



Optical Character Recognition (OCR) Of License Plates Using the KNN Method

Abim Tisanarada ¹, Yo Ceng Giap ^{*1}

¹Prodi Teknik Informatika, Universitas Buddhi Dharma, Tangerang, Banten

Email: pingchen115@gmail.com^{1*}, cenggiap@ubd.ac.id¹

Article Info

Received: 17-07-2025

Revised: 23-07-2025

Accepted: 24-07-2025

Available online : 07-31-2025

Keywords :

OCR;
KNN;
license plate recognition;
image preprocessing;
real-time detection

ABSTRACT

This study aims to implement an IoT-based security system with character recognition (OCR). The OCR system utilizes a webcam and the KNN method to recognize vehicle license plate text in real-time. This prototype was tested using six samples of the latest Indonesian license plates. The character detection process involves steps such as capturing images from the webcam, preprocessing images to improve contrast and convert them to grayscale, and applying calibrated transformations. Image inversion and thresholding are performed to separate characters from the background. Character segmentation and filtering criteria are also performed to clean the character image from noise and remove inappropriate backgrounds. The detected characters are identified using Region of Interest (ROI) detection to ensure the validity of the characters. The validated contours are sorted from left to right to form the complete license plate number. Subsequently, KNN implementation is used to recognize the detected characters. Test results indicate that the KNN-based webcam license plate detection system, with K set to 1, performs well and achieves a sufficiently high level of performance. Testing at camera-to-license plate distances of 60 cm, 70 cm, and 80 cm shows an average accuracy rate of 100% within 5 seconds. This research contributes to the development of an efficient and accurate vehicle license plate recognition system for various applications, including parking systems and access control.

This is an open-access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



*Corresponding Author:

Yo Ceng Giap

Prodi Teknik Informatika, Universitas Buddhi Dharma, Tangerang, Banten

Email: cenggiap@ubd.ac.id

1. Introduction

The rapid expansion of urban populations and increased vehicular mobility in Indonesia have contributed to a notable surge in the number of registered motor vehicles. While this development reflects infrastructural and economic growth, it also introduces critical challenges concerning vehicular security, particularly the persistent issue of vehicle theft. To address these challenges, there is an increasing reliance on intelligent automated systems that incorporate both hardware and software for access control and surveillance. Among such systems, automatic gate mechanisms integrated with access cards and vehicle license plate recognition technology have emerged as

<https://doi.org/xx.xxxxx/iarcixxx>

promising security solutions [1].

At the core of these systems lies Optical Character Recognition (OCR), a digital image processing technique that converts printed or handwritten characters into machine-encoded text. When combined with imaging devices such as CCTV cameras or webcams, OCR enables real-time recognition of vehicle license plates, facilitating applications in toll collection, automated parking, traffic law enforcement, and smart access control [2][3]. Digital image processing is applied as a process of manipulating or editing images digitally. One form of digital image processing is Optical Character Recognition (OCR). OCR is a technology that recognizes and converts printed or handwritten text into a digital format, performed by a computer [4]. Meanwhile, tools for recording text using a camera are available. Webcams in OCR enable real-time text recognition, allowing them to be used in various automation systems. This application, commonly referred to as Automatic Number Plate Recognition (ANPR), eliminates the need for manual data entry, thereby reducing human error and enhancing system efficiency.

Despite the growing deployment of ANPR systems globally, several challenges remain, particularly in achieving high recognition accuracy under diverse lighting conditions, plate rotation, skewed alignment, and font variability. To overcome these issues, machine learning-based classification methods have been adopted in the OCR pipeline. Among them, the K-Nearest Neighbor (KNN) algorithm is notable for its simplicity and effectiveness in pattern recognition [5]. KNN is a non-parametric, instance-based algorithm that classifies data points based on their proximity to labeled training samples in the feature space. Its low computational complexity and resilience to noisy input make it particularly suitable for OCR tasks in constrained or embedded environments [6].

