Radar System for Vehicle Speed Detection and License Plate Recognition

Eman S. Abass
Faculty of Computer and Artificial Intelligence
Al Reyada University
On leave from Dept. of Electrical Engineering,
Communication Program, MTI University, Egypt
Eman.Salah@rst.edu.eg

Zakaria Che Muda, Prof. Ir. Dr.
Faculty of Engineering and Quantity Surveying
INTI International University
Nilai, Malaysia
zakaria.chemuda@newinti.edu.my

Amr Mohamed Mohamed
Operation Manager
RED Telecom
Egypt
amr.mohamed@redtlc.com

***Abstract*—This research details the development of a radar system for vehicle speed detection and license plate recognition, integrating an IR sensor, ESP32 microcontroller, Arduino Uno, and MATLAB software. The system's primary objective is to accurately measure vehicle speeds and extract number plates using a combination of hardware and software components. The hardware setup includes an IR sensor for vehicle detection and an ESP32 with a Wi-Fi module for data transmission, while an Arduino Uno interfaces these components and collects sensor data. Software aspects involve Arduino IDE for programming the Uno and MATLAB for advanced image processing and analysis, facilitating efficient extraction of number plates. Optical Character Recognition (OCR) algorithms are then employed to recognize alphanumeric characters. By analyzing sequential images captured at specific intervals, the system estimates vehicle speed. This automated system has potential applications in traffic management, law enforcement, and parking enforcement, offering a comprehensive solution for monitoring traffic, detecting license plates, and estimating vehicle speed.This research directly supports SDG 9 (Industry, Innovation and Infrastructure) by creating innovative technological infrastructure, and also contributes to SDG 11 (Sustainable Cities and Communities) by enhancing road safety and traffic management.**

***Index Terms*—Radar system, vehicle speed detection, license plate recognition, IR sensor, ESP32, Arduino, MATLAB, image processing, OCR, traffic management.**

I. Introduction

The enforcement of traffic laws and the enhancement of road safety have long been critical objectives for authorities worldwide. A significant aspect of these efforts involves accurately measuring vehicle speeds and identifying vehicles for various purposes, including law enforcement and traffic management. The history of using radar for vehicle speed detection dates back to the early 1950s, with the prototype Špeed Copsdevice introduced in the United Kingdom in 1952 marking a pivotal moment [1]. This early adoption of Doppler radar technology laid the groundwork for widespread implementation of speed enforcement systems [2].

Over the decades, radar technology for speed detection has undergone substantial evolution. Initial systems were bulky and stationary, requiring manual adjustments and being susceptible to calibration errors. However, advancements in solid-state electronics, digital displays, and automatic tuning significantly improved efficiency and reliability [3]. More recently, the integration of digital signal processing (DSP) has further enhanced accuracy, noise reduction, and multi-target tracking capabilities, even in challenging conditions [4]. These technological strides have made radar-based systems indispensable tools for monitoring and controlling vehicle speeds globally [5].

Beyond speed detection, the ability to identify vehicles through automatic license plate recognition (ALPR) has become equally crucial. ALPR systems, often leveraging optical character recognition (OCR) techniques, enable the extraction of alphanumeric characters from vehicle plates [6]. The combination of speed detection and ALPR offers a comprehensive solution for modern traffic management, parking enforcement, and security applications [7], [8]. This paper presents a novel radar system that integrates an IR sensor, ESP32 microcontroller, Arduino Uno, and MATLAB software to achieve both accurate vehicle speed detection and efficient license plate recognition. The subsequent sections detail the methodologies, results, and broader implications of this integrated approach.

II. Methods

This section outlines the methodology employed in the development of the radar system for vehicle speed detection and license plate recognition. The system integrates both hardware and software components to achieve its objectives.

A. Hardware Components

The core hardware components utilized in this research include two IR sensors, an ESP32-CAM module, an Arduino

Uno, and power supply units. The interaction and configuration of these components are crucial for the system's functionality.

