Vehicle detection and alert generation in no parking area

Sanika Kanase,Hase Siddhi,Dond Dhanashree,Jadhav Vedika

How to Cite this Article:

Kanase, S., Siddhi, H., Dhanashree, D. & Vedika, J. (2026). Vehicle detection and alert generation in no parking area. International Journal of Creative and Open Research in Engineering and Management, 02(04). https://doi.org/10.55041/ijcope.v2i4.259

License:

This article is published under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.

© The Author(s). Published by International Journal of Creative and Open Research in Engineering and Management.

https://doi.org/10.55041/ijcope.v2i4.259

ABSTRACT

The No Parking Vehicle Detection System aims to automate the process of identifying vehicles parked in unauthorized or restricted areas using advanced computer vision and deep learning techniques. The system utilizes live streaming from surveillance cameras installed near “No Parking” zones. The captured video is divided into individual frames, which are then subjected to pre-processing operations such as noise reduction, resizing, and normalization to enhance image quality.

A MobileNet-based deep learning model is trained on a large vehicle dataset containing labeled images of cars captured in various lighting and environmental conditions. The region of interest (ROI) is defined to focus the detection process only on the area where parking violations are likely to occur. Once the model identifies an illegally parked vehicle, a decision-making module triggers a buzzer alert to notify nearby authorities and simultaneously sends a digital message to the RTO office with relevant details such as location, time, and vehicle image evidence.

This automated detection system reduces the need for manual monitoring, helps improve traffic discipline, and ensures faster enforcement of parking rules. The integration of IoT, computer vision, and machine learning makes the system efficient, cost-effective, and scalable for deployment in smart city infrastructure. Additionally, the project can be expanded by incorporating license plate recognition and GPS-based mapping for enhanced accuracy and reporting.

Keywords: *Mobile net,Parking system,RTO,Decision making.*

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Parking management has become one of the major challenges in modern cities due to the increasing number of vehicles and limited parking spaces. Illegal parking in “No Parking” areas not only disrupts the smooth flow of traffic but also causes inconvenience to pedestrians, emergency vehicles, and public transport. Manual monitoring of such violations by traffic police is often inefficient and labor-intensive. To overcome these limitations, there is a growing need for an automated and intelligent system that can detect and report parking violations in real-time.

The No Parking Vehicle Detection System utilizes computer vision and deep learning techniques to identify vehicles that are parked in restricted areas. The process begins with live video streaming from surveillance cameras placed in specific locations. The video is divided into frames and pre-processed to remove noise and improve quality. Using a MobileNet model, trained on a labeled vehicle dataset, the system identifies vehicles within the region of interest (ROI) — the defined no-parking zone.

Once the system detects an unauthorized vehicle, it performs decision-making operations to confirm the violation. It then triggers a buzzer alert to inform local authorities or security personnel and sends a notification message to the RTO office with relevant details, such as time, location, and image evidence.

This automated approach significantly enhances traffic law enforcement, reduces the dependency on human monitoring, and ensures timely action against violators. Moreover, the system contributes to the broader goal of developing smart cities by integrating AI, IoT, and real-time surveillance for efficient urban management. In the future, the system can be further improved by incorporating license plate recognition, GPS-based tracking, and cloud-based reporting to provide a more robust and scalable parking management solution.

By automating the detection process, the system reduces the need for manual monitoring and enhances the efficiency of traffic enforcement agencies. It can also be integrated into smart city infrastructure, providing continuous surveillance and supporting urban traffic management. The approach not only saves manpower but also ensures a faster, more reliable, and scalable way to maintain discipline in public parking spaces.

The increasing number of vehicles in urban areas has led to frequent traffic congestion and illegal parking issues, especially in restricted zones. Traditional methods of monitoring such violations rely heavily on manual supervision, which is inefficient and time-consuming. To overcome this, the No Parking Vehicle Detection System provides an intelligent and automated solution using Artificial Intelligence (AI) and Computer Vision technologies.

This project aims to reduce human intervention, enhance traffic rule enforcement, and contribute to smart city development by automating the detection of parking violations. The proposed system is accurate, efficient, and scalable, making it suitable for deployment in busy city areas and public places.

1.2 Problem Definition

Illegal parking in restricted areas causes traffic congestion and inconvenience. Manual monitoring is inefficient and error-prone. Hence, there is a need for an automated system that can detect and alert authorities about vehicles parked in “No Parking” zones using Mobile net model and computer vision.

CHAPTER 2
LITERATURE SURVEY

[1] Adesh Pawar et al., The increase in the overall population and unplanned architectures of cities are the main reasons for decreased parking spaces which is becoming backbone of irritating traffic problems. so, if there is a problem, we need a solution. Hence many researchers and tech enthusiasts have started their work to find worthy solutions in this particular area. The paper gives an elaborative comparison of available research on smart parking systems and talks about various algorithms with their results, pros, and cons. We have tried to give a quick look at the study that we have done throughout our research. Paper provides the classification of research papers based on methodologies used. In this paper we have deployed two algorithms on two datasets and measured their performance in terms of Precision score, Recall score and F1 score . After analyzing all study it is found that the Smart parking system deployed using multiple approaches will be more efficient and have high accuracy in terms of its results. The use of IoT with deep learning algorithms and giving a touch of wireless networking to the core system i.e. object detection and image processing will be more efficient in terms of result but somewhat not in terms of its overall cost. But with the use of IoT, sensor selection will be a major task if one wants to build the system with a low budget. still problems like deem light, occlusion and bad climate needs to be address.

[2] H. Canli et al., Deep learning methods yield successful results in parking space occupancy prediction as in many areas. In this study, a new cloud and deep learning based architecture and a new mobile application are proposed for parking space occupancy prediction. To predict the parking space occupancy rate, a deep learning based LSTM model, which models multi variate and large data sets almost seamlessly, was used. In order to demonstrate the effectiveness of the model, models were created in different parameters and scenarios and compared with SVM, Random Forest and ARIMA models. The results are given in Table 3 and Table 4. When the dataset was trained with a combination of Capacity, Density, Time, Day and Holiday, it resulted in an accuracy rate of 99.57%. This result demonstrates that the architecture and application created yielded successful results. Today, traffic density is the biggest problem of metropolitan cities. Naturally, waiting in front of the car park, exiting from the car park and driving slowly while searching for a parking space are thought to cause this traffic density to increase. In subsequent studies, the effect of car parks on traffic density with different parameters will be examined and a new measure will be presented to reduce traffic density by using deep learning methods. In addition, by realizing reservation-based architecture in parking lots, both traffic density and cost can be reduced. At the same time, the layout of the parking lots can be realized by using deep learning and optimization methods in smart city planning.

[3] In Hwan Jung et al., In this paper, a smart parking management system using AI technique was presented. The implemented system recognizes the vehicle number and uses it as an object ID, and tracks the vehicle by applying YOLO technology. Training and learning algorithms based on CNN deep learning algorithm were applied to detect whether a vehicle was parked or an accident occurred. A number of experiments was conducted to check the detection accuracy and it was confirmed that the deep learning algorithm works effectively after training reasonable number of images. Experimental results show that the detection accuracy of parking and accident detection increases as the number of training images increases. The accident detection needed more training images because it has more diversity. In both experiments, greater that 95% of detection accuracy was observed. The smart parking app allows drivers to easily check parking information and accident information. As a conclusion, the system implemented in this study can be utilized as an AI-based unmanned parking management system.

[4] Joseph, Leobin et al., These system can counter the parking problems that arise due to the unavailability of a reliable, efficient and modern Parking system. Such system can help the economic, social and safety based aspects of the society. It also helps in preserving the environment, fuel and time. The economic analysis can help us find the feasible project so that we can have a better parking system without making the economy suffer.

[5] W. Kim et al., In this study, we proposed a smart parking lot based on edge cluster computing for full self-driving vehicles. The proposed parking lot consists of xed edges using Raspberry Pi as a physical model, and a mobile edge as a full self-driving vehicle. To evaluate the smart parking lots performance, we compared the moving distance and time in a smart parking lot with those in a general parking lot using the RaSim simulator. We also checked battery usage for data transmissions between edges and determined the most effective number of edges for a smart parking lot service. As a result of the experiment, When 13 edge terminals were placed, the smart parking lot reduced moving distance and time by 47% compared to a general parking lot. Therefore, the most effective number of edges for the smart parking lot was 13.

[6] B. Benjdira et al., This study presented a methodology leveraging Deep Learn ing and Image Processing to better analyze and manage parking areas. We began by introducing the importance of this subject in modern cities. Then we summarized the research efforts that intersect with the matter presented. In the third section, we described the methodology and the four main Fig. 4. Heatmap representing the occupancy of every parking slot in seconds during a specific period of time. Fig. 5. Heatmap representing the number of cars that occupied every parking slot during a specific period of time. building blocks constructing it. We formulated the pipeline and the algorithm used to extract occupancy statistics of the parking. Then, in the Experimental Results, we applied this pipeline on a sample video to show a real-time occupancy analysis of the parking. This makes a better visualization of the parking status and helps to better guide entering drivers to choose the right slot for their vehicles. Then, we extract more elaborated insights from the generated video, like curves and heatmaps. To conclude, our solution is to solve one of the most recurring problems that drivers face daily in modern cities. Moreover, our solution is beneficial to solve many problems related to the management of parking areas: space management of the parking areas, avoiding misuse of the parking and long-lasting cars, design of the parking slots in the parking areas. Nevertheless, our efforts need to be improved by working on randomly oriented camera viewpoints. This is a challenging problem because it primarily affects the accuracy of the extracted statistics. Also, the solution can be extended to target covered parking slots or partly occluded parking areas. Solving these two challenges helps better to adopt the solution on a more significant number of parking areas.