Recent studies reinforce the effectiveness of KNN in OCR-based license plate recognition. For instance, Purwanto demonstrated that KNN achieves high character recognition accuracy (>93%) when applied to Indonesian license plates, even under real-world variations such as lighting inconsistency and low-resolution imagery [7]. Meanwhile, the evolution of deep learning models, such as YOLOv4 and YOLOv5 for license plate localization, combined with Convolutional Neural Networks (CNNs) for character classification, has further enhanced the performance of ANPR systems. Park et al. developed an integrated model capable of both vehicle type recognition and license plate decoding using YOLOv4, achieving state-of-the-art performance in real-time applications [8]. Moreover, Haryanto et al. conducted a comparative analysis of OCR performance between old and new Indonesian plate formats using deep learning, obtaining F1-scores exceeding 97%, thereby underscoring the necessity of adapting OCR methods to the changing regulatory plate standards [3].

While deep learning models offer high accuracy, their implementation often requires significant computational resources, making them less feasible for lightweight applications. In contrast, KNN-based OCR systems provide a practical and cost-efficient alternative for deployment in residential gate systems, parking facilities, and other embedded platforms. Accordingly, this study aims to design and evaluate an OCR system for Indonesian license plates using the KNN method, encompassing key stages such as image acquisition, preprocessing, character segmentation, feature extraction, and classification. OCR uses the KNN (K-Nearest Neighbor) method to detect characters on license plates. KNN is a method used to classify objects based on training data that is closest in distance to the object in question [9]. This work not only contributes to the national literature on vehicle security but also offers a scalable solution for smart transportation infrastructure.

2. Related Work

2.1. Optical Character Recognition (OCR)

According to N. Ali, OCR is a field of computer science that covers the process of reading text from images or pictures and converting it into a form that can be manipulated by computers, such as ASCII code [4]. OCR is the process of reading text from printed or scanned handwritten documents. Currently, OCR is applied in the capture of filled-out forms, passports, ID cards, license plates, and other documents. Not only in desktop

applications, but also uploading or capturing images, OCR is widely applied on smartphones. The quality of the input image is the primary factor affecting recognition performance and influencing the accuracy of OCR [10]. OCR methods are divided into several working arrangements [11]:

- 1) Reset image (Auto Deskewing). Will reset the image to correct any skew if the scan results show skewing. The goal is to make the image straight.
- 2) Analysis. The software performs analysis and separates the text from the image in the image file.
- 3) Automatic orientation adjustment. The software selects a section of the image file and recognizes the correct orientation of the text. The image is then rotated as specified.
- 4) Single-character separation. The software separates each character in the image, creating separate characters for numbers and letters.
- 5) Image identification. The software recognizes each separated character and compares it with the software database. This is done to determine the corresponding letters or numbers.
- 6) Recognition output results. After the image has been converted into letters or numbers, the software generates a final file in a text format such as .docx, .txt, or .xlsx, according to the previously specified settings.

Before the existing data can be used, it must be processed using the Optical Character Recognition algorithm. First, the data will be processed through several stages of data processing.

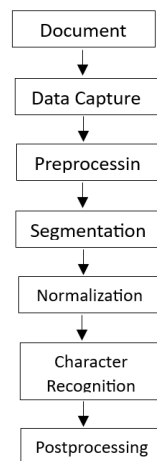


Fig 1. OCR Workflow Diagram [12]

- 1) Data Capture Stage. The process of converting an object/document into a digital image file [13].
- 2) Preprocessing Stage. This stage aims to separate single characters from scanned text with good image quality and clarity, thereby facilitating the character recognition process [14].
- 3) Segmentation Stage. Grouping the observation areas of each character to be recognized separately [13].
- 4) Normalization Stage. The font thickness dimension will be changed using the scaling technique [13].
- 5) Feature Extraction Stage. Aims to identify the characteristics of patterns that are significant and differ significantly traits characteristics, thereby enabling accurate classification [14].
- 6) Character Recognition Process Stage. After character patterns are converted into vector values, the next challenge is how to combine or classify characters with similar valuable vectors. To overcome this problem, a classification process is used, which is the core of the OCR process, focusing on grouping representations. The OCR process focuses on the representation and grouping of characters based on their valuable vectors [14].
- 7) Post-Processing. Even the most advanced character identification systems often make mistakes, which means that not all characters that are read can be correctly converted to the characters they should be. Therefore, this stage is conducted to enhance the accuracy of OCR. [14].

