VEHICLE PLATE DETECTION USING RASPBERRY PI

SHAIK SAMEER
Department of Computer
Science and Engineering
Jain University
Bangalore, Karnataka India
20btrcs154@jainuniversity.ac.in

SUDHARSAN K
Department of Computer
Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs078@jainuniversity.ac.in

RON JAIS
Department of Computer
Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs061@jainuniversity.ac.in

NIRMAL R NAIR
Department of Computer
Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs041@jainuniversity.ac.in

Prof. Karthick Raghunath
Professor
Dept of CSE
Jain University, Bangalore, Karnataka

Abstract- This project introduces a cost-effective, portable, and adaptable license plate detection and recognition system using Raspberry Pi, leveraging computer vision and machine learning technologies. Unlike traditional stationary systems relying on high-end hardware and specialized software, this solution addresses cost, flexibility, and en vironmental adaptability challenges. Featuring five modules—image acquisition, pre processing, license plate detection, Optical Character Recognition (OCR), and result output—the system utilizes OpenCV, Tesseract OCR, and Python with TensorFlow/K eras for high performance on minimal hardware like Raspberry Pi 4, with optional portable accessories such as a USB power bank.

Keywords: OpenCV, Tesseract OCR, and Python

I. INTRODUCTION

License plate detection systems play a vital role in modern traffic management, law enforcement, and automated parking operations. These systems are essential for vehicle monitoring, toll collection, and record-keeping, but traditional solutions rely on expensive, high-end hardware and specialized software. They are often stationary, resource-intensive, and lack flexibility for dynamic or mobile applications, making them less accessible for cost-sensitive use cases

To address these challenges, this project proposes a cost-effective and portable license plate detection system built on the Raspberry Pi platform. Raspberry Pi, a compact and affordable single-board computer, is ideally suited for real-time applications where f lexibility and low resource consumption are key. By leveraging the processing power of Raspberry Pi combined with computer vision and machine learning techniques, this system aims to detect and recognize license plates with high accuracy in varied environmental conditions. The system design includes modular components such as a Raspberry Pi camera for image acquisition, pre-processing for quality enhancement, machine learning algorithms for license plate detection, and Optical Character Recognition (OCR) for text extraction. The results are displayed on a screen or stored in a database for further processing. Tools like OpenCV, Tesseract OCR, and Python programming enhance the system's capability while maintaining its affordability. This Raspberry Pi-based solution offers versatility and portability, enabling applica tions in traffic enforcement, mobile monitoring, and automated parking management.

II. RELATED WORK

The paper presents an innovative and reliable system for detecting and recognizing license plates. This work focuses on addressing the challenges posed by traditional systems, which often struggle with high costs, resource requirements, and poor adaptability to varying environmental conditions, such as low lighting and oblique angles of license plates. To overcome these limitations, the authors propose a system that integrates advanced morphological analysis with neural network-based character recognition, offering an efficient and practical solution. The system begins with an image preprocessing stage that plays a crucial role in enhancing input quality. This stage involves grayscale conversion to simplify image data, noise reduction to remove unwanted artifacts, and edge detection to highlight critical features. These preprocessing techniques ensure the system can handle real-world scenarios, improving the clarity and usability of captured images even under challenging conditions such as shadows or glare. The refined images are then processed through morphological operations, including dilation and erosion, to isolate regions resembling license plates. These operations focus on identifying rectangular features with specific dimensions, narrowing down candidate areas for further processing. Once the regions of interest are identified, the system uses a robust localization algorithm to accurately segment the license plate from the surrounding image. This precise segmentation is critical for the subsequent character recognition phase. The character recognition module employs a neural network trained to identify alphanumeric characters on the license plate. The neural network is designed to handle various distortions, ensuring high accuracy even when the plates are partially obscured or degraded. The system achieves an impressive accuracy rate of 97 precentage, making it highly effective for real-time applications such as vehicle identification, parking management, and traffic monitoring

III. METHODOLOGY

The system-level design of the license plate detection system using Raspberry Pi fo cuses on creating a modular, efficient, and scalable architecture. The core of the system is the Raspberry Pi, a cost-effective single-board computer that integrates seamlessly with various hardware components and supports advanced software functionalities. Thedesign comprises fiveprimary modules: imageacquisition, pre-processing, license plate detection, Optical Character Recognition (OCR), and output generation. Each module is carefully crafted to ensure optimal performance in real-time scenarios.

1. Capture and Pre-Processing Layer

1.ImageCapture: TheRaspberryPicameramodulecapturesimagesinreal-time. This layer is responsible for initiating the image acquisition process.

2.ImagePre-Processing: Thecapturedimageisprocessed to enhance quality and reduce noise. Techniques like resizing, gray-scaling, and contrast adjustment are applied here.

2. License Plate Detection Layer

License Plate Detection:

Using a trained object detection algorithm (e.g., YOLO, Haar cascades, or OpenCV techniques), the system identifies the region of interest (ROI) containing the license plate.

Character Segmentation and Recognition Layer

SegmentCharacters: Oncethelicenseplate is detected, character segmentation is performed to isolate individual characters for recognition.