[7] Amara Aditya et al., The suggested IPS consists of an IoT framework that transmits real-time data to the cloud and enables us to learn whether parking spaces are available in a specific area. The system also comes with a smartphone app that enables us to look up nearby parking availability and reserve a spot accordingly. The current study also discusses several scenarios in which a person finds a parking spot and parks their vehicle appropriately. The results of the suggested system's implementation utilizing Raspberry-Pi, NodeMCU, RFID, and IR sensors are discussed at the end, concluding that the proposed Intelligent Parking system is correct and efficient.

Future Work - The proposed research can be further enhanced by adding the route direction feature. The route will be displayed to the user directing to the available space and also ensuring the slot remains reserved until the user reaches the location. Further, additional functionality of alarm generation can be added to the proposed system that will detect if the user has parked incorrectly (i.e., partially occupying two slots) and generates a warning alarm.

[8] Vo, Van-An et al., This paper discusses a proposed model for a smart parking system that combines RFID technology and a Deep Learning algorithm. The proposed system is designed with low-cost electronic components to re duce manufacturing costs compared to existing com mercial products while ensuring essential features and stable operation like other systems. Our research team has implemented a proposed algorithm based on deep learning artificial intelligence on the Raspberry Pi 3 embedded computer hardware platform. Experiments have demonstrated the proposed system’s ability to accurately recognize license plate numbers of vehicles such as motorbikes, cars, trucks, etc. The experimental results also show that the proposed smart parking system has been well designed to achieve an overall solution with low cost, satisfactory system performance, and help users save time while reducing the workforce in the parking lot. Furthermore, the price

announced by our research team shows that this is a highly feasible solution that can be fully applied to public parking lot models. Finally, our research team will research to improve the algorithm so that the proposed system can recognize many types of license plates with different background colours, such as blue and yellow, compared to currently only recognizing license plates with a white background colour.

[9] Ming Li et al., This research paper highlights the significance of License Plate Recognition (LPR) in IoT smart parking systems. It explores the use of traditional and deep learning-based methods for LPR detection, with a focus on why deep learning approaches are preferred. The challenges faced by deep learning-based LPR methods, particularly accuracy rate and computation cost, are addressed using insights from previous studies. The proposed method introduces a solution based on the YOLOv6 algorithm to overcome these challenges. The process entails creating a custom dataset, carrying out testing, validation, and training, and assessing the suggested model's performance. The experimental results demonstrate that the suggested method outperforms existing state-of-the-art methods, offering improved accuracy and reduced computation costs. The research contributes to enhancing LPR technology in IoT smart parking systems, enabling more efficient and reliable parking management solutions. For future work, an interesting direction would be to extend the proposed YOLOv6-based LPR system to incorporate real-time tracking of vehicles within the smart parking system. This would enable continuous monitoring and tracking of vehicles, providing valuable information for parking space availability and management. Cross-Domain License Plate Recognition: Another potential avenue for future research is to explore the applicability of the proposed method in cross-domain scenarios. Investigating the adaptation of the YOLOv6-based LPR system to different environments, such as night-time conditions, adverse weather, or different camera viewpoints, would contribute to the robustness and generalization capabilities of the model. This would expand the practical use cases of the system beyond traditional smart parking systems.

[10] Kalyani, G. et al., The proposed system was designed and tested. The proposed system can detect the availability of free parking spot through a real time camera feed. The proposed system collects real time data using camera for processing the parking slot information. The coordinates of parking slot are given first, then the availability is detected. Using the coordinates, we detect the change in the pixels and know if the parking slots are available or not. Although there are a number of existing systems, the proposed system provides a unique feature by using camera instead of regular sensors. This system helps in reducing the cost, manpower, and implementation that which are used in a sensor-based parking system.

CHAPTER 3

SCOPE OF THE PROJECT

No Parking vehicle detection system

3.1 Qtanalytics

Smart parking systems typically get information only regarding accessible parking areas. All smart cities commonly face a key problem related to parking facilities and traffic management systems. Due to vehicles in a no-parking zone, several folksought to face traffic problems. Therefore, to reduce this, projected work combines IoT and cloud. The target of the projected
work is to create a model, "Detection and Identification of Vehicle's No-Parking area using Io and cloud", that helps the drivers to acknowledge the present parked space. No parking area is detected by making use of geolocation of vehicles with the assistance of a real-time cloud server. Projected work does not simply reduce the traffic jam, but it is collectively giving
user authentication, cost-efficient and real-time. This method is enforced by putting geolocation, real-time cloud server, a microprocessor for instant data assortment and no-parking detection mobile-application into service. There will be no wastage time anymore searching for a parking space because the parking space will find them.
Reference URL: https://qtanalytics.in/journals/index.php/JREAS/article/view/4843

CHAPTER 4
METHODOLOGY

The methodology for No Parking vehicle detection system is developed under waterfall model architecture as shown in the below figure 1.

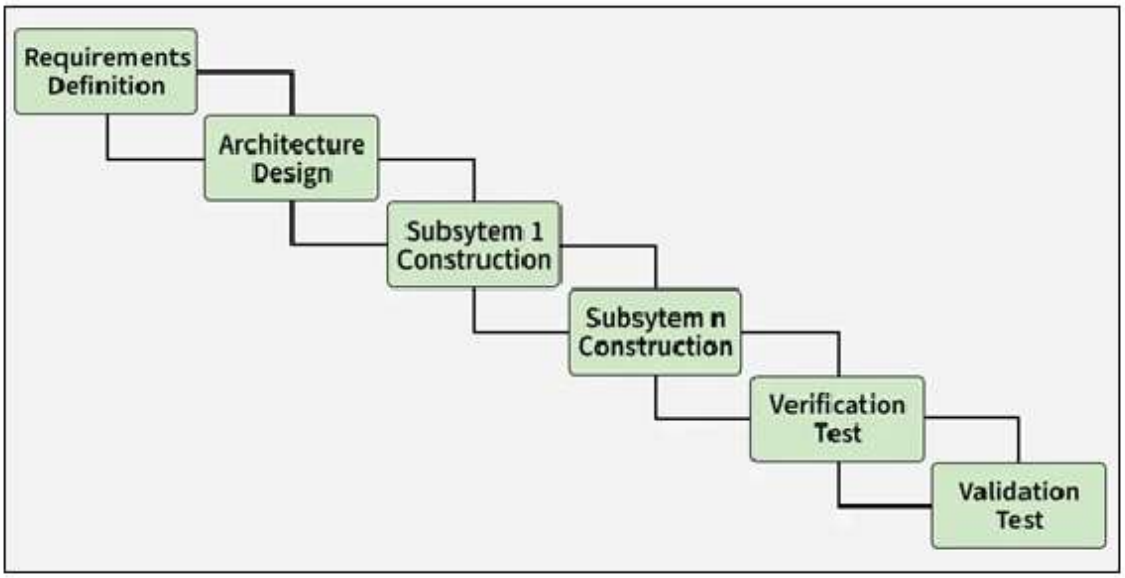


Fig 1: Water fall model Architecture

The sequence phases in water fall model according to our project are mentioned below.

4.1 Requirement Analysis – Here requirement analysis is done based on following points

✓ Base paper for No Parking vehicle detection system

4.2 System Design: The System of No Parking vehicle detection system is designed by using the following hardware and software

Minimum Hardware Specification:

- CPU : Intel Core i5
- RAM : DDR 8 GB
- HDD : 500 GB
- Camera : 48 MP

Software Specification:

Coding Language : Python

- Development Kit : Python 3.8.
- Front End : Tkinter
- Development IDE : Spyder 3.9
- External API : OpenCV, Keras, Tensorflow

4.3 Implementation:

The proposed No Parking vehicle detection system is designed using the following modules:

4.3.1 Module A: Camera, Live image capture

- Instant Frame Collection
- Frame Processing
- Frame normalization
- Extracted Frames

4.3.2 Module B: Mobile net

- Input layer initialization
- Hidden Layer evaluation
- Output Layer estimation
- Trained data

4.3.3 Module C: Preprocessing & Image Normalization

- Image resizing
- Pixel value normalization
- Image conversion
- Normalized and preprocessed image output
- Output - Parking status detection

4.5 Deployment of the System:

The developed software is deployed on a laptop with the specified configuration and integrated with a mobile camera for image capture. The trained MobileNet model is loaded into the system for detecting vehicles parked in no-parking zones in real time. The system processes the captured images and identifies unauthorized parked vehicles to support monitoring and enforcement activities.

4.6 Maintenance of the System:

The system is designed for easy maintenance and efficient operation. Since the software tools and libraries used (such as Python, OpenCV, and the MobileNet deep learning model) are open-source, there are no licensing issues. Regular updates of the model, periodic camera checks, and continuous dataset improvements help maintain accurate detection and ensure reliable long-term operation of the no-parking vehicle detection system.

