# VIDYAVARDHAKA COLLEGE OF ENGINEERING

**GOKULAM III STAGE, MYSURU-570 002**

**Accredited by NAAC with A 'Grade', *Autonomous institution affiliated to***

**Visvesvaraya Technological University, Belagavi**

# *ORGANIZED PARKING SYSTEM USING MACHINE LEARNING*

**A Project Report submitted in partial fulfillment for the award of degree**

**BACHELOR OF ENGINEERING**

**IN**

**INFORMATION SCIENCE AND ENGINEERING**

**BY**


| | |
|---|---|
| **NATARAJ S LAKKUNDI** | **[4VV21IS406]** |
| **NAVYASHREE S** | **[4VV21IS407]** |
| **POOJA S** | **[4VV21IS408]** |


**Under the guidance of**

**Prof. Chayashree G**

**Assistant Professor**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

**Accredited by NBA, New Delhi**

**2023-24**

**VIDYAVARDHAKA COLLEGE OF ENGINEERING**

**GOKULAM III STAGE, MYSORE- 570 002**

**DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING**

VVCE

# CERTIFICATE

This is to certify that the final year project report entitled **"ORGANIZED PARKING SYSTEM USING MACHINE LEARNING"** is a bonafide work carried out by **Nataraj S Lakkundi** (4VV21IS406), **Navyashree S** (4VV21IS407) and **Pooja S** (4VV21IS408) student of 8th-semester Information Science and Engineering, **Vidyavardhaka College of Engineering, Mysuru** in partial fulfillment for the award of the degree of **Bachelor of Engineering** in **Information Science & Engineering** of the **Visvesvaraya Technological University, Belagavi**, during the academic year **2023-2024**. It is certified that all the suggestions and corrections indicated for the internal assessment have been incorporated in the report deposited in the department library. The report has been approved as it satisfies the requirements in respect of project work prescribed for the said degree.

| **Signature of the Guide** | **Signature of the HOD** | **Signature of the Principal** |
|---|---|---|
| **(Prof.Chayashree G)** | **(Dr.Ravi Kumar V)** | **(Dr. B Sadashive Gowda)** |

**Name of the Examiners** **Signature with Date**

**1)**

**2)**

# ACKNOWLEDGEMENT

We would like to acknowledge with pride and humility the kind of help and guidance received from several people in the preparation of this report. Without their invaluable help, cooperation, and guidance, it could never have been possible to prepare this report in this form.

Firstly, we thank the management of this college and our beloved Principal, Dr. B. Sadashive Gowda, Vidyavardhaka College of Engineering, Mysuru, for their constant support and motivation in preparing this report, for making the needed library and laboratory facilities available in preparing this report.

We want to thank Dr. Ravi Kumar V, Professor and Head, Department of Information Science and Engineering, VVCE, for all his valuable suggestions and guidance in the project.

My heartfelt thanks go to my guide, Prof. Chayashree G, Assistant Professor, Department of Information Science and Engineering, VVCE, Mysuru, who has given us constant motivation and guidance from the beginning through the end of the project work. I am deeply grateful for her meticulous and thoughtful discussions and many valuable suggestions. For the guidance, inspiration, and encouragement received from her, I owe her a lot.

**NATARAJ S LAKKUNDI** **4VV21IS406**

**NAVYASHREE S** **4VV21IS407**

**POOJA S** **4VV21IS408**

# ABSTRACT

Parking management is very important in an urban setup but, on the contrary, it is very inefficient and mostly congested. This paper presents the proposed innovative technique for optimizing the utilization of the parking lot using machine learning. The system uses a set of sensors, such as cameras and ultrasonic devices, for the collection of real-time data on parking space availability and vehicle entrance and exit within parking lots. The preprocessing of the collected data and the extraction of the features are carried out to come up with a very strong machine-learning model. The effective model identifies and predicts parking slots using object detection, classification, and predictive analysis algorithms, among many others. The information is displayed to users in a friendly interface to enable drivers to make informed decisions regarding where to find parking. The user interface is also updated in real time, thus showing to drivers where the available parking space is through visual signals or navigational directions. The system is adaptive and scalable; hence it can be installed in preexisting infrastructures without any challenges and be fit for installations in different parking lot sizes and configurations. The system is functional at the core of its being in continuous optimization and maintenance—it is constantly updated and fine-tuned based on feedback and changing patterns of parking. In addition, privacy and security methods have been ensured to maintain the protection of sensitive data, compliance with regulations, and build trust among users. So, this proposed parking management system leverages machine learning to reduce congestion, optimize space, and improve efficiency in city parking with the ultimate goal of making urban parking much smarter and sustainable.

# TABLE OF CONTENTS

# CHAPTER 1

# INTRODUCTION

The one thing that the major urban centers of the world have in common: inefficiencies in parking that are a source of congestion, frustration, and wastage of resources. Traditional parking management systems do not allow for the efficient use of available space, hence encouraging higher traffic in the area, thereby developing a higher environmental impact.

This research would be redefining the traditional parking management with the exploitation of machine learning. The current system combines sensor technology with advanced algorithms to make possible the provision of insight in real-time on the availability of parking space in urban and commercial parking lots.

The importance of this system is that it can process massive data collected from various sensors, such as cameras, ultrasonic devices, and RFID readers. Through sophisticated data processing techniques and

Combining those with the power of machine learning algorithms, this system can determine patterns and predict the availability of parking spaces, directing the driver to the available parking spot. It optimizes the space for parking and, at the same time, interfaces with the drivers by providing real-time information with user-friendly interfaces, such as mobile applications or display panels. In guiding to and through available spots, this solution reduces the congestion of cars and, therefore, the build-up of pollution brought about by time-intensive searches.

The proposed solution is scalable, flexible—meaning it it can vary in the features of the parking lot—to fit different configurations and sizes of parking lots, hence applicable in various urban settings. In this way, continuous data-driven optimization combined with strict privacy assurance ensures reliability and the highest level of user trust.

This research would go a long way toward redefining parking management paradigms and, therefore, an all-inclusive solution in sync with changing needs for urban infrastructures that promote sustainability and raise the general quality of life in urban areas.

## 1.1 MOTIVE

The key objective of this machine learning-based organized parking system project is to increase the utility of parking space and improve the user experience. The application of machine learning algorithms to predict availability and allocate it efficiently actually streamlines the parking process. Not only does it save users' valuable time that they waste looking for parking spaces, but it also decreases the opportunity for traffic congestion and the environmental impact caused by circling until a parking spot becomes available. Ultimately, the goal is to make parking easier and more sustainable for drivers and communities.

## 1.2 OBJECTIVES

Develop an efficient and smart parking management solution, which makes use of machine-learning algorithms in the most effective way to utilize the provided parking space, enhance user experience, and reduce parking area congestion. Highlighted below is the main objective of the project:

- **Optimize Parking Space Allocation:** By the development of machine learning models through historical data and real-time inputs, the system can accurately predict how many parking spaces are available at any given time. Parking spaces will, therefore, be promptly allocated to reduce the time taken to find parking.
- **Enhanced User Experience:** The driver and the parking lot operators must be comfortable with the user interface design. The system should provide real-time information of availability of parking space, guide users to the nearest available parking space, and integrated modes of payment to ease the parking process for users.
- **Reduce Congestion:** Utilize machine learning algorithms to predict peak hours and high-traffic periods. By analyzing these patterns, the system can implement dynamic pricing strategies or offer incentives to encourage drivers to park during off-peak hours, thereby reducing congestion and improving traffic flow in parking areas.

- **Improve Revenue Generation:** Implement a robust revenue management system that optimizes parking pricing based on demand, time of day, and other relevant factors. By maximizing revenue generation while ensuring fair pricing for users, the system can enhance the overall profitability of parking operations.

- **Scalability and Adaptability:** Develop a scalable solution that can be easily deployed in various parking environments, including indoor and outdoor parking lots, parking garages, and urban areas. The system should also be adaptable to accommodate future enhancements and technological advancements in machine learning and parking management.
- **Data security and privacy:** Data should be given the tightest security measures to ensure protection of the sensitive user information, such as the license plate numbers, payment information, among others. Ensuring data is compliant with relevant regulations on data privacy will also build trust with users and stakeholders**.**

## 1.3 PROPOSED SYSTEM

The proposed organized parking system to apply machine learning is all about having a seamless and effective parking experience. The system will employ various machine learning techniques in parking space optimization, entry, and exit streamlining, and general management. It will tap into sensors and cameras to collect real-time information on available parking slots, types of cars, and traffic flows in the parking facility. Machine learning algorithms would then analyze such data to make predictions about parking demand patterns, identify the best-fit parking spot for incoming vehicles on the basis of size and proximity to exits or entrances, and keep changing prices to incentivize optimal use of space. The system could also be embedded with a predictive maintenance algorithm, thus predicting and avoiding possible equipment failures in real time, and thereby ensuring that the operation is never interrupted. The organized parking system involves machine learning in order to provide maximum comfort to the driver, better space utilization for the operator, and reduced congestion on the roads of a city.

## 1.4 PROBLEM STATEMENT

This project aims at designing a vehicle-counting-and-classification system for a systematically organized parking lot through the application of machine learning techniques. This system will make it possible to detect and track vehicles entering and leaving parking lots, classifying them into different categories: cars, motorcycles, trucks, buses. Furthermore, the system will be real-time in providing data on the availability of the parking lot occupancy and free spaces for an appropriate parking management function.

- Vehicle Detection: Develop algorithms for the detection of vehicles from video feeds or images captured by surveillance cameras installed at entry and exit points of parking lots.

- Vehicle Tracking: Vehicle tracking mechanisms should be put in place to monitor movements within the parking lot with a high degree of precision in recording entry and exit times.

- Vehicle Classification: Apply machine-learning models to classify vehicles into multiple classes on the basis of their size, shape, or any other features.

- Real-Time Monitoring: Develop an interface that will present real-time information on the occupancy and availability of parking lots to the parking attendant and drivers for a better decision-making process.

- Performance Evaluation: System performance will be evaluated in terms of accuracy, speed, and scalability for varied environmental conditions such as changes in illumination, weather, and traffic.

.

# CHAPTER 2 LITERATURE SURVEY

This paper, therefore, attempts a literature survey that aims to sum up everything that is out there in the form of existing research and scholarly works, including work around the fringes on blockchain-based voting systems. The survey is conducted based on a wide range of academic papers, journal articles, conference proceedings, and like sources and serves the purpose of outlining the status of knowledge, identifying key trends and developments, and reviewing critically the strengths and limitations of blockchain technology within the context of electoral processes.

This survey elaborates principles, technical architectures, challenges of implementation, and applications of blockchain-based voting systems. The survey, therefore, aggregates and analyzes findings from the literature on the possible benefits of blockchain technologies, such as tamper-proof audit trails, decentralized consensus mechanisms, and increased voter trust

## 2.1 A Fast and Accurate Real-Time Vehicle Detection Method Using Deep Learning.

**Authors: Annam Farid. Farhan Hussain. Khurram Khan. Mohsin Shahzad. Uzair Khan**

**Published: - 2023**

These deep learning-based classification and detection algorithms have recently been potentiated in the domain of vehicle detection for intelligent transportation systems. Single vehicle detection methods, which are based on a small number of high-quality labeled training samples, are not able to accomplish acceptable accuracy for the task of road vehicle detection. The current paper presents available datasets for detection and classification of vehicles using the YOLO-v5 architecture. The finding of this paper uses the concept of transfer learning through fine-tuning the weights of the pre-trained YOLO-v5 architecture. For being able to use the concept of transfer learning, the authors had to collect extensive datasets of images and videos showing the crowding traffic patterns. These datasets were made more comprehensive by pointing different attributes—for instance, high- and low-density traffic patterns, occlusions, and different weather circumstances. All these accumulated datasets were manually annotated. Eventually, the enhanced YOLO-v5 architecture is then trained over any complex traffic patterns. Training the fine-tuning of the pre-trained network on our datasets made our proposed YOLO-v5 to outperform several other traditional approaches of vehicle detection in terms of detection accuracy as well as execution time.

## 2.2 Performance Evaluation of Machine Learning and Neural Network-Based Algorithms for Predicting Segment Availability in AIoT-Based Smart Parking

**Authors: Issa Dia. Ehsan Ahvar. Gyu Myoung Lee**

**Published: 2022**

Parking is a challenging task for a driver in a large-size smart city. In an AIoT smart parking application, time-related prediction of parking place availability will save searching time and automotive fuel. However, the performance of different Machine Learning and Neural Network-based (MLNN) algorithms in predicting parking segment availability might differ. Performance of a set of well-known MLNN algorithms and different combinations of them, known as Ensemble Learning or Voting Classifier, has been undertaken by using a real dataset. Datasets used in this report have about five million records of the measured parking availability in San Francisco. In addition to the cross-validation scores, in the evaluation, resource requirements, and simplicity of the algorithms, including their execution time for training and testing, are considered. The results suggest that even if a portion of the ensemble learning algorithms might have great scores in terms of validation, they do consume significant computing and time resources. The simple decision tree (DT) algorithm, on the other hand, maintains much faster execution time compared to ensemble learning algorithms while it still has acceptable performance. To illustrate, DT accuracy is less than 1% smaller compared to the best ensemble algorithm.

## 2.3 Optimized real-time parking management framework using deep learning

**Authors: Sarmad Rafique, Saba Gul, Kaleemullah Jan, Gul Muhammad Khan**

**Published: 2023**

In this context, a new intelligent parking management system was proposed, one that sought to address the challenges and overpass the limitations of the existing hardware-based approach. The previously developed system extended the contemporary intelligent parking management system, using the YOLO v5 object detection algorithm to detect vehicles, instead of distinguishing between vacant and occupied slots, as done by the previous IPMS. The status of the parking lots is recognized by the framework of vehicle detection.

## 2.4 multi-camera vehicle counting using edge-AI

**Authors: Luca Ciampi, Claudio Gennaro, Fabio Carrara, Fabrizio Falchi, Claudio Vairo, Giuseppe Amato.**

**Published: 2022**

The paper describes a novel system to count vehicles automatically in the parking area through captured images by the smart cameras. This work is different from most in the literature dealing with the task in the base of the use of multiple visual sources for the wide area of observation in the parking area from different perspectives. The described multicamera system offers the possibility of performing an automatic estimation of the number of cars in the whole parking lot directly on the edge devices. It is composed of an on-device deep learning-based detector for localizing and counting the vehicles in the captured images and a decentralized geometric-based approach that is capable of analyzing intercamera shared areas and merging data acquired by all the devices. We tested our experimental prototype with the extended version of the CNRPark-EXT dataset, that is, a set of images captured from the parking lot in the campus of the National Research Council (CNR) in Pisa, Italy. From our experiments, it could be seen that our system was strong and took advantage of the redundant information derived from the different cameras, thus improving the overall performance without any extra geometrical information regarding the monitored scene.

## 2.5 Detecting Heavy Goods Vehicles in Rest Areas in Winter Conditions Using YOLOv5

**Authors: Margrit Kasper-Eulaers. Nico Hahn.Stian Berger.Tom Sebulonsen.ystein Myrland.**

**Published: 2021**

Therefore, optimizing the planning of rest breaks by provisioned parking lots at rest areas is of utmost concern to haulage companies and traffic and road administrations. Here, we provide an example where You Only Look Once (YOLO) v5, an object detection framework, is deployed to ascertain the availability of heavy goods vehicles in rest areas over the course of a winter for real-time prediction of parking spot occupancy. The winter, snowy condition and the polar night usually present some challenges to image recognition; therefore, we use thermal network cameras. Because images from such cameras usually have a lot of overlaps and cut-offs of vehicles, we applied the transfer learning approach has been adopted for

YOLOv5 in order to further investigate whether the features of the front cabin and the rear are good features for heavy goods vehicles. Our results show that the algorithm trained is able to detect the front cabin of Confidence is high in detecting trucks and buses, but the rear part is hard to detect, especially if it is located far away from the camera. Conclusion: We will show that using the front and rear instead of the whole vehicle when winter conditions result in challenging images with many overlaps and cut-offs will lead to an improvement in detecting the heavy goods vehicles. We will also show that the thermal network imaging is promising in detecting vehicles.