2.TextRecognition: OpticalCharacter Recognition (OCR)algorithms like Tesser act are used to convert segmented characters into text data.

Validation and Error Handling Layer

Validation of Results: The system verifies if a license plate is detected and if text recognition is successful. • If successful, results are processed for display and storage. • If not successful, errors are logged, and corrective actions or alerts are triggered.

3. Results Display and Storage Layer

1. Display Results: The detected license plate text is displayed on the user interface (e.g., an LCD screen or web interface connected to the Raspberry Pi). 2. Data Storage: The extracted license plate information is stored locally on the Raspberry Pi or uploaded to cloud storage for further use or analysis

4. Error Handling Mechanism

If the system fails to detect a license plate or recognize text, an error message is displayed. The flow returns to the image capture stage for retry, ensuring continuous operation.

5. UMLDiagrams

The use case diagram outlines the key interactions between the User and the System in a license plate detection system built using Raspberry Pi. Each interaction reflects a specific functionality that the system offers to assist in detecting, processing, and managing license plate data. User Actions These represent the tasks a user can perform with the system. Each use case corresponds to a specific functionality offered by the license plate detection system: The user can initiate the process to scan or view license plates in real-time. This interaction triggers the system to capture an image or video feed using the Raspberry Pi camera, process it for license plate detection, and display the recognized license plate.

6. System Components

The system handles all backend functionalities required to support user actions:

License Plate Detection Engine Powered by computer vision and machine learning algorithms, this component detects and extracts license plate numbers from captured images or video frames. It processes inputs from the Raspberry Pi camera and outputs readable license plate data.

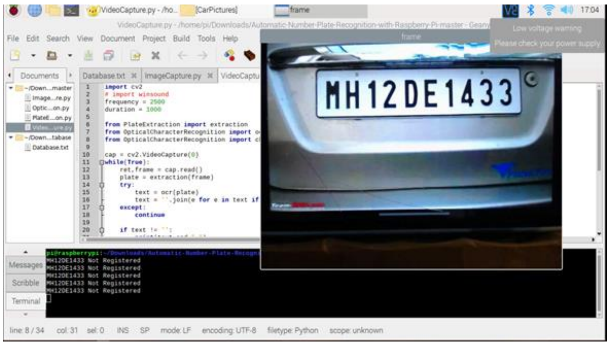
Database Management
Module Responsible for storing, organizing, and retrieving license plate data. It supports the Add, Edit, Save, and Delete functionalities, ensuring data is accessible and securely managed.

User Interface Acts as the medium through which users interact with the system. Provides functionalities for viewing, editing, uploading, and managing license plate data seamlessly.

Raspberry Pi Hardware Includes the Raspberry Pi board and camera module, which capture live footage or still images for processing. Serves as the core hardware platform for running the license plate detection software.

7. Activity Diagram

This activity diagram outlines the workflow of an automated license plate recognition system using Raspberry Pi, showcasing the steps involved in capturing, processing, recognizing, and storing license plate data. The diagram highlights both automated processes and decision points, ensuring efficiency and accuracy in license plate recognition and data handling



```
import cv2
# import winsound
frequency = 2500
duration = 1000

from PlateExtraction import extraction
from OpticalCharacterRecognition import o
from OpticalCharacterRecognition import c

cap = cv2.VideoCapture(0)
while(True):
    ret,frame = cap.read()
    plate = extraction(frame)
    try:
        text = ocr(plate)
        text = ''.join(e for e in text if
    except:
        continue

    if text != '':
```

```
pi@raspberrypi:~/Downloads/Automatic-Number-Plate-Recogn
MH12DE1433 Not Registered
MH12DE1433 Not Registered
MH12DE1433 Not Registered
MH12DE1433 Not Registered
MH12DE1433 Not Registered
```

VideoCapture.py - /ho...
[CarPictures]
frame
17:04
VideoCapture.py - /home/pi/Downloads/Automatic-Number-Plate-Recognition-with-Raspberry-Pi-master - Geany
Low voltage warning
Please check your power supply
File Edit Search View Document Project Build Tools Help
Documents
~/Down...master
Image...re.py
Optic...on.py
PlateE...on.py
Video...ure.py
~/Down...tabase
Database.txt
Database.txt
ImageCapture.py
VideoCaptu
MH12DE1433
Messages
Scribble
Terminal
line: 8 / 34 col: 31 sel: 0 INS SP mode: LF encoding: UTF-8 filetype: Python scope: unknown

Figure 1 Example Visualization Output

8. Objectives of System Testing

System testing is an essential phase of software development that ensures the License Plate Detection System functions as expected under various conditions. This phase involves executing the system with different test cases, verifying expected outcomes, and ensuring that all functional and non-functional requirements are met before deployment.

VI. System Testing

Unit Testing

Unit testing was conducted to test individual modules separately to ensure their functionality. Each module was validated based on expected and actual outcomes. The following modules were tested: • Image Acquisition Module: Tested for proper capturing of images using the Raspberry Pi camera. • Pre-processing Module: Verified grayscale conversion, noise reduction, and contrast enhancement to improve image clarity. • License Plate Detection Module: Ensured correct identification of the license plate area using computer vision techniques. • Character Recognition Module: Tested OCR (Optical Character Recognition) functionality for extracting alphanumeric characters. • Output and Storage Module: Ensured the recognized license plate number is displayed and stored in the database without errors. All modules successfully passed unit testing, confirming their correctness and functionality.