2.2. K-Nearest Neighbour (KNN)

This method is a machine learning technique in pattern recognition. This algorithm is considered one of the 10 most influential data mining algorithms in the research community [15]. According to Thirumuruganathan in [16], K-Nearest Neighbor is a non-parametric algorithm, which means it does not require any assumptions about its environment. The number of parameters in KNN depends on the amount of training data. Classification using KNN involves calculating the average or majority distance of the test vector to its neighboring training vectors.

KNN is a classification method. It classifies new query instances based on the nearest distance by comparing the data to be evaluated with its nearest values from the nearest candidates, which are generally determined by the previous k parameter. The purpose of using this method is to classify new objects by comparing their attributes with existing training data.

Classification is performed in a model-free manner, meaning it is based solely on the existing training data's memory. The algorithm works by classifying test data into specific classes based on the number of nearest neighbors using Euclidean distance calculations. In the calculation, the distance between the test data and the entire dataset, including its class labels, is stored in a vector, which is then sorted based on the shortest distance [17].

3. Methodology

The system is designed to detect objects in license plate images captured in real time using a webcam. Characters in the object image will be identified using the KNN (K-Nearest Neighbors) method in OCR technology. The KNN method was chosen because it is capable of producing a high level of accuracy in image detection.

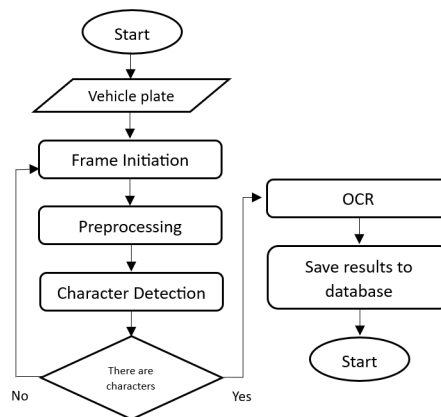


Fig 2. License Plate Detection Flowchart

4. Results

4.1. Discussion of Algorithms KNN

4.1.1 Character Detection

The flowchart in the image below shows how training data images are entered into a folder that will later be used for training. After obtaining the training data, the next step is to capture images using a webcam. The character detection process involves several steps, including image preprocessing, identifying contours that may represent characters, finding matching character groups, removing overlapping characters, and finally recognizing the characters using the KNN algorithm.

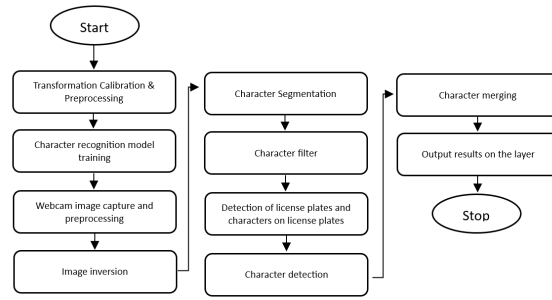


Fig 3. Overall flowchart of the detection system

The following is an explanation of the steps in the image above:

- 1) In the first step, adjust the transformation and preprocessing parameters using the trackbar provided. By using the trackbar, users can adjust the calibration values and parameters to improve the accuracy and quality of the image processing performed by the program.
- 2) Prepare a training dataset containing characters on vehicle license plates. The character recognition model will be saved in a .txt file.
- 3) Use the OpenCV library to capture images from the webcam. Perform preprocessing on the image by converting it to grayscale and enhancing the contrast. Apply the calibrated transformation from the first step.
- 4) Inverting the preprocessed image, so that when thresholding is performed, pixels darker than the pixel threshold are considered part of the character (object) and converted to black, while pixels lighter than the pixel threshold are considered background and converted to white. This results in the characters appearing black in the binary image, while the background appears white.
- 5) Perform character segmentation, such as thresholding or edge detection, to separate the characters on the license plate. Character segmentation will ensure that the separated characters are well separated, no characters are lost or merged, and no noise is mistakenly considered as characters.
- 6) Apply filtering criteria to the segmented characters. Clean the character image of noise, remove the inappropriate background, or correct the character shape.
- 7) Detect the vehicle license plate in an image and extract the license plate image from the original image.
- 8) Implement character detection techniques, such as character segmentation or template matching, to identify characters on the license plate.
- 9) Use the character recognition model trained in step 2.
- 10) Arrange the detected characters in the correct order to form the complete license plate number.
- 11) Display the detected license plate number on the layer.

