International Journal of
Advanced Science Computing and Engineering

Journal Homepage www.ijasce.org/index.php/ijasce

Efficient Vehicle Registration Recognition System: Enhancing Accuracy and Power Efficiency through Digital Image Processing

R. Madhumitha [a,*]

[a] Department of Electronics and Communication Engineering, St.Joseph's College of Engineering, OMR, Chennai-600119, India
Corresponding author: *madhumitha.r14@gmail.com

***Abstract*— Number-plate recognition technology uses optical character recognition on images to read vehicle registration plates using OpenCV, PyTesseract OCR Engine, and YOLOv4 model. Images of the license plates are provided to the YOLOv4 model using a web app. The model is trained with the help of images from the Kaggle dataset. Then the number plate is extracted from the vehicle using the model. The string from the image is extracted using PyTesseract OCR Engine. Tested on several datasets this method gave us a success rate of 94%. This method can also be used in real-time incidents.**

***Keywords*— License plate; number plate recognition; OpenCV; optical character recognition; pytesseract.**

Manuscript received 15 Oct. 2020; revised 29 Jan. 2021; accepted 2 Feb. 2021. Date of publication 17 Feb. 2021.

I. INTRODUCTION

Number Plate Recognition (NPR) is a technology that utilizes an Optical Character Recognition (OCR) Engine and the YOLOv4 model to accurately 'read' vehicle number plates. In essence, NPR cameras capture images of passing vehicles' number plates. These images are then processed by a computer system to extract details about the vehicles themselves. NPR typically comprises cameras connected to a central computer system. NPR uses cameras mounted in mobile units, built into traffic vehicles, or inside Closed-Circuit Television (CCTV) systems to take pictures of Vehicle Registration Marks, or number plates, as they pass by. The NPR system converts these digital photos into data, which is subsequently examined.

The main components of our suggested approach are OCR operations inside a vehicle image and the YOLOv4 model. Having a car these days is more than just a status symbol—it's a need. But when it comes to cars, unanticipated disastrous events can happen. To prevent accidents, the necessary policies must be put in place to improve security, safety, and vehicle monitoring. By identifying the number plates of vehicles that break traffic laws, NPR can be used to identify them and so increase road safety. In an online Automatic License Plate Recognition (ALPR) framework [6], license plate recognition and interpretation occur in real-time from incoming video frames, facilitating immediate tracking via surveillance cameras. Example: Open ALPR CloudWatch. Conversely, an offline ALPR framework captures images of vehicle number plates, storing them in a centralized data server for subsequent processing, i.e., license plate interpretation. Example: Open ALPR Library.

Vehicle number plate standards are closely guarded in developed countries. Strict adherence is maintained to parameters such as vehicle number plate number of lines, script, font face, size, and color of each character, plate size, color, and space between characters [10]. The pictures below show some examples of standard plates used in industrialized countries. However, the current procedure for registering cars upon entry for guests, employees, or students in many academic institutions and parking lots [8] entails a security guard manually verifying membership details by looking for a membership sticker on the vehicle's windscreen or by examining the driver's identification card. This manual process is tedious, time-consuming, and prone to inaccuracies. Additionally, the backup and sharing of vehicle information are challenging due to the reliance on hard-copy data.

For instance, a city like Bangalore with numerous apartment complexes and gated communities often relies on similar manual verification methods, typically by checking for a membership sticker on the vehicle's windscreen.

Registering unknown or guest vehicles is time-consuming, and many complexes consider it unsafe, as tracking the movement of vehicle occupants becomes challenging once they enter the premises. Security concerns are exacerbated by the risk of car theft, especially in parking lots where vehicles may be left unattended for extended periods [7]. Managing records of vehicles entering and exiting during peak usage times becomes increasingly difficult. Therefore, considering these limitations of the traditional system, we aim to proactively address each of these challenges by developing a more efficient and secure solution. Automatic license plate recognition relies on two crucial technological components: a) Quality of License Plate Recognition Algorithms: The effectiveness of license plate recognition algorithms significantly influences the overall performance of the system. Better algorithms offer higher recognition accuracy, ensuring accurate identification of license plates, faster processing speed, allowing for swift analysis of images and detection of license plates, wider compatibility with varying picture qualities, enabling the system to perform well under different illumination conditions and camera specifications. b) Quality of Image Acquisition (Camera and Illumination Conditions): The quality of image acquisition, including the camera used and the illumination conditions, plays a pivotal role in the success of license plate recognition. High-quality images captured under optimal lighting conditions facilitate more accurate recognition of license plates. c) Varying Indian Number Plate Formats: One significant challenge in license plate recognition systems is the variability in number plate formats across different regions or countries. Typically, an LPR program is designed to recognize plates from a specific nation due to variations in plate geometry, orientation, fonts, and syntax. The geometrical structure of the plate, including character distribution, spacing, color, and dimension ratios, is crucial for accurate recognition. Without prior knowledge of the plate geometry, the algorithm may struggle to locate the plate within the captured image and accurately identify the characters. In summary, effective license plate recognition systems require both advanced algorithms and high-quality image acquisition techniques, while also considering the variability in number plate formats, such as those encountered in India. Understanding the geometrical characteristics of the plates is essential for successful recognition and identification.