CHAPTER 5

DETAILS OF DESIGN, WORKING AND PROCESSES

5.1 DETAILS OF DESIGN

5.1.1 Data Flow Diagrams

5.1.1.1 DFD level 0

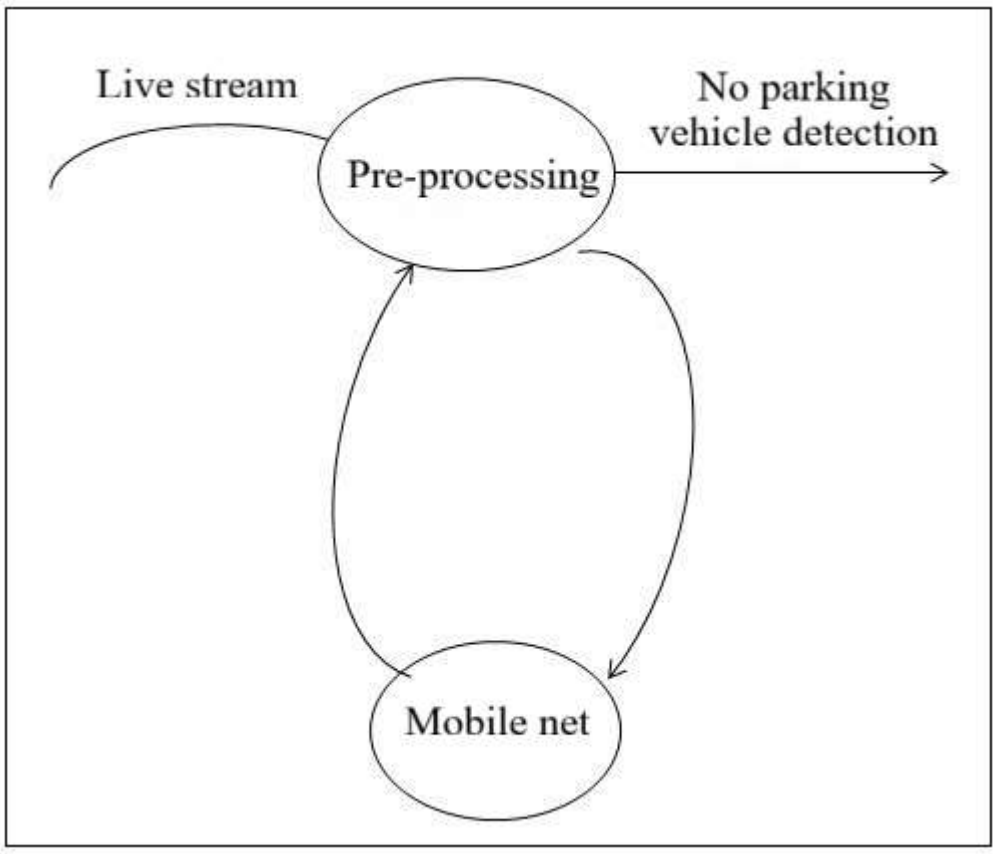
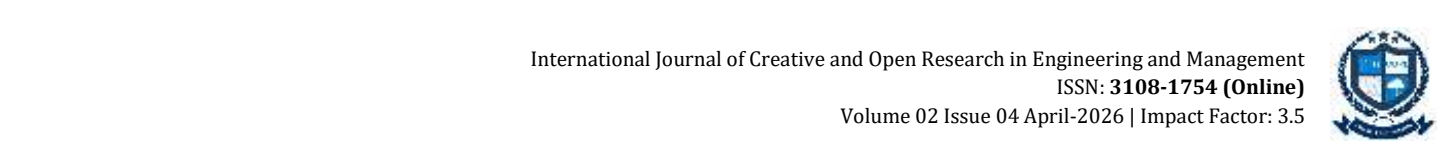


Fig 2 DFD level 0

The DFD 0 diagram for the data flow diagrams describes the flow of the approach. The DFD diagram provides the simplest flow where in the user provides the live image which is preprocessed and MOBILE NET is deployed after which the no parking vehicle detection.



5.1.1.2 DFD level 1

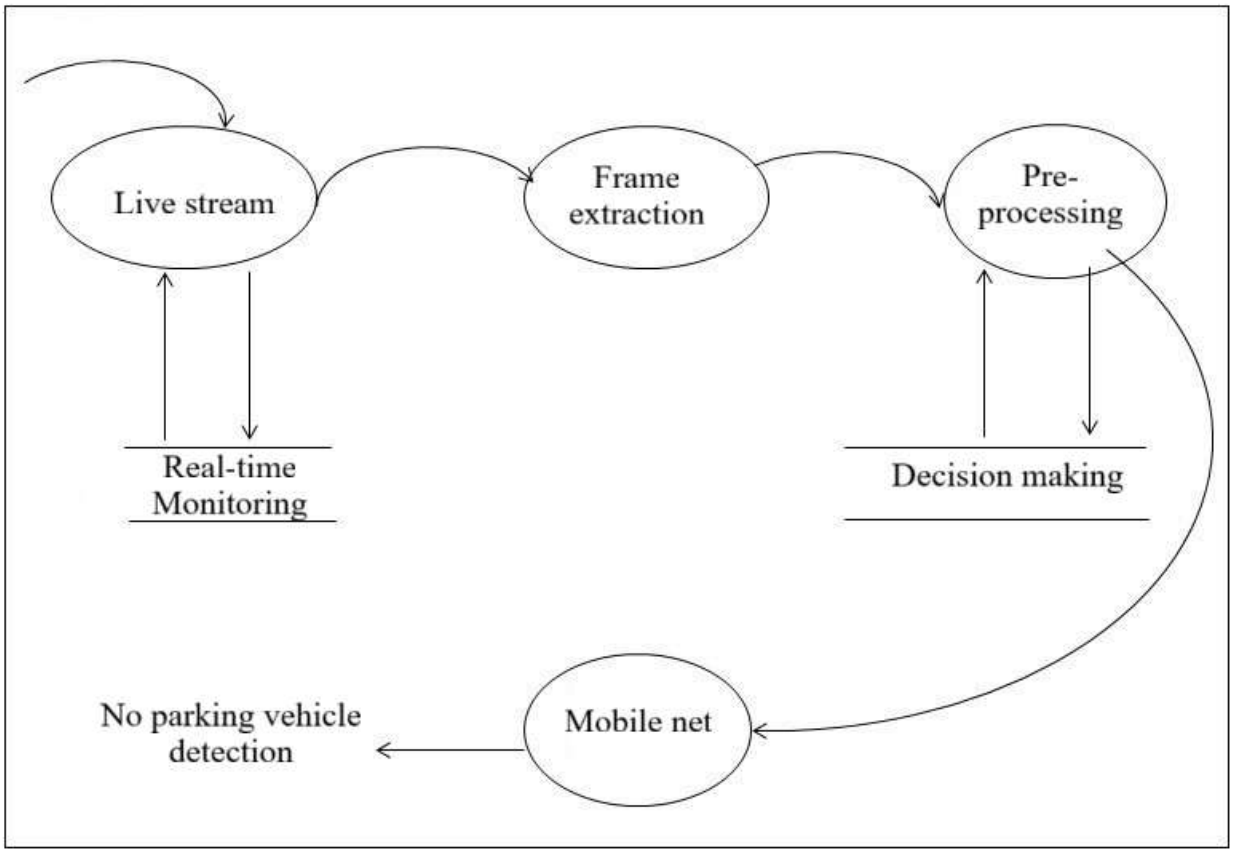
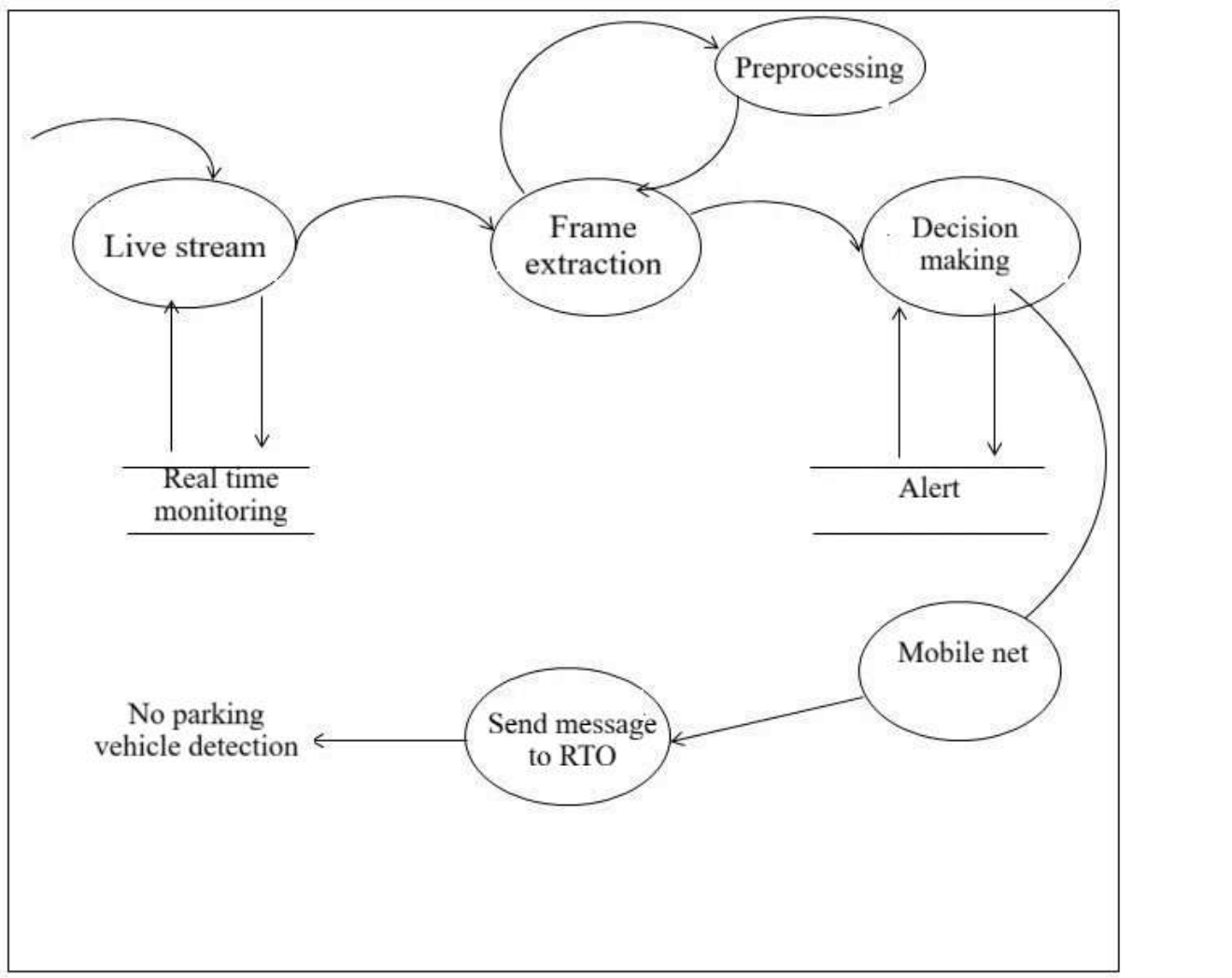