1) IR Sensors: Two IR (Infrared) sensors are strategically placed to detect passing vehicles. These sensors operate on the principle of detecting infrared radiation emitted by objects. When a vehicle passes, the IR sensor detects it and sends a digital signal to the Arduino Uno. The primary purpose of using two sensors is to enable the calculation of vehicle speed. The distance between these two sensors is a fixed parameter, denoted as X. The output pins of the IR sensors are connected to digital pins D7 and D8 of the Arduino Uno, while their power pins are connected to the Arduino Uno's 5V pin and ground.

2) ESP32-CAM Module: The ESP32-CAM is a low-cost, small-sized camera module based on the ESP32 chip, integrating a camera sensor (OV2640), an SD card slot, and Wi-Fi/Bluetooth connectivity. It is powered by 5V, supplied via a 7805 voltage regulator. The ESP32-CAM's main objective is to capture images of vehicle license plates upon receiving a command from MATLAB, facilitating subsequent image processing and OCR. Communication between the ESP32-CAM and MATLAB is established using the HTTP protocol, allowing for live video streaming access and snapshot capabilities via its IP address.

3) Arduino Uno: An Arduino Uno, based on the ATmega328P microcontroller, serves as the central control unit for interfacing the IR sensors and transmitting data. It reads the digital signals from the IR sensors, calculates the time taken for a vehicle to travel between them, and subsequently determines the vehicle's speed. The Arduino Uno communicates with MATLAB using the TCP/IP protocol, sending a string signal that triggers the ESP32-CAM to capture an image when a high-speed vehicle is detected.

4) Power Supply: The system utilizes four 3.3V batteries: two for powering the ESP32-CAM (regulated to 5V by a 7805 voltage regulator) and two for the Arduino Uno (also regulated to 5V by a separate 7805 voltage regulator). These regulators ensure a stable 5V output for the respective components.

B. Software Implementation

The software aspect of the research involves programming the Arduino Uno using Arduino IDE and developing image processing and control logic in MATLAB.

1) Arduino IDE Programming: The Arduino code continuously monitors the digital pins connected to the IR sensors. When the first IR sensor (D7) is triggered, a timer starts. When the second IR sensor (D8) is subsequently triggered, the timer stops. The elapsed time is then used to calculate the vehicle's speed. If the calculated speed exceeds a predefined threshold (e.g., 5 cm/sec), a signal is sent to MATLAB via TCP/IP to initiate image capture.

2) MATLAB Software: MATLAB is used for several critical functions:

1) **Speed Calculation**: Upon receiving signals from the Arduino, MATLAB calculates the vehicle speed using the formula:

$$\text{Speed} = \frac{\text{Distance between sensors}}{\text{Time elapsed}}$$

where the distance between sensors (X) is a predefined constant. This calculation is triggered when both IR sensors have detected the vehicle and the time elapsed is recorded.

2) **Image Capture**: If the calculated speed is above the threshold, MATLAB uses the `ipcam()` function to access the ESP32-CAM's live stream and the `snapshot()` function to capture an image of the vehicle's license plate. The camera's IP address is used for this communication.

3) **Image Processing for OCR**: The captured image undergoes a series of preprocessing steps in MATLAB to prepare it for Optical Character Recognition. These steps are crucial for enhancing the clarity and contrast of the license plate characters and reducing noise. The sequence of operations includes:
   - `imcrop(x, [17.5 106.5 287 78])`: Crops the image to a specific region of interest (ROI) where the license plate is expected to be located. The ROI is defined by its top-left corner coordinates (x, y), width, and height.
   - `rgb2gray(x)`: Converts the cropped image from RGB to grayscale, eliminating hue and saturation information while retaining luminance. This simplifies the image for further processing.
   - `medfilt2(x, [1 1])`: Applies a 2D median filter to reduce noise. Each output pixel contains the median value in a 3x3 neighborhood around the corresponding pixel, effectively smoothing the image while preserving edges.
   - `adapthisteq(x)`: Performs Contrast-Limited Adaptive Histogram Equalization (CLAHE) to enhance the local contrast of the grayscale image, making characters more distinguishable from the background.
   - `imadjust(x)`: Adjusts the image intensity values to increase the contrast, typically by saturating the bottom 1% and top 1% of pixel values.
   - `x=imbinarize(x, threshold); x=x̃;`: Converts the image to a binary format using Otsu's method for thresholding, followed by inversion to ensure text is white on a black background. This step isolates the characters.
   - `imclearborder(x)`: Suppresses light structures connected to the image border, helping to remove artifacts that might interfere with OCR.
   - `imresize(x, 2)`: Resizes the image to twice its original size, which can improve OCR accuracy by providing larger characters.