## 2.6 A Review of Machine Learning and IoT in Smart Transportation

**Authors: Fotios Zantalis, Grigorios Koulouras, Sotiris Karabetsos, Dionisis Kandris**

**Published: 2019**

With the rise of the Internet of Things, applications are now smarter, and connected devices are used to exploit all aspects of a modern city. ML techniques applied further to enhance intelligence and the capabilities of an application with the increase of volume of the collected data. There has been a lot of research and approach with techniques from both ML and IoT regarding smart transportation. In this review, by smart transportation, I shall treat the field as an umbrella term that includes applications in the field of route optimizations, parking, street lights, accident detection and prevention, road anomalies, and infrastructures. The purpose of this paper is to provide a self-contained review of ML techniques and IoT applications into Intelligent Transportation Systems (ITS) and get a clear view of trends in aforementioned fields, spotting possible coverage needs. From the reviewed articles, it becomes evident that there is a possible lack of ML coverage for the Smart Lighting Systems and Smart Parking applications. Among the most popular ITS applications, researchers have found route optimization, parking, and accident/detection applications.

## 2.7 Automated parking surveys from a LIDAR equipped vehicle

**Authors: Douglas A. Thornton a b, Keith Redmill b, Benjamin Coifman b c**

**Published: 2014**

For this reason, parking surveys can function as a potential source for quantitative data which will provide an illustration of the spatial and temporal use of parking spaces within an area of interest. The outputs from these types of surveys have proven key tools for the management of supplies in car parks and the planning of related infrastructure. The main drawback in carrying out parking studies has been that manual tabulation of observations is very labour intensive and, consequently, allows only for low temporal resolution. In this paper, the potential for data collection and information extraction is investigated with a proof-of-concept study in the context of two-dimensional scanning.

The ranging measurements are processed to estimate the position of the curb and the presence of objects in the road. This paper discusses work done for parallel parking in an opposing direction of travel. The sensor data comes from a Light Detection and Ranging (LIDAR) sensor mounted on the vehicle; although the work is compatible with other ranging sensors, for example, stereo vision. Occlusion and location reasoning are then used to differentiate which of the objects are vehicles and whether a given vehicle is parked or is in the traffic-stream. Occupancy of the parking area, vehicle size, and vehicle-to-vehicle gaps are then measured. The next section applies the algorithm to an area with unmarked, on-street parking near a large university campus. Overall, vehicle counts from 29 trips over 4 years were compared against concurrent ground truth with favourable results. In addition, it can be applied to monitoring parking in the direction of travel because it rules out occlusion and facilitates processing.

## 2.8 Estimating the occupancy status of parking areas by counting cars and non-empty stalls

**Authors: - D. Di Mauro a b, A. Furnari a, G. Patanè b, S. Battiato a, G.M. Farinella**

**Published: -2019**

This work presents and compares different vision-based approaches to estimate the occupancy status of parking areas by counting cars and non-empty parking stalls. Our investigation considers both the scenario in which parking stalls are marked on the ground and the more challenging one in which no assumption on

the presence or position of stalls is assumed. We carry out an experimental analysis on a real-world dataset of videos collected in different parking areas. Specifically, this work compares solutions based on image classification, vehicle detection and semantic segmentation. Our analysis highlights that:

(1) methods based on image classification can be effectively leveraged when the position of parking stalls is known in advance.

(2) methods based on image segmentation should be preferred over methods based on object detection when the geometry of the scene is not known, (3) temporal smoothing can be effectively used to improve predictions over time**.**

## 2.9 Smart Traffic Monitoring Through Pyramid Pooling Vehicle Detection and Filter-Based Tracking on Aerial Images

**Authors: - Adnan Ahmed Rafique; Amal Al-Rasheed; Amel Ksibi; Manel Ayadi; Ahmad Jalal; Khaled Alnowais**

**Published: - 2022**

The traffic monitoring system shall be a practicable option for the reduction in traffic snarls. The principal function of the traffic monitoring system is to maintain the data about the traffic, such as the number of cars, the types of vehicles, and the speed of the vehicles. It carries out the traffic analysis with the data collected, in order to use the road network effectively, estimate future transportation needs, and ensure traveler safety. In most countries, it is costly to implement and maintain traffic monitoring systems.

## 2.10 Parking Time Violation Tracking Using YOLOv8 and Tracking Algorithms

**Authors: - Nabin Sharma, Sushish Baral, May Phu Paing, Rathachai Chatchai**

**Published: - 2023**

In Thailand, the biggest violation in parking is a time violation. The vehicle cannot be parked for more than a certain amount of time. The current remedy on the road is closed-circuit television (CCTV) surveillance cameras with human labor. This paper proposes a scheme that can open doors to a low-cost time violation tracking system using the CCTV, deep learning models, and object tracking algorithms.

# CHAPTER 3

# SOFTWARE REQUIREMENTS SPECIFICATIONS

The software requirement specifications (SRS) for the Organized Parking Vehicle Counting and Classification System bring a whole solution to effective parking space management realized through machine learning approaches. This set of software requirements involves a combination of features enabling accurate counting and classification of vehicles, ultimately to enhance the user's parking experience. This is where computer vision algorithms should be put into place so that the system can detect whether a vehicle is entering or exiting the parking space. The monitoring of the vehicle movement within the area is very crucial for continuous monitoring. The software should have the ability to count the exact number of vehicles at any given time in the parking lot. It is for this reason that this is an important feature in an effective parking space management process and use of resources to be utilized to the maximum. Classify between several types of vehicles such as cars, motorcycles, trucks, and so on. Such differentiation shall be crucial to generate insights into the types of vehicles that visit the parking facility with regularity and apply different pricing strategies whenever implemented.

## 3.1 SYSTEM REQUIREMENT

The efficient management of parking spaces is crucial for minimizing congestion and maximizing utilization. Traditional parking management systems often rely on manual intervention or basic sensors, leading to inefficiencies and inaccuracies. To address these challenges, an automated solution leveraging machine learning (ML) for vehicle counting and classification offers significant advantage.

### 3.1.1 HARDWARE SPECIFICATIONS

#### 1) Camera System:

- **High-resolution cameras:** Cameras with high resolution (HD or higher) are crucial for capturing clear images of vehicles entering and exiting the parking area.
- **Wide-angle lenses**: Wide-angle lenses help capture a broader field of view, allowing the camera to cover a larger area of the parking lot.

## 2) Computer or Embedded System:

- **Processing Power:** A computer or an embedded system with enough processing power to perform the running of machine learning models for real-time vehicle detection and classification.
- **Graphics Processing Unit (GPU):** GPUs can significantly accelerate the computation required for machine learning tasks, such as image processing and deep learning inference.
- **Memory (RAM):** Sufficient RAM is needed to store image data and intermediate computations during the processing pipeline.
- **Storage:** Adequate storage space is required to store captured images and video footage for later analysis or archival purposes.

## 3) Network Connectivity:

- **Ethernet or Wi-Fi connection—** the system should be connected to a local network or the Internet, which will support remote monitoring, data exchange, and software update access.
- **Optional:** Cellular connectivity can be integrated if you find the need to do so, covering locations where regular network connectivity is not available.

## 4) Mounting Hardware:

- **Mounts or brackets:** Sturdy mounts or brackets are needed to securely install the cameras in optimal positions for capturing vehicle movements.
- **Weatherproof enclosures:** If the cameras are installed outdoors, weatherproof enclosures are necessary to protect them from the elements.

## 5) Optional Add-ons:

- **Lighting:** Additional lighting, such as floodlights, can be necessary to supplement visibility in low light or poor lighting conditions.
- **Environmental sensors:** Such sensors can monitor the atmospheric conditions such as temperature and humidity, and this information can be used to provide further information for the analysis of the parking facility during optimization**.**

## 3.1.2 SOFTWARES SPECIFICATIONS

i. **Programming Language**: Python is the most commonly used, preferable, and in-demand programming language for most machine learning tasks primarily because of its wide range of libraries, such as TensorFlow, Keras, and OpenCV. Also, Python allows easy interoperability with other technologies and platforms.

**ii. Machine Learning Framework**

- **TensorFlow:** It provides some powerful tools to design and deploy machine learning models, including neural networks, for any image recognition task, vehicle classification included.
- **OpenCV:** Open-Source Computer Vision Library comprises many such tools and algorithms within the image processing domain that, in fact, stand very much responsible for detecting and counting vehicles.

**iii. Deep Learning Model:**

- **Convolutional Neural Network (CNN):** CNNs are designed to do image classification. One can leverage pre-trained CNN architectures like VGG, ResNet, or a custom one to classify the vehicles based on their features.

**iv. Vehicle Detection and Tracking:**

- **YOLO (You Only Look Once):** This is a real-time object-detection system popular in real-time detection, which provides a very high accuracy level and speed in vehicle recognition in pictures or video streams.
- **Kalman Filters:** The use of Kalman filters could be incorporated in tracking vehicles across frames, thus maintaining a steady count even with occlusions or partial visibility.

**v. Graphical User Interface (GUI**

- **Tkinter or PyQt:** These are Python libraries that can be used to build a human-friendly GUI for human interactions with the system. The GUI will display real-time video feeds, count statistics, and present options to configure system parameters.

**vi. Web Interface (Optional)**

- **Django or Flask:** Using the Python web frameworks, you can develop a web interface in such a way that remote access/monitoring can be given. Users should be able to view the parking lot status and historical data and receive notifications using the web.

## 3.2 NON-FUNCTIONAL REQUIREMENT

Non-functional requirements describe the attributes of a system beyond its specific functionalities. For a project like "Vehicle Counting and Classification for Organized Parking using Machine Learning," non-functional requirements are crucial to ensure that the system operates effectively, efficiently, and securely.

**1) Performance:**

- The system is expected to be capable of processing and categorizing vehicles in real time, with the least possible delay in the identification and counting of vehicles entering and exiting the parking area.
- It should be capable of handling the volume of vehicles during peak hours with no significant loss in performance.
- Classification accuracy should meet predefined thresholds to ensure high reliability in vehicle type identification.

**2) Scalability:**

- The system should be designed to scale horizontally to accommodate an increasing number of parking spaces or entrances/exits.
- It should be capable of handling future expansions or modifications to the parking infrastructure without significant reengineering.

**3) Reliability:**

- A high reliability system that should minimize failures or errors in vehicle detection and classification.
- In other words, it must include mechanisms to deal with unexpected scenarios from adverse weather conditions, poor lighting, or occlusions.

**4) Security:**

- Data privacy and security are paramount. The system should comply with relevant regulations regarding the handling of personal data (e.g., license plate numbers).
- Access to the system's backend, data storage, and administrative interfaces should be appropriately secured to prevent unauthorized access.

**5) Usability:**

- The system should have a user-friendly interface for administrators to configure settings, monitor performance, and generate reports.
- It should provide clear feedback to users (e.g., parking attendants) about the status of vehicle counts and classifications.

**6) Maintainability:**

- The system should be designed with modularity and extensibility in mind, allowing for easy maintenance and updates.
- Proper documentation should be provided for system components, APIs, and data formats to facilitate troubleshooting and future enhancements.

**7) Compatibility:**

- The system should be compatible with several types of cameras, sensors, and hardware that are common in the parking lot.
- It should support integration with existing parking management systems or databases for easy information interchange.

**8) Environmental Constraints:**

- The system should be designed to operate under typical environmental conditions encountered in parking areas, such as varying lighting conditions, weather effects (rain, snow), and temperature fluctuations.

# CHAPTER 4

## SYSTEM DESIGN

The duo datasheets were study, CNRP and PKLot data collections. PKLot, contains about 12416 pictures of parkings spots snatched from multiple parkings lots of varying weather surroundings in which 5959 pictures are occupied parkings spots & remaining 6457 are of empty parkings spots. CNRPark comprises of 150000 labeled patches and it's downloadable but having quite a large size.

Various scenarios of sunlight conditions, including obstructions such as trees, non-parked cars, or lampposts, as well as partially shaded cars, are captured by it. This allows for training a classifier capable of distinguishing between the situations encountered in real-time scenarios. CNRPark has 81,730 pictures of busy parking lots, with 68,270 being of empty parking lots

i)PKLot has pictures spanning across months, capturing various weather conditions throughout the year. In CNRPark, parking spot pictures are generally square and cannot be rotated. They are not capable of precisely or entirely covering the volume of the parking space.

ii) CNRPark consists of occluded areas for the most part, where almost all the parts are occluded due to trees and the shadows of the lampposts. These images are not accounted for in the data collection of PKLot. Lower geographical points account for the images of PKLot, which results in higher occlusions in the images. We compared and validated both algorithms by training and testing them many times while studying two completely varying datasheets!

## 4.1 SYSTEM ARCHITECTURE

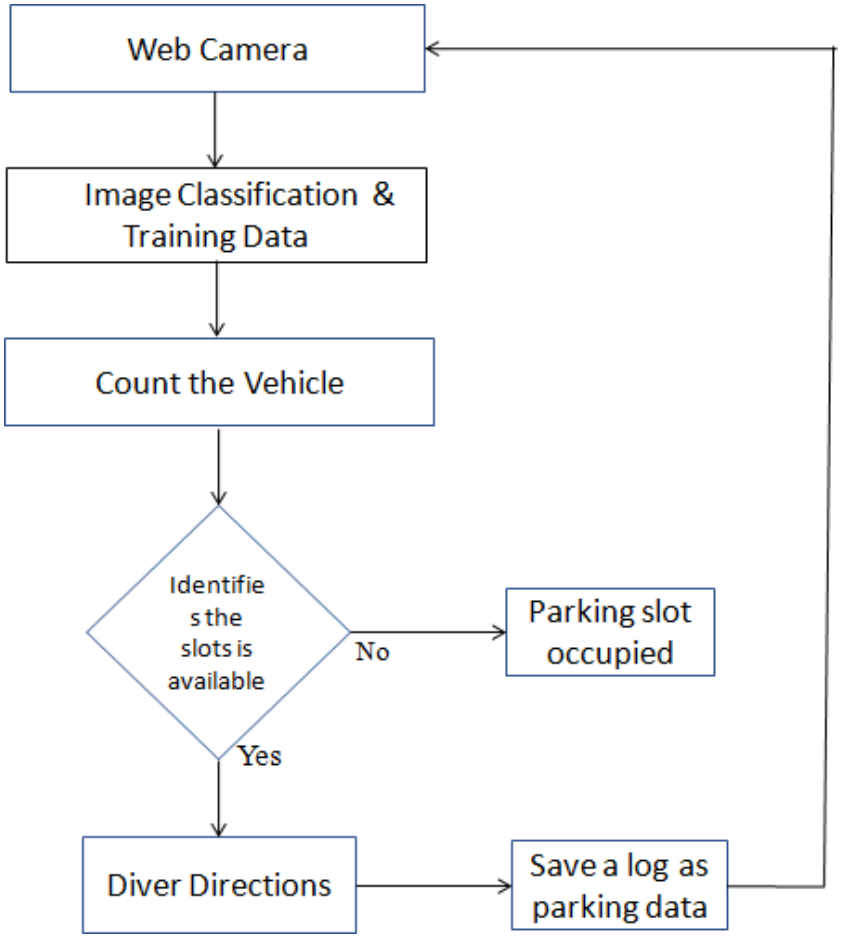



**Fig 4.1 Flowchart of the proposed parking model**

**1) Data Collection and Preprocessing**

- Get a large dataset of images, or video recordings, on the entrance and exit of vehicles in and out of the parking space. In this dataset, multiple types of lighting conditions, angles, and vehicle types should be represented.
- Preprocess the data in order to normalize the images, denoise, and provide image annotations with labels that reflect the type of vehicle (e.g., car, truck, motorcycle) and its entrance/exit status.

