License Plate Detection Using OCR Method with Raspberry Pi

Annisa Firasanti
Electrical Engineering Department
Universitas Islam "45"
Bekasi, *Indonesia*
annisa_firasanti@unismabekasi.ac.id

Tiara Eka Ramadhani
Electrical Engineering Department
Universitas Islam "45"
Bekasi, Indonesia
tiaraeka@unismabekasi.ac.id

Muhammad Amin Bakri
Electrical Engineering Department
Universitas Islam "45"
Bekasi, Indonesia
amin@unismabekasi.ac.id

Eki Ahmad Zaki Hamidi
Electrical Engineering Department
UIN Sunan Gunung Djati
Bandung, Indonesia
ekiahmadzaki@uinsgd.ac.id

***Abstract*— The application of The Automatic License Plate Recognition (ALPR) to overcome the weakness of reading vehicle license numbers manually is largely determined by the choice of segmentation techniques in processing the detected object image. This study shows a comparison of the performance of two segmentation methods in detecting license plate edges, namely Canny Edge and Otsu Thresholding. The license plate image data is processed through several stages before the license plate text is detected by the OCR and Teserract Library methods. Data processing is done using Rasberry Pi. The performance test of the two methods compared was carried out on 30 samples of vehicles in three time segments, namely morning, afternoon, and evening. From the results of the experiments carried out, Canny Edge shows better performance because it can detect 100% of edges compared to Otsu Thresholding which is only able to detect 70% of edges of the entire data. In general, the system built successfully detects number plates with an average accuracy of 72%.**

Keywords*— *Canny Edge, Otsu Thresholding, Vehicular License Plate, Optical Character Recognition

I. INTRODUCTION

Manual recording of vehicular license plates in parking areas has many weaknesses, such as writing errors, queue times, driver inconvenience, and loss of the history of the number and time of vehicles going in and out. Meanwhile, the automatic recognition of vehicle plate numbers known as The Automatic License Plate Recognition (ALPR) offers many technology options, both from the aspect of reading methods, image processing techniques and the algorithms used[1].

Mostly, character recognition uses Optical Character Recognition (OCR), a system that can read the original text and convert it into digital data[2][3][4]. The use of OCR is very wide, such as handwriting recognition for historical documents [5], medical text for medical records[6], etc. The data processing uses either Arduino Uno or Raspberry Pi[2]. While the interface that displays the results of machine readings is web-based, desktop [4], or mobile[2][3][7]. The character recognition techniques used in OCR also vary, including template matching [8][9], neural network[10], and use an existing library such as Tesseract as this research does.

The technique used in the image processing will determine the success of the machine in recognizing characters based on the number plate image generated by the camera. Based on these needs, this article is intended to show the performance test results of two segmentation methods in image processing, namely Canny Edge and Otsu Thresholding. A good understanding of the advantages and disadvantages of the two methods is expected to make a practical contribution to the selection of image processing techniques for vehicle plate number detection cases and other similar needs.

II. RESEARCH METHOD

This study aims to detect the vehicular license plate character automatically using a Raspberry Pi. The identification method used is OCR with Tesseract Library. This work is the initial stage of the smart parking design, which connects the detected license plates with a database for calculating parking costs and perhaps other future uses.

The object of this research is the license plate image data captured by the raspberry camera and then continued with image processing. The number of license plates tested is 30 plates.

Fig 1. Sample of License Plate Image

The image is captured by the camera after initializing the camera object at a resolution (640, 480) and a frame rate of 30 fps. The distance between the camera and the license plate is 60-160 cm [11]. The experiment is done in the morning, afternoon, and evening to find the right lighting for the camera. The camera and license plate are in a straight line and the vehicle is stationary. The process is continued by using the capture_continuous function to start capturing frames from the Raspberry PI camera. To capture the desired image in this study, the button used here is the "S" button.

A. Pra-processing Image

Before reading the license plate, several processes are carried out to convert the raw image into an image that is

2021 15th International Conference on Telecommunication Systems, Services, and Applications (TSSA) | 978-1-6654-2892-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/TSSA52866.2021.9768252

978-1-6654-2892-7/21/$31.00 ©2021 IEEE

ready to be detected by OCR. These stages are grayscale, bilateral filters, edge detection, and then cropping.

The captured image is resized via the Raspberry Pi Camera, then the results are processed using the grayscale function by converting all the colors in the image into grayscale. This grayscale process is very common in all image processing steps. This process speeds up the next process because we no longer have to deal with the color details in the image when processing the image. After the grayscaling process is conducted, the data image in Fig. 1 change into Fig. 2.

Fig 2. Result of grayscale process

Each image has useful and unused information. In this experiment, only the location of the license plate information is useful while the rest is useless to this experimental program. This unused information or location is called noise. In this case, a bilateral filter (Blurring) is used to remove unwanted details from the image to be detected.

This filter can increase the sigma color and sigma space to a higher value to obscure more background information. However, this work must be done carefully so that the useful parts do not become blurred. The output image is shown in Figure 3, which shows that the background detail of the car body is slightly blurred when compared to Figure 2. The Bilateral Filter function is used to remove unwanted details from the captured image or smooth out the noise in Figure 3.