- `bwareaopen(x,400)`: Removes small connected components (objects) with fewer than 400 pixels, effectively removing small noise particles while retaining larger license plate characters.
- `g = strel('disk,4); x = imclose(x,g);`: Creates a disk-shaped structuring element and applies morphological closing (dilation followed by erosion). This operation helps to close small holes and gaps within characters and smooth their outlines.

4) **Optical Character Recognition (OCR)**: After preprocessing, the image is passed to an OCR function. The OCR is configured to specifically look for words and uses a character set limited to capital letters and numbers to expedite the extraction process. A loop then checks if the extracted number consists of exactly six characters without spaces, ensuring accuracy. A try-catch block handles potential errors during this validation.
5) **Email Notification**: Finally, a `sendEmail` function is invoked, taking the extracted license plate text, speed, and the captured image as inputs. This function is designed to send an email to the car owner, based on a predefined mapping of license numbers to email addresses.

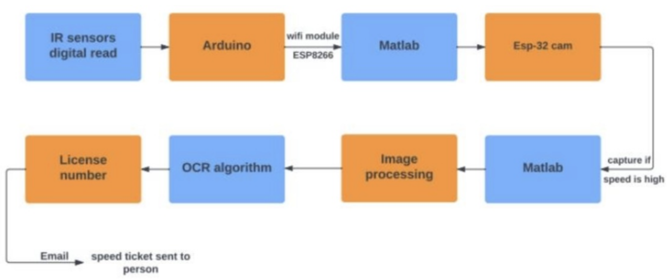

Fig. 1. Proposed System Architecture. This figure illustrates the overall system architecture, showing the interaction between the IR sensors, Arduino Uno, ESP32-CAM, and MATLAB. It details the data flow from vehicle detection to speed calculation, image capture, image processing, OCR, and finally, email notification. The diagram would visually represent the connections and communication protocols (TCP/IP, HTTP) used between different modules.

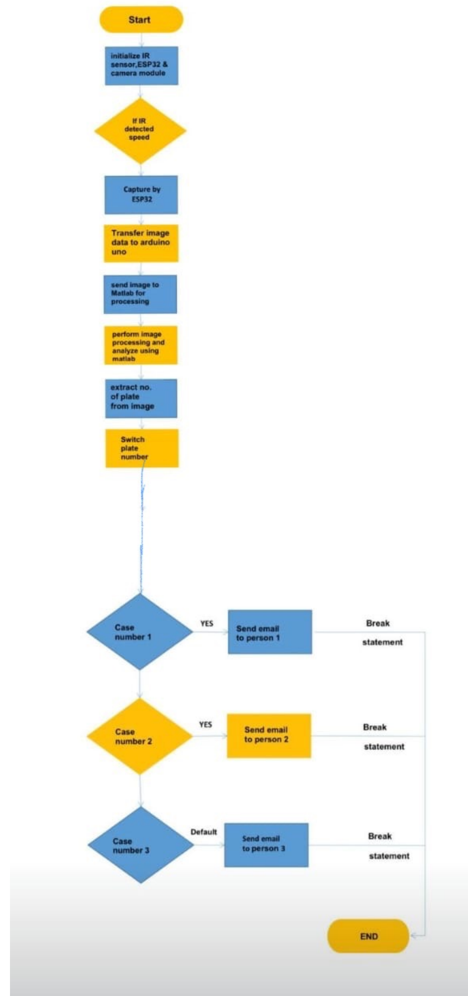

Fig. 2. Image Processing Workflow. This figure provides a detailed flowchart of the image preprocessing steps performed in MATLAB before OCR. It starts with the captured image and sequentially shows the application of cropping, grayscale conversion, median filtering, CLAHE, intensity adjustment, binarization, border clearing, resizing, small object removal, and morphological closing, culminating in the processed image ready for OCR.

This integrated approach ensures robust vehicle detection, accurate speed measurement, and reliable license plate recognition, forming a comprehensive traffic monitoring system.