**2) Feature Extraction:**

- Extract relevant features of the images or video frames, such as the color histograms, edge features, texture features, etc.
- In more complex systems, you can also make use of deep learning methods for automatic feature extraction; a few examples include convolutional neural networks.

**3) Model Training**

- Machine learning-based models taught to count and categorize the types of vehicles based on the extracted features.
- Counting vehicles with object detection or object segmentation approaches that localize and count individual vehicles in images or frames of a video.
- Use a classifier to classify the types of vehicles based on its characteristic**.**

**4) Integration with Sensor Systems**

- Deploy the machine learning models onto the sensor systems located at the entry and exit points of the parking space.
- These may be cameras, LIDAR, or any other potential sensors capable of collecting vehicle data in real-time.

**5) Reak time Processing**

- Operationalization of the real-time processing of the data captured by the sensors.
- Develop efficient algorithms and data structures to process the incoming data quickly and with accuracy.

**6) Vehicle Tracking**

- Implement vehicle tracking algorithms to monitor vehicle movement in the parking space. • It can be achieved by attaching, with every vehicle entry and exit events, the update of the status of vehicles in terms of position inside the parking facility**.**

**7) User Interface and Reporting:**

- Develop a user interface that provides real-time information about the number and types of vehicles in the parking area.
- Generate reports and analytics to help parking managers optimize parking space allocation and monitor traffic flow.

**8) Testing and Evaluation:**

- Test the system extensively under different conditions to verify that it is accurate, reliable, and scalable.
- Use performance metrics such as precision, recall, and the F1 score to carry out assessment of the vehicle counting and classification algorithms.

**9) Deployment and Maintenance:**

- Deploy the system in the parking facility and monitor its performance in real-world conditions.
- Regularly update and maintain the system to address any issues and improve its performance over time.

### 4.1.1 YOLO ALGORITHM

YOLO is an abbreviation for "You Only Look Once," and it is a pioneering object detection system in machine learning and deep learning but more so in computer vision tasks.

YOLO is a single-shot object detection algorithm, which means it looks at the whole image to predict objects and their bounding boxes, rather than relying on region proposals, as used by all prior methods.

CNN Architecture: YOLO architecture is based on implementing the backbone of a Convolutional Neural Network (CNN) using architectures like Darknet or any other deep neural network.

It divides the input image into a grid and predicts the bounding boxes, confidence scores, and class probabilities for each grid cell.

Real-Time Detection: YOLO is known for its speed and efficiency, especially in real-time object detection applications. It can process images swiftly, which is good for scenarios where predictions are supposed to be real-time.

The trade-off between speed and accuracy: While YOLO maintains strength in speed, it may somewhat compromise in accuracy compared to slower, more accurate methods.

Different Variants: A number of versions of YOLO—such as YOLOv2, YOLOv3, YOLOv4—work on a very fine balance between speed and accuracy; with every new release, there is betterment in architecture and enhancement in the performance of the.

**1) Image Detection**

- Image detection by the YOLO (You Only Look Once) is a technique for object detection in computer vision that is famous for its speed and accuracy.
- YOLO is devised to perform real-time object detection in an image or video frame. Unlike traditional methods including multiple region proposals and stages of classification, YOLO does this in a single forward pass, making it much faster.
- YOLO divides the input image into a grid, thereby making predictions based on this grid. It is the responsibility of individual cells within the grid to make predictions for bounding boxes, with their corresponding confidence scores and class probabilities.
- YOLO is typically architecture with a convolutional neural network backbone followed by detection layers responsible for prediction of bounding boxes and class probabilities.

**2) Vehicle Counting**

- Vehicle counting with YOLO: In here, the YOLO algorithm is used for accurate detection and counting of vehicles in different scenarios.
- Object Detection for Vehicles: Object detection capability is precisely why YOLO is preferred; it can be trained or used to detect vehicles from images or video frames.
- YOLO detects vehicles by predicting bounding boxes around them in the given input. Each bounding box represents a detected vehicle instance.
- Counting Mechanism: Based on the output of the YOLO model, the counting process considers the number of bounding boxes detected by the algorithm, which represents a vehicle. In another sense, the bounding box which will be detected in a frame will be counted to make a decision on the count of vehicles within that scene or frame.

### 3) Slot Identification

- YOLO V3 makes use of the network structure of Darknet53, which contains a total of 53 convolution layers. Darknet53 utilizes a total of five residual blocks, by borrowing the idea of the ResNet neural network.
- Each residual block has many residual units, and a residual unit is created by feeding residual operations with two input DBL units.
- Within which, the DBL unit has convolution, batch normalization, and leaky ReLU activation functions. The depth of the network can be made deeper, and the vanishing gradient can be avoided by introducing a residual unit.

### 4) Driver Directions

- Audio Output System: Implement an audio output system, such as speakers strategically placed within the parking area or connected to a central guidance system. Ensure these speakers are easily audible to drivers.
- Voice Guidance Integration: Develop or integrate a voice guidance system capable of providing clear and concise instructions to drivers. This system should relay information regarding available spots, their locations, and directions to reach them.
- Real-Time Updates: Continuously update the audio guidance system in real-time as parking spots become available or occupied. The system should dynamically adapt directions based on changing parking conditions

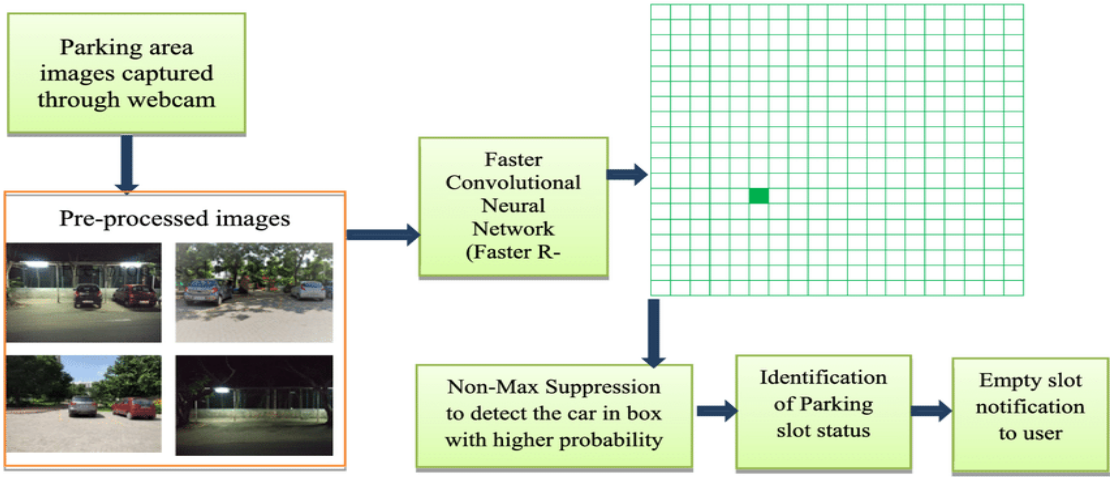



**Fig 4.1.1 Design of parking space detection**

## 4.2 METHODOLOGY

The applied algorithms are mostly designed based on neural network models, where the neurons are organized in a stacked layer. The system is designed for managing the incoming and outgoing vehicles by collecting and processing the images and data on passenger information to update the parking status with the news of empty lots.

1. **Data Collection**
    - The first step is to collect a sufficiently large dataset of images or videos for vehicles entering the parking lot and exiting the parking lot. This dataset should encompass various lighting conditions, weather conditions, vehicle types, and angles.
2. **Preprocessing**
    - The collected data are pre-processed to improve the quality and make it ready for training. These are done by resizing images, normalization, and noise reduction, as well as increasing data augmentation to improve the diversity of the dataset.
3. **Object Detection:**
    - Object identification within the images or video frames and vehicle localization: some popular algorithms that can be used for identifying the vehicles are YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), and Faster R-CNN (Region-based Convolutional Neural Network).
4. **Classification**
    - Following the detection of the vehicles, it is pertinent that they are classified to fall into one of the many available categories, such as cars, trucks, and motorbikes, among others. This can be implemented using convolutional neural networks trained on the collected dataset.
5. **Counting**
    - The system calculates the number of vehicles in a parking lot by detecting and classifying them. This may range from a very simple count of classified vehicles to a much more sophisticated method that uses object tracking to avoid double counting.

6. **Integration and Deployment**
    - The developed model is integrated into the parking management system. It could be deployed on-premises or on a cloud platform based on the requirements. If the parking space environment is dynamic, real-time processing capability might be needed.

7. **Testing and Evaluation**
    - The system is moved through rigorous testing using synthetic and real-world data to ensure veracity, robustness, and efficiency. Performance can be assessed using metrics such as precision, recall, and F1 score.

8. **Iterative Improvement**
    - The fine-tuning and optimization of the model are carried out iteratively with the evaluation results to overcome the shortcomings shown and to improve performance progressively over time. This can be through data collection, refining prepossessing techniques, or model parameter tweaking.

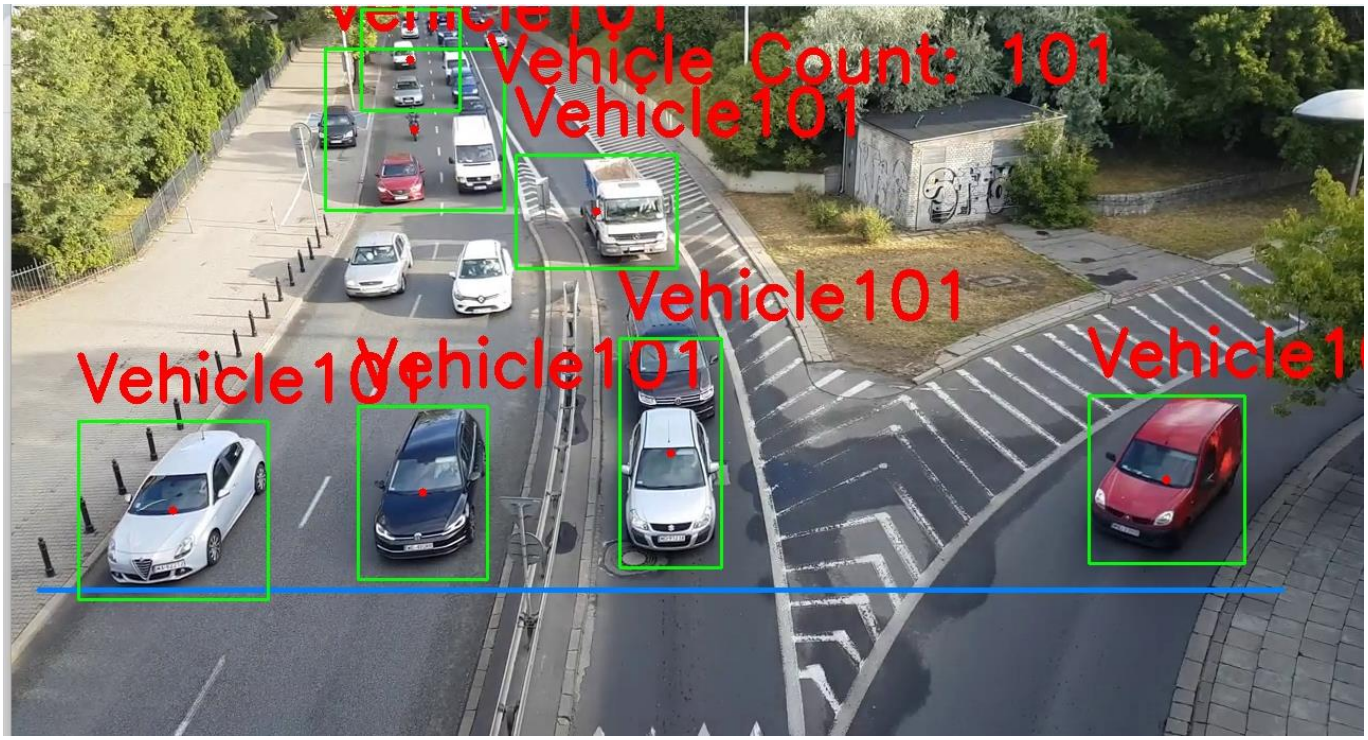



**Fig 4.2 Vehicle counting and Classification**

# CHAPTER 5

# IMPLEMENTATION

## 5.1 Data Collection:

- Collect a dataset consisting of pictures and/or videos of vehicles in a parking lot. The dataset should cover the different conditions of vehicles, including various vehicle types such as cars, trucks, bikes, etc., under different illumination conditions, various viewpoints, and occlusions.
- Annotate the dataset with bounding boxes to each vehicle and tag each of the classes appropriately (i.e., car, truck, bike, etc.).

## 5.2 Preprocessing:

- Size all images equally for consistency.
- Standardize the pixel values of the images to have zero mean and unit variance.
- Increase the diversity and robustness of the dataset through techniques like rotation, flipping, brightness adjustments, etc.

## 5.3 Model Selection:

- Choose an eligible deep learning model capable of identifying and classifying objects. In general, the used models include YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), or Faster R-CNN to deal with this kind of problem.
- Consider the trade-off between speed and accuracy based on your project requirements.

## 5.4 Training

- Splitting the dataset into training, validation, and test sets.
- Train the model chosen on the training data using an appropriate loss function, like cross-entropy loss for classification, or else a joint loss of localization loss and confidence loss for object detection.
- Tuning hyperparameters like the learning rate, batch size, number of epochs to achieve optimized model performance.
- Track the model's performance on the validation set and adjust the hyperparameters if needed to avoid overfitting.

## 5.5 Evaluation:

- Evaluate the trained model on the test set, where performance is judged by accuracy, precision, recall, and F1 score for both vehicle detection and classification.
- Use metrics, e.g., mAP (Mean Average Precision), to quantify the performance level of the object detection model.
- Analyze the performance of the model on different types of vehicles and under varying conditions, such as day vs. night or a crowded vs. empty parking lot.

## 5.6 Deployment:

- Deploy the trained model into a software system that is deployable with real-time inference.
- Build an interface, could be a website or mobile application, that will allow users to interact with the system.
- Implement the system in the parking lot environment and test it to know how it will perform in the field.
- Implement features such as live streaming of video feeds, real-time counting of vehicles, and parking violation alerts when deemed necessary.

## 5.7. Continuous Improvement:

- Gather user feedback and observe system performance in production.
- Model fine-tuning with in-operation data means, for example, the use of techniques such as transfer learning.
- Continue updating the model continuously to guarantee improvement in accuracy and adaptability to the changing conditions.