Fig.3 Result of Filter Bilateral

The next step is edge detection, where this study compares two methods, namely Canny Edge and Otsu thresholding. Each image is processed using both methods, so it will be known which method has better performance in detecting the edge of the license plate. Examples of edge detection results on number plates are shown in Figures 4 and 5.

The next step is finding contours. Raspberry can find several contours, so a filter must be performed to find rectangular contours with four sides and closed images between the results obtained, such as plates that have a closed list and are not damaged. In the image, the counter can be anything that has a closed surface but from 50% of the results obtained, the plate number will also appear because it is a closed surface. After the contour is found, it is presented in the form of a bounding box as shown in Figure 6.

Fig 4. Edge detection with Canny Edge method

Fig 5. Edge detection with Otsu Thresholding method

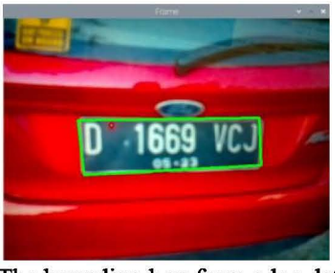

Fig 6. The bounding box from edge detection

After making the bounding box, all the colors in the image are removed, except for the location of the color on the number plate which has been detected and outlined from the previous step after the position of the number plate is known. The display of the results of this process is shown in Figure 7.

Fig 7. Removing all colors outside the bounding box

The final step is the character segmentation process, by cropping the image from Figure 7. This is done so that the recognition process carried out through the OCR process makes the letters more effective by ensuring the image to be recognized is in a very readable condition. After covering the entire image except for the license plate area, the area of the license plate that has been lined up is cut out and saved as a new image as shown in Figure 8.

Fig 8. Cropped License Plate Image

B. *Plate Number Identification*

The identification process is carried out using the OCR method and the pytesseract package, successfully reading the characters from the license plate image using the Tesseract library and storing the recognized characters as 'text' variables which will be read by raspberries. The detected characters from figure 8 are then printed on the console. When compiling the program that has been run, the resulting image is displayed as shown in Figure 9.

```
top left  (94, 152)
button right  (479, 265)
Detected Number is: D -1669 VCJ
```

Fig 9. License Plate Character Identification

The percentage of detection accuracy is calculated by comparing the number of characters correctly detected and the total number of characters on the number plate.

III. RESULT AND DISCUSSION

A. *Edge Detection*

The Canny Edge method works well on plates that have a clear white list and good plate quality. This method will work well by cropping the plate before text recognition is performed. Based on [12], Canny Edge is the method that shows the best result for edge detection among other methods namely Sobel, Prewitt, and Thresholding.

Table 1. Comparison of Edge Detection method

No.	Picture of Plate Number	Canny Edge	Thresholding
1		Detected	Detected
2		Detected	Detected
3		Detected	Undetected
4		Detected	Undetected
5		Detected	Undetected

The experimental results show that Thresholding has several weaknesses in detecting edges, especially if an image has different shape conditions on one background in one frame. This causes thresholding to sometimes detect objects that are not the destination or objects that are not a vehicle number plate. Some examples of edge detection results by these two methods are shown in Table 1.

B. *Translation results*

Based on the experiment, we found that that the license plates that are damaged or of poor quality cannot be detected by Thresholding. Canny edge has better performance [12] because it can detect 100% of edges compared to thresholding which is only able to detect 70% of the 30 samples used in this study (Fig. 11). Therefore, to test the accuracy of the OCR method on the Tesseract library, the Canny Edge method is then used to detect the edges on the license plate.

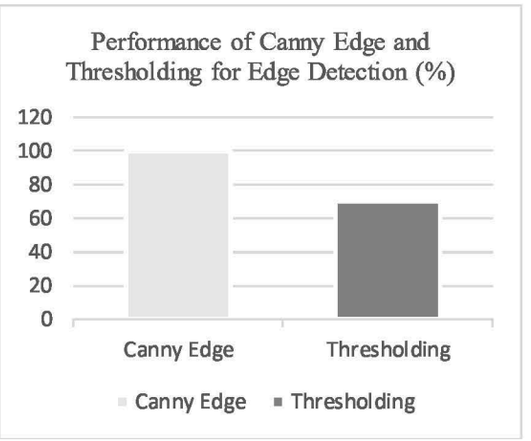

Fig 11. Comparison between Canny Edge and Otsu Thresholding

C. *License Plate Detection*

From 30 samples, the average number plate character reading accuracy is 72%. A total of 21 number plates were detected with good accuracy, namely a minimum of 85% and a maximum of 100% (Table 2). However, the rest have poor accuracy, which is below 50% (table 3).