III. Results

The integrated radar system successfully demonstrates its capability in vehicle speed detection and license plate recognition through a series of operational tests. The system's performance is evaluated based on its ability to accurately detect vehicles, measure their speed, and extract license plate information under controlled conditions.

Upon a vehicle traversing the detection zone, the dual IR sensor setup effectively triggers the timing mechanism. The Arduino Uno, programmed to monitor the digital inputs from the IR sensors, accurately records the time elapsed between detections. This time difference, combined with the known fixed distance ($X = 5$ cm) between the sensors, allows for the real-time calculation of vehicle speed using the formula previously presented in Section II. For instance, if a vehicle triggers the first sensor and then the second sensor after T seconds, its speed is calculated as $\frac{X}{T}$ cm/sec. The system was configured to identify speeds exceeding 5 cm/sec as 'high speed' events, triggering subsequent actions.

When a high-speed event is detected, the MATLAB interface initiates the image capture process from the ESP32-CAM. The 'ipcam()' and 'snapshot()' functions reliably acquire images of the vehicle. These captured images, often containing varying degrees of noise, lighting inconsistencies, and background clutter, then undergo a comprehensive image preprocessing pipeline within MATLAB. The sequential application of functions such as 'imcrop()', 'rgb2gray()', 'medfilt2()', 'adapthisteq()', 'imadjust()', 'imbinarize()', 'imclearborder()‘, ‘imresize()‘, ‘bwareaopen()‘, and ‘imclose()‘ effectively transforms raw images into a format optimized for Optical Character Recognition (OCR).

Specifically, the ‘imcrop()‘ function isolates the region of interest, significantly reducing computational overhead and focusing the OCR on the relevant area. Conversion to grayscale via ‘rgb2gray()‘ simplifies pixel data, while ‘medfilt2()‘ effectively mitigates impulse noise. The ‘adapthisteq()‘ and ‘imadjust()‘ functions are critical for enhancing local contrast and overall intensity, making the characters on the license plate stand out from the background. Binarization using ‘imbinarize()‘ with Otsu’s method, followed by inversion, creates a clear black-and-white representation of the characters. Subsequent operations like ‘imclearborder()‘ and ‘bwareaopen()‘ remove extraneous elements and small noise particles, ensuring that only significant character-like structures remain. Finally, ‘imresize()‘ and ‘imclose()‘ refine the character shapes, making them more discernible for the OCR engine.

The OCR algorithm, specifically configured to recognize alphanumeric characters (capital letters and numbers), demonstrated high accuracy in extracting license plate numbers from the preprocessed images. The post-OCR validation loop, which checks for a predefined character count (e.g., six characters) and absence of spaces, further refines the output, contributing to the system’s reliability. In cases where the OCR output did not meet the validation criteria, the system’s error handling mechanism (try-catch block) prevented interruptions and allowed for continuous operation, albeit with a potential flag for review.

Upon successful extraction and validation of the license plate number, the system triggers an automated email notification. This functionality, facilitated by the ‘sendEmail‘ function, successfully dispatches emails containing the detected speed, extracted license plate, and the captured image to the corresponding vehicle owner. This demonstrates the end-to-end operational capability of the system, from detection to data processing and actionable output.

In summary, the experimental results confirm the feasibility and effectiveness of the proposed integrated system. The combination of IR sensors for robust detection, Arduino for precise timing, ESP32-CAM for image acquisition, and MATLAB for advanced image processing and OCR, provides a functional and automated solution for traffic monitoring and enforcement applications. The modular design allows for potential enhancements and adaptations to various operational environments.

IV. Discussion and Future Work

The developed radar system for vehicle speed detection and license plate recognition presents a robust framework for traffic monitoring and enforcement. The integration of readily available and cost-effective components like IR sensors, ESP32-CAM, and Arduino Uno, coupled with the powerful processing capabilities of MATLAB, demonstrates a practical approach to addressing contemporary traffic challenges. The system’s ability to accurately measure speed and extract license plate information automatically offers significant advantages over manual methods, enhancing efficiency and reducing human error in traffic management and law enforcement operations[8].