## 5.8 Code Snippet

### 5.8.1. Update_Mapper.py

```
from flask import Flask, request, redirect, url_for, flash, send_from_directory, render_template,
render_template_string
import os
from werkzeug.utils import secure_filename
import cv2
from parkingDetections.models.models import ParkingSpace, db
from parking_space_mapper import mark_parking_lots, save_to_text, save_to_text_uploadFolder
from flask_sqlalchemy import SQLAlchemy
import logging
# Configure logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
UPLOAD_FOLDER = 'uploads'
ALLOWED_EXTENSIONS = {'mp4'}
app = Flask(__name__)
app.secret_key = 'POOJAVVCE'  # Set your secret key here
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///parking.db'
db.init_app(app)
# Ensure the uploads folder exists
if not os.path.exists(UPLOAD_FOLDER):
    os.makedirs(UPLOAD_FOLDER)
def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS
@app.route('/', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        if 'file' not in request.files:
            flash('No file part')
            return redirect(request.url)
        file = request.files['file']
        if file.filename == '':
            flash('No selected file')
            return redirect(request.url)
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
            file.save(file_path)
            # Call the marking function
            cap = cv2.VideoCapture(file_path)
            ret, frame = cap.read()
            if not ret:
                flash('Error reading video frame.')
                return redirect(request.url)
            frame = cv2.resize(frame, (1020, 500))
```

```
        marker_locations = mark_parking_lots(frame.copy())
        # Log marker_locations
        logging.info(f"Marker Locations: {marker_locations}")
        # save for the file
        save_to_text_uploadFolder(marker_locations, filename, app.config['UPLOAD_FOLDER'])
        cap.release()
        # Create database entries for parking spaces
        db.create_all()
        # Delete all records from the ParkingSpace table
        ParkingSpace.query.delete()
        # Assuming marker_locations is a list of tuples of coordinates
        for i in range(0, len(marker_locations), 4):
            try:
                # Extract coordinates for each parking space
                corner1, corner2, corner3, corner4 = marker_locations[i:i + 4]
                x1, y1 = corner1
                x2, y2 = corner2
                x3, y3 = corner3
                x4, y4 = corner4

                # Concatenate the coordinates into a single string
                coordinates = f"{x1} {y1} {x2} {y2} {x3} {y3} {x4} {y4}"
                parking_space = ParkingSpace(coordinates=coordinates)
                db.session.add(parking_space)
            except ValueError:
                flash('Error parsing coordinates.')
                return redirect(request.url)
        db.session.commit()
        # Retrieve parking spaces from the database
        spaces = ParkingSpace.query.all()
        # Render the marked spaces on a webpage
        return render_template('parking_spaces.html', spaces=spaces)
    return '''
    <!doctype html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Upload Video</title>
    </head>
    <body>
        <form action="/" method="POST" enctype="multipart/form-data">
            <input type="file" name="file">
            <input type="submit" value="Upload">
        </form>
    </body>
    </html>
@app.route('/parking-spaces')
def parking_spaces():
```

```
    spaces = ParkingSpace.query.all()
    return render_template('parking_spaces.html', spaces=spaces)


if __name__ == '__main__':
    app.run(debug=True)
```

### 5.8.2 Parking_entrance_detection.py

```
import cv2
import numpy as np
from ultralytics import YOLO
from models.models import ParkingSpace, db
from pathlib import Path
from flask import Flask
UPLOAD_FOLDER = 'uploads'
app = Flask(__name__)
app.secret_key = 'POOJAVVCE'  # Set your secret key here
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///parking.db'
db.init_app(app)
# Load parking spaces from the database
def load_parking_spaces():
    with app.app_context():
        parking_spaces = []
        for space in ParkingSpace.query.all():
            coords = list(map(int, space.coordinates.split()))
            parking_spaces.append([(coords[i], coords[i + 1]) for i in range(0, len(coords), 2)])
        return parking_spaces
# Read entrance line coordinates from file
def read_entrance_line_from_file(video_filename):
    entrance_line_filename = f"{video_filename}_entrancemap.txt"
    try:
        with open(entrance_line_filename, 'r') as file:
            line = file.readline().strip()
            coords = list(map(int, line.split()))
            entrance_line = [(coords[0], coords[1]), (coords[2], coords[3])]
            return entrance_line
    except FileNotFoundError:
        print(f"Entrance line file '{entrance_line_filename}' not found.")
        return None
# Check database for available parking spaces
def check_available_spaces():
    with app.app_context():
        available_spaces = [space.id for space in ParkingSpace.query.filter_by(status='empty').all()]
    return available_spaces
def main(input_video_path):
    # Load the YOLOv8 model
```

```
model = YOLO('yolov8n.pt')
cap = cv2.VideoCapture(input_video_path)
if not cap.isOpened():
    print(f"Error: Failed to open video '{input_video_path}'")
    return
video_filename = Path(input_video_path).stem
parking_spaces = load_parking_spaces()
entrance_line = read_entrance_line_from_file(video_filename)
if entrance_line is None:
    return
# Initialize variables for tracking
previous_midpoints = {}  # Dictionary to store previous midpoints of tracked vehicles
line_position = entrance_line
crossed_vehicles = set()  # Set to track vehicles that have already crossed the line
# Process the video frames
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame, (1020, 500))
    # Draw the crossing line on the frame
    cv2.line(frame, line_position[0], line_position[1], (0, 255, 0), 2)
    # Run tracking on the current frame
    results = model.track(frame, persist=True)
    # Check if tracking IDs are available
    if hasattr(results[0].boxes, 'id') and results[0].boxes.id is not None:
        boxes = results[0].boxes.xyxy.cpu().numpy().astype(int)
        ids = results[0].boxes.id.cpu().numpy().astype(int)

        # Check for line crossing by vehicle midpoints
        for box, id in zip(boxes, ids):
            midpoint_x = (box[0] + box[2]) // 2
            midpoint_y = (box[1] + box[3]) // 2
            if id in previous_midpoints:
                previous_midpoint_x = previous_midpoints[id][0]
                previous_midpoint_y = previous_midpoints[id][1]
            else:
                previous_midpoint_x = midpoint_x
                previous_midpoint_y = midpoint_y
            # Check if the midpoint crosses the line and the vehicle has not crossed before
            if (midpoint_y > line_position[0][1] and previous_midpoint_y <= line_position[0][1]) or \
            (midpoint_y < line_position[0][1] and previous_midpoint_y >= line_position[0][1]):
                if id not in crossed_vehicles:
                    crossed_vehicles.add(id)
                    print(f"Car {id} is entering the parking lot. Checking parking space status...")
                    available_spaces = check_available_spaces()
                    print(f"Parking Spaces Available: {available_spaces}")
                    # Draw a bounding box around the vehicle that crossed the line
                    cv2.rectangle(frame, (box[0], box[1]), (box[2], box[3]), (255, 0, 0), 2)
```

```
                cv2.putText(frame, f"Car {id} - Crossed", (box[0], box[1]), cv2.FONT_HERSHEY_SIMPLEX, 0.8,
(255, 0, 0), 2)
                # Pause the video until a key press
                cv2.waitKey(0)
            # Update previous midpoint for the current vehicle
            previous_midpoints[id] = (midpoint_x, midpoint_y)
        # Draw boxes and IDs on the frame for all tracked vehicles
        for box, id in zip(boxes, ids):
            cv2.rectangle(frame, (box[0], box[1]), (box[2], box[3]), (0, 255, 0), 2)
            cv2.putText(frame, f"Car {id}", (box[0], box[1]), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 2)
    else:
        print("Tracking IDs not available.")
    # Display the frame
    cv2.imshow("frame", frame)
    # Check for 'q' key press to exit
    if cv2.waitKey(1) & 0xFF == ord("q"):
        break
# Release resources
cap.release()
cv2.destroyAllWindows()
if __name__ == "__main__":
    input_video_path = 'parking1_entrance.mp4'
    main(input_video_path)
```

### 5.8.3 Parking_entrance_mapper.py

```
import cv2
def mark_entrance_line(image):
    clone = image.copy()
    marker_locations = []
    entrance_line = None
    def draw_entrance_line(clone, marker_locations):
        if len(marker_locations) == 2:
            cv2.line(clone, marker_locations[0], marker_locations[1], (0, 128, 0), 2)
    def mouse_callback(event, x, y, flags, param):
        nonlocal entrance_line
        if event == cv2.EVENT_LBUTTONDOWN:
            marker_locations.append((x, y))
            if len(marker_locations) == 2:
                entrance_line = marker_locations[:]  # Copy the coordinates for the line
                draw_entrance_line(clone, marker_locations)
    cv2.namedWindow('Mark Entrance Line')
    cv2.setMouseCallback('Mark Entrance Line', mouse_callback)
    while True:
        cv2.imshow('Mark Entrance Line', clone)
        key = cv2.waitKey(1) & 0xFF
        if key == ord('r'):  # Press 'r' to reset markers
            clone = image.copy()
```

```
            marker_locations.clear()
            entrance_line = None
        elif key == ord('c') and entrance_line is not None:  # Press 'c' to capture entrance line
            break
    cv2.destroyAllWindows()
    return entrance_line
def save_entrance_line_to_text(entrance_line, video_filename):
    if entrance_line:
        map_filename = f"{video_filename.split('.')[0]}_entrancemap.txt"
        with open(map_filename, 'w') as file:
            file.write(f"{entrance_line[0][0]} {entrance_line[0][1]} {entrance_line[1][0]}
{entrance_line[1][1]}")
        print(f"Entrance line saved to '{map_filename}'")
    else:
        print("No entrance line marked.")
def main():
    video_filename = 'parking1_entrance.mp4'  # Change this to your video filename
    cap = cv2.VideoCapture(video_filename)
    ret, frame = cap.read()
    if not ret:
        print("Error: Failed to read frame from video.")
        return
    frame = cv2.resize(frame, (1020, 500))
    entrance_line = mark_entrance_line(frame.copy())
    save_entrance_line_to_text(entrance_line, video_filename)
    cap.release()
if __name__ == "__main__":
    main()
```