Fig 3 DFD level 1

The DFD 1 diagram provides even more details wherein the user provides the live image which is preprocessed. The preprocessing is performed and then provided to the image normalization which is then utilized by the MOBILE NET. Following this the Training dataset is achieved which is implemented to achieve the No parking vehicle detection.



5.1.1.3 DFD level 2

Fig 4 DFD level 2

The DFD 2 diagram is the most detailed wherein the user provides the test image which is preprocessed. The preprocessing is performed before the image normalization procedure after which the image is resized. The preprocessed and resized image is then provided to the MOBILE NET module which is then utilized using the adecision making Following this the dense layer is implemented to achieve the trained module and send message to RTO which leads to No parking vehicle detection.

5.1.2Activity Diagram

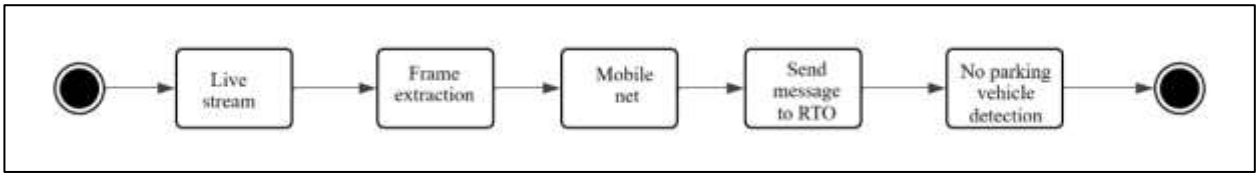


Fig 5.1 Activity Diagram

The activity diagram lists the various activities that are performed in the proposed methodology. The start state is initiated and the live image is providing which leads preprocessing, image normalization, MOBILE NET, and No parking vehicle identification which leads to the stop state.

5.1.3 Usecase Diagram

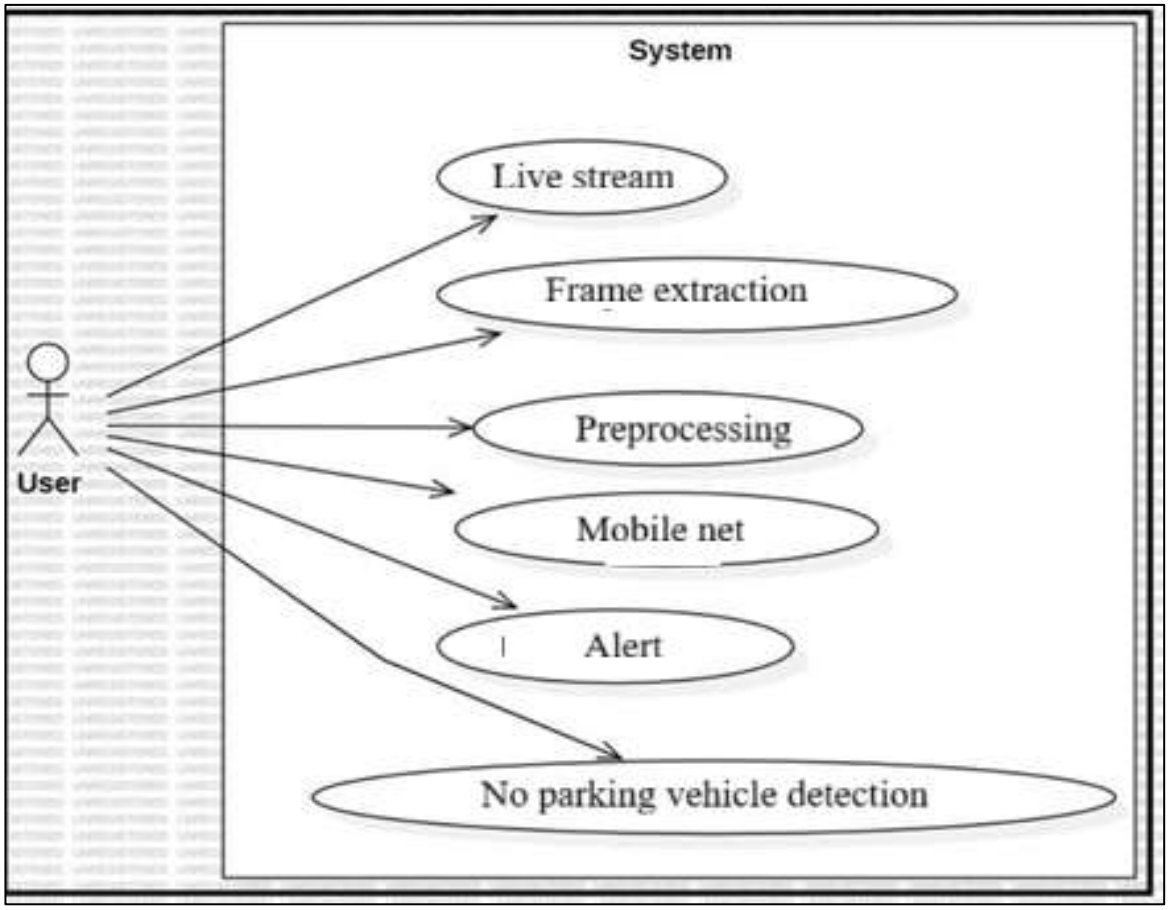


Fig 6 Usecase Diagram

The Use case Diagram depicts the various use cases that are performed by the user in the proposed model. The use cases include feeding the live image, frame extraction, preprocessing, image normalization, MOBILE NET, and No parking vehicle detection.