4.1.2. Detection ROI (*Region Of Interest*)

Validated contours will be sorted from left to right. The contour detection process in an image serves to identify and find contours or closed shapes in a binary image. Contour criteria are provided to validate whether a contour is considered valid or not, with aspect ratio as a potential valid characteristic, where the contour area is greater than 80, width > 2, and length > 8, as well as an aspect ratio with length divided by width between 0.25 and 1.

Once the contours are found, specific contours or areas can be selected as the region of interest (ROI). The program performs character recognition to identify numbers in the image using the K-Nearest Neighbors (KNN) algorithm.

4.1.3. Classification K-NN

The KNN classification process involves recognizing characters that have been detected through the previous method. The training data is in the form of a .txt file. This data will be used to train the KNN algorithm, enabling it to be applied for character recognition. The detected characters are resized to a specified size (20x30 pixels) and reshaped into a 1D numpy array vector, converting the previously integer array into a float format [18].

This reshaping process is performed to ensure that the data conforms to the format required by the KNN method. The training data and labels must be in the appropriate format. By reshaping, we ensure that the data will have the correct form and can be used correctly by the KNN model. Determine the value of k and calculate the nearest neighbor distance. Each detected character is compared with the training data.

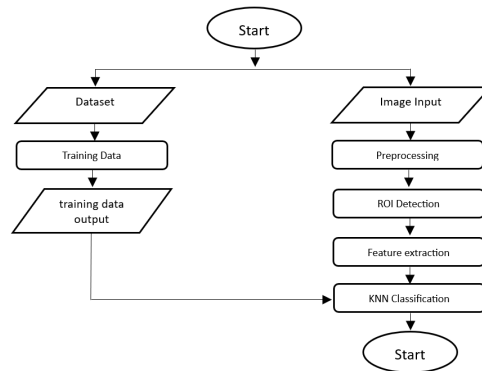


Fig 4. KNN Classification Flowchart [18]

4.1.4. Webcam Display

The following is the webcam display on the system that was created. This display contains video frames that are being processed in color (RGB: Red, Green, Blue) format, allowing it to be used for the webcam display in the program, where the license plate will appear on the webcam display. Then, the display from this webcam will be further processed through the preprocessing and vehicle license plate detection stages.

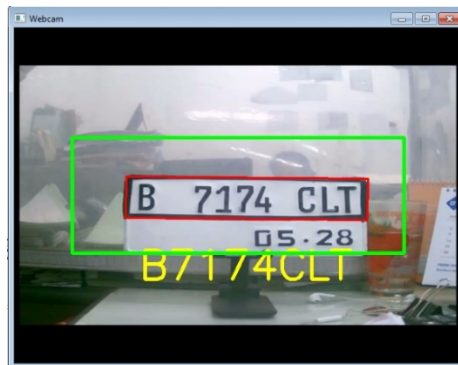


Fig 5. Webcam Display on License Plate

The green box is drawn around the area considered to be the ROI, which is the area around where the license plate is likely to be located. This green box provides a visual indicator to show the potential location of the license plate in the image. Its function is to provide a clear visualization of the detected vehicle license plate in the image, making it easier for users or monitors to identify the relevant area during the vehicle license plate detection process.

The red box serves as a visual indicator to show the position of the detected license plate within the image. The purpose of this red box is to provide a clear and prominent visualization of the detected license plate in the image. Meanwhile, the text on the license plate is colored yellow to display the characters of the detected vehicle license plate over the original image. This function helps in visualizing the characters of the vehicle license plate that have been successfully identified.

4.1.5. Grayscale Display

The following is a grayscale display. It is used as one of the stages in preprocessing in the vehicle license plate detection system in this study. The purpose of the grayscale display is to convert the input image from a color format to a grayscale format that contains only grayscale levels, eliminating color information. This display can then be used in the preprocessing or vehicle license plate detection stages.