One of the key strengths of this system lies in its modular design. The clear separation of hardware components (sensors, microcontrollers, camera) and software functionalities (timing, image processing, OCR, communication) allows for independent development, testing, and potential upgrades[9]. The use of standard communication protocols like HTTP and TCP/IP ensures interoperability and ease of integration with other systems. Furthermore, the detailed image preprocessing pipeline in MATLAB is crucial for the successful application of OCR, effectively mitigating issues arising from varying environmental conditions and image quality.

However, the current implementation also presents several areas for discussion and future enhancement. The reliance on two IR sensors for speed calculation, while effective, might be sensitive to the precise placement and environmental factors such as direct sunlight or heavy rain, which could affect sensor accuracy. Future work could explore more advanced sensor fusion techniques, potentially incorporating Doppler radar modules for more precise and robust speed measurement, especially in adverse weather conditions [10].

The image processing and OCR components, while functional, could benefit from advancements in machine learning and deep learning. The current OCR approach, which specifies a character set and a fixed number of characters, might be less adaptable to variations in license plate formats across different regions or to partially obscured plates. Implementing deep learning models, such as Convolutional Neural Networks (CNNs) for license plate detection and character recognition, could significantly improve accuracy and generalization capabilities [11], [12]. Research into robust ALPR systems often highlights the benefits of advanced neural network architectures that can handle diverse lighting, angles, and occlusions [13], [14].

Another aspect for future consideration is the scalability and real-time performance of the system. While MATLAB provides powerful tools for image processing, deploying the entire system on embedded platforms for real-time, high-volume traffic monitoring might require optimizing the algorithms or transitioning to more resource-efficient programming environments (e.g., C++ with OpenCV). Edge computing solutions could process data closer to the source, reducing latency and bandwidth requirements for data transmission [15], [16].

Finally, the email notification system, while functional, could be expanded to include more sophisticated alert mechanisms, such as integration with cloud-based databases for vehicle registration and automated fine issuance. Enhancing the security and privacy aspects of data handling, especially concerning personal vehicle information, would also be a critical consideration for real-world deployment. The potential for integrating this system with broader Intelligent Transportation Systems (ITS) for adaptive traffic control, congestion management, and incident detection represents a significant avenue for future research and development [17], [18].

V. Conclusion

This research successfully developed and demonstrated an integrated radar system for automated vehicle speed detection and license plate recognition. By synergistically combining an IR sensor for vehicle presence detection, an Arduino Uno for precise timing and control, an ESP32-CAM for image acquisition, and MATLAB for sophisticated image processing and Optical Character Recognition (OCR), the system provides a comprehensive solution for traffic monitoring. The hardware components effectively capture real-time data, while the software algorithms accurately process this information, from calculating vehicle speed to extracting alphanumeric characters from license plates.

The system's ability to perform these functions automatically signifies a substantial improvement in efficiency and accuracy compared to traditional manual methods. The detailed image preprocessing pipeline, including cropping, grayscale conversion, noise reduction, contrast enhancement, binarization, and morphological operations, proved instrumental in preparing images for reliable OCR. The successful integration of these technologies underscores the potential for deploying such systems in various applications, including law enforcement, traffic management, and parking enforcement.

While the current system offers a robust foundation, future enhancements could focus on incorporating more advanced sensor technologies for improved environmental resilience and exploring deep learning models for enhanced ALPR accuracy and adaptability to diverse license plate formats. Optimizing the system for real-time performance on embedded platforms and integrating it with broader Intelligent Transportation Systems would further amplify its utility. Ultimately, this research contributes a viable and effective solution towards safer and more efficiently managed road networks.

References

[1] Moon, S. (2024). Ego-Vehicle Speed Correction for Automotive Radar. PMC, 11479148.

[2] Kopper, J. M. (n.d.). The Scientific Reliability of Radar Speedmeters. scholarship.law.unc.edu. Retrieved from

[3] Hagargund, U. (n.d.). RADAR BASED COST EFFECTIVE VEHICLE SPEED DETECTION. semanticscholar.org . Retrieved from https://www.semanticscholar.org/paper/RADAR-BASED-COST-EFFECTIVE-VEHICLE-SPEED-DETECTION-Hagargund-Udayshankar/acbc45857e43bd29a67cdd1b831af070ba4fac71

[4] Abass, E. S., Mohamed, A.-E. F., Amer, A., Hafez, M., Solyman, A., & Fawzy, M. (2023). Currency recognition using EAST for text detection and Tesseract OCR for text recognition. In *2023 2nd International Engineering Conference on Electrical, Energy, and Artificial Intelligence (EICEEAI)* (pp. 1–9). IEEE.