Table 2. Samples of License plates with high accuracy

No.	License Plate Image	Real License Number	Extraction Result	Luminance (Lux)	Accuration
1		B 3016 TSU	B 3016 TSU	9668	100%
2		AD4 18U	AD4 18U	10915	100%
3		B 3270 FNL	3270 FAL	9175	85%
4		B 3763 KVH	t 3763 KVH	12998	90%

Table 3. Samples of License plates with low accuracy

No.	License Plate Image	Real License Number	Extraction Result	Luminance (Lux)	Accuration
1		B 3064 KQS	B Crers KAS	6871	40%
2		B 3016 TSU	PEO	9668	0%
3		B 3355 FZG	F26	197	25%
4		B 3577 KYI	7	9766	15%

Another factor that affects the performance of the Tesseract is the physical condition of the license plate itself. The characteristics of a number plate that can be detected properly are a plate that has a list on the edge, has no bends, clean from stains or stickers, has not been modified, the writing is clear and not outdated. An example of a number plate that has been successfully read 100% as in Table 2 can be seen in Fig 12.

Fig 12. Sample of license plates that are well-detected by Tesseract

Meanwhile, a license plate that is poorly detected is a plate that does not have a list on the edge of the, or a plate that has a list but because it is outdated, the quality of the plate and the color of the writing is a little dim, and if the plate is affixed to a sticker or plate that is modified by adding a casing as in Fig. 13.

Fig 13. Sample of license plates that are poor-detected by Tesseract

IV. CONCLUSION

This study has succeeded in designing a number plate detection system with an average accuracy of 72%. In detecting edges, the Canny Edge method has better performance than Otsu Thresholding. Canny edge successfully detects 100% of the edges of all data while Thresholding is only 70%.

Based on the experiment, we found that there are several factors affect the success rate of character detection by Tesseract. If the edge is not detected or detected incorrectly, definitely Teserract will not be able to read the characters in the image. Nevertheless, if the edge is detected correctly, then the accuracy of Tesseract in reading characters still depends on several other determining factors, namely the intensity of light and the physical condition of the license plate itself.

The minimum light required to read license plates is 500 lux, otherwise, the Tesseract will not be able to work. High-intensity light, such as flashlight, is also not allowed because it will cause reflections so that the license plate looks blurry when captured by the camera. The maximum distance needed to capture the image is 60 cm, otherwise, the system cannot find the location and identify the license plate.

To enhance the capabilities of this detection system, future works is to focus on camera performance and the optimal angle of image reading.

REFERENCES

[1] A. Budianto, "Automatic License Plate Recognition: A Review with Indonesian Case Study," *Sci. J. Informatics*, vol. 5, no. 2, pp. 258–270, 2018.

[2] W. Sugeng, R. K. Utoro, and M. T. Prabowo, "Identifikasi Plat Nomor Kendaraan Dengan Metode Optical Character Recognition Menggunakan Raspberry Pi," *J. Inform.*, vol. 7, no. 2, pp. 116–125, 2020.

[3] N. D. W. I. Cahyo, "Pengenalan Nomor Plat Kendaraan Dengan Metode Optical Character Recognition," *Ubiquitous Comput. its Appl. J.*, vol. 2, pp. 75–84, 2019.

[4] H. Diwanti, I. S. Sumaryo, and C. Setianingsih, "Real Time Smart CCTV Untuk Mendeteksi Plat Nomor Kendaraan Menggunakan Optical Character Recognition," *e-Proceeding Eng.*, vol. 6, no. 2, pp. 2–9, 2019.

[5] A. Gupta *et al.*, "Automatic assessment of OCR quality in historical documents," *Proc. Natl. Conf. Artif. Intell.*, vol. 3, pp. 1735–1741, 2015.

[6] P. Thompson, J. McNaught, and S. Ananiadou, "Customised OCR correction for historical medical text," pp. 35–42, 2016.

[7] A. Solichin and Z. Rahman, "Aplikasi Identifikasi Nomor Kendaraan Berbasis Android Dengan Metode Learning Vector Quantization," *Tek. Inform.*, vol. 3, no. 3, pp. 216–222, 2015.

[8] P. Hidayatullah, N. Syakrani, I. Suhartini, and W. Muhlis, "Optical character recognition improvement for license plate recognition in Indonesia," *Proc. - UKSim-AMSS 6th Eur. Model. Symp. EMS 2012*, pp. 249–254, 2012.

[9] K. K., "Number Plate Recognition Using Ocr Technique," *Int. J. Res. Eng. Technol.*, vol. 02, no. 09, pp. 286–290, 2013.

[10] B. V. Kakani, Di. Gandhi, and S. Jani, "Improved OCR based automatic vehicle number plate recognition using features trained neural network," *8th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2017*, 2017.
[11] D. Triyandi and J. Adler, "Sistem Otomatisasi Gerbang Dengan Pengolahan Citra Membaca Nomor Plat Kendaraan," *Univ. Komput. Indones.*, pp. 1–7, 2014.
[12] J. Kaur, S. Agrawal, and R. Vig, "A Comparative Analysis of Thresholding and Edge Detection Segmentation Techniques," *Int. J. Comput. Appl.*, vol. 39, no. 15, pp. 29–34, 2012.