5.1.4 Sequence Diagram

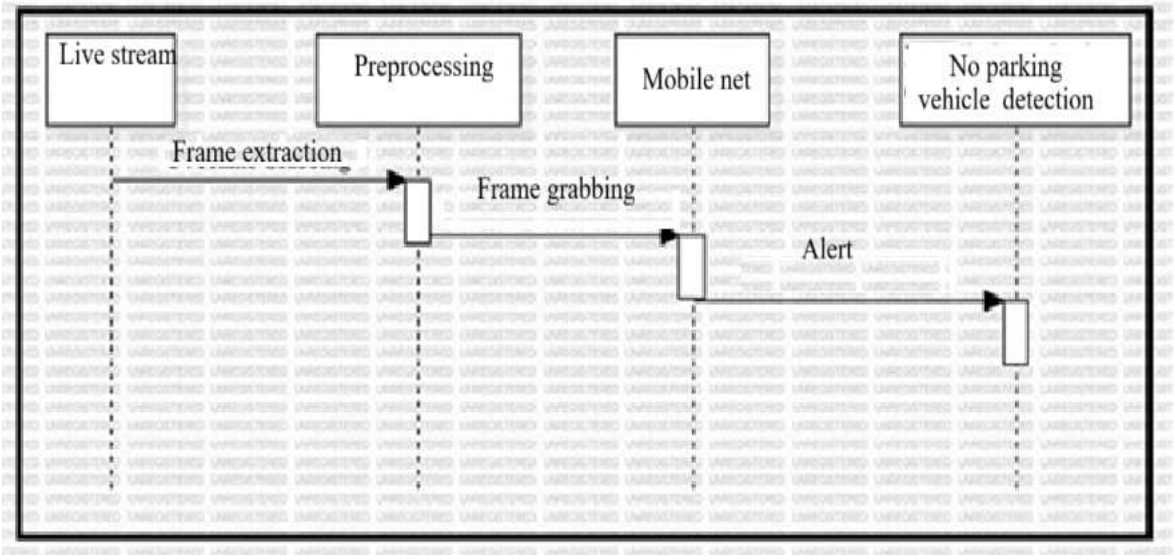
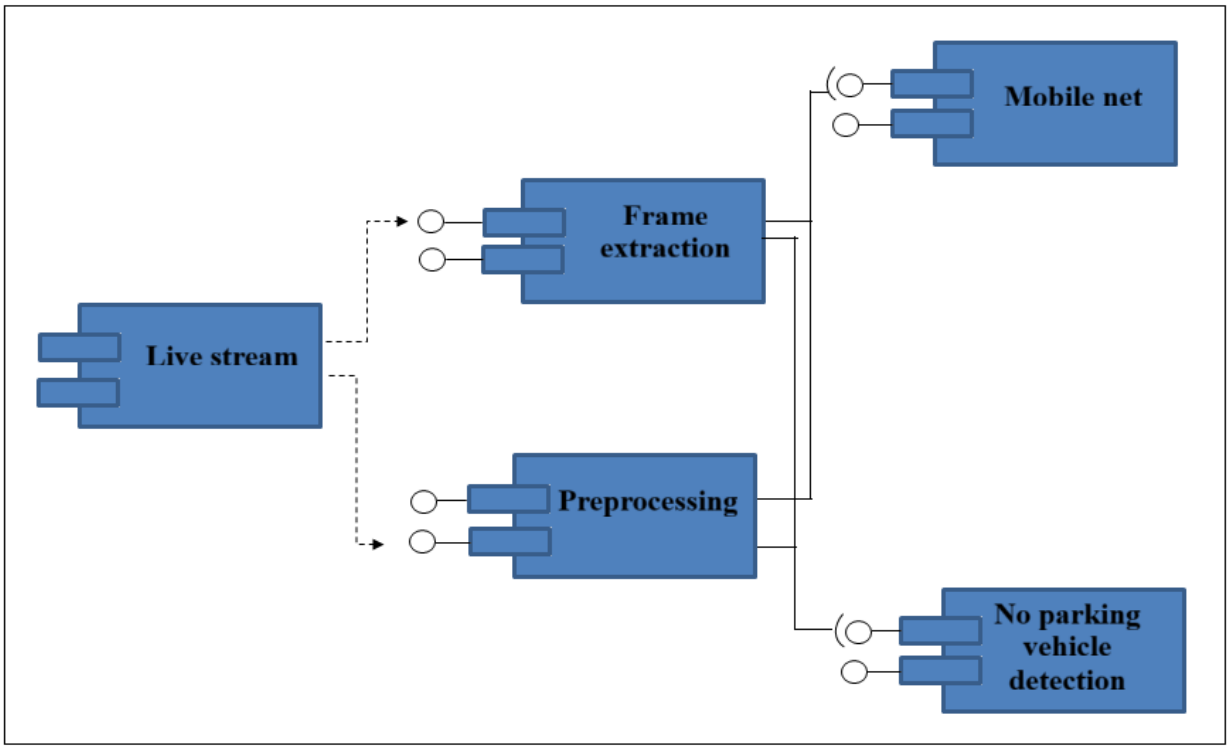


Fig 7 Sequence Diagram

The sequence role diagram provides a sequence of the approaches as well as the various roles performed in the intermediate. In this approach the live image which is utilized for frame extraction after that the preprocessing is done which utilize image normalization, after the Mobile net and the training dataset generation procedure which leads to the No parking vehicle detection.



5.1.5 Component Diagram

Fig 8 Component Diagram

The component diagram illustrates the important components in the proposed system. In our approach the important components consist of the live image which is interlinked with frame extraction and image normalization, these two modules are further linked to the MOBILE NET and No parking vehicle detection.

5.1.6 Deployment Diagram

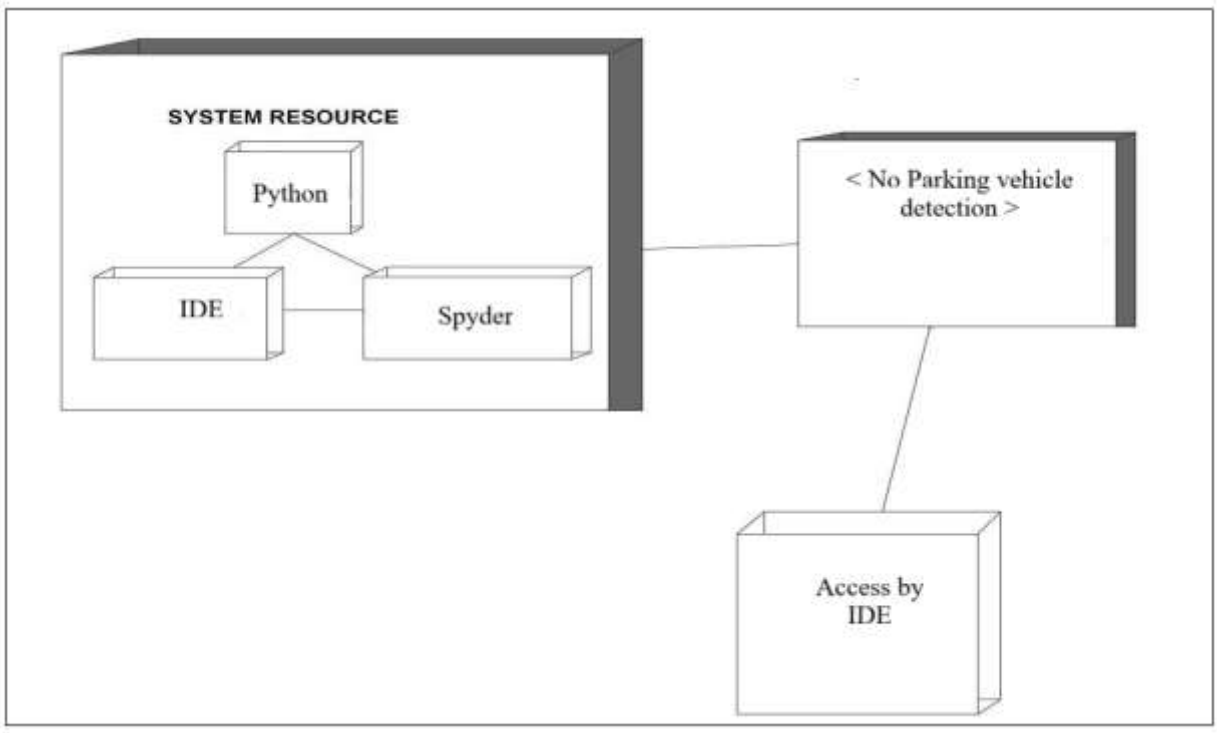


Fig 9 Deployment Diagram

The deployment diagram illustrates the important resources that are utilized for the deployment purposes. In our approach the system resources consist of camera, Training Dataset and the Keras and TensorFlow libraries along with the system for No parking vehicle detection Through AI and the access to the system using the IDE.

5.1.7 Package Diagram

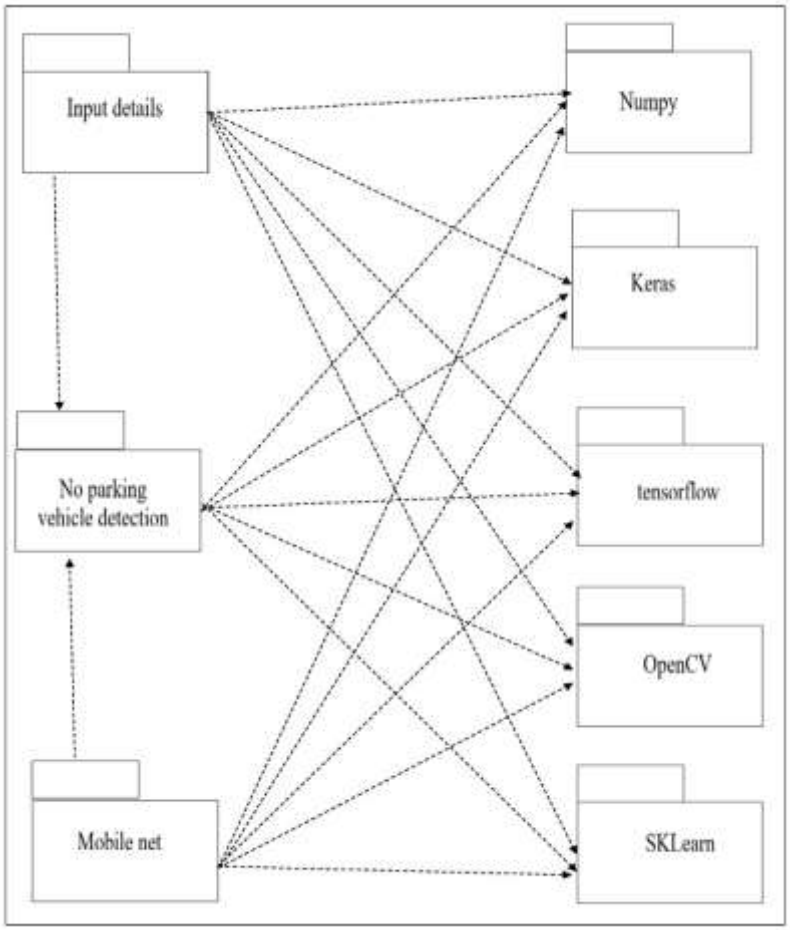
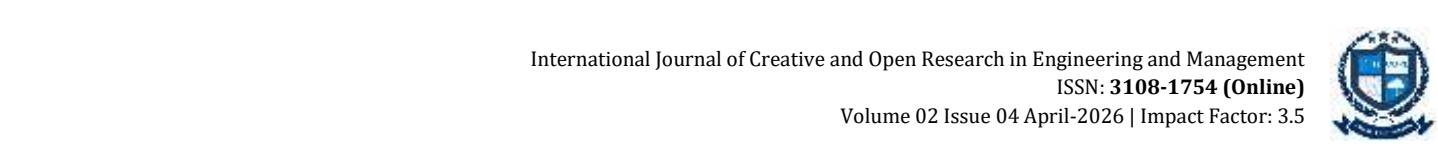


Fig 10 Package Diagram

The package diagram lists the important modules and the relevant packages that are interconnected with each other. The important modules include, Live Image, No parking vehicle detection and MOBILE NET and the packages include TensorFlow, Keras, Sklearn, OpenCV and Numpy.