Fig 6. Grayscale display of License Plates

4.1.6. Threshold Display

The threshold display in the image below illustrates the results of the thresholding process applied to the original image. The thresholding process separates objects from the background based on the grayscale level of the pixels and is one of the stages in preprocessing.

In the context of vehicle license plate detection in this study, this is done by inverting the colors in the grayscale image. In this context, the original grayscale image has a black background and white license plate characters. However, the character recognition algorithm used requires black license plate characters and a white background to align with the latest Indonesian vehicle license plate regulations set by the National Police Traffic Corps (Korlantas). Therefore, by inverting the image colors, the background becomes black and the license plate characters become white, facilitating the character recognition process.

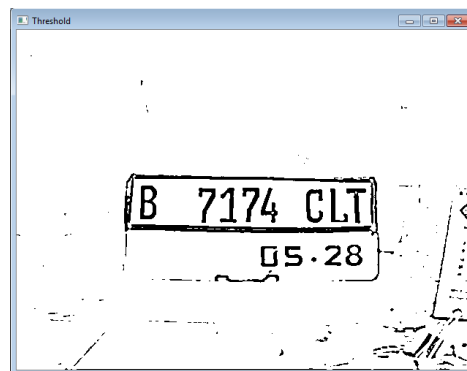


Fig 7. Threshold Display on License Plate

4.2. License Plate Testing Data Results

In processing image data using the KNN algorithm, the system successfully detected the majority of the license plates tested. Nine license plates were tested. The accuracy level column is the result of

$Accuracy\ Level = \frac{Number\ of\ Recognition\ Results}{Number\ of\ Texts\ on\ Plate} \times 100\%$ [19]. The following is a table of test results from several

plates with predetermined distances and a duration of 5 seconds:

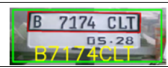
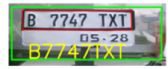
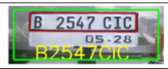
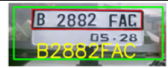
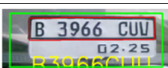
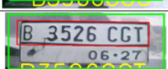
Number plate	Text on Plate	Recognition results	Accuracy Level	Information
	B 7174 CLT	B 7174 CLT	100%	Success
	B 7747 TXT	B 7747 TXT	100%	Success
	B 2547 CIC	B 2547 CIC	100%	Success
	B 2882 FAC	B 2882 FAC	100%	Success
	B 3966 CUU	B 3966 CUU	100%	Success
	B 3526 CGT	B 3526 CGT	100%	Success

Fig 8. Results of license plate detection testing from a distance of 60 cm

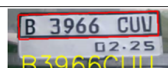
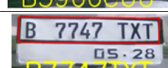
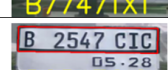
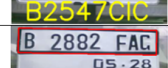
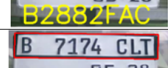
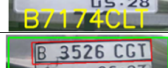
Number plate	Text on Plate	Recognition results	Accuracy Level	Information
	B 3966 CUU	B 3966 CUU	100%	Success
	B 7747 TXT	B 7747 TXT	100%	Success
	B 2547 CIC	B 2547 CIC	100%	Success
	B 2882 FAC	B 2882 FAC	100%	Success
	B 7174 CLT	B 7174 CLT	100%	Success
	B 3526 CGT	B 3526 CGT	100%	Success

Fig 9. Results of license plate detection testing from a distance of 70 cm

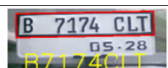
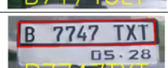
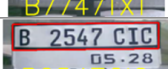
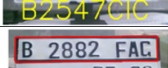
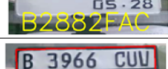
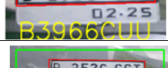
Number plate	Text on Plate	Recognition results	Accuracy Level	Information
	B 7174 CLT	B 7174 CLT	100%	Success
	B 7747 TXT	B 7747 TXT	100%	Success
	B 2547 CIC	B 2547 CIC	100%	Success
	B 2882 FAC	B 2882 FAC	100%	Success
	B 2966 CUU	B 2966 CUU	100%	Success
	B 3526 CGT	B 3526 CGT	100%	Success