### 5.8.4 Updated_parking_space_detections.py

```
import cv2
import numpy as np
from ultralytics import YOLO
# from yolov5 import YOLOv5
import time
from pathlib import Path
from flask import Flask, request, redirect, url_for, flash, send_from_directory, render_template,
render_template_string
from flask_sqlalchemy import SQLAlchemy
import logging
from models.models import ParkingSpace, db
UPLOAD_FOLDER = 'uploads'
app = Flask(__name__)
```

```
app.secret_key = 'POOJAVVCE'  # Set your secret key here
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///parking.db'
db.init_app(app)

def load_parking_areasDatabase():
    parking_areas = []
    for space in ParkingSpace.query.all():
        coords = list(map(int, space.coordinates.split()))
        parking_areas.append([(coords[i], coords[i + 1]) for i in range(0, len(coords), 2)])
    return parking_areas, len(parking_areas)
#this will update parking space
def update_parking_space_status(available_spaces):
    for i, is_available in enumerate(available_spaces):
        space = ParkingSpace.query.get(i + 1)
        if is_available:
            space.status = 'empty'
        else:
            space.status = 'occupied'
    db.session.commit()
def print_parking_space_status():
    for space in ParkingSpace.query.all():
        print(f"Parking Space {space.id}: {space.status}")
#end of new functions
def load_parking_areas(video_filename):
    map_filename = f"{video_filename}_map.txt"
    parking_areas = []
    try:
        with open(map_filename, 'r') as file:
            lines = file.readlines()
            for line in lines:
                coords = list(map(int, line.strip().split()))
                parking_areas.append([(coords[i], coords[i + 1]) for i in range(0, len(coords), 2)])
    except FileNotFoundError:
        print(f"Map file '{map_filename}' not found.")
    return parking_areas, len(parking_areas)
#old model , changing to new parkingDetections
def detect_cars(frame):
    model = YOLO('yolov8s.pt')
    results = model.predict(frame, conf=0.35)
    return results[0].boxes.data

def calculate_overlap(area, car):
    x1, y1, x2, y2 = map(int, car[:4])  # Convert coordinates to integers
    rect = np.array(area, np.int32)
```

```
    rect = rect.reshape((-1, 1, 2))
    intersection = cv2.pointPolygonTest(rect, ((x1 + x2) // 2, (y1 + y2) // 2), False)
    if intersection >= 0:
        return cv2.contourArea(rect)
    else:
        return 0
def count_available_spaces(parking_areas, car_boxes, car_label=2):
    NUM_SPACES = len(parking_areas)
    available_spaces = [True] * NUM_SPACES
    total_cars_detected = 0
    total_occupied_spaces = 0
    cars_in_parking = []  # List to store bounding boxes of cars in parking spaces
    car_boxes_filtered = []
    if car_boxes is not None and len(car_boxes) > 0:
        car_boxes_filtered = [box for box in car_boxes if int(box[5]) == car_label]
        occupied_spaces = set()  # Set to store indices of parking spaces occupied by cars
        # Identify occupied parking spaces
        for car in car_boxes_filtered:
            total_cars_detected += 1
            max_overlap = -1
            max_index = -1
            for i, area in enumerate(parking_areas):
                overlap = calculate_overlap(area, car)
                if overlap > max_overlap:
                    max_overlap = overlap
                    max_index = i
            if max_overlap > 0:  # Only consider cars overlapping with parking spaces
                occupied_spaces.add(max_index)
                total_occupied_spaces += 1
                cars_in_parking.append(car)  # Store bounding box of car in parking space
        # Mark occupied parking spaces as unavailable
        for space_index in occupied_spaces:
            available_spaces[space_index] = False
    total_free_spaces = NUM_SPACES - total_occupied_spaces
    return available_spaces, total_cars_detected, total_occupied_spaces, total_free_spaces,
car_boxes_filtered, cars_in_parking

def main():
    # Create an application context
    with app.app_context():
        input_video_path = 'occupiedParkingSpace.mp4'
        input_video_name = Path(input_video_path).stem
        cap = cv2.VideoCapture(input_video_path)
        # Now you can safely call load_parking_areasDatabase() within the application context
        PARKING_AREAS, NUM_SPACES = load_parking_areasDatabase()
```

```
    frame_count = 0
    skip_frames = 5 # Adjust this value to skip more or fewer frames
    while True:
        ret, frame = cap.read()
        if not ret:
            break
        frame = cv2.resize(frame, (1020, 500))
        # Skip processing for some frames to speed up the process
        if frame_count % skip_frames == 0:
            car_boxes = detect_cars(frame)
            available_spaces, total_cars_detected, total_occupied_spaces, total_free_spaces,
car_boxes_filtered, cars_in_parking = count_available_spaces(
                PARKING_AREAS, car_boxes)
            # Update parking space status in the database
            update_parking_space_status(available_spaces)
            # Print parking space status
            print_parking_space_status()
            # Display the result
            for i, area in enumerate(PARKING_AREAS):
                color = (0, 255, 0) if available_spaces[i] else (0, 0, 255)
                cv2.polylines(frame, [np.array(area, np.int32)], True, color, 2)
                overlay = frame.copy()
                cv2.fillPoly(overlay, [np.array(area, np.int32)], color)
                alpha = 0.3
                frame = cv2.addWeighted(overlay, alpha, frame, 1 - alpha, 0)
                cv2.putText(frame, str(i + 1), ((area[0][0] + area[1][0] + area[2][0] + area[3][0]) // 4,
                                        (area[0][1] + area[1][1] + area[2][1] + area[3][1]) // 4),
                            cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)

            cv2.putText(frame, f'Total cars detected: {total_cars_detected}', (20, 40),
cv2.FONT_HERSHEY_SIMPLEX, 1,
                        (255, 255, 255), 2)
            cv2.putText(frame, f'Total occupied parking spaces: {total_occupied_spaces}', (20,
80),
                        cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
            cv2.putText(frame, f'Total free parking spaces: {total_free_spaces}', (20, 120),
                        cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
            cv2.imshow('Processed Video', frame)
            cv2.waitKey(1)
        frame_count += 1
    cap.release()
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

# CHAPTER 6

## RESULTS & SNAPSHOTS


**Fig 6.1 Uploading the empty parking slot video**

This Flask application takes a video file recording a parking lot. When the video uploaded into this application gets processed, the application detects parking places in that video using computer vision. It then saves the detected parking space coordinates to a database. It also consists of a web interface in which the detected parking spaces are overlayed on the parking lot image. The main functionalities of the application include:

- **Upload Functionality:** Users can upload a video file through a web form.
- **Video Processing:** The uploaded video is processed frame by frame to detect parking spaces.
- **Database Integration:** Detected parking space coordinates are stored in a SQLite database.
- **Web Interface:** Users can view the detected parking spaces on a webpage rendered using Flask's templating system.
- **Logging:** The application logs important events such as errors and marker locations.
- **Error Handling**: Flash messages are used to notify users about errors during the upload and processing of the video.


**Fig 6.2 Mapping the slots**

**Setup and Initialization:** The script initializes necessary components such as the Flask web application, database connection, and file upload configuration.

**Functions for Parking Lot Management:** It includes functions to load parking space coordinates from a database, read entrance line coordinates from a file, and check the availability of parking spaces.

**Main Processing Loop:** The main () function processes each frame of the input video. It resizes the frame, draws a line representing the entrance, and tracks vehicles using the YOLOv8 model. It checks for vehicles crossing the entrance line and identifies available parking spaces when a vehicle enters. Bounding boxes and IDs are drawn around detected vehicles.

**Video Processing and Display:** The script iterates through each frame of the video, performs object tracking, and updates the display with bounding boxes and IDs. It waits for a key press to proceed to the next frame.

**Resource Cleanup:** Once video processing is complete, the script releases the video capture resources and closes any open windows.

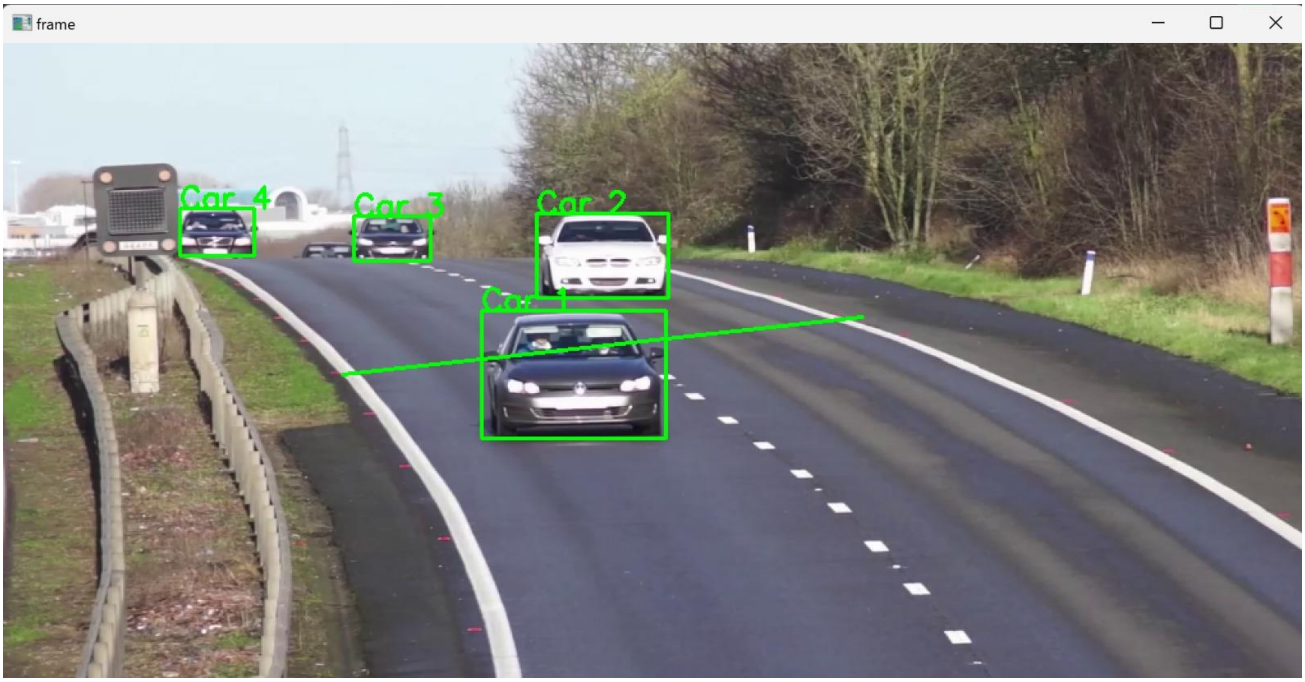



**Fig 6.3 Parking entrance detection**

- This project endeavors to automate the observation of a parking area through computer vision techniques. YOLO is used to detect cars in a video stream of a parking lot. Detected cars are then matched with predefined parking areas for occupancy status.
- The designed system has the features to load parking areas from a database and update the status of a parking space dynamically. The results will be displayed in real time on a video feed.
- It also uses Flask in developing a web application where users can interact with the system in a user-friendly manner. Overall, the project aims at effectual monitoring and management of parking spaces.


**Fig 6.4 Available slots**

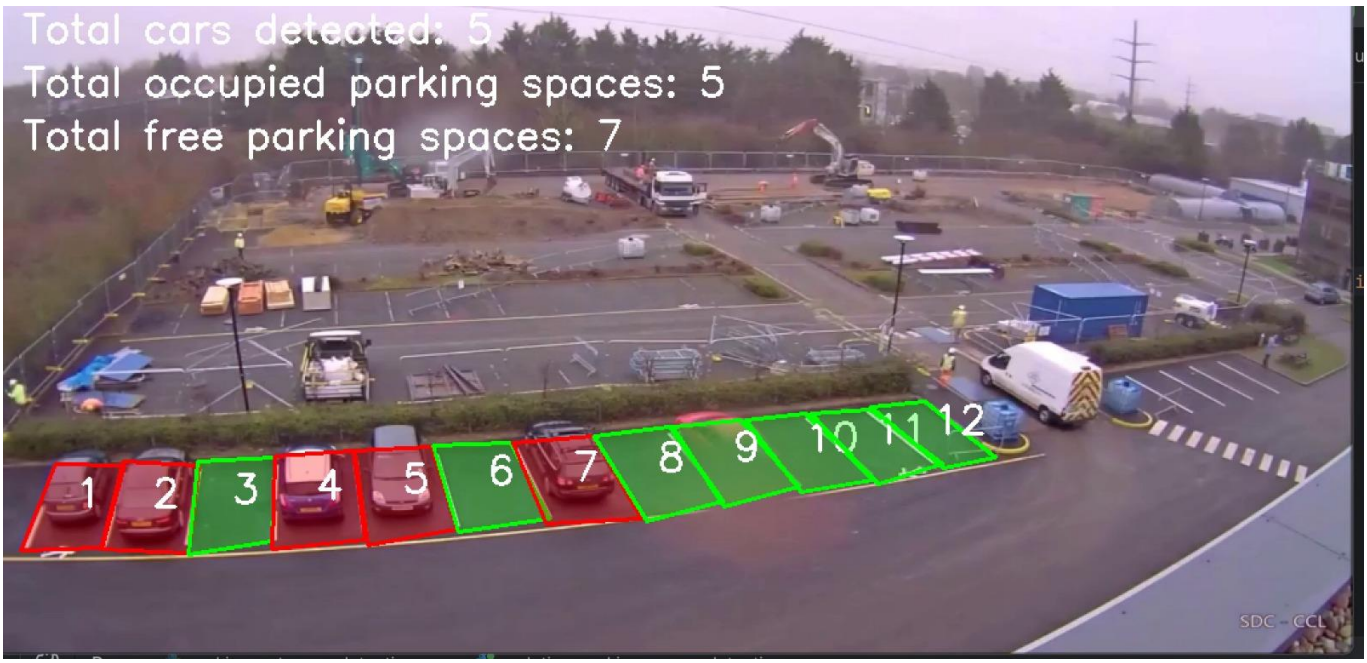



**Fig 6.5 Updated parking space detection**

- Libraries Importing: The code begins by importing the OpenCV library, which will be utilized for image and video processing.
- Function mark_entrance_line: This function takes an image, basically one frame of video, and allows a user to mark an entrance line on this image by clicking twice on the image. It will open a window in which the user is supposed to click on two points to mark an entrance line. These two points are saved to the list marker locations.