5.1.8 State Transition Diagram

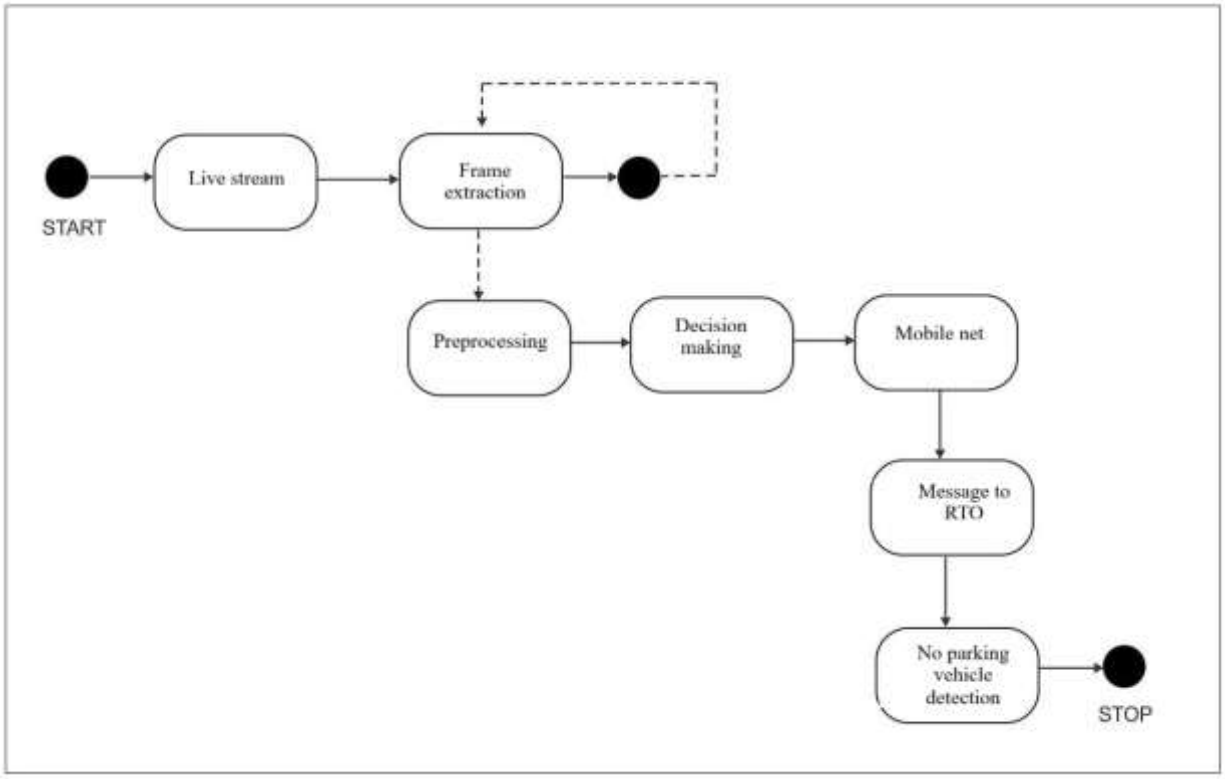


Fig 11: State Transition

The state transition diagram provides the various states that the proposed system goes through. Initially the start state wherein the user provides the live image which is preprocessed. The preprocessing is performed; frame extraction is done before the image normalization procedure after which the MOBILE NET module which is then utilized using the activation function. Following this the trained data is achieved which leads to No parking vehicle detection and then reaches the stop state.

5.1.9 Action Plan

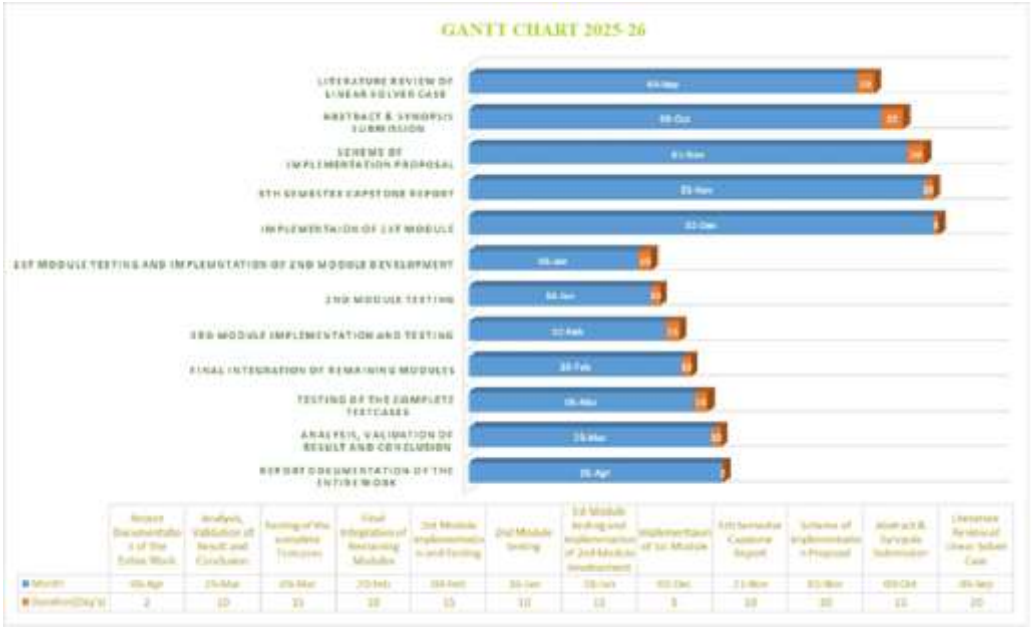
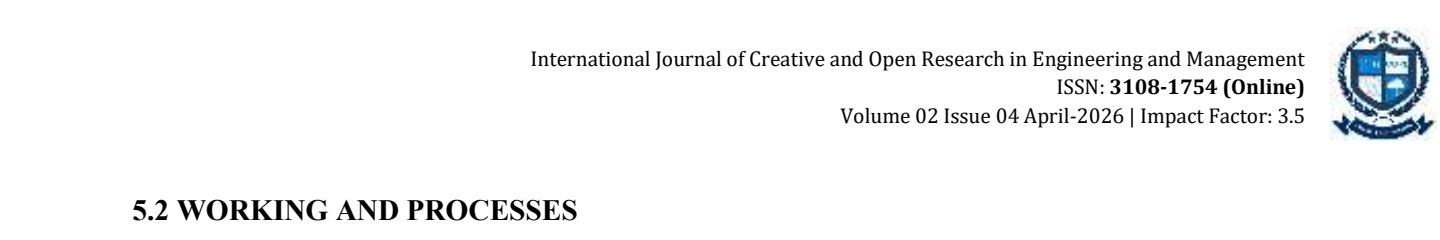