II. Material and Method

Number Plate Recognition (NPR) leveraging an efficient OCR engine such as Pytesseract, in conjunction with the YOLOv4 object detection model and OpenCV libraries for image processing, presents a robust solution to various challenges in vehicle monitoring and security [5]. While NPR addresses many existing issues, there is potential to further enhance its scope and capabilities. One prominent challenge in number plate detection is the presence of noise in captured images, stemming from environmental factors or the image capture process itself. Our system aims to mitigate this challenge by offering adaptability to diverse environments, including adverse weather conditions like rain or low light situations. When presenting our solution to potential clients, a common concern is the integration of new features into their existing infrastructure. Our system addresses this concern by offering seamless integration with pre-existing systems, ensuring minimal disruption to operations. Moreover, our web application provides users with the flexibility to submit their images or videos for number plate detection. Additionally, it offers analytics and solutions based on the extracted data, enhancing the overall utility of the system. Advantages of the Proposed System include successful and efficient preprocessing of raw RGB images, utilization of YOLOv4, OpenCV, and PyTesseract frameworks to enhance detection and recognition of vehicle license plates, thereby improving system reliability, accurate determination of number plates based on Indian standards, ensuring compliance with local regulations and ability to correctly identify number plates from other countries, expanding the applicability of the system to a global context. In summary, our proposed NPR system offers numerous advantages, including adaptability to various environments, seamless integration with existing infrastructure, user-friendly web interface, and accurate license plate detection and recognition capabilities. These features position our system as a comprehensive solution for vehicle monitoring and security needs.

NPR is a type of mass surveillance technology that takes images of cars and identifies them by their license plate. This paper proposes a method that uses support vector machines (SVM) to locate and interpret Indian car number plates in digital photos. In this proposed model pre-processing and number plate localization are performed by using ―Otsu's methods and feature-based localization methods respectively. It gives reliability and time optimization. Finally, the character reorganization is performed using the Support Vector Machine [2]. The images of various vehicles have been acquired manually and converted into grayscale images. Then the Wiener2 filter is used to remove the noise present in the plates. The segmentation of grayscale image generated by finding edges using the Sobel filter [9] for smoothing the image is used to reduce the number of connected components and then a Bilateral filter is used to calculate the connected component. Finally, a single character is detected [3].

Images from input files or CCTV video are used. To extract the vehicle number from the input image, the CCTV footage must be clear. After converting these input photos to grayscale, OCR is used to segment and identify the characters. For this software to function, a few prerequisites must be met: i) White license plates must adhere to the guidelines set forth by the Indian government. ii) The image should have a suitable amount of contrast and brightness: This involves the building of software that uses MATLAB to identify the vehicle's license plate number [1].

Signal processing is a discipline in electrical engineering and mathematics that deals with the analysis and processing of analog and digital signals, and deals with storing, filtering, and other operations on signals. These signals include speech or sound signals, image signals, transmission signals, and other signals, among others. Among all these signals, image processing is the area that deals with the kind of signals where the input is an image and the output is also an image. Since taking a picture with a camera involves physical processes, sunlight is employed as an energy source. An array of sensors is employed to capture the image. Thus, when an object is exposed to sunlight, the sensors detect how much light it reflects, and the amount of data they sense creates a continuous voltage signal. We must transform this data into a digital format to produce a digital image. This yields a digital image in the form of a two-dimensional array or matrix of numbers.

