IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547

Literature review on "AUTOMATIC GATE CONTROL SYSTEM USING NUMBER PLATE RECOGNITION WITH OCR"

Chethan G [1], Rahul R Nadig [2], Supriya V [3], Praveen Andrew [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: License plate detection, automatic gate control, Optical character recognition (OCR).

I. INTRODUCTION

Vehicle License Plate detection is a type of automatic vehicle recognition. Almost everything in the modern world is going automatic, we have built this project to increase the convenience and security at the entrance gate. This technology can be used in various security and traffic applications, controlling access to car parks and gathering traffic flow statistics. The purpose of this paper is to develop an automatic gate control application which recognizes the license plate of a vehicle at the entrance gate and an action is taken to let vehicles enter or not. The system consisting of Raspberry pi and video camera, catches video frames which include a visible car license plate and processes them. A new computer vision algorithm of number plate detection is used. After reading the vehicle number the system will allow only the authorized vehicles.

II. METHODOLOGY

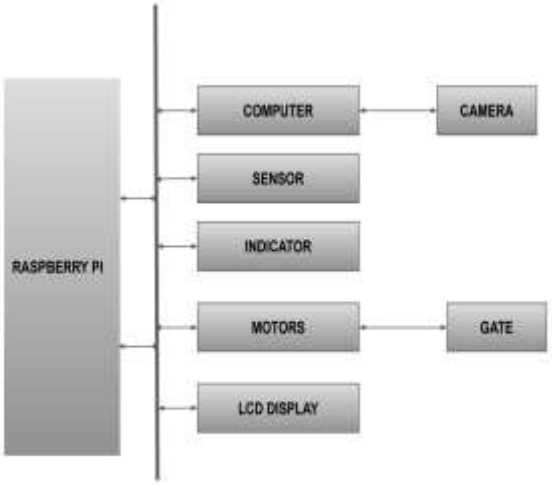
At First, the car will stand in front of the barrier, then the IR sensor sends a signal to the Raspberry Pi and it will send a message to the python, then LCD will display a Welcome Message. Then, the image (license plate) acquired from the hardware components by the Camera will be analysed in data analysis part which is mostly done in python. Then the analysed image will be compared with the information stored in the database, if this image is matched with the information stored in the database, then python sends message to the Raspberry Pi to open the barrier gate and after some time delay the barrier gate will be closed again. But if the image doesn't match with any of those in the database then python will send a message to the Raspberry Pi, and the Raspberry Pi will turn on the alarm and LCD will display a message "you are not allowed to enter, please go back".

IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology
Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547



Block Diagram of Automated Gate

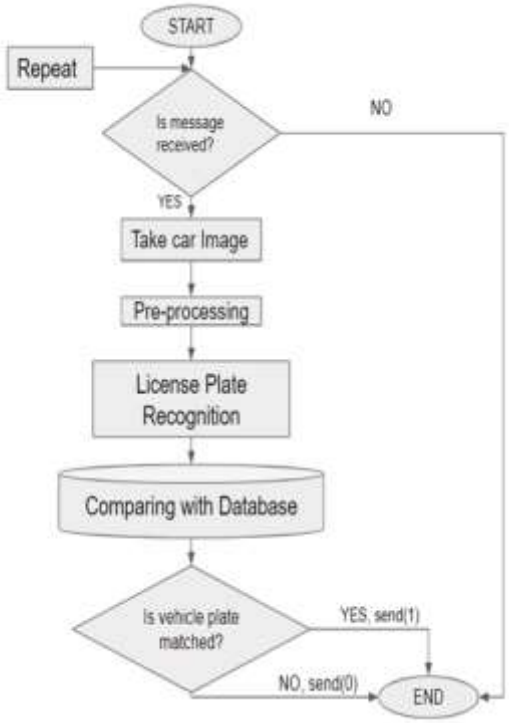


Fig.2 General Flowchart

Figure 2 shows the flowchart of the plate recognition process. Upon obtaining the plate image, the contours within the plate were computed and evaluated if they were valid characters according to their size. Upon validation, the plates were segmented into the detected contours.