Fig 12: Action Plan



5.2 WORKING AND PROCESSES

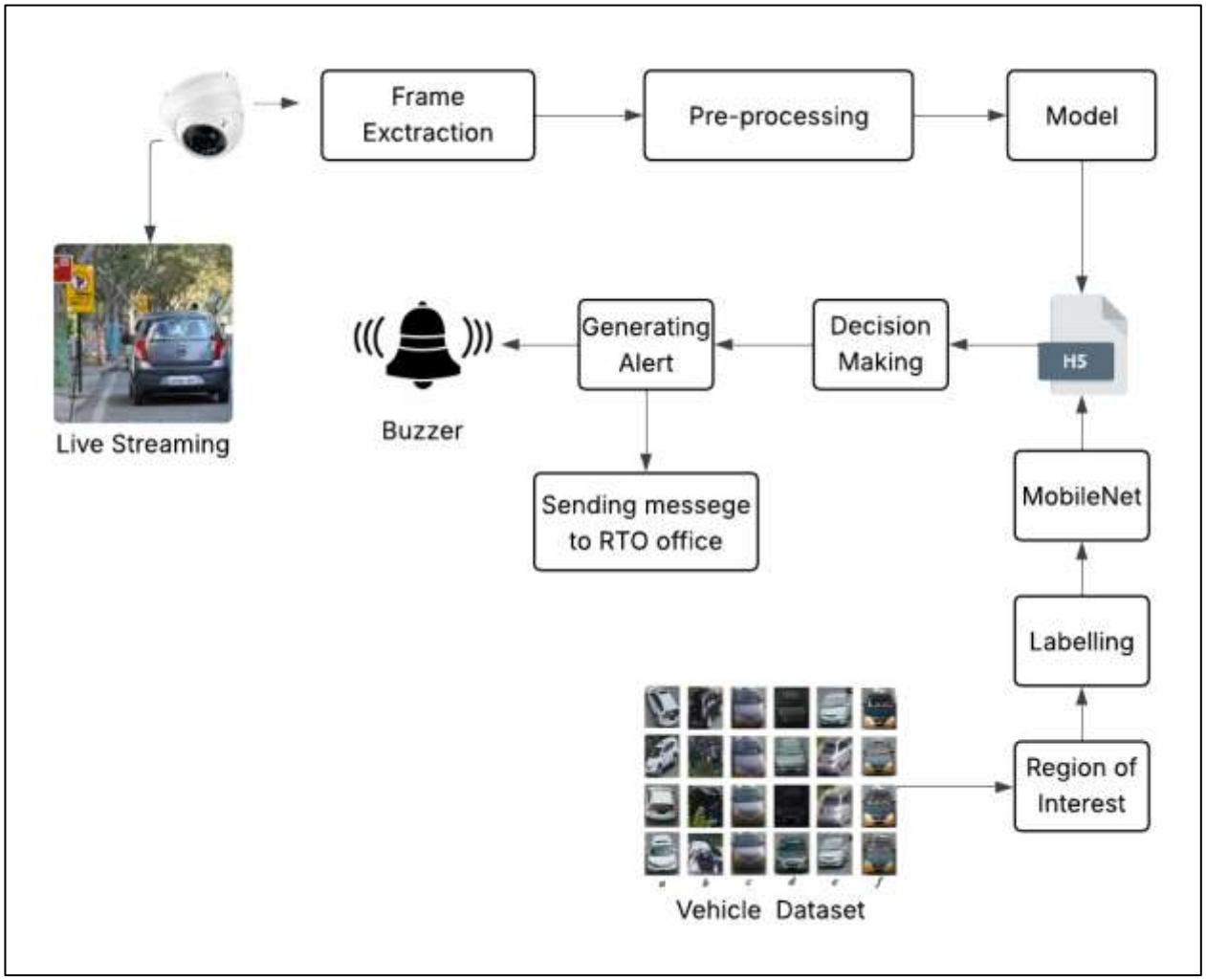


Figure 13 System Overview

The steps of the overview diagram that meant to No Parking vehicle detection system can be described with the below mentioned steps.

Step 1: Dataset Generator: Capturing images of vehicles violating no-parking rules using OpenCV is the initial stage. To test the model, the CV2 package's Video Capture function is used to capture real-time images from the camera. The dataset folder is accessed to identify and analyze various types of vehicles. The model is trained to detect different vehicle categories such as cars, bikes, and trucks. A separate folder is created to store the collected no-parking violation images for further processing and model improvement.

Step 2: Data Labelling: In this step, the previously captured images are annotated using a labeling tool. The user marks the vehicles parked in no-parking zones by drawing bounding boxes around them. The coordinates of the bounding box (x1, y1, x2, y2) are generated by selecting the top-left and bottom-right corners of the object after importing the image into the labeling program. These coordinates are then saved in an XML file format, which is used to train the MobileNet deep learning model for accurate no-parking vehicle detection.

Step 3: Installation of APIs

This Google Colab instance requires the TensorFlow Object Detection API to be installed beforehand. To begin, a copy of the TensorFlow Models repository (https://github.com/tensorflow/models) along with the required installation scripts is needed. By running the provided code cells, the setup process is executed step by step. The next step involves configuring the environment and installing necessary dependencies, including downloading and setting up cuDNN for GPU support. After that, the TensorFlow Models repository is cloned from GitHub. Finally, the TensorFlow Object Detection API is installed and configured, preparing the system for training the no-parking vehicle detection model.

Step 4: Upload Image Dataset and Prepare Training Data

To prepare the dataset for training the no-parking vehicle detection model, the TF Record generation scripts must be executed and the training images uploaded properly. First, upload your dataset and organize the images into three folders: train, validation, and test. These folders should contain images of vehicles in no-parking zones along with their corresponding XML annotation files. Next, on your local system, create a compressed folder named images.zip that includes all the images and XML files. Ensure that all required files are placed correctly inside this zip folder. After uploading the zip file to Google Colab, use the necessary commands to extract its contents and set up the directory structure. The extracted folders will typically be created inside the /content directory of the Colab file system. You can easily navigate and verify these files by clicking on the Files icon on the left panel. Finally, run the TF Record generation scripts to convert the dataset into a format suitable for training the TensorFlow Object Detection model for no-parking vehicle detection.

Step 5: Split images into train, validation, and test folders

After uploading the images.zip file, the next step is to extract its contents and organize them into separate folders for training the no-parking vehicle detection model. These folders are created inside the /content directory in Google Colab, and you can view them using the Files icon on the left panel.
Once the dataset is extracted, the images and their corresponding XML annotation files must be divided into three categories: train, validation, and test.
Train: This set contains the images used to train the model. During each training iteration, images from this folder are fed into the neural network. The model detects vehicles in no-parking zones, estimates their locations, and updates its internal weights using loss calculation and backpropagation to improve accuracy.
Validation: This set is used to monitor the model's performance during training. It helps in tuning hyper parameters such as learning rate and avoiding overfitting. Unlike training images, validation images are used occasionally to evaluate how well the model is generalizing.
Test: This set consists of images that the model has never seen during training. These images are used for final evaluation to measure the real-world performance and accuracy of the no-parking vehicle detection system.
Properly organizing the dataset into these three folders ensures effective training, validation, and evaluation of the model.

Step 6: Create TF Records
The final step is to convert the dataset into TF Record format, which is the preferred input format for TensorFlow during training. This process is carried out using Python scripts that automatically transform the images and their corresponding XML annotations into TF Records.
Before running these scripts, it is necessary to create a label map file that defines the classes for the no-parking vehicle detection system. Create a file named labelmap.txt and list all the vehicle classes (for example: car, bike, truck), assigning each class a unique ID on a new line.
After creating the label map, run the provided code cells in Google Colab. This will generate the labelmap.txt file and then convert the dataset into TF Record files for the train and

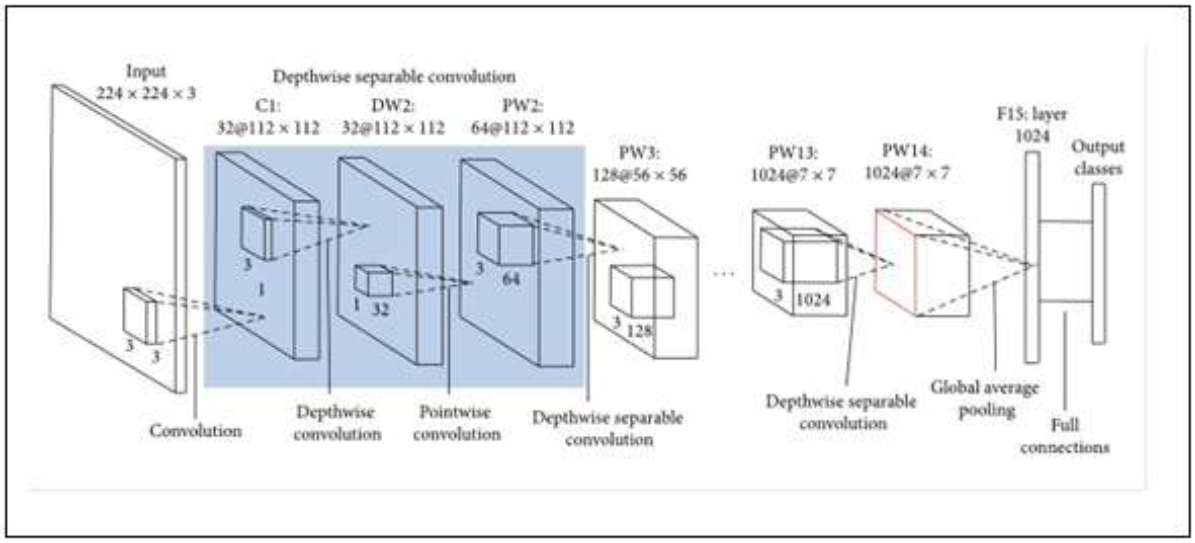
validation sets. These TF Record files are then used by the TensorFlow Object Detection API to train the model efficiently for detecting vehicles parked in no-parking zones**.**

Step 7: Set Up Training Configuration

In this step, the SSD-MobileNet model is configured for training the no-parking vehicle detection system. A pre-trained model from the TensorFlow 1 Object Detection Model Zoo is selected as the base model. Each model comes with its own configuration file, which defines important training parameters such as learning rate, number of steps, and file paths.
Initially, the list of available models in the TF1 Model Zoo is displayed. From this list, the required model is selected by specifying its name in the "chosen_model" field. In this case, “ssd-mobilenet-v1-quantized” is chosen. The corresponding model files and configuration file are then downloaded and prepared for use. After downloading, the configuration file is modified to suit the custom no-parking vehicle detection task. Key training parameters are updated as follows:
num_steps: Defines the total number of training steps. A value of around 40,000 is a good starting point. Training can be extended if the loss continues to decrease.
batch_size: Specifies the number of images processed in one training step. A value of 16 is suitable for most Google Colab GPU environments.
quant_delay_steps: Used for quantization-aware training. Typically set to half of the total training steps.
Additional details such as the number of classes (e.g., car, bike, truck), paths to the dataset (TF Records), label map file, and pre-trained model checkpoint are also configured in the file.