Optical Character Recognition (OCR) is a technology that converts images of typed, handwritten, or printed text into machine-encoded text. Scanned documents, images of documents, and scene shots with text on billboards and signage can all be processed using this method. For data entry from a variety of printed paper documents, including business cards, invoices, bank statements, passports, and more, OCR is frequently employed. It makes it possible to digitize printed texts so they can be edited, searched, and compactly stored electronically. OCR also makes it easier for them to be displayed online and used in machine processes like text mining, machine translation, cognitive computing, and text-to-speech conversion. OCR research sits at the nexus of computer vision, artificial intelligence, and pattern recognition. Early OCR systems needed to be trained using character-by-character images and could only process one font at a time. But since then, more sophisticated systems have been developed that can recognize a wide variety of fonts with a high degree of accuracy. These days' systems can replicate structured output that nearly resembles the original document layout, including graphics, columns, and other non-textual elements. They can handle a variety of digital image file types. All things considered, OCR technology is essential to the digitization and information extraction of printed documents, enabling effective data processing and administration. OCR techniques consist of the following: (i) Pre-processing; (ii) De-skew: reorienting scanned texts to guarantee horizontal or vertical alignment; (iii) Despeckle: eliminating positive and negative areas and softening edges for improved recognition(iv) Line removal: To improve text extraction, clean out non-glyph components like boxes and lines. (v) Analysis of layout, or "zoning": recognizes discrete blocks, such as paragraphs, columns, and captions, to enhance identification, especially in intricate layouts. Text recognition (vi) and matrix matching (vii): for typewritten text and fonts that are comparable, this technique, known as "pattern matching" or "image correlation," compares the incoming image to the stored glyph pixel by pixel. (viii) Post-processing: Lexicon constraints, which limit output to a predetermined list of terms and increase OCR accuracy, are helpful in specialist sectors but may have trouble with proper nouns. Maintaining original layout: Sophisticated OCR systems can provide annotated PDFs that contain searchable text and graphics while preserving the original page layout.

OCR (Optical Character Recognition) technology finds diverse applications across various domains. It serves as a crucial tool for efficient data entry in business contexts, facilitating the extraction of information from a plethora of documents such as checks, passports, invoices, bank statements, and receipts. Additionally, OCR is integral to automatic number plate recognition systems deployed for law enforcement and traffic management purposes. In airport settings, OCR enables passport recognition and facilitates the extraction of relevant information. Moreover, OCR is leveraged in insurance document processing to automatically extract key data points. Beyond document-centric applications, OCR is employed for traffic sign recognition and the extraction of contact information from business cards for seamless integration into contact lists. Furthermore, OCR expedites the digitization of printed materials, enabling faster creation of textual versions of documents, and enhancing accessibility through real-time handwriting conversion for individuals with visual impairments. In the digital realm, OCR plays a pivotal role in defeating CAPTCHA systems, while also assisting in the identification of CAD images and the creation of searchable PDFs from scanned documents. Thus, OCR emerges as a versatile technology with far-reaching implications, spanning from enhancing operational efficiency to promoting accessibility and innovation across diverse domains. These applications showcase the versatility and utility of OCR technology in various domains, from document processing and data entry to accessibility and security.

The graphical user interface (GUI) revolutionized computing by offering users an intuitive and visually engaging way to interact with computers and electronic devices. Originating in the late 1970s at the Xerox Palo Alto research laboratory and later commercialized by Apple's Macintosh and Microsoft's Windows operating systems, GUIs resolved the limitations of text-based command-line interfaces, democratizing computing for a wider audience. Key features of GUIs include Direct manipulation of graphical icons like buttons, scroll bars, windows, tabs, menus, cursors, and the mouse. Integration of touchscreen and voice-command interaction capabilities in modern interfaces. Adherence to the model-view-controller software pattern, which segregates internal data representation from user presentation, enhancing system organization and flexibility. Use of visual widgets for user interaction, dynamically responding to user inputs and actions. Flexibility in appearance ("skin") independent from application functions, facilitating easy customization and redesign. Adoption of standard formats for graphics and text representation, enabling seamless data sharing between applications. GUI testing plays a pivotal role in software development, ensuring both functionality and design consistency. This process involves generating test cases to evaluate system behavior and GUI elements comprehensively. GUI testing tools, available in both manual and automated forms, come with various licenses and platform support to assist developers in this endeavor. These tools help identify and resolve any issues with the GUI, thereby ensuring a seamless and enjoyable user experience.