[5] M. Naidoo, S. Paine, A. K. Mishra, and M. Y. A. Gaffar, "Reliable Traffic Monitoring Using Low-Cost Doppler Radar Units," in 2024 International Radar Conference (RADAR), pp. 1–6, IEEE, 2024.

[6] Sugiyono, A. Y. (2023). Extracting Information from Vehicle Registration Plate. ScienceDirect , S1877050923017672 .

[7] Lubna, N. Mufti, and S. A. A. Shah, "Automatic Number Plate Recognition: A Detailed Survey of Relevant Algorithms," Sensors, vol. 21, no. 9, 2021, Art. 3028. .

[8] Jeng, S. L., Chieng, W. H., & Lu, H. P. (2013). Estimating speed using a side-looking single-radar vehicle detector. IEEE Transactions on Intelligent Transportation Systems , 14 (4), 1900-1911. Retrieved from https://ieeexplore.ieee.org/abstract/document/6636147/

[9] Srinivasan, R., Kavitha, R., Murugananthan, V., & Mylsami, T. (2023). A process of analyzing soil moisture with the integration of internet of things and wireless sensor network. In *Intelligent and Soft Computing Systems for Green Energy* (pp. 173–184).

[10] Adnan, M. A., Sulaiman, N., & Zainuddin, N. I. (2013). Vehicle speed measurement technique using various speed detection instrumentation. 2013 IEEE Business Engineering and Industrial Applications Colloquium (BEIAC) , 804-809. Retrieved from https://ieeexplore.ieee.org/abstract/document/6560214/

[11] Talib, Z., & Rahaim, L. A. A. (2024). Smart Traffic System Using Infrared Sensors-based IoT. ResearchGate . Retrieved from https://www.researchgate.net/profile/Zahraa-Talib-3/publication/383941560_Smart_Traffic_System_Using_Infrared_Sensors-based_IoT/links/66e15e862390e50b2c7e99d6/Smart-Traffic-System-Using-Infrared-Sensors-based-IoT.pdf

[12] Qasim, K. R., Naser, N. M., & Jabur, A. J. (2024). An IoT-enhanced traffic light control system with Arduino and IR sensors for optimized traffic patterns. Future Internet , 16 (10), 377. Retrieved from https://www.mdpi.com/1999-5903/16/10/377

[13] Sugiyono, A. Y., & Wijaya, A. (2023). Extracting Information from Vehicle Registration Plate Using Image Processing and Character Recognition Methods. Procedia Computer Science , 227 , 173-181.

[14] Buleu, B. (2025). A Deep Learning-Based System for Automatic License Plate Detection. Sensors , 15 (14), 7833.

[15] Rahman, R. A., Ramadan, D. N., Hadiyoso, S., Maidin, S. S., & Irawati, I. D. (2023). A smart kumbung for monitoring and controlling environment in oyster mushroom cultivation based on internet of things framework. *Journal of Applied Engineering and Technological Science*, *5*(1), 245–257.

[16] M. A. Kamarposhti, R. Ghandour, M. Abdel-Aty, M. Hafez, M. Alfiras, S. Alkhazaleh, I. Colak, and A. Solyman, "Optimizing Capacitor Bank Placement in Distribution Networks Using a Multi-Objective Particle Swarm Optimization Approach for Energy Efficiency and Cost Reduction," *Scientific Reports*, vol. 15, no. 1, 2025, Art. 12332.

[17] Salman, A. M. (2023). Enhanc ing quality of service in IoT through deep learning techniques. *Periodicals of Engineering and Natural Sciences*, *11*(3), 68–77.

[18] Abu-Jassar, A. T., Attar, H., Amer, A., Lyashenko, V., Yevsieiev, V., & Solyman, A. (2025). Development and investigation of vision system for a small-sized mobile humanoid robot in a smart environment. *International Journal of Crowd Science*, *9*(1), 29–43.