The updated configuration file (pipeline.config) is then saved and used for training. It includes all necessary settings related to the dataset, model architecture, and training process. Furthermore, checkpoint saving frequency is adjusted (e.g., every 1000 steps) to monitor training progress and allow recovery if needed.
This configuration step ensures that the model is properly set up and optimized for detecting vehicles in no-parking zones.

Mobile net Architecture



Step 8: Train Custom TF1 Object Detector: In this step, the custom object detection model for the no-parking vehicle detection system is trained. The training process is executed using the "model_main_tf2.py" script from the TensorFlow Object Detection API. All required parameters and arguments, such as model configuration, dataset paths, label map, and training steps, have already been defined in the previous steps of the Colab setup.
Once the training script is executed, the model begins learning to detect vehicles parked in no-parking zones based on the provided dataset. The training duration typically ranges from 2 to 6 hours, depending on factors

such as the selected model (e.g., SSD-MobileNet), batch size, number of training steps, and available GPU resources.
During training, the system continuously updates model weights and saves checkpoints at regular intervals, allowing progress monitoring and recovery if needed. After the training is completed, the final trained model is exported and converted into a TensorFlow Lite (.tflite) format.
This. tflite file is then used to test and deploy the model for real-time no-parking vehicle detection on edge devices or integrated systems.

Step 9: Testing the Model for No-Parking Vehicle Detection:

Here, the Python program uses the camera to capture live video and frames with the help of the DroidCam app, which is compatible with both mobile phones and laptops. The trained model file (.tflite) is used to detect vehicles parked in no-parking zones from the live video stream. When a violation is detected, the system identifies the location and sends a notification to the concerned authorities via WhatsApp along with the map, direction, and landmark of the camera location.

CHAPTER 6

RESULT AND APPLICATIONS

6.1 RESULTS

PLEASE PUT THE SCREENSHOTS OF YOUR PROJECT HERE AND CONTINUE THE FIGURE NUMBERS

(PUT AS MUCH AS CAN)

6.2 Applications

- Traffic management and congestion control
- Automated parking violation detection
- Smart city surveillance systems

6.3 TEST CASES

6.3.1 Performance Testing:
The performance of the system is measured by the accuracy for the given input Dataset.

6.3.2 System testing:
Checked the steadiness of the space provided by the system for the maximum number of dataset. This is done by feeding the maximum number of dataset to learn and so that set the threshold by the Virtual machine option of the Spyder and Jupyter IDE.
6.3.3 Recovery System
Our system can be recovered in span of 2 hour after crashing. Within two hour we can install all the Software and deploy our system to work as in the past.

6.3.4 Security Testing

6.3.4.1 Stress Testing:
The System is well equipped to stand against the breakdown point of maximum number of datasets decided by the IDE's virtual machine settings, beyond that the memory overflow exception may arise.

6.3.4.2 Unit Testing:
All the modules are independently handled developed and ran to get proper output and finally they are integrated to get the whole output.

6.3.4.3 Black Box Testing:
Compatibility analysis is performed by passing the output of one module to another and verifying the expected results of the no-parking vehicle detection system using deep learning.

6.3.5 Integration Testing:
When all the individual modules are integrated, they form a complete no-parking vehicle detection system using deep learning, which is then cross-checked to ensure the desired output is achieved.

❑ 6.3.6 Test Cases

ID	TEST CASE	INPUT	PASS CRITERIA
D_CAP	Dataset capturing	Images	All images of vehicles parked in no-parking zones are being captured and successfully stored in a designated folder.
D-LAB	Dataset labelling	Image	All images of vehicles parked in no-parking zones are labeled and stored successfully.
D-TRAIN	Dataset training	Labelled images	Images are successfully trained and obtained .tflite file.
U_ACTIVATE	Activate the engine for the no-parking vehicle detection system using deep learning.	Activation of Code	The user successfully activates the no-parking vehicle detection system using deep learning.

Table 9.1: Test Cases

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

The no-parking vehicle detection system using deep learning provides an efficient and automated solution for monitoring and managing illegal parking. By leveraging real-time image capture, a trained MobileNet model, and the TensorFlow Object Detection API, the system can accurately detect vehicles violating no-parking rules. The integration of modules from data collection, labeling, and TF Record generation to model training and deployment ensures a robust workflow. Notifications with location details allow authorities to respond

promptly, improving traffic management and road safety. Overall, this system demonstrates how AI and computer vision can enhance urban monitoring, enforcement, and smart city initiatives.

Future Work

In the future, the no-parking vehicle detection system can be expanded and enhanced in several ways. The system could be integrated with city-wide CCTV networks to enable large-scale monitoring and improve coverage across urban areas. It can be extended to detect additional vehicle types, including buses, trucks, and electric vehicles, while also incorporating vehicle tracking to identify repeat offenders. Real-time cloud processing can be implemented to speed up detection and notifications, and automatic e-challan generation could streamline enforcement. The model can be further improved using advanced architectures like YOLOv8 for higher accuracy. Edge devices can be utilized for on-site detection without relying on constant cloud connectivity, and traffic flow analysis could be combined with the system to optimize parking and reduce congestion. Additionally, a mobile application for authorities can provide instant alerts, and multilingual, location-based notifications can enhance smart city management and public awareness.

CHAPTER 8

APPENDIX

8.1 No Parking vehicle detection system

- ✓ https://www.ijraset.com/research-paper/a-novel-approach-for-detecting-vehicles-parked-in-no-parking-zone-using-image-processing
- ✓ https://arxiv.org/abs/1710.02546
- ✓ https://islab.ulsan.ac.kr/pvd/
- ✓ https://universe.roboflow.com/vehicle-detection-iva8f/illegal-parking-detection-i7jxz
- ✓ https://universe.roboflow.com/search?q=no%20parking
- ✓ https://github.com/topics/parking-detection
- ✓ https://github.com/8harath/Car-Parking-Detection

CHAPTER 9

REFERNCES AND BIBLIOGRAPHY

[1]Adesh Pawar , Ajay Pawar , Ashish Pawar , Ganesh Pawar, Anagha Chaudhari, 2021, An Elaborative Study of Smart Parking Systems, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 10 (October 2021)

[2]H. Canli and S. Toklu, "Deep Learning-Based Mobile Application Design for Smart Parking," in IEEE Access, vol. 9, pp. 61171-61183, 2021, doi: 10.1109/ACCESS.2021.3074887.

[3] In Hwan Jung, Jae Moon Lee, Kitae Hwang, “Smart Parking Management System Using AI,” ISSN: 1735-188X, DOI: 10.14704/WEB/V19I1/WEB19307

[3]Joseph, Leobin & Krishna, Ajay & Berty, Maschio & P, Pramod & A, Velusamy. (2022). Advanced Parking Slot Management System Using Machine Learning. International Journal of Advanced Research in Science, Communication and Technology. 497-502. 10.48175/IJARSCT-3299.

[4]W. Kim and I. Jung, "Smart Parking Lot Based on Edge Cluster Computing for Full Self-Driving Vehicles," in IEEE Access, vol. 10, pp. 115271-115281, 2022, doi: 10.1109/ACCESS.2022.3208356.

[5]B. Benjdira, A. Koubaa, W. Boulila and A. Ammar, "Parking Analytics Framework using Deep Learning," 2022 2nd International Conference of Smart Systems and Emerging Technologies (SMARTTECH), Riyadh, Saudi Arabia, 2022, pp. 200-205, doi: 10.1109/SMARTTECH54121.2022.00051.

[7] Amara Aditya, Shahina Anwarul, Rohit Tanwar, Sri Krishna Vamsi Koneru, An IoT assisted Intelligent Parking System (IPS) for Smart Cities, Procedia Computer Science, Volume 218,2023,Pages 1045-1054, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2023.01.084.

[8] Vo, Van-An & Phan, Van-Duc & Bui, Vu-Minh & Do, Tri-Nhut. (2023). Design Of Deep Learning Model Applied For Smart Parking System. Advances in Electrical and Electronic Engineering. 21. 258-267. 10.15598/aeee.v21i4.5366.

[9]Ming Li and Li Zhang, “Deep Learning-based License Plate Recognition in IoT Smart Parking Systems using YOLOv6 Algorithm” International Journal of Advanced Computer Science and Applications(IJACSA), 14(12), 2023. http://dx.doi.org/10.14569/IJACSA.2023.0141223.

[10] Kalyani, G., Jyothi, K., & Likhitha, M. (2023). SMART PARKING SYSTEM BASED ON COMPUTER VISION TECHNIQUE

****