OpenCV (Open-Source Computer Vision Library) is a widely used open-source software library for computer vision and machine learning. Developed to provide a common platform for computer vision applications, OpenCV accelerates the integration of machine perception into commercial products. Thanks to its BSD license, OpenCV offers businesses the flexibility to utilize and modify its code. Key features of OpenCV include Optimized Algorithms: OpenCV boasts over 2500 optimized algorithms, encompassing a wide range of classic and state-of-the-art computer vision and machine learning techniques. These algorithms enable tasks such as face detection and recognition, object identification, human action classification in videos, camera movement tracking, motion object tracking, 3D object modeling, 3D point cloud extraction from stereo cameras, image stitching, image database search, red-eye removal, eye movement tracking, scenery recognition, and augmented reality marker establishment. User Community: With a user community exceeding 47 thousand people and an estimated download count exceeding 18 million, OpenCV enjoys widespread adoption across various industries, research groups, and governmental bodies worldwide. Industrial and Research Applications: OpenCV is utilized extensively in established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, and Toyota, as well as startups such as Applied Minds, VideoSurf, and Zeitera. Its applications range from stitching street view images and surveillance video analysis to robotic navigation, drowning accident detection, interactive art, debris inspection on runways, product label inspection in factories, and rapid face detection. Supported Platforms: OpenCV supports multiple programming languages, including C++, Python, Java, and MATLAB, and is compatible with Windows, Linux, Android, and macOS operating systems.

Real-time Vision Applications: Primarily designed for real-time vision applications, OpenCV leverages MMX and SSE instructions when available. It also offers interfaces for CUDA and OpenCL, which are actively developed to harness GPU acceleration. C++ Native Implementation: OpenCV is written natively in C++ and features a templated interface that seamlessly integrates with STL containers. It offers over 500 algorithms and approximately ten times as many supporting functions. In summary, OpenCV is a versatile and powerful tool for developing computer vision applications, with a wide range of features, extensive user support, and broad industrial and research applications.

Tesseract Engine: Among OCR developers, Tesseract is an open-source OCR engine that has grown in popularity. There have long been strong, free OCR alternatives available, even though they can be difficult to use and adjust at times. Tesseract started as a Ph.D. research project at Bristol's HP Labs. HP focused on developing it further between 1984 and 1994 after it attracted attention. Tesseract was made available as free software by HP in 2005, and Google has been actively working on it since 2006. A text recognition (OCR) engine called Tesseract is licensed under the Apache 2.0 license. It allows for direct usage or, for programmers, via an API to extract printed text from images. Notably, Tesseract supports a wide variety of languages. While it lacks a built-in GUI, several third-party GUI options are available. Moreover, Tesseract is compatible with numerous programming languages and frameworks through wrappers, which can be found on its website. One of Tesseract's strengths is its ability to integrate with existing layout analysis to recognize text within large documents. Additionally, it can be utilized alongside an external text detector to recognize text from an image of a single text line. Overall, Tesseract provides a robust solution for OCR needs, offering flexibility and support for various languages and frameworks. The Tesseract

LSTMs excel at learning sequences but may experience significant slowdowns with large numbers of states. Empirical evidence suggests that LSTMs perform better when learning long sequences rather than short sequences with many classes. Tesseract, which originated from the OCR opus model in Python, was initially a fork of CLSTM, an LSTM implementation in C++. The legacy version of Tesseract (3. x) relied on a multi-stage process, with distinct steps including word finding and line finding. Word finding involved organizing text lines into blobs, analyzing lines and regions for fixed pitch or proportional text, and breaking text lines into words based on character spacing. Recognition then proceeded through a two-pass process: in the first pass, each word was attempted to be recognized individually, with satisfactory words passed to an adaptive classifier for further training. This classifier then had an opportunity to more accurately recognize text lower down the page. The modernization of the Tesseract tool involved code cleaning and the introduction of a new LSTM model. In this updated version, the input image is processed in boxes (rectangles) line by line, which are then fed into the LSTM model for output. The diagram below illustrates this process visually. [Diagram: Visualization of Tesseract processing input image in boxes line by line and feeding into LSTM model for output]