Table 10. Results of license plate detection testing from a distance of 80 cm

Based on figure 8, 9, and 10, the tests conducted with the specified parameters, it was found that the optimal parameters are a Gaussian Filter parameter with a threshold value of 255 and a value of $k = 1$, achieving an accuracy rate of 100%. The accuracy rate obtained from distances of 60 cm, 70 cm, and 80 cm with a time interval of 5 seconds resulted in an average accuracy rate of 100%. From these three distance comparisons, the ideal distances for camera placement and vehicle positioning are evident. The Gaussian filter is preferable because it works by

blurring or reducing noise in the image.

The result of this operation is a smoother image with reduced noise, making it more suitable for subsequent processes such as segmentation or object recognition. The threshold value analysis is based on the RGB values converted to grayscale, with the condition that the resulting grayscale image resulting from the conversion is inverted (i.e., the pixel values are reversed). This is done to convert the image into a negative (inverted intensity), where pixels with high intensity become low intensity, and vice versa.

The purpose of this step is to help distinguish between the characters and the background of the license plate during the thresholding process. In this case, a license plate with black characters and a white background is used. However, it is essential to note that using a high threshold requires sufficient light intensity, meaning it should neither be too bright nor too dark. If the light intensity is too high/bright, the characters will become unreadable. When the threshold value is too high, the resulting binary image will have a larger white area and a smaller black area. This causes the loss of details in the image, including thin lines, textures, and other essential elements.

5. Conclusions and Recommendations

5.1. Conclusions

Based on the tests conducted, several conclusions were drawn:

- 1) The webcam, as an OCR tool utilizing the KNN method, has successfully recognized and detected vehicle license plates in real-time for a dual security system, in addition to RFID access cards.
- 2) Based on testing with the specified parameters, it was found that the optimal parameters are a Gaussian Filter, a threshold value of 255 and a k value of 1, yielding an accuracy rate of 100%. The accuracy rate, measured at distances of 60 cm, 70 cm, and 80 cm with a 5 second time interval, yielded an average accuracy rate of 100%.
- 3) The ideal distance for camera placement and vehicle position is between 60 and 80 cm.

5.2. Recommendation

Based on the tests that have been conducted, the following recommendations can be made:

- 1) Sufficient lighting intensity during license plate recording; if there is too much light, it will result in a lack of accuracy in detecting the characters on the license plate.
- 2) For further research, it is hoped that more test samples can be used and that a test environment with consistent lighting can be used.
- 3) The need to determine the ideal shooting distance to obtain a high level of accuracy.
- 4) Font thickness, font type, and font layout greatly affect character detection. Therefore, the font format used must comply with police standards.

References

- [1] C. B. Asaju, P. A. Owolawi, C. Du, and E. Van Wyk, "Enhancing Security with Automated Boom Gate Access Through License Plate Recognition Utilising YOLOv8 Model," in *Communications in Computer and Information Science*, Springer Science and Business Media Deutschland GmbH, 2025, pp. 181–197. doi: 10.1007/978-3-031-85856-7_15.
- [2] Lubna, N. Mufti, and S. A. A. Shah, "Automatic number plate recognition: A detailed survey of relevant algorithms," May 01, 2021, *MDPI AG*. doi: 10.3390/s21093028.
- [3] J. F. Haryanto, J. M. Kristian, and R. Sutoyo, "Automatic Number Plate Recognition Performance Comparison In Old And New Indonesian License Plates Using Deep Learning," *ICIC Express Letters*, vol.