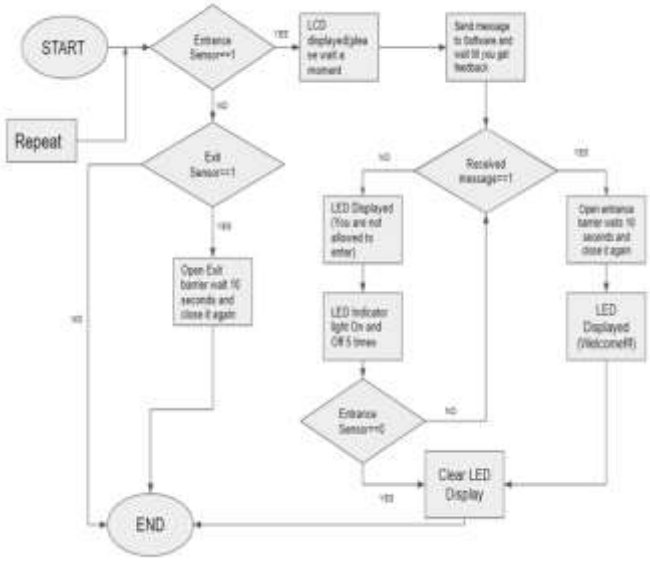


Fig.3 Flowchart of gate operation

II.A *Import Libraries and Image*

To implement the project first various python tools and libraries are imported. We will import a few libraries that are OpenCV for image processing, Numpy for mathematics and Pytesseract for optical character recognition (OCR). After the libraries are imported, I import the image using its path and store the image in a variable named as image.

IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547

Fig.4 Original Image

II.B *Pre-processing*

A coloured image is an image in which each pixel is specified by three values, one each for the red, blue, and green components of the pixel scalar. M*N*3 array of classes. To store a single colour pixel of an RGB colour image we will need m*n*3 bits, but when we convert an RGB image to a grayscale image, only m*n bits are required for storage of a single-pixel of an image. So we will need 33 percent more memory for the storage of grayscale images than to store an RGB image. Grayscale images are much easier to work within a variety of tasks like In many morphological operations and image segmentation problems, it is easier to work with the single-layered image (Grayscale image) than a three-layered image (RGB colour image). It is also easier to distinguish features of an image when we deal with a single-layered.

Fig.5 Grayscale Image

After gray scaling we will blur the grey image to reduce the background noise. Image blurring is done by passing an image with the low-pass filter kernel. It is very useful for removing noise. It removes high-frequency content from the image. So, edges are blurred in this operation but there are also blurring techniques that don't blur the edges. There are different blurring methods that can be used to blur the gray image. Averaging (first method) is done by convolving an image with a normalized box filter. This method takes an average of all the pixels under the kernel area and assigns the central element. In the Gaussian Blurring method (second method), instead of a box filter, a Gaussian kernel is used. We specify the height and width of the kernel which should be odd and positive. We also specify the standard deviation in the Y and X directions, sigma X, and sigma Y respectively. Median Blurring (third method) takes the median of all the pixels under the kernel area and the central element is assigned with this median value. This is highly effective against pepper-and-salt noise in the image. Its kernel size should be an odd and positive integer. Bilateral Filtering (fourth method) is highly effective in noise removal and keeping edges sharp. This operation is slower as compared to other filters. Bilateral filtering takes a Gaussian filter, but one more Gaussian filter which is a function of pixel difference so it does not affect the edges. We have used the bilateral filter to blur the image because it actually preserves all strength, it removes noise quite well and strengthens the edges in the image when we deal with a single-layered image.

IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547

Fig.6 Bilateral conversion

After blurring we will do edge detection. It is a very important part of computer vision, especially when we are dealing with contours. Edges are defined as sudden changes in an image. They can encode just as much information as pixels. Edges are also defined as the boundaries of the images. There are three main types of Edge Detection. Sobel Edge Detection (first method) is a way to avoid the gradient calculated about an interpolated point between the pixels which uses 3 x 3 neighbourhoods for the calculations of the gradient. It finds vertical or horizontal edges. LaplacianEdge Detection (Second method) builds a morphing algorithm that operates on features extracted from target images. It is a good method to find the edges in the target images. Canny Edge Detection (Third method) follows the series of steps and is a very powerful edge detection method. First it smoothens an image with the Gaussian filter. Then it computes the gradient magnitude and orientation using finite-difference approximations for the partial derivatives. Then it applies non-maxima suppression to the gradient magnitude. After this in the next step uses the double threshold algorithm to link and detect edges. Canny edge detector approximates the operator that optimizes the product of localization and signal-to-noise ratio. It is generally the first derivative of a Gaussian. We Will use the canny edge detection to extract the edges from the blurred image because of its optimal result, well-defined edges, and accurate detection.

Fig.7 Canny edges

After finding edges we will find the contours from an edged image. Contours are the continuous curves or lines that cover or bound the full boundary of the object in the image. Contours play a very important role in object detection shape identification. There are mainly two important types of contours Retrieval Modes. The first one is the cv2.RETER_LIST which retrieves all the contours from an image and second is cv2.RETER_EXTERNAL which retrieves external or outer contours from an image. There are two types of Approximation Methods. The first method is the cv2.CHAIN_APPROX_NONE stores all the boundary points. But we don't necessarily need all bounding points. If the points form a straight line, we only need the start and ending points of that line. The second method is the cv2.CHAIN_APPROX_SIMPLE instead only provides these start and endpoints of bounding contours, thus resulting in much more efficient storage of contour information.

IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547

Fig.8 All counters

Fig.9 Top 10 counters

After finding contours we will sort the contours. Sorting contours is quite useful when doing image processing. We will sort contours by area which will help us to eliminate some small and useless contours made by noise and extract the large contours which contain the number plate of a vehicle. We will take the top 10 contours to find our number plate as we are only using the frontal image of the car and we can expect the area of the number plate covers most of the region in an image.

II.C *Detecting Plate*

After we sort the contours, we will now take a variable plate and store a value none in the variable recognizing that we did not find the number plate till now. Now we iterate through all the contours we get after sorting from the largest to the smallest having our number plate in there so we should be able to segment it out. Now to that, we will look through all the contours and go to calculate the perimeter for each contour. Then we will use cv2.approxPolyDP () function to count the number of sides. The cv2.approxPolyDP () takes three parameters. First one is the individual contour which we wish to approximate. Second parameter is Approximation Accuracy Important parameter is determining the accuracy of the approximation. Small values give precise- approximations, large values give more generic approximations. A good rule of thumb is less than 5% of the contour perimeter. Third parameter is a Boolean value that states whether the approximate contour should be open or closed. I had used contour approximation and it approximated a contour shape to another shape with less number of that is dependent on the position I specify so the 0.02 is the precision that worked. After that we will compare if the edge count is equal to 4 so we found our number plate. After that we will find the coordinates of the rectangle formed using cv2.boundingRect(c) and store the one coordinate in x, y and store width and height of the contour in another. After that we put the image of the detected rectangle in the plate variable.

Fig.10 Detected License Plate

IARJSET

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

International Advanced Research Journal in Science, Engineering and Technology

Vol. 8, Issue 5, May 2021

DOI: 10.17148/IARJSET.2021.8547

II.D *Text Recognition*

After detecting the license plate of the vehicle, we will recognize the characters on the license plate using tesseract. Python-tesseract is an (OCR) optical character recognition tool for python. That is, it will recognize and read‖ the text embedded in images. It is a wrapper for Google's TesseractOCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Leptonica imaging and Pillow libraries, including png, jpeg, gif, BMP, tiff, and others. If Python-tesseract is used as a script it will print the recognized text instead of writing it to a file. Optical character recognition (OCR) is a conversion of printed text images or handwritten text scanned copy, into editable text for further processing. This technology gives an ability to the machine to recognize the text automatically. It is like a combination of the mind and eyes of the human body. An eye can only view the text from an image but the brain actually processes as well as interprets that extracted text read by eye.