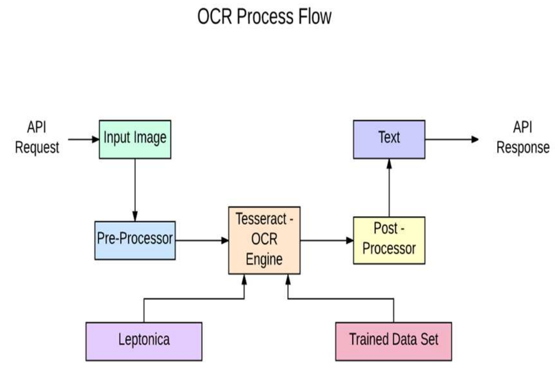

Fig. 1 Tesseract Architecture

III. Results and Discussion

The architecture of a number plate recognition system and an information extractor is depicted in Figure 2. The proposed system is designed to detect vehicle number plates and extract the characters from them. In the vehicle number plate recognition module, the input image undergoes preprocessing, followed by segmentation of characters. The segmented characters are then cropped and recognized. Finally, the recognized characters are assembled into a string. The subsequent module entails a web crawler, which retrieves information about the vehicle. The fetched information is converted into JSON format and stored in a database or presented on a dashboard for further analysis and processing.

A. Number Plate Recognition System

The system captures the input image from a camera feed or an image file. It then undergoes pre-processing, converting the image to grayscale and applying a bilateral filter to smoothen it. Edge detection using the Canny algorithm follows, with contours containing four edges ranked. The contour with the highest rank is cropped as the number plate region. Pytesseract, an OCR tool, performs character recognition in this region.

B. Implementation

The system employs the following algorithms to fulfill functional and non-functional requirements. Algorithm for Number Plate Recognition: Input: Image file uploaded from the camera, Output: Vehicle number plate characters. The process begins with reading or capturing the original image, which is then resized and converted to grayscale for further analysis. A Bilateral Filter is applied to the grayscale image to reduce noise while preserving edges. Canny edge detection is then employed to highlight prominent edges. Subsequently, the algorithm identifies and sorts the top 30 contours, focusing on those with four corners indicative of potential text regions. Other parts of the image are masked out, revealing the final processed image. Utilizing Tesseract OCR, the text within the selected regions is extracted. Finally, the extracted text is standardized to adhere to the format of Indian vehicle number plates, thus completing the process. Upon uploading the image file, the number plate recognition system executes its functions to generate the output. The experimental results are shown in figure 3.

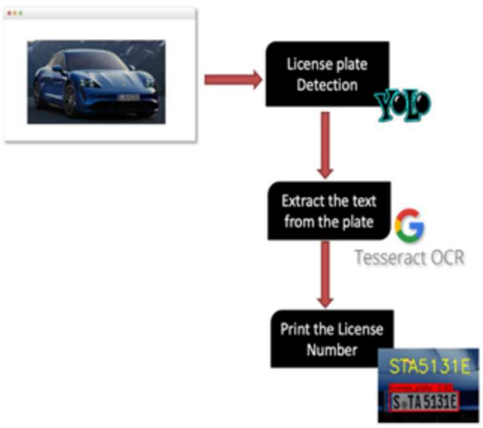

Fig. 2 System Architecture

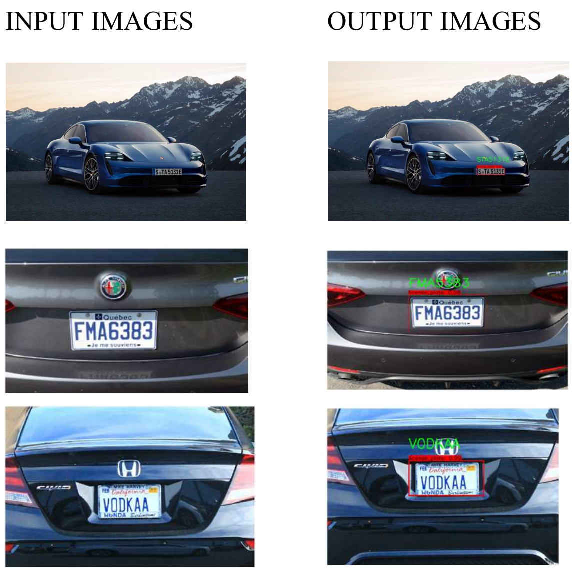

Fig. 3 Experimental results

Testing verifies that the software system is error-free by comparing the actual results to the predicted results. To assess one or more properties of interest, software or system components must be executed. Software testing also aids in finding inconsistencies, holes, or missing requirements between the required requirements and the actual requirements. It is an important stage in the life cycle of product development and can be carried out manually or with the aid of automated tools. To ensure software reliability and quality assurance, testing is essential. Every test has a specific purpose, but taken as a whole, they all confirm that all system components are correctly integrated and carry out their assigned tasks. In the end, testing makes sure that the product works exactly as planned. It represents the final verification and validation activity within the development environment.