- 18, no. 8, pp. 793–799, Aug. 2024, doi: 10.24507/icicel.18.08.793.
- [4] N. Ali, M. Isheawy, and H. Hasan, “Optical Character Recognition (OCR) System,” *IOSR Journal of Computer Engineering Ver. II*, vol. 17, no. 2, pp. 2278–661, 2015, doi: 10.9790/0661-17222226.
 - [5] M. Arul Selvan, A. Gowtham, T. Haarish, K. Ram Kishore, S. Vignesh Ram, and S. Vignesh, “Real-Time License Plate Recognition and Validation System Through Image Processing and KNN,” *International Journal of Innovative Research in Engineering*, vol. 5, no. 6, pp. 8–10, 2024, [Online]. Available: www.theijire.comhttp://creativecommons.org/licenses/by/4.0/
 - [6] M. Lall and J. A. Van Der Poll, “A Z Specification for Reliability Requirements of a Service-based System,” 2020. [Online]. Available: www.ijacsa.thesai.org
 - [7] W. Purwanto and M. Septiani, “Implementasi Automatic License Plate Recognition untuk mengurangi pelanggaran lalu lintas berbasis Artificial Intelligence,” *Informatics for Educators And Professionals : Journal of Informatics*, vol. 8, no. 2, pp. 148–157, 2023.
 - [8] S. H. Park, S. B. Yu, J. A. Kim, and H. Yoon, “An All-in-One Vehicle Type and License Plate Recognition System Using YOLOv4,” *Sensors*, vol. 22, no. 3, Feb. 2022, doi: 10.3390/s22030921.
 - [9] B. H. Y. S. H. Dinda Rizki Taningrum, “Sistem Pengidentifikasian Plat Nomor Kendaraan Mobil Menggunakan Principal component Analysis Dan Klasifikasi K-NN,” *e-Proceeding of Engineering*, vol. 3, pp. 1868–1876, Aug. 2016.
 - [10] L. R. Mursari and A. Wibowo, “The Effectiveness of Image Preprocessing on Digital Handwritten Scripts Recognition with The Implementation of OCR Tesseract,” *Computer Engineering and Applications*, vol. 10, no. 3, pp. 177–186, 2021.
 - [11] N. D. Cahyo, “Pengenalan Nomor Plat Kendaraan Dengan Metode Optical Character Recognition,” *Ubiquitous: Computers and its Applications Journal*, vol. 2, pp. 75–84, 2019, doi: 10.51804/ucaiaj.v2i1.75-84.
 - [12] D. Rohpandi, A. Sugiharto, and G. A. Winara, “Aplikasi Pengolahan Citra Dalam Pengenalan Pola Huruf Ngalagena Menggunakan MATLAB,” *Konferensi Nasional Sistem & Informatika*, pp. 772–777, 2015.
 - [13] Kusnantoro, T. Rohana, and D. S. Kusumaningrum, “Implementasi Metode Tesseract OCR (Optical Character Recognition) untuk Deteksi Plat Nomor Kendaraan Pada Sistem Parkir,” vol. III, no. 1, pp. 59–67, 2022.
 - [14] A. Firdaus, S. Kurnia, T. Shafera, and W. I. Firdaus, “Implementasi Optical Character Recognition (OCR) Pada Masa Pandemi Covid-19,” *Jurnal JUPITER*, vol. 13, no. 2, pp. 188–194, 2021.
 - [15] X. Wu *et al.*, “Top 10 algorithms in data mining,” *Knowl Inf Syst*, vol. 14, no. 1, pp. 1–37, 2008, doi: 10.1007/s10115-007-0114-2.
 - [16] V. Ong and D. Suhartono, “Using K-Nearest Neighbor In Optical Character Recognition,” *ComTech*, vol. 7, no. 1, pp. 53–65, 2016.
 - [17] R. M. R. Clinton and R. Sengkey, “Purwarupa Sistem Daftar Pelanggaran Lalulintas,” *Jurnal Teknik Elektro dan Komputer Vol.8*, vol. 8, no. 3, pp. 181–192, 2019.
 - [18] T. Dewangca, A. L. Prasasti, and R. E. Saputra, “Identifikasi Plat Nomor Kendaraan Menggunakan Metode K-Nearest Neighbor,” *e-Proceeding of Engineering*, vol. 7, no. 2, pp. 4650–4658, 2020.
 - [19] Burhanuddin, P. H. Siregar, and M. Rigayatsyah, “Deteksi Plat Nomor Kendaraan Bermotor Menggunakan Algoritma K-Nearest Neighbors (KNN),” *Jurnal Teknologi Terapan & Sains*, pp. 461–469, 2020.