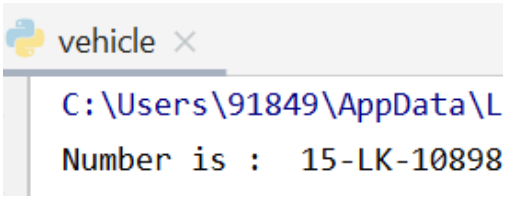


Fig.13 Output Text.

III. CONCLUSION AND FUTURE SCOPE

In this paper, 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 machine learning models that can be used, the one with best accuracy can be considered.

REFERENCES

[1] M. S. Uddin, A. K. Das and M. A. Taleb, "Real-time Area Based Traffic Density Estimation by Image Processing for Traffic Signal Control System: Bangladesh Perspective," in 2nd Int'l Conf. on Electrical Engineering and Information & Communication Technology, Dhaka, 2015.
[2] J. Vijverberg, N. A. H. M. de Koning and J. Han, "HIGHLEVEL TRAFFIC-VIOLATION DETECTION FOR EMBEDDED TRAFFIC ANALYSIS," in IEEE International Conference on Acoustics, Speech and Signal Processing, Honolulu, 2007.
[3] R. A. Bedruz, E. Sybingco, A. R. Quiros, A. C. Uy, R. R. Vicerra and E. Dadios, "Fuzzy Logic Based Vehicular Plate Character Recognition System Using Image Segmentation and ScaleInvariant Feature Transform," in TENCON 2016 - 2016 IEEE Region 10 Conference, Singapore, 2016.
[4] S. Ezell, "Explaining International IT Application Leadership: Intelligent Transportation Systems," The Information Technology & Innovation Foundation, pp. 1-5, January 2010.
[5] "Intelligent transport systems: EU-funded research for efficient, clean and safe road transport," European Commission, pp. 5-8, 2010.
[6] R. A. Bedruz and A. R. F. Quiros, "Comparison of Huffman Algorithm and Lempel-Ziv Algorithm for audio, image and text compression," in Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), 2015 International Conference, Cebu, 2015.
[7] A. R. F. Quiros, A. C. Abad and E. P. Dadios, "Object locator and collector robotic arm using artificial neural networks," 2015 International Conference on Humanoid, Nanotechnology, Technology, Communication and Control, Environment and Management (HNICEM), Cebu City, 2015, pp. 1-6.
[8] J. R. F. Regidor, "Intelligent Transport Systems: Leveraging to Address Transport and Traffic Challenges in the Philippines," Makati, 2010.
[9] B. F. Momin and T. M. Mujawar, "Vehicle detection and Attribute based search of vehicles in video surveillance system," in International Conference on Circuit, Power and Computing Technologies, Nagercoil, 2015.
[10] P. Gupta, M. Rathore and G. N. Purohit, "Detection of Direction Deviation of vehicle using CCTV Cameras," in IEEE International Conference on Recent Advances and Innovations in Engineering, Jaipur, 2014.
[11] R.-T. Lee and K.-C. Hung, "Real-Time Vehicle License Plate Recognition Based on 1-D Discrete Periodic Wavelet Transform," in International Symposium on Computer, Consumer and Control, Taichung, 2012.
[12] X. Li, L. Liu, J. Xu and Y. Li, "The Research of Vehicle Model Feature Extraction," in the 4th International Conference on New Trends in Information Science and Service Science, Gyeongju, 2010.
[13] W. L. Hsu, H. Y. Mark Liao, B. S. Jeng and K. C. Fan, "Realtime Vehicle Tracking on Highway," IEEE Intelligent Transportation Systems, vol. 2, pp. 909-914, 2003.
[14] A. R. F. Quiros, A. Abad, R. A. Bedruz, A. C. Uy and E. P. Dadios, "A Genetic Algorithm and Artificial Neural Network Based Approach for the Machine Vision of Plate Segmentation and Character Recognition," in 8th IEEE International Conference Humanoid, Nanotechnology, Information Technology, Cebu, 2015.
[15] A. C. P. Uy, R. A. Bedruz, A. R. Quiros, A. Bandala and E. P. Dadios, "Machine vision for traffic violation detection system through genetic algorithm," 2015 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Cebu City, 2015, pp. 1-7.