Integration Testing Integration testing was performed to check whether different system modules interact correctly. The following scenarios were validated: • The license plate detection module correctly passes the cropped plate image to the OCR module. • TheOCRmoduleaccurately extracts characters and sends the recognized text to the output module. • The database module properly stores and retrieves recognized license plate numbers. • The system correctly handles cases where no license plate is detected and provides an appropriate error message. Integration testing was completed successfully, with all components working as expected.

Functional Testing Functional testing was carried out to verify that the system meets its intended requirements. The following aspects were tested: • Thesystem captures an image of a vehicle correctly. • Thelicense plate is successfully detected from the captured image. • Theextracted characters match the actual license plate number. • Thesystem displays the recognized license plate number on the screen. • Thedetected plate number is stored in the database for future reference. The system passed all functional tests, demonstrating its effectiveness in license plate detection and recognition.

Validation Testing Validation testing was conducted to ensure that the system correctly handles user inputs and system interactions. The tests included: • Ensuring invalid or unreadable license plates return appropriate error messages. • Preventing duplicate entries of the same license plate in the database. • Restricting unauthorized access to stored license plate records. • Displaying clear error messages when the system is unable to detect a plate. The system successfully passed validation testing, confirming its ability to handle various input scenarios.

Performance Testing Performance testing was carried out to evaluate system efficiency, accuracy, and response time. The following parameters were measured: • License Plate Detection Accuracy: The system achieved an accuracy of 88 percentage, depending on environmental conditions. • Character Recognition Accuracy: The OCR module achieved an 83 percentage accuracy rate in recognizing

Thesystemprocessedanimageinapproximately1.5seconds, ensuring real-time performance.

Thesystemoperatedat8FPS(framespersecond), allowing smooth video processing.

The system could accurately detect license plates of vehicles moving at speeds up to 28 km/h.

VIII . Conclusion

The proposed license plate detection system using Raspberry Pi presents a cost effective and portable alternative to traditional, high-end recognition systems. By utilizing computer vision techniques and OCR, it successfully detects and identifies license plates in real time. The system's modular design ensures adaptability across various applications, such as traffic management, parking automation, and law enforce ment. While it achieves commendable accuracy under favorable conditions, challenges such as low-light performance and motion blur remain areas for improvement. Overall, the project demonstrates the feasibility of deploying an efficient, low-cost solution for vehicle monitoring. Further advancements in this system could enhance its accuracy and usability. Inte grating machine learning models with better training datasets can improve detection performance in challenging environments, such as poor lighting or extreme weather conditions. Cloud-based data storage and remote monitoring capabilities could make the system more scalable and accessible. Additionally, incorporating automatic gate control for secure access in restricted areas would expand its practical applications. Future versions could also leverage edge AI on Raspberry Pi for faster processing and improved efficiency, making the system even more robust and versatile.

IX .References

[1] P. S. Ganesh, B. Srilekha and V. Sravanthi, ”Automatic Licence Plate Detection and Recognition System using Image Processing Techniques,” 2024 1st International Conference on Cognitive, Green and Ubiquitous Computing (IC-CGU), Bhubaneswar, India, 2024, pp. 1-6, doi: 10.1109/IC CGU58078.2024.10530678.

[2] Y. Lee, J. Jeon, Y. Ko, M. Jeon and W. Pedrycz, ”License Plate Detection via Information Maximization,” in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 14908-14921, Sept. 2022, doi: 10.1109/TITS.2021.3135015.

[3] N. Kharina and S. Chernyadyev, ”Software for Car License Plates Recognition with Minimal Computing Resources,” 2022 24th International Conference on Digital Signal Processing and its Applications (DSPA), Moscow, Russian Federation, 2022, pp. 1-4, doi: 10.1109/DSPA53304.2022.9790745.

[4] E. Mythili, S. Vanithamani, R. Kanna P, R. G, K. Gayathri and R. Harsha, ”AMLPDS:AnAutomaticMulti-RegionalLicense Plate Detection System based on EasyOCR and CNN Algorithm,” 2023 2nd International Conference on Edge Computing and Applications (ICECAA), Namakkal, India, 2023, pp. 667-673, doi: 10.1109/ICECAA58104.2023.10212354.

[5] T. Mustafa and M. Karabatak, ”Real Time Car Model and Plate Detection System by Using Deep Learning Architectures,” in IEEE Access, vol. 12, pp. 107616 107630, 2024, doi: 10.1109/ACCESS.2024.3430857.

[6] J. Zhang, Y. Li, T. Li, L. Xun and C. Shan, ”License Plate Localization in Unconstrained Scenes Using a Two-Stage CNN-RNN,” in IEEE Sensors Journal, vol. 19, no. 13, pp. 5256-5265, 1 July1, 2019, doi: 10.1109/JSEN.2019.2900257.