicles inside or not. The system consists of Raspberry pi along with a video camera that captures video frames which include an image of the vehicle license plate and processes them. The proposed system has been implemented using python and Optical character recognition (OCR).

Keywords: Automatic gate control, Optical character recognition (OCR),Python,Rasberry pi

I. INTRODUCTION

In this paper we have conducted a survey of most authentic techniques of license plate detection of a vehicle and an automatic gate control system that will increase convenience and security at entrance of all the important places that require protection and Security. Here the gate will work automatically without the need of human beings and also the system will be able to recognize license plates from vehicles at the entrance gate and decide whether to let vehicles inside or not. The system consists of Raspberry pi along with a video camera that captures video frames which include an image of the vehicle license plate and processes them. The proposed system has been implemented using python and Optical character recognition (OCR).

II. METHODOLOGY

Initially, the car arrives in front of the barrier. Sensing this, the IR sensor sends a sign to the Raspberry Pi which in turn sends a message to the python program. The program displays the "Welcome" message on the LCD. Then the image, that is, the license plate acquired from the camera will be analyzed in the data analysis part which is mostly done in python. The analyzed image will be compared with the information stored in the database, if this image is matched with the information stored in the database, then the python program sends the message to the Raspberry Pi to open the barrier gate and after some time delay, the barrier gate will be closed again. If the image doesn't match with any of those in the database, the python program will send a message to the Raspberry Pi, which in turn displays "you are not allowed to enter, please go back" on LCD.

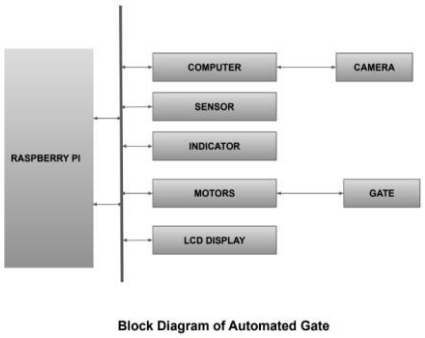

Figure 1: Block Diagram of Automated Gate

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 6, June 2021
DOI: 10.17148/IARJSET.2021.86146

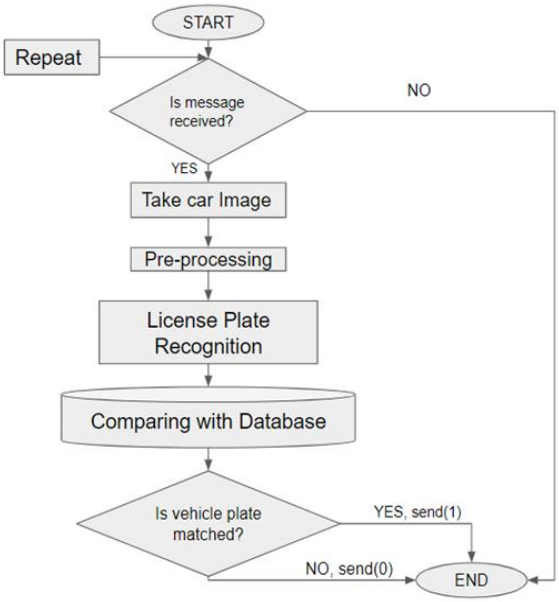

Figure 2: General Flowchart

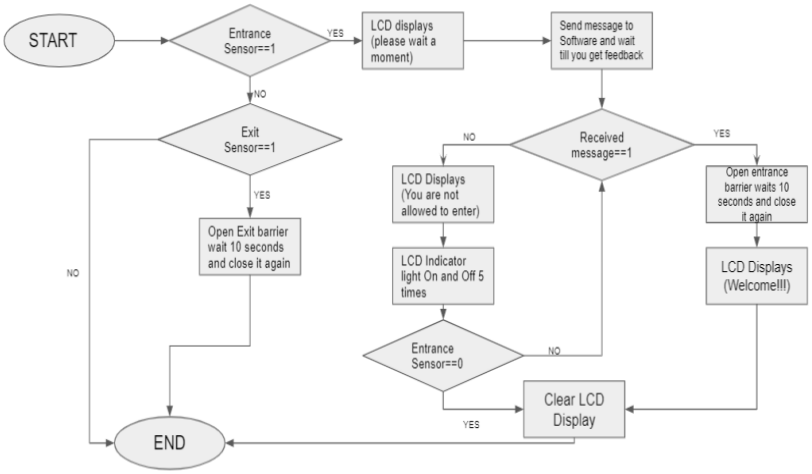

Figure 3: Flowchart of gate operation

Figure 4 shows the principle of plate recognition process. Upon obtaining the plate image, the contours within the plate were computed and evaluated if they were valid characters consistent with their size. The plates were segmented into the detected contours based on validation.