```
0: 320x640 7 cars, 3 trucks, 185.1ms
Speed: 3.3ms preprocess, 185.1ms inference, 1.2ms postprocess per image at shape (1, 3, 320, 640)
Parking Space 1: occupied
Parking Space 2: occupied
Parking Space 3: empty
Parking Space 4: occupied
Parking Space 5: occupied
Parking Space 6: empty
Parking Space 7: occupied
Parking Space 8: empty
Parking Space 9: occupied
Parking Space 10: empty
Parking Space 11: empty
Parking Space 12: empty
```

**Fig 6.6 Updated slots**

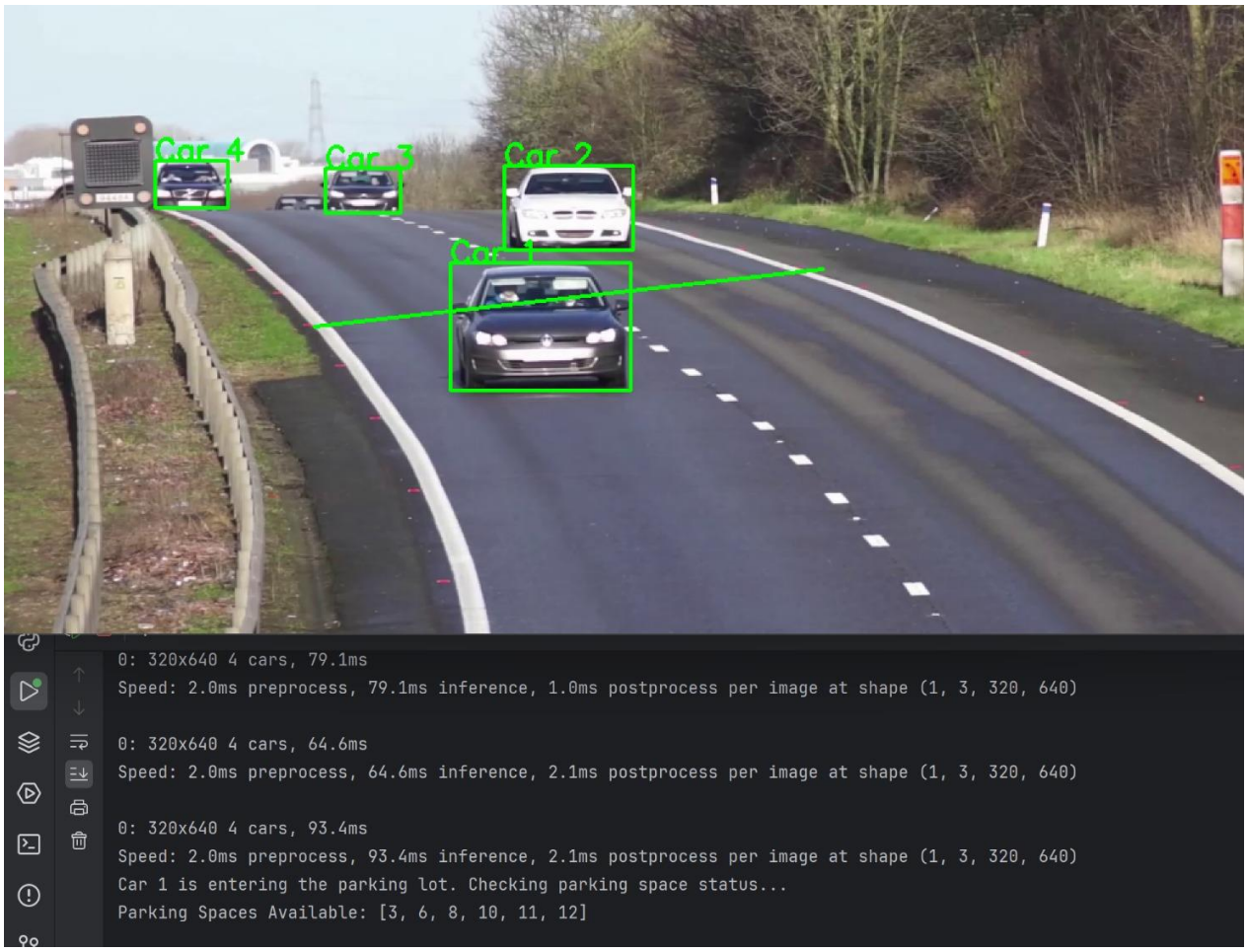



**Fig 6.7 Updated parking entrance and available slots**

- draw_entrance_line Function: This function draws a line between two points where the user specifies the.
- mouse_callback Function: A function called whenever a mouse event happens within the window. For mouse click events, it picks coordinates and appends them into marker_locations. When two points are marked, it sets an entrance line to the coordinates of the line and calls draw_entrance_line to draw the line on the image.
- save_entrance_line_to_text Function: Reads an entrance line coordinates and video filename, and saves the entrance line coordinates to a file with a filename obtained from the video filename.

# CHAPTER 7

# CHALLENGES

## 7.1 COMPLEXITY

• Variability in Vehicle Types: Vehicles come in all shapes, sizes, and types, be it cars, trucks, buses, motorcycles, bicycles, and many more. Each one of them may have distinctive features, some of which are hard to be distinguished and classified with the highest precision.

• Different Camera Viewpoints and Resolutions: Cameras may view the vehicles from different angles, distances, or under different lighting conditions. This variation of perspectives may not allow the model to be robust and perform well across different scenarios.

• Crowded Scenes: Roads can be crowded in urban environments, with multiple vehicles moving closely one behind the other. This also can lead to occlusions where vehicles partially block each other, making accurate counting and classification a major challenge.

• Dynamic Environments: The traffic conditions are dynamic in nature, as the vehicles are seen in different speeds, changing lanes, or entering and leaving the scene. In such dynamic environments, the ability to process information on the fly and adapt the model in view is important for accurate tracking and classification of the vehicles.

• Weather Conditions: Unfavorable weather conditions, such as rain, fog, or snow, may introduce significant quality degradation in the image and visibility, which poses problems for the correct detection and classification of vehicles.

## 7.2 BACKGROUND CLUTTER

- Background Clutter: All the non-vehicle objects—pedestrians, bicyclists, trees, buildings, signs, and so forth—contribute to background clutter that could cause a false detection or a wrong classification.
- Scale Variation: A vehicle can appear at various scales in the image, due to its location with respect to the camera. Doing this well is important so that counting and classification can be done correctly in the presence of scale variation.
- Annotated Data Availability: There is a requirement for a large amount of annotated data to make accurate machine learning models for vehicle counting and classification. This may have to be acquired, labeled, and is time-consuming and costly.

- Computational Resources: The need for real-time processing of high-resolution video feeds from multiple cameras requires very significant computational resources. All algorithms used must be very efficient and possibly require hardware acceleration to meet their performance requirements.
- Privacy Concerns: In general, vehicle counting and classification systems deployed in public spaces raise a plethora of privacy concerns, more so when it comes to the collection and processing of personally identifiable information. This has to be ensured that it is compliant with privacy regulations.

# CHAPTER 8

# FUTURE ENHANCEMET

## 8.1 SCOPE

This project is on vehicle counting and classification using machine learning. The project delves into many aspects of vehicle counting and classification to achieve better traffic management, surveillance, and urban planning systems. This project, with the help of machine learning algorithms, will aim to develop a robust framework that can detect, count, and classify vehicles from various sources such as CCTV cameras, drones, or road-embedded sensors. There are also elements that go on to involve the application of computer vision techniques that analyze the video feeds or images in real time in order to enable the system to identify the vehicles involved, such as cars, buses, trucks, motorcycles, and bicycles. The project can further include some algorithms optimized for scalability, adaptability to various environments, and efficient use of computational resources. It will further be important to integrate the solution into the existing traffic management systems and, in general, the smart city infrastructure to enable decision-making processes based on data, relieve congestion, improve safety, and further urban mobility. Lastly, privacy, security of data, and issues of ethical consideration should make part of the scope so as to ensure responsible deployment and operation of the system.

## CONCLUSION

Parking management systems leverage technology to address contemporary parking challenges, offering intelligent and seamless solutions. By employing advanced tech, they enhance parking experiences, promoting convenience, safety, and environmental sustainability. These systems optimize parking methods, leading to time and cost savings for all stakeholders. Furthermore, automated car parking systems efficiently maneuver vehicles to mitigate congestion, enabling the optimal utilization of parking facilities. In summary, automated car parking systems deliver a range of advantages, including enhanced safety, reduced costs, and maximized space utilization.

# REFERENCES

1) https://www.mdpi.com/2076-3417/13/5/3059

2) https://www.mdpi.com/2673-8732/2/2/15

3) https://www.sciencedirect.com/science/article/abs/pii/S0957417423001872

4) https://doi.org/10.1016/j.eswa.2022.117929

5) https://www.mdpi.com/1999-4893/14/4/114

6) https://www.mdpi.com/1999-5903/11/4/94

7) https://doi.org/10.1016/j.trc.2013.11.014

8) https://doi.org/10.1016/j.jvcir.2019.05.015

9) https://ieeexplore.ieee.org/abstract/document/10006795

10) https://www.mdpi.com/1424-8220/23/13/5843

# Organized Parking System Using Machine Learning

NATARAJ S L, NAVYASHREE S, POOJA S
Department of Information Science and Engineering
Vidyavardhaka College of Engineering Mysuru, India

**Abstract: - Parking management is a critical aspect of urban infrastructure, often plagued by inefficiencies and congestion. This research proposes a novel approach to optimize parking space utilization using machine learning techniques. The system integrates various sensors, such as cameras and ultrasonic devices, to collect real-time data on parking space availability, vehicle entry, and exit within parking lots. The data collected undergoes preprocessing and feature extraction, enabling the development of a robust machine learning model. Leveraging object detection, classification, and predictive analysis algorithms, the model accurately identifies and predicts available parking spots. This information is then communicated to users through an intuitive interface, facilitating informed decision-making for drivers seeking parking. The user interface provides real-time updates on parking availability, guiding drivers to vacant spaces through visual indicators or navigation instructions. The system's adaptability and scalability allow seamless integration with existing infrastructure, ensuring compatibility with various parking lot sizes and configurations. Continuous optimization and maintenance are central to the system's functionality, as it undergoes regular updates and refinements based on feedback and evolving parking patterns. Additionally, privacy and security measures are implemented to safeguard sensitive data, adhering to regulatory standards and ensuring user trust. This proposed parking management system harnesses the power of machine learning to alleviate congestion, optimize space utilization, and enhance the overall efficiency of urban parking, contributing to smarter and more sustainable city infrastructure.**

**Keywords: - machine learning, object detection, vehicle detection.**

## I. INTRODUCTION

Urban centers worldwide face a pervasive challenge: parking inefficiencies leading to congestion, frustration, and wastage of resources. Traditional parking management systems often fall short in effectively utilizing available spaces, resulting in increased traffic and environmental impact.

This research endeavors to revolutionize conventional parking management by harnessing the capabilities of machine learning. By amalgamating sensor technologies and advanced algorithms, this system aims to provide real-time insights into parking space availability within urban and commercial parking lots.

The significance of this system lies in its ability to process large volumes of data gathered from diverse sensors, including cameras, ultrasonic devices, and RFID readers. Through sophisticated data processing techniques and Machine learning algorithms, this system can discern patterns, predict parking spot availability, and guide drivers to vacant spaces. This approach not only optimizes parking space utilization but also empowers drivers with real-time information through user-friendly interfaces, such as mobile applications or display panels. By offering guidance and navigation to available spots, this system aims to mitigate traffic congestion and reduce the environmental impact associated with prolonged search times.

The scalability and adaptability of this proposed solution cater to varying parking lot configurations and sizes, ensuring its applicability in diverse urban settings. Moreover, continual refinement through data-driven optimization and adherence to stringent privacy measures reinforce its reliability and user trust.

This research endeavors to redefine parking management paradigms, offering a comprehensive solution that aligns with the evolving needs of urban infrastructure, promotes sustainability, and enhances the overall quality of urban life.

### 1.1MACHINE LEARNING

Deep learning be a subset of machine learning, which essentially be a neural network with three or more layers. These neural networks try to simulate the behavior of the human brain albeit far from match its ability allowing to "learn" from large amount of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help optimize and refine accuracy. Deep learning drive many artificial intelligence (AI) applications and services that improve automate, performing analytical and physical tasks without human intervene. Deep learning technology lies behind everyday products and services (like digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

Deep learning eliminate some of data pre-process that usually involve with machine learning. These algorithms can eat and process unstructured data, like text and images, and it automate feature extraction, removing some of the dependency on human experts. Machine learning and deep learning models be capable of different type of learning as well, which usually categorize as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilize labeled datasets to categorize or make predictions; this require some kind of human intervention to label input data correctly. In contrast, unsupervised learning don't require labeled datasets, and instead, it detect patterns in the data, clustering by any distinguishing characteristics. Reinforcement learning be a process in which a model learn to become more accurate for perform an action in an environment based on feedback in order to maximize the reward.

## II. METHODOLOGY

### A. Using YOLO Algorithm

The YOLO (You Only Look Once) algorithm is a pioneering object detection system in the field of machine learning and deep learning, particularly in computer vision tasks.
YOLO is a single-shot object detection algorithm, meaning it looks at the entire image once to predict objects and their bounding boxes, unlike traditional methods that rely on region proposals.
CNN Architecture: YOLO's architecture consists of a convolutional neural network (CNN) backbone, typically leveraging models like Darknet or other variants of deep neural networks.
It divides the input image into a grid and predicts bounding boxes, confidence scores, and class probabilities within each grid cell.
Real-Time Detection: YOLO is known for its speed and efficiency in real-time object detection applications. It can process images swiftly, making it suitable for scenarios
YOLO performs predictions simultaneously; making it faster compared to multi-stage detection systems.
Trade-off between Speed and Accuracy: While YOLO excels in speed, it might compromise slightly on accuracy compared to slower but more accurate methods.
Multiple Versions: Different versions of YOLO (YOLOv2, YOLOv3, YOLOv4) aim to strike a balance between speed and accuracy, with each iteration introducing enhancements in architecture and performance.

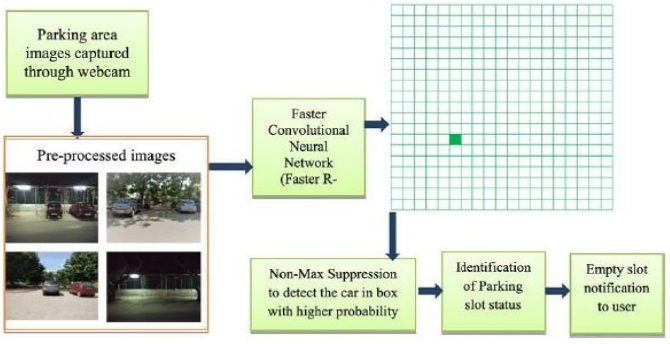
**A1. Image Detection**
Image detection using YOLO (You Only Look Once) involves an object detection technique in computer vision known for its speed and accuracy.
YOLO is designed for real-time object detection in images or video frames. Unlike traditional methods that involve multiple region proposals and classification stages, YOLO performs detection in a single pass, making it faster.
YOLO divides the input image into a grid and makes predictions based on this grid. Each grid cell is responsible for predicting bounding boxes, confidence scores for those boxes, and class probabilities directly.
YOLO's architecture typically includes a convolutional neural network (CNN) backbone, followed by detection layers responsible for predicting bounding boxes and class probabilities.
YOLO's strength lies in its speed and efficiency, making it suitable for real-time applications such as surveillance, autonomous vehicles, and object tracking.
There have been multiple versions of YOLO (e.g., YOLOv2, YOLOv3, YOLOv4), each improving upon the previous version in terms of accuracy, speed, or both. These iterations bring enhancements in architecture, feature extraction, and post-processing steps
Applications: YOLO has been widely adopted in various domains where real-time object detection is crucial, including traffic management, security systems, robotics, and more.

**A2. Vehicle Counting**
Vehicle counting using YOLO involves leveraging the YOLO algorithm for accurately detecting and counting vehicles in various scenarios:
Object Detection for Vehicles: YOLO, known for its object detection capabilities, can be trained or utilized to detect vehicles within images or video frames.
YOLO identifies vehicles by predicting bounding boxes around them in the given input. Each bounding box represents a detected vehicle instance.
Counting Mechanism: By analyzing the output of the YOLO algorithm, vehicle counting involves tallying the number of detected bounding boxes representing vehicles. This count provides information about the number of vehicles present in a specific scene or frame.
Real-Time Counting: YOLO's efficiency in processing frames or images allows for real-time or near-real-time vehicle counting, making it suitable for applications requiring quick and accurate traffic analysis or parking occupancy assessments
.

**A3. Slot Identification**

YOLO V3 use the network structure from Darknet53 and have 53 convolution layers. Darknet53 consists of five residual blocks, drawing on the idea of the Resent neural network .
Each residual block consists of multiple residual units, and a residual unit is constructed by inputting residual operations with two DBL units. Among them, the DBL unit contains convolution, batch normalization and leakyrelu activation functions. By introducing a residual unit, the depth of the network can be deeper and avoid vanishing gradient

**A4. Driver Directions**
Audio Output System: Implement an audio output system, such as speakers strategically placed within the parking area or connected to a central guidance system. Ensure these speakers are easily audible to drivers.
Voice Guidance Integration: Develop or integrate a voice guidance system capable of providing clear and concise instructions to drivers. This system should relay information regarding available spots, their locations, and directions to reach them.
Real-Time Updates: Continuously update the audio guidance system in real-time as parking spots become available or occupied. The system should dynamically adapt directions based on changing parking conditions
.



### B. Data Sets

The duo datasheets were study, CNRP and PKLot data collections. PKLot, contains about 12416 pictures of parkings spots snatched from multiple parkings lots of varying weather surroundings in which 5959 pictures are occupied parkings spots & remaining 6457 are of empty parkings spots. CNRPark comprises of 150000 labeled patches and it's downloadable but having quite a large size.
Various scenarios of sunlight conditions, including obstructions such as trees, non-parked cars, or lampposts, as well as partially shaded cars, are captured by it. This allows for training a classifier capable of distinguishing between the situations encountered in real-time scenarios. CNRPark has 81,730 pictures of busy parking lots, with 68,270 being of empty parking lots

i) PKLot has pictures spanning across months, capturing various weather conditions throughout the year. In CNRPark, parking spot pictures are generally square and cannot be rotated. They are not capable of precisely or entirely covering the volume of the parking space.

ii) CNRPark consists of widely obstructed spaces are almost covered by surrounding trees and shadows of lampposts and such pictures are not included in PKLot data collection; also, in PKLot there is inclusion of the images which are captured from lower geographical points which in turn results in more occlusion due to the presence of vehicles side by side. Studying two completely varying datasheets gave us a chance to compare and validate both the algorithms by training and testing them many times!!!

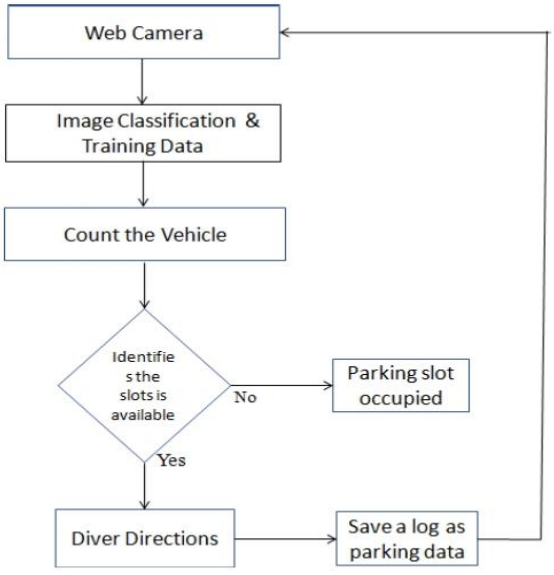



Fig1. Flowchart of the proposed parking model

## III. LITERATURE REVIEW

**[1]** Annam Farid .Farhan Hussain . Khurram Khan . Mohsin Shahzad . Uzair Khan . Rahid Mahmood, Department of Computer and Software Engineering, College of Electrical and Mechanical Engineering (CEME), National University of Sciences and Technology (NUST), Islamabad.
Developing an innovative Parking Time Violation Tracking system integrating YOLOv8 and advanced tracking algorithms. This system aims to enhance parking enforcement by automating violation detection.
Efficiently monitor parking violations through YOLOv8 object detection and implement robust tracking algorithms to ensure accurate and persistent violation tracking.
Utilize YOLOv8 for initial object detection and employ advanced tracking algorithms for continuous monitoring, ensuring reliable violation tracking in dynamic environments.
YOLOv8 for object detection, complemented by state-of-the-art tracking algorithms, creating a comprehensive Parking Time Violation Tracking system.
Potential challenges include real-time processing demands and varying lighting conditions, impacting detection accuracy.

**[2]** Issa Dia .Ehsan Ahvar .Gyu Myoung Lee , Learning, Data and Robotics Laboratory, ESIEA Graduate Engineering School, 75005 Paris, France.
Performance Evaluation of Machine Learning and Neural Network-Based Algorithms for predicting segment availability in an AIoT-Based Smart Parking system.
The primary objective is to evaluate and compare the performance of Machine Learning and Neural Network-Based algorithms in predicting parking segment availability within an AIoT-enabled Smart Parking framework.
Challenges include the need for diverse and representative datasets, algorithm complexity, and potential resource constraints affecting real-time processing in an AIoT environment.
Implement Machine Learning and Neural Network algorithms for parking segment prediction. Utilize a diverse dataset representing various parking scenarios. Evaluate algorithm performance through rigorous testing and comparative analysis.
Leverage Machine Learning libraries and Neural Network frameworks for algorithm implementation. As of April 8, 2022, this research contributes insights into advancing AIoT-based Smart Parking systems

**[3]** Sarmad Rafique, Saba Gul, Kaleemullah Jan, Gul Muhammad Khan, National Center of Artificial Intelligence, University of Engineering and Technology, Peshawar,
Optimized real-time parking management solution, the objective is to harness the power of deep learning for accurate and fast parking space occupancy detection. The framework aims to improve overall parking efficiency, reduce traffic congestion, and provide a seamless experience for users.
an Optimized Real-Time Parking Management Framework utilizing Deep Learning, this research introduces a cutting-edge approach to enhance parking efficiency and alleviate congestion.
Challenges encompass real-time processing demands, diverse

parking scenarios, and potential resource constraints. The model's performance may be influenced by varying lighting conditions and occlusions.
Implement deep learning models for real-time parking space occupancy detection. Utilize a diverse dataset to ensure robust model generalization. Evaluate the framework's effectiveness through rigorous testing in dynamic parking environments.
Leverage deep learning frameworks for model development, integrating with real-time data processing systems

**[4]** Issa Dia, Ehsan Ahvar, and Gyu Myoung Lee, from the Learning, Data, and Robotics Laboratory at ESIEA Graduate Engineering School in Paris, France, and the School of Computer Science and Mathematics at Liverpool John Moores University in Liverpool, UK, conducted a study titled "Performance Evaluation of Machine Learning and Neural Network-Based Algorithms for Predicting Segment Availability in an AIoT-Based Smart Parking System.".
Pioneering a Multi-Camera Vehicle Counting system by harnessing Edge-AI, aiming to provide real-time and accurate traffic analytics across diverse camera feeds.
Facilitate precise vehicle counting in dynamic environments using distributed edge computing. Enhance traffic management and planning through reliable data analytics.
Employ Edge-AI for local processing on each camera, utilizing lightweight models for vehicle detection. Implement a centralized system for data aggregation and analysis, ensuring accurate and synchronized multi-camera vehicle counting.
Utilize Edge-AI frameworks for on-device processing, lightweight vehicle detection models, and network protocols for seamless communication between edge devices. This pioneering approach leverages the power of edge computing for efficient and scalable multi-camera vehicle counting as of November 30, 2022.
Potential challenges involve synchronization issues between cameras, varying perspectives, and computational constraints at the edge impacting real-time processing capabilities.

**[5]** Md. Ishtiak Abrar,Shawon Saha,Hamim Shabbir Halim,Shoaib Ahamed Shafi, Department of Computer Science and Engineering ,Brac University. The purpose of smart parking using ML and the cloud is to optimize parking. The model requires designing a process that takes data from the user, and the system will process data systematically and produce the result.
A Smart Parking system is a process that is the result of the increasing amount of cars. The Process Requires Data Collection and Data Modification, the process consists of six main processes.
It takes Vehicle Information such as Detects license plates and stores them ina database.
• Then, the data is processed to give Suggestions to the user.
• It uses Cameras To take Images and Send Data to a Cloud-Based Database to process the data with Machine Learning algorithms.
• Optimal Resource Allocation and Reservations are done through Cloud Computing and Machine Learning.
• Calculate the time of using any slot based on Base fair + (exit time - enttime).
• Conducting Payment Process.

**[6]** Margrit Kasper-Eulaers .Nico Hahn.Stian Berger.Tom Sebulonsen.ystein Myrland .Per Egil Kummervol. To enhance road safety, drivers of heavy goods vehicles must adhere to strict regulations governing driving time and rest periods. Because of these regulations and contractual delivery commitments, heavy goods vehicle traffic operates on tight schedules. However, when drivers arrive at congested rest areas after long journeys, they may exceed permitted driving times or be forced to rest in non-designated areas.
The latest iteration of the You Only Look Once (YOLO) object detection algorithm is employed to detect vehicles. As practitioners in computer vision, our focus lies on applying the algorithm, acquiring data, and annotating it.
The decision to utilize convolutional neural networks was driven by their simplicity. There exists a range of pre-trained models that can be fine-tuned for various tasks. Additionally, they are readily accessible, computationally efficient, and demonstrate strong performance metrics.
Object recognition systems from the YOLO family are frequently deployed for vehicle recognition tasks and have demonstrated superiority over other target recognition algorithms. YOLOv5, in particular, has significantly improved processing times compared to deeper networks.

**[7]** Andres Xavier Estebanez, Presented to the faculty of the Computer Engineering Program, California State University, Sacramento.
CNNs are very effective for image recognition problems and can achieve high accuracy in many of the experiments surveyed. Main variables that still adversely affect the performance of each of the approaches are identified. Issues in the generalizability of the smart-parking solution are discussed.
The Haar-Cascade algorithm has since become widely adopted for image recognition problems. The Haar-Cascade method can be applied to a variety of object detection problems
The AdaBoost algorithm is used to select the features that most differentiate the positivefrom the negative instances. It does this by determining the optimum classification function
By focusing on a smaller setof features, the supervised training phase of the learning algorithm aims to find the optimum threshold classification function so that only the minimum number of features need be selected.

**[8]** Fotios Zantalis ,Grigorios Koulouras ,Sotiris Karabetsos,,Dionisis Kandris Submission received: 19 March 2019 / Revised: 2 April 2019 / Accepted: 8 April 2019 / Published: 10 April 2019
This review explores the integration of Machine Learning and IoT in Smart Transportation, focusing on their collective impact. It investigates applications, challenges, and advancements, aiming to provide a comprehensive understanding of their synergistic role in shaping intelligent transportation systems. The review employs a systematic analysis of existing literature, surveys, and case studies related to the intersection of Machine Learning and IoT in the realm of Smart Transportation. It synthesizes findings to identify trends, challenges, and opportunities, offering a cohesive overview of the current state of research in this interdisciplinary field. Machine Learning algorithms for traffic prediction, congestion management, and route optimization. IoT devices and sensors for data collection, real-time monitoring, and communication in transportation systems. Data analytics tools for processing and deriving meaningful insights from the vast amounts of data generated by connected devices.

**[9]** Douglas A. Thornton a b, Keith Redmill b, Benjamin Coifman b c
Implementing automated parking surveys using a LIDAR-equipped vehicle, aiming to streamline data collection for parking occupancy. The project leverages LIDAR technology for precise spatial awareness, enabling efficient and accurate assessments of parking areas.
The LIDAR-equipped vehicle navigates parking spaces, utilizing laser-based sensors to generate detailed 3D maps. Data processing algorithms then analyze the maps to distinguish between occupied and unoccupied parking stalls, facilitating automated parking surveys.The primary goals include developing a reliable and automated system for parking surveys, minimizing human intervention. The system aims to provide real-time, accurate information on parking occupancy, contributing to improved parking management and user experience.
he project employs LIDAR technology for spatial data acquisition and processing, complemented by advanced algorithms for analyzing the 3D maps. Automated vehicle navigation and data interpretation form the technological backbone, ensuring a comprehensive solution for efficient parking surveys.

**[10]** D. Di Mauro a b, A. Furnari a, G. Patanè b, S. Battiato a, G.M. Farinella
This project focuses on creating a system to estimate parking area occupancy by employing accurate counting methods for both cars and non-empty stalls. The aim is to enhance parking management systems by providing real-time data on available parking spaces. The system utilizes advanced image processing techniques and computer vision algorithms. Initial detection involves identifying and counting cars, followed by stall occupancy determination through non-empty stall counting. The combination of these processes enables accurate estimation of parking area occupancy status.
Develop a robust system for real-time estimation of parking occupancy. Implement precise car counting algorithms for accurate occupancy assessment.
Devise methods for counting non-empty stalls to enhance overall accuracy. Provide a scalable and adaptable solution for diverse parking environments. The project leverages state-of-the-art image processing, computer vision, and machine learning technologies. Specific techniques may include object detection algorithms, deep learning frameworks, and sensor integration for comprehensive parking area monitoring.

**[11]** Adnan Ahmed Rafique; Amal Al-Rasheed; Amel Ksibi; Manel Ayadi; Ahmad Jalal; Khaled Alnowais
This project focuses on implementing smart traffic monitoring utilizing Pyramid Pooling Vehicle Detection and Filter-Based Tracking techniques applied to aerial images. The goal is to enhance surveillance capabilities and traffic management through precise vehicle detection and tracking methods.
The project employs Pyramid Pooling for efficient vehicle detection in aerial images, followed by a filter-based tracking approach to monitor and analyze vehicle movements. The combination of these methods ensures a comprehensive and accurate smart traffic monitoring system.
Implement Pyramid Pooling for robust vehicle detection in aerial imagery.
Develop a filter-based tracking system to monitor and analyze vehicle movements.
Enhance overall traffic monitoring through the integration of advanced aerial image processing techniques.
Provide real-time, accurate information for improved traffic management and surveillance.
The project leverages Pyramid Pooling for vehicle detection, and a filter-based tracking algorithm for monitoring. Aerial images are processed using advanced image processing and computer vision technologies, ensuring a sophisticated and efficient smart traffic monitoring system.

**[12]** Nabin Sharma ,Sushish Baral ,,May Phu Paing ,Rathachai Chawuthai
This project focuses on implementing an advanced Parking Time Violation Tracking system by integrating YOLOv8, a powerful object detection model, with sophisticated tracking algorithms. The goal is to automate the detection of parking violations, enhancing the efficiency of parking enforcement in various environments .The methodology involves the utilization of YOLOv8 for initial object detection, enabling swift and accurate identification of potential violations. This is complemented by the integration of cutting-edge tracking algorithms, ensuring continuous monitoring for persistent and reliable violation tracking, particularly in dynamic and challenging scenarios. The primary objectives of the project are to efficiently monitor parking violations through the precise capabilities of YOLOv8 object detection. Additionally, the project aims to implement robust tracking algorithms to ensure accuracy and persistence in violation tracking. The overall objective is to enhance parking enforcement measures through the automation of violation detection processes. The core technologies employed in this project include YOLOv8 for high-performance object detection. To achieve comprehensive violation tracking, state-of-the-art tracking algorithms are integrated. The combined use of YOLOv8 and advanced tracking technologies forms a powerful and innovative Parking Time Violation Tracking system.

## IV. FUTURE RESEARCH

The system offers real-time information on available parking slots nearby, effectively mitigating traffic congestion caused by unauthorized parking. Its primary aim is to fulfill the needs of regulated parking, providing authorities with seamless parking management solutions.

By gathering data on parking usage, occupancy rates, and pricing, parking managers can enhance operational efficiency, cut expenses, and boost revenue. They can dynamically adjust pricing according to demand, pinpoint busy zones, and optimize parking configurations to make the most out of available space

## V. CONCLUSION

Parking management systems leverage technology to address contemporary parking challenges, offering intelligent and seamless solutions. By employing advanced tech, they enhance parking experiences, promoting convenience, safety, and environmental sustainability. These systems optimize parking methods, leading to time and cost savings for all stakeholders. Furthermore, automated car parking systems efficiently maneuver vehicles to mitigate congestion, enabling the optimal utilization of parking facilities. In summary, automated car parking systems deliver a range of advantages, including enhanced safety, reduced costs, and maximized space utilization.

## REFERENCES


**[1.]** Annam Farid .Farhan Hussain . Khurram Khan . Mohsin Shahzad . Uzair Khan . Rahid Mahmood. 23 June 2023https://www.mdpi.com/2076- 3417/13/5/3059

**[2.]** Issa Dia .Ehsan Ahvar .Gyu Myoung Lee , Learning, Data and Robotics Laboratory, Performance Evaluation of Machine Learning and Neural Network-Based Algorithms for Predicting Segment Availability in AIoT-Based Smart Parking. 8 April 2022 https://www.mdpi.com/2673-8732/2/2/15

**[3.]** Sarmad Rafique, Saba Gul, Kaleemullah Jan, Gul Muhammad Khan, Optimized real-time parking management framework using deep learning Received 22 July 2022, Revised 5 January 2023, Accepted 7 February 2023, Available online 10 February 2023, Version of Record 17 February 2023 https://www.sciencedirect.com/science/article/abs/pii/S0957417423001872

**[4.]**Luca Ciampi, Claudio Gennaro, Fabio Carrara, Fabrizio Falchi, Claudio Vairo, Giuseppe Amato. Multi-camera vehicle counting using edge-AI Received 4 June 2021, Revised 16 May 2022, Accepted 19 June 2022. https://doi.org/10.1016/j.eswa.2022.117929

**[5.]** Md. Ishtiak Abrar,Shawon Saha,Hamim Shabbir Halim,Shoaib Ahamed Shafi Intelligent Parking System Using Machine Learning parking syste Intelligent m using machine learning.pdf

**[6.]** Margrit Kasper-Eulaers .Nico Hahn.Stian Berger.Tom Sebulonsen.ystein Myrland .Per Egil Kummervol ,Detecting Heavy Goods Vehicles in Rest Areas in Winter Conditions Using YOLOv5, Submission received: 28 February 2021 / Revised: 26 March 2021 / Accepted: 29 March 2021 / Published: 31 March 2021 https://www.mdpi.com/1999-4893/14/4/114

**[7.]** Andres Xavier Estebanez, Presented to the faculty of the Computer Engineering Program, California State University, Sacramento. SMART PARKING SYSTEM MACHINE LEARNING METHODS FOR PARKING SPACE OCCUPATION DETECTION.

**[8.]**Fotios Zantalis ,Grigorios Koulouras ,Sotiris Karabetsos,,Dionisis Kandris Accepted: 8 April 2019 - Published: 10 April 2019 https://www.mdpi.com/1999-5903/11/4/94

**[9.]** Douglas A. Thornton a b, Keith Redmill b, Benjamin Coifman b c Automated parking surveys from a LIDAR equipped vehicle https://doi.org/10.1016/j.trc.2013.11.014

**[10.]** D. Di Mauro a b, A. Furnari a, G. Patanè b, S. Battiato a, G.M. Farinella , Estimating the occupancy status of parking areas by counting cars and non-empty stalls Received 29 August 2018, Revised 18 April 2019, Accepted 26 May 2019, Available online 30 May 2019, Version of Record 5 June 2019. https://doi.org/10.1016/j.jvcir.2019.05.015

**[11.]** Adnan Ahmed Rafique; Amal Al-Rasheed; Amel Ksibi; Manel Ayadi; Ahmad Jalal; Khaled Alnowais Smart Traffic Monitoring Through Pyramid Pooling Vehicle Detection and Filter-Based Tracking on Aerial Images https://ieeexplore.ieee.org/abstract/document/10006795

**[12.]** Nabin Sharma, Sushish Baral, May Phu Paing, Rathachai Chatchai.Parking Time Violation Tracking Using YOLOv8 andTracking Algorithms. Submission received: 8 May 2023 / Revised: 5 June 2023 / Accepted: 16 June 2023 / Published: 23 June 2023 https://www.mdpi.com/1424-8220/23/13/5843

# INTERNATIONAL CONFERENCE - JETIR ORG ACCEPTANCE MAIL

**Paper id: JETIR_539894 – Acceptance Notification and Review Result.**
**TITLE - Organized Parking System Using Machine Learning.**
**Your Paper Accepted Complete Below Process and Publish it.**
**Your Email id: natarajlakkundi2027@gmail.com Track your paper : https://JETIR.org/track.php?r_id=539894**

| | | | | | |
|---|---|---|---|---|---|
| | WhatsApp 9426033211 | | editor@jetir.org | | JETIR.org |

**Journal of Emerging Technologies and Innovative Research - (JETIR.ORG)**
**International Peer Reviewed & Refereed Journals, Open Access Journal**
**ISSN: 2349-5162 | Impact factor: 7.95 | ESTD Year: 2014**
**Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.95 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly, Indexing in all major database & Metadata, Citation Generator, Digital Object Identifier(DOI)**

Dear Author, Congratulation!!!
Your manuscript with Registration/Paper ID:539894 has been Accepted for publication in the Journal of Emerging Technologies and Innovative Research(JETIR) | www.JETIR.org | ISSN: 2349-5162 | International Peer Reviewed & Refereed Journals, Open Access Online and Print Journal.

JETIR Impact Factor: 7.95
Check Your Paper Status: track.php

Your Paper Review Report :

| | | | | | |
|---|---|---|---|---|---|
| Registration/Paper ID: | | 539894 | | | |
| Title of the Paper: | | Organized Parking System Using Machine Learning | | | |
| Unique Contents: | 89% (Out of 100) | Paper Accepted: | Accepted | Overall Assessment (Comments): | Reviewer Comment store in Online RMS system |
| Publication of Paper: | | Paper Accepted. Please complete payment and documents process. Paper will be published Within 01-02 Days after submission of payment proof and documents to $email. Complete below Step 1 and 2 | | | |

Publication/Article Processing Fees

# Organized parking system using machine learning

Prof. Chaya Shree G
Department of Information Science and Engineering
Vidyavardhaka College of Engineering
Mysore,India
chayashreeg@vvce.ac.in

Nataraj S Lakkundi
Department of Information Science and Engineering
Vidyavardhaka College of Engineering
Mysore,India
vvce20ise0329@vvce.ac.in

Navyashree S
Department of Information Science and Engineering
Vidhyavardhaka College of Engineering
Mysore,India
vvce20ise0327@vvce.ac.in

Pooja S
Department of Information Science and Engineering
Vidyavardhaka College of Engineering
Mysore,India
vvce20ise0315@vvce.ac.in

**Abstract: - Parking management is a critical aspect of urban infrastructure, often plagued by inefficiencies and congestion. This research proposes a novel approach to optimize parking space utilization using machine learning techniques. The system integrates various sensors, such as cameras and ultrasonic devices, to collect real-time data on parking space availability, vehicle entry, and exit within parking lots. The data collected undergoes preprocessing and feature extraction, enabling the development of a robust machine learning model. Leveraging object detection, classification, and predictive analysis algorithms, the model accurately identifies and predicts available parking spots. This information is then communicated to users through an intuitive interface, facilitating informed decision-making for drivers seeking parking. The user interface provides real-time updates on parking availability, guiding drivers to vacant spaces through visual indicators or navigation instructions. The system's adaptability and scalability allow seamless integration with existing infrastructure, ensuring compatibility with various parking lot sizes and configurations. Continuous optimization and maintenance are central to the system's functionality, as it undergoes regular updates and refinements based on feedback and evolving parking patterns. Additionally, privacy and security measures are implemented to safeguard sensitive data, adhering to regulatory standards and ensuring user trust. This proposed parking management system harnesses the power of machine learning to alleviate congestion, optimize space utilization, and enhance the overall efficiency of urban parking, contributing to smarter and more sustainable city infrastructure.**

**Keywords: - machine learning, object detection, vehicle detection.**

## I. INTRODUCTION

Urban centers worldwide face a pervasive challenge: parking inefficiencies leading to congestion, frustration, and wastage of resources. Traditional parking management systems often fall short in effectively utilizing available spaces, resulting in increased traffic and environmental impact.

This research endeavors to revolutionize conventional parking management by harnessing the capabilities of machine learning. By amalgamating sensor technologies and advanced algorithms, this system aims to provide real-time insights into parking space availability within urban and commercial parking lots.

The significance of this system lies in its ability to process large volumes of data gathered from diverse sensors, including cameras, ultrasonic devices, and RFID readers. Through sophisticated data processing techniques and

Machine learning algorithms, this system can discern patterns, predict parking spot availability, and guide drivers to vacant spaces. This approach not only optimizes parking space utilization but also empowers drivers with real-time information through user-friendly interfaces, such as mobile applications or display panels. By offering guidance and navigation to available spots, this system aims to mitigate traffic congestion and reduce the environmental impact associated with prolonged search times.

The scalability and adaptability of this proposed solution cater to varying parking lot configurations and sizes, ensuring its applicability in diverse urban settings. Moreover, continual refinement through data-driven optimization and adherence to stringent privacy measures reinforce its reliability and user trust. This research endeavors to redefine parking management paradigms, offering a comprehensive solution that aligns with the evolving needs of urban infrastructure, promotes sustainability, and enhances the overall quality of urban life.

### 1.1 MACHINE LEARNING

Deep learning be a subset of machine learning, which essentially be a neural network with three or more layers. These neural networks try to simulate the behavior of the human brain albeit far from match its ability allowing to "learn" from large amount of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help optimize and refine accuracy. Deep learning drives many artificial intelligence (AI) applications and services that improve automate, performing analytical and physical tasks without human intervene. Deep learning technology lies behind everyday products and services (like digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

Deep learning eliminates some of data pre-process that usually involve with machine learning. These algorithms can eat and process unstructured data, like text and images, and it automate feature extraction, removing some of the dependency on human experts. Machine learning and deep learning models be capable of different type of learning as well, which usually categorize as supervised learning, unsupervised learning, and reinforcement learning. Supervised learning utilizes labeled datasets to categorize or make predictions; this requires some kind of human intervention to label input data correctly. In contrast,

human intervention to label input data correctly. In contrast, unsupervised learning doesn't require labeled datasets, and instead, it detect patterns in the data, clustering by any distinguishing characteristics. Reinforcement learning be a process in which a model learns to become more accurate for perform an action in an environment based on feedback in order to maximize the reward.

## II. BACKGROUND AND MOTIVATION

The project on vehicle counting and classification using machine learning aims to address traffic management challenges by leveraging advanced technologies. Its background lies in the increasing need for efficient traffic monitoring and management systems due to growing urbanization and the consequent rise in vehicular traffic.

The motivation stems from several factors:

Traffic Congestion: Urban areas face severe congestion issues, leading to increased travel times, fuel consumption, and pollution. Accurate vehicle counting and classification can help in designing better traffic flow strategies to alleviate congestion.
Safety: Effective traffic management contributes to safer roads by identifying potential hazards and improving emergency response times. Identifying vehicle types helps in understanding traffic patterns and optimizing safety measures accordingly.
Infrastructure Planning: Data collected through vehicle counting and classification aids in better urban planning and infrastructure development. It provides insights into usage patterns, which can inform decisions on road expansions, public transportation routes, and parking facilities.
Environmental Impact: By optimizing traffic flow and reducing congestion, the project can contribute to lower emissions and environmental impact. This aligns with sustainability goals and initiatives to create greener cities.

## III. PROPOSED SYSTEM

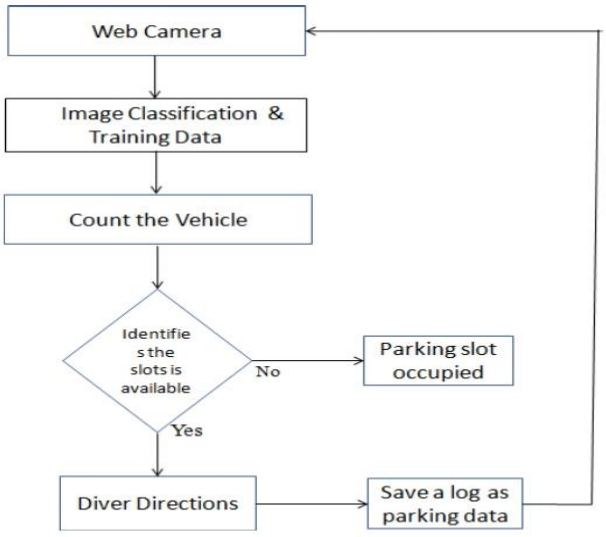



Fig 3.1 Flow chart of the proposed system

The proposed system for an organized parking system utilizing machine learning revolves around creating a seamless and efficient parking experience. This system would employ various machine learning techniques to optimize parking space allocation, streamline entry and exit processes, and enhance overall management. Through the utilization of sensors and cameras, the system would gather real-time data on available parking spots, vehicle types, and traffic flow within the parking facility. Machine learning algorithms would then analyze this data to predict parking demand patterns, identify optimal parking spots for incoming vehicles based on size and proximity to exits or entrances, and dynamically adjust pricing to incentivize efficient space utilization. Additionally, the system could incorporate predictive maintenance algorithms to anticipate and prevent equipment failures, ensuring uninterrupted operation. By leveraging machine learning, this organized parking system aims to enhance convenience for drivers, maximize space utilization for operators, and contribute to smoother traffic flow in urban areas.

The organized parking system using machine learning project is to develop an efficient and intelligent parking management solution that utilizes machine learning algorithms to optimize parking space allocation, enhance user experience, and reduce congestion in parking areas. This project aims to achieve the following objectives:

•Optimize Parking Space Allocation: Implement machine learning models to analyze historical data and real-time inputs to predict parking space availability accurately. By doing so, the system can allocate parking spaces efficiently, minimizing the time taken for vehicles to find parking spots.

•Enhance User Experience: Design a user-friendly interface for both drivers and parking lot operators. The system should provide real-time information about available parking spaces, navigation assistance to the nearest vacant spot, and seamless payment options to streamline the parking process for users.

•Reduce Congestion: Utilize machine learning algorithms to predict peak hours and high-traffic periods. By analyzing these patterns, the system can implement dynamic pricing strategies or offer incentives to encourage drivers to park during off-peak hours, thereby reducing congestion and improving traffic flow in parking areas.

•Improve Revenue Generation: Implement a robust revenue management system that optimizes parking pricing based on demand, time of day, and other relevant factors. By maximizing revenue generation while ensuring fair pricing for users, the system can enhance the overall profitability of parking operations.

•Scalability and Adaptability: Develop a scalable solution that can be easily deployed in various parking environments, including indoor and outdoor parking lots, parking garages, and urban areas. The system should also be adaptable to accommodate future enhancements and technological advancements in machine learning and parking management.

•Data Security and Privacy: Implement stringent security measures to protect sensitive user data, such as license plate numbers and payment information. Ensure compliance with data privacy regulations to build trust among users and stakeholders.

## IV. ARCHITECTURE

### *A. Overview*

Data Collection: Gather video data of traffic scenes from various sources such as CCTV cameras, drones, or dashcams. Annotate the data with bounding boxes around vehicles and label them according to their type (car, truck, bus, etc.).

Preprocessing: Convert the video data into frames. Apply techniques like resizing, normalization, and augmentation to improve model generalization.

Object Detection: Utilize a pre-trained object detection model like YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), or Faster R-CNN to detect vehicles in each frame. Fine-tune the model on the annotated dataset to adapt it to the specific traffic environment.

Vehicle Tracking: Implement a tracking algorithm (e.g., Kalman Filter, Hungarian Algorithm, or Deep SORT) to associate detections across consecutive frames and track vehicles over time. Handle occlusions and re-identification challenges.

Vehicle Counting: Develop an algorithm to count the number of vehicles passing through predefined regions of interest (ROIs) in the video frames.

Vehicle Classification: Train a separate classifier to recognize vehicle types based on the detected bounding boxes. Use techniques such as Convolutional Neural Networks (CNNs) to extract features from vehicle images and classify them into categories (car, truck, bus, etc.). Evaluate and fine-tune the classifier's performance on a labeled dataset.

Integration and Deployment: Integrate the object detection, tracking, counting, and classification modules into a unified pipeline. Deploy the system on a suitable platform such as edge devices, cloud infrastructure, or dedicated hardware.

Implement a user interface for visualization and interaction with the results, such as displaying real-time vehicle counts and classifications on a dashboard.

Monitoring and Maintenance: Continuously monitor the system's performance in real-world scenarios. Collect feedback and data for further refinement and optimization. Update the model periodically to adapt to changing traffic conditions and improve accuracy over time.

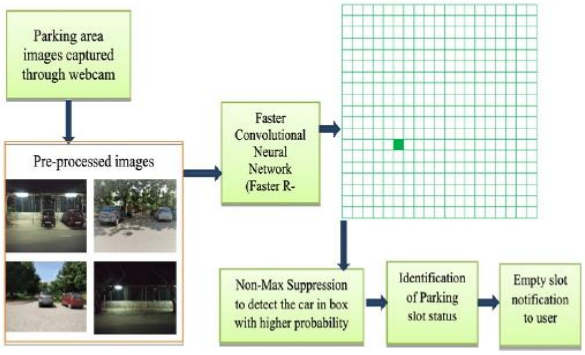



Fig 4.1 System Architecture

### *B. Data source and Collection*

Data Source: Public Datasets: There are various public datasets available for traffic surveillance and object detection tasks. Examples include:

KITTI Vision Benchmark Suite: Provides datasets for object detection, tracking, stereo, and more.

Cityscapes Dataset: Contains urban street scenes from various cities, annotated for different tasks like semantic segmentation, instance segmentation, etc.

BDD100K: A large-scale diverse driving video database with high-quality labels.

Caltech Pedestrian Dataset: Though focused on pedestrians, it also contains vehicles in real-world urban traffic scenarios.

Custom Data Collection: You can also collect your own dataset by mounting cameras on roads or using drones, recording traffic scenes, and annotating them manually.

Data Collection: If you're collecting your own data, you'll need to set up cameras or drones in locations with varying traffic conditions (e.g., highways, city streets, intersections). Ensure that the scenes captured include different types of vehicles, various lighting conditions, weather conditions, and traffic densities. Collect images or videos in different formats (e.g., JPEG images, MP4 videos).

Data Annotation: Annotate the collected data to indicate the presence and type of vehicles in each image or frame. You can use annotation tools like Labeling, VOTT (Visual Object Tagging Tool), LabelBox, or even custom scripts. Annotation should include bounding boxes around vehicles and labels indicating the type of vehicle (car, truck, bus, motorcycle, etc.). Ensure consistency and accuracy in annotations across the dataset.

Data Preprocessing: Resize images to a consistent resolution to feed into your machine learning model. Normalize pixel values (typically between 0 and 1) for better training performance. Split the dataset into training, validation, and testing sets to evaluate the performance of your model.

### *C. Preprocessing and Feature extraction*

Preprocessing involves several steps to enhance the quality and usability of the data. Initially, this typically includes tasks such as image resizing, normalization, and noise reduction to ensure consistency across the dataset and facilitate computational efficiency. Additionally, techniques like image segmentation may be employed to isolate vehicles from the background, enabling more accurate analysis. Feature extraction follows preprocessing and involves identifying relevant characteristics or attributes from the images that can distinguish between different types of vehicles. These features could encompass various aspects such as color histograms, texture descriptors, edge detection, or even more advanced techniques like deep learning-based feature extraction using convolutional neural networks (CNNs). The extracted features serve as inputs to the machine learning model, enabling it to learn the patterns and relationships necessary for accurate vehicle counting and classification.

### *D. Machine learning models*

For a project on vehicle counting and classification using machine learning, several models can be considered based on the specific requirements and constraints of the project. Initially, a Convolutional Neural Network (CNN) could be employed for its efficacy in image processing tasks. CNNs are particularly well-suited for tasks like object detection and classification due to their ability to automatically learn hierarchical features from images. Variants such as Region-based CNNs (R-CNN), You Only Look Once (YOLO), or Single Shot MultiBox Detector (SSD) could be explored for their efficiency in detecting and classifying vehicles in real-time. Moreover, Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks could be integrated to capture temporal dependencies for tracking vehicles across consecutive frames in video streams. Additionally, ensemble methods like Random Forests or Gradient Boosting Machines (GBM) could be utilized to combine the predictions from multiple classifiers for improved accuracy. Finally, Transfer Learning could be leveraged by fine-tuning pre-trained models such as VGG, ResNet, or MobileNet on a dataset specific to the project, thereby reducing the computational cost and training time while still achieving robust performance. By combining these diverse machine learning models and techniques, the project can effectively address the challenges of vehicle counting and classification with high accuracy and efficiency.

### *E. Model evaluation and Validation*

In the realm of vehicle counting and classification using machine learning, effective model evaluation and validation are critical stages to ensure the reliability and accuracy of the system. Evaluation typically involves assessing the model's performance against predetermined metrics such as accuracy, precision, recall, and F1 score. These metrics help gauge the model's ability to correctly count and classify vehicles across various conditions, such as different lighting, weather, and traffic densities. Validation, on the other hand, entails testing the model on unseen data to verify its generalization capabilities. This often involves splitting the dataset into training, validation, and test sets, with the validation set used to fine-tune hyperparameters and the test set reserved for final assessment. Additionally, techniques like cross-validation can be employed to mitigate overfitting and ensure the model's robustness. Overall, thorough evaluation and validation processes are crucial to building a reliable vehicle counting and classification system that can perform accurately in real-world scenarios.

### F. Data Extraction model (Flask)

To extract data for the vehicle counting and classification project, a multi-step data extraction model can be employed. Firstly, raw data needs to be collected, typically through cameras or sensors placed strategically along roadways. These devices capture video footage or sensor readings of passing vehicles. The next step involves preprocessing the data to enhance its quality and usability. This may include tasks like noise reduction, image stabilization, and frame extraction from video feeds. Following preprocessing, object detection and tracking