IV. Conclusion

This work presents a system capable of recognizing vehicle registration numbers through digital image processing. The system has yielded several noteworthy results: Identification of blacklisted vehicles, efficient traffic monitoring by a single user, facilitates the detection of traffic violations, easy storage and transfer of data, enhancing system efficiency. Because of the system's modular design, changing or upgrading its various sub-modules is a simple process. Its versatility for a broad range of vision applications is improved by its modularity. Furthermore, its performance makes it a competitive option, especially when budget is of the utmost importance. Its adaptability and versatility are further increased by the modular architecture. Because digital image processing was used, the designed recognition system performs more accurately and efficiently than previous approaches, with a 94% accuracy rate under typical conditions. This work focuses on automatic vehicle license plate recognition and emphasizes improving system efficiency, particularly in terms of power consumption. Successful implementation of power-saving measures will significantly enhance its utility in traffic management and security systems, including automobile theft prevention and parking lot management. Initial software algorithm implementations have demonstrated promising results. The system's robustness can be further improved by employing high-precision cameras to enhance overall accuracy in real-time applications. Additionally, the system's power consumption can be reduced by designing sensors to trigger image capture only when necessary.

Nomenclature

ALPR	Automatic License Plate Recognition
CAD	Computer-Aided Design
CCTV	Closed Circuit Television
CUDA	Compute Unified Device Architecture
GPU	Graphics Processing Unit
GUI	Graphical User Interface
MMX	Multimedia Extension
NPR	Number Plate Recognition
OpenCV	Open-Source Computer Vision
OCR	Optical Character Recognition
SVM	Support Vector Machine
YOLOv4	You Only Look Once

Acknowledgment

The author has contributed entirely to this paper.

Conflict of Interest

The author has no conflict of interest.

REFERENCES

[1] Abhay Singh, Anand Kumar Gupta, Anmol Singh, Anuj Gupta, Sherish Johri, “Vehicle Number Plate Detection Using Image Processing”, Department of IT, Volume: 05 Issue: 03 | Mar-2018

[2] Amninder Kaur, Sonika Jindal, Richa Jindal “License Plate Recognition Using Support Vector Machine (SVM)” Dept. Of Computer Science, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 7.

[3] Anish Lazarus, Siddhartha Choubey, Sinha G.R.,” An Efficient Method of Vehicle Number Plate Detection and Recognition” Department of Computer Science, International Journal of Machine Intelligence, Volume 3, Issue 3.

[4] Ganesh R. Jadhav, Kailash J. Karande, “Automatic Vehicle Number Plate Recognition for Vehicle Parking Management System”, IISTE, Vol.5, No.11, 2014.

[5] Hsieh, Jun-Wei, Shih-Hao Yu and Yung-Sheng Chen. "Morphology-based license plate detection from complex scenes." Object recognition supported by user interaction for service robots. Vol. 3. IEEE, 2002.

[6] Wikipedia, https://en.wikipedia.org/wiki/Automatic_number-plate_recognition

[7] Jin, Lisheng, et al. "License plate recognition algorithm for passenger cars in Chinese residential areas." Sensors 12.6 (2012): 8355-8370.

[8] Mutua Simon Mandi, Bernard Shibwabo, Kaibiru Mutua Raphael, “An Automatic Number Plate Recognition System for Car Park Management”, International Journal of Computer Applications, Volume 175 – No.7, October 2017

[9] Wenshuo Gao, Xiaoguang Zhang, Lei Yang, and Huizhong Liu, "An improved Sobel edge detection," 2010 3rd International Conference on Computer Science and Information Technology, 2010,

[10] Yan, Qing. "Method of license plate location based on license plate texture and HSV color space." Information Engineering and Applications. Springer, London, 2012. 962-970.