A. IMPORT LIBRARIES AND IMAGE

First we import the python tools and libraries necessary to implement the project. Among these are OpenCV for image processing, Numpy for mathematics, and Pytesseract for optical character recognition (OCR).Once libraries have been imported,the image is imported using its path, and the image is stored in a variable with the name image.

Figure 4: Original Image

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 6, June 2021
DOI: 10.17148/IARJSET.2021.86146

B. PREPROCESSING

In a colored image, each pixel is defined by three values, one each for the red, blue, and green components of the pixel scalar. M*N*3 array of classes. In order to store a single pixel of an RGB image, m*n*3 bits are necessary, but when the RGB image is converted to grayscale, only m*n bits are needed to store a single pixel. Therefore, grayscale images require 33 percent more memory than RGB images. When dealing with many morphological operations and image segmentation problems, it is much easier to work with a single-layered image (grayscale) than a three-layered image (RGB color). Furthermore, the features of an image are more easily discernible when we are dealing with a single-layered image.

Figure 5: Grayscale Image

In order to reduce the background noise of the grayscale image, we will blur the image after gray scaling. This is accomplished by passing them through a low-pass filter kernel. The low-pass filter kernel filters out high-frequency content from the image, blurring the edges, although there are other methods that don't blur edges. Blurring the gray image can be achieved using different methods. Averaging (first method) is done by combining an image with a normalized box filter. This method takes the average of all pixels in and around the kernel area and assigns it to the central element. Instead of using a box filter, we use a Gaussian kernel in the Gaussian Blurring method (second method). The kernel's height and width must be odd and positive. In addition, we specify the standard deviations in the Y and X directions, sigma X and sigma Y, respectively. The median blurring (third method) assigns a value to the central element based upon the median of the pixels within the kernel area. This method is highly effective against pepper-and-salt noise in the image, and its kernel size should be an odd and positive integer. Bilateral Filtering (fourth method) is highly effective at removing noise and keeping edges sharp. It is slower than the other methods. This technique takes a Gaussian filter, but then adds a Gaussian filter which is a function of pixel differences. That way the edges are not affected.

Figure 6: Bilateral conversion

Blurring is followed by edge detection, which is an important part of computer vision, especially when we are dealing with contours. In an image, edges, which represent sudden changes, hold just as much information as pixels. They also serve as the image's boundaries. Edge detection can be broken down into three different types. The Sobel Edge Detection (first method) is a way to avoid the gradient calculated about an interpolated point between pixels, which uses 3 x 3 neighbourhoods. It finds vertical or horizontal edges. Laplacian Edge Detection (Second method) uses the features extracted from the target image to construct a morphing algorithm. It is a good method for discovering the edges in the target image. In addition to this, the Canny Edge Detection (Third method) is also a very good edge detection method. It begins by smoothing the image with the Gaussian filter. Next, the gradient magnitude and orientation are derived from partial derivatives, via finite difference approximations, with non-maxima suppression for the gradient magnitude. In the next step, the double threshold algorithm is used to link and detect edges. The Canny edge detector represents an approach to optimize the multiplicative product of localization and signal-to-noise ratio. In general, it is the first derivative of a Gaussian. In order to extract the edges from a blurred image, we will use Canny edge detection since it offers the best results, well-defined edges, and accurate detection methods.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 6, June 2021

DOI: 10.17148/IARJSET.2021.86146

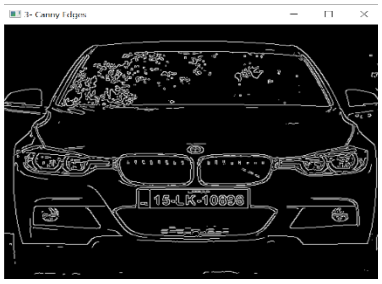

Figure 7: Canny edges

Having determined the edges of an image we will then determine the contours. Contours are the continuous curves or lines covering or bounding the entire boundary of the image. Object detection and shape recognition are highly dependent on contours. There are two important types of contour retrieval. Traditionally there are two main types of contour retrieval modes. The first one is cv2.RETER_LIST which retrieves all contours from an image and the second one is cv2.RETER_EXTERNAL which retrieves external contours from an image. First, cv2.CHAIN_APPROX_NONE stores all the boundary points. However, we don't always need all of the boundary points since a straight line only has to be defined at its beginning and end. Alternatively, you can use the cv2.CHAIN_APPROX_SIMPLE method. This method provides only the starting and ending points of the contours, making it much more efficient for storing contour information.

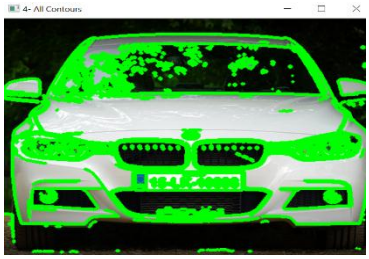

Figure 8: All counters

Figure 9: Top 10 counters

Once contours have been found, they are sorted. Sorting contours is highly useful when processing images. It is important to sort contours by area in order to remove small and unnecessary contours created by noise and find the large contour containing the license plate of a vehicle. For the purpose of locating the number plate, the top 10 contours were used, since there's only the frontal view of the car to work on, and the number plate covers most of its region.

C. Detecting Plate

After sorting the contours, a variable plate is created and a value of none is stored in the variable, indicating that we have not yet located the number plate. The contours obtained after sorting from the largest to the smallest with our number plate in there are then iterated in order to segment them out. After that, all the contours are taken into consideration, and the perimeter for each contour is calculated. Approximation Accuracy is the second parameter, which determines the accuracy of the approximation. Small values provide more precise approximations, while large values present more generic approximations. It is advisable to keep the box perimeter to less than 5% of the contour perimeter. The third parameter is a Boolean value that specifies whether an approximate contour should be open or closed. A contour approximation is used, which approximates a contour shape to another shape with a smaller number depending on what the user specifies, this value here is 0.02. A check is done to see if the edge count is equal to four and the plate is located. Then the coordinates of the rectangle formed using cv2.boundingRect(c) are found and stored in x, y, and width and height of the rectangle are stored in another array variable. Lastly, the image of the rectangle is stored in the table variable.

IARJSET
ISSN (Online) 2393-8021
ISSN (Print) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 6, June 2021
DOI: 10.17148/IARJSET.2021.86146

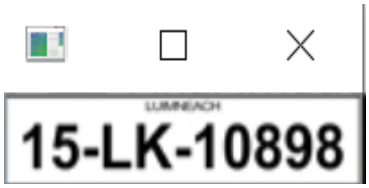

Figure 10: Detected License Plate

D. TEXT RECOGNITION

Following detection of the vehicle's license plate, the characters are recognized using tesseract. Python-tesseract is a wrapper for Google's TesseractOCR Engine. It enables optical character recognition (OCR) of images in Python. In addition, it can be used as a stand-alone invocation script for tesseract, since it is capable of reading all image formats supported by Leptonica imaging and Pillow, including PNG, JPG, GIF, BMP, and TIFF. Python-tesseract is a script configured to print the recognized text, instead of writing it to a file. Optical character recognition (OCR) converts a scanned handwritten or printed document into editable text for further processing. With this technology, the machine can recognize text automatically. It is like combining the mind and eyes of a human being. A human eye can read text only from an image, but the brain is actually processing and interpreting what the eye reads.

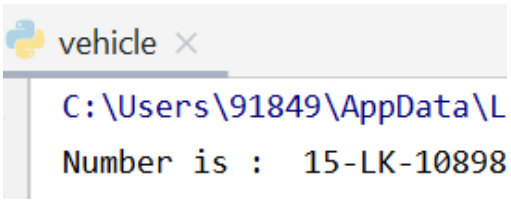

Figure 11: Output

E. VEHICLE AUTHENTICATION

Once the characters are recognized from the number plate using Tesseract, the string output is extracted and is searched through the database to check its authorization.

The database is created using SQLite, a table is created called Vehicle_Details, the table has three columns: Name, Vehicle_Number, Vehicle_model. All the three columns are of TEXT storage class.

A representative of the organization has to update the database of registered vehicle details, this updated table is used by the system to authenticate the vehicle going to enter the parking space.

If the vehicle number is found in the database, the vehicle is tagged authorized and a message is displayed on the LCD screen as “Welcome”.

If the vehicle number is not found in the database the vehicle is tagged as unauthorized and the message displayed is “You are not allowed to enter”.

	Name	Vehicle_Number ▾1	Vehicle_model
	Filter	Filter	Filter
1	Agni	KA01G0100	Police Jeep
2	Aravind	KA01P4568	Audi R8
3	Chethan G	KA05KD0223	McLaren P1
4	Supriya V	KA05M4597	TVS XL100
5	Praveen Andrew	KA05P4628	Mercedes-Benz C-Class
6	Rahul Nadig	KA18X7557	Ferrari Roma

Figure 12: Database

III.RESULT

Several scenarios of toy cars were simulated to test the system. Displays the interface of our application with the corresponding results. The sensor detected the toy car that was stopped. The program successfully recognized the vehicle and recognized the plate KA51MD4175. The number is verified with the database. If the vehicle number is found in the database, the vehicle is tagged authorized and a message is displayed on the LCD screen as “Welcome”. If the vehicle number is not found in the database the vehicle is tagged as unauthorized and the message displayed is “You are not allowed to enter”. Since the car being tested was of registered user, the car is allowed access by enabling the gate to open.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 6, June 2021

DOI: 10.17148/IARJSET.2021.86146

Figure 6.1: Registered Vehicle

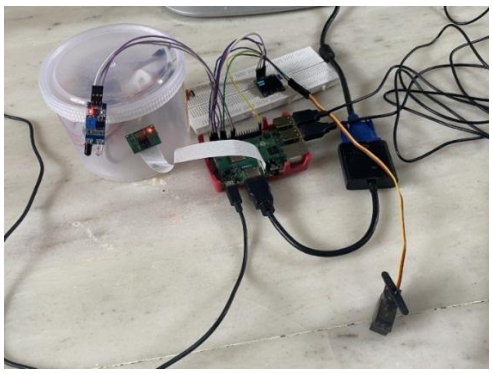

Figure 6.2: Project Setup

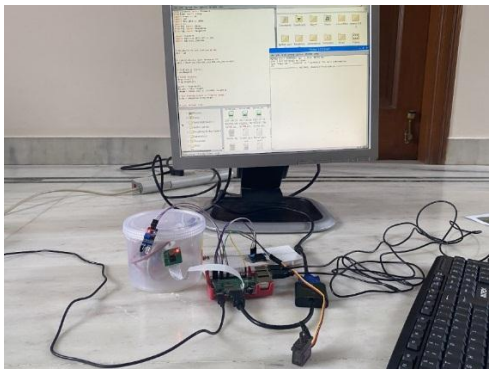

Figure 6.3: Pi Interface using Monitor

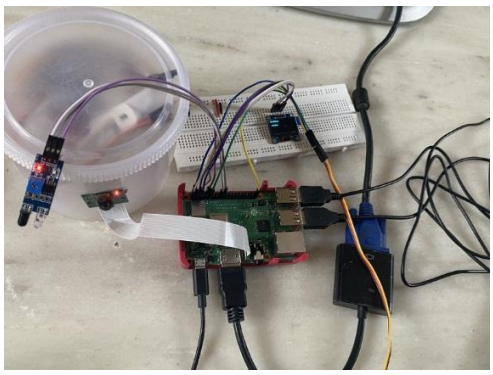

Figure 6.4: Top View of Setup

According to the test results in the dataset, 93% accuracy was achieved in recognizing the car. However, there were two incorrect license plate detections, which may have been caused by light reflections on the windshield. Apart from this prototype, more accurate algorithms can be explored for the recognition and detection phases which would help the

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 6, June 2021

DOI: 10.17148/IARJSET.2021.86146

system to be implemented in a real-life situation. The prototype takes about two seconds to recognize and detect. Optimized algorithms result in comparable processing times, something that is really helpful in areas such as international airports where we can leave our cars for extended periods without fear of theft. The assurance of security is also useful in areas like long-term parking lots when leaving our cars overnight. Security systems in military compounds can facilitate authorized vehicles' entries without requiring security personnel. These systems are more secure and unbiased.

IV. FUTURE WORK

ANPR (automatic license plate recognition) systems are very useful to identify traffic violations and limit their impact. Furthermore, ANPR systems can ensure the safety of people and cars as well as control theft. Today, ANPR systems are commonly used in public places like parks, shopping malls, etc. The ANPR may not provide enough recognition of certain vehicles in certain cases [2], since the metal can be easily forged. For example, a thief might replace a stolen car license plate with another one to gain access to a secured area. For this reason, we cannot rely solely on the ANPR systems in more heavily secured areas such as military bases, governmental offices, etc. In order to demonstrate the real identity of the vehicle, the face of the driver can be added as a second confirmation. An automated gate control system with a multi-level authentication can take into account multiple cues on the vehicle. It can thus prevent unauthorized access.

V. CONCLUSION

In this project, we are able to design an authentic technique of license plate recognition with an accurate reading of vehicle number plates. Here the system is able to identify authorized vehicles and let them through the gate. In conclusion, this system can be incorporated in places where security is important, it is easy to use and accurate. The further work is on its accuracy, a machine learning model can be integrated to automate and detect vehicles position along with fetching its license plate number, even if the image captured is not that clear There are many machines learning models that can be used, the one with best accuracy can be considered.

REFERENCES

[1] National Police Chief's Council. "Automatic Number Plate Recognition (ANPR)".Online.Available: https://www.npcc.police.uk/FreedomofInformation/ANPR.aspx.[Acce ssed: Nov 28, 2019].
[2] C. Parker. (2 December 2018). "Warning to Brits over fake number plates as simple tips could stop YOU becoming a victim". [Online]. Available: https://www.thesun.co.uk/uncategorized/7879382/warningbrits-fake-number-plates/.[Accessed: Nov 28, 2019].
[3] S. Almaadeed, A. Bouridane, D. Crookes, O. Nibouche, "Partial shoeprint retrieval using multiple point-of-interest detectors and SIFT descriptors," Integrated Computer-Aided Engineering, vol. 22, (1), pp. 41-58, 2015.
[4] M. Zahedi and S. M. Salehi, "License plate recognition system based on SIFT features," Procedia Computer Science, vol. 3, pp. 998-1002, 2011.
[5] F. A. da Silva, A. O. Artero, M. S. V. de Paiva, and R. L. Barbosa. (2013), "ALPRs-A new approach for license plate recognition using the SIFT algorithm". [Online]. Available: https://arxiv.org/abs/1303.1667. [Accessed: Nov 28, 2019].
[6] K. M. A. Yousef, M. Al-Tabanjah, E. Hudaib, and M. Ikrai, "SIFT based automatic number plate recognition," in 2015 6th International Conference on Information and Communication Systems (ICICS), 2015, pp. 124-129.
[7] R. Azad, F. Davami and B. Azad, "A novel and robust method for automatic license plate recognition system based on pattern recognition," Advances in Computer Science: An International Journal, vol. 2, (3), pp. 64-70, 2013.
[8] K. Bhosale, J. Jadav, S. Kalyankar, R. Bhambare, "Number Plate Recognition System for Toll Collection," International Journal of Emerging Technology and Advanced Engineering, vol. 4, (4), pp. 729- 732, 2014.
[9] G. Lekhana and R. Srikantaswamy, "Real time license plate recognition system," International Journal of Advanced Technology & Engineering Research, vol. 2, (4), pp. 5-9, 2012.
[10] A. Kaur, S. Jindal and R. Jindal, "License plate recognition using support vector machine (SVM)," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 2, (7), 2012.
[11] R. Chen, "Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning," Image Vision Comput., vol. 87, pp. 47- 56, 2019.
[12] L. Connie, C. K. On and A. Patricia, "A Review of Automatic License Plate Recognition System in Mobile based Platform," Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 10, (3-2), pp. 77-82, 2018.
[13] I. Zafar, E. A. Edirisinghe, S. Acar, H. E. Bez, "Two-dimensional statistical linear discriminant analysis for real-time robust vehicle-type recognition," in Real-Time Image Processing, 2007, pp. 649602. Figure 1. Sample images from our database 83 Authorized licensed use limited to: Newcastle University. Downloaded on May 18,2020 at 05:14:38 UTC from IEEE Xplore. Restrictions apply.
[14] H. Emami, M. Fathi and K. Raahemifar, "Real time vehicle make and model recognition based on hierarchical classification," International Journal of Machine Learning and Computing, vol. 4, (2), pp. 142, 2014.
[15] D. F. Llorca, D. Colás, I. Daza, I. Parra, M. A. Sotelo, "Vehicle model recognition using geometry and appearance of car emblems from rear view images," in 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2014, pp. 3094-3099.
[16] R. Baran, T. Rusc and P. Fornalski, "A smart camera for the surveillance of vehicles in intelligent transportation systems," Multimedia Tools Appl, vol. 75, (17), pp. 10471-10493, 2016.
[17] R. Baran, A. Glowacz and A. Matiolanski, "The efficient real-and nonreal-time make and model recognition of cars," Multimedia Tools Appl, vol. 74, (12), pp. 4269-4288, 2015.
[18] S. Al-Maadeed, R. Boubezari, S. Kunhoth, A. Bouridane, "Robust feature point detectors for car make recognition," Comput. Ind., vol. 100, pp. 129-136, 2018.
[19] H. J. Lee, I. Ullah, W. Wan, Y. Gao, Z. Fang , "Real-time vehicle make and model recognition with the residual SqueezeNet architecture," Sensors, vol. 19, (5), pp. 982, 2019.

IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 6, June 2021

DOI: 10.17148/IARJSET.2021.86146

[20] S. Al-Maadeed, M. Bourif, A. Bouridane, R. Jiang, "Low-quality facial biometric verification via dictionary-based random pooling," Pattern Recognit, vol. 52, pp. 238-248, 2016.
[21] P. Viola and M. J. Jones, "Robust real-time face detection," International Journal of Computer Vision, vol. 57, (2), pp. 137-154, 2004.
[22] Y. Ban, S. K. Kim, S. Kim, K. A. Toh, S. Lee, "Face detection based on skin color likelihood," Pattern Recognit, vol. 47, (4), pp. 1573-1585, 2014.
[23] K. Seo, W. Kim, C. Oh, J.-J. Lee, "Face detection and facial feature extraction using color snake," in Proceedings of the 2002 IEEE International Symposium on Industrial Electronics, L'Aquila, Italy, 2002, pp. 457-462.
[24] C. Tsai, W.C. Cheng, J.S. Taur and C.W. Tao, "Face detection using eigenface and neural network," in 2006 IEEE International Conference on Systems, Man and Cybernetics, 2006, pp. 4343-4347.
[25] A. Mahmood, M. Uzair and S. Al-Maadeed, "Multi-order statistical descriptors for real-time face recognition and object classification," IEEE Access, vol. 6, pp. 12993-13004, 2018.
[26] S. S. Farfade, M. J. Saberian and L. Li, "Multi-view face detection using deep convolutional neural networks," in Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, 2015, pp. 643- 650.
[27] X. Sun, P. Wu and S. C. Hoi, "Face detection using deep learning: An improved faster RCNN approach," Neurocomputing, vol. 299, pp. 42- 50, 2018.
[28] K. Sundus and S. Mamare, "Using Digital Image Processing to Make an Intelligent Gate," (IJACSA) International Journal of Advanced Computer Science and Applications, vol. 5, (5), 2014.
[29] M. M. Aziz, "Ishik University Gate Control based on Kurdistan License Plat," International Journal of Enhanced Research in Science, Technology & Engineering (2319-7463), vol. 5, (1), pp. 34-38, 2016.
[30] M. M. Rashid, A. Musa, M. A. Rahman, N. Farahana, and A. Farhana, "Automatic parking management system and parking fee collection based on number plate recognition," International Journal of Machine Learning and Computing, vol. 2, (2), pp. 94, 2012.
[31] A. U. Ahmed, T. M. Masum and M. M. Rahman, "Design of an automated secure garage system using license plate recognition technique," International Journal of Intelligent Systems and Applications, vol. 6, (2), pp. 22, 2014.
[32] L. T. A. Al-Mahbashi, N.A. Yusof, S. Shaharum, M. S. Karim, A. A. Faudzi, "Development of automated gate using automatic license plate recognition system," in Proceedings of the 10th National Technical Seminar on Underwater System Technology, 2019, pp. 459-466.
[33] P. Bongard, Automated Security Gate Attendant, United States patent US 8,254,631. Aug 28 2012.
[34] C. Anagnostopoulos, I. Giannoukos, T. Alexandropoulos, A Psyllos, V. Loumos, and E. Kayafas , "Integrated vehicle recognition and inspection system to improve security in restricted access areas," in 13th International IEEE Conference on Intelligent Transportation Systems, 2010, pp. 1893-1898.
[35] Mathworks, Vision Cascade Object Detector. [Online]. Available: https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdete ctor-system-object.html [Accessed: Nov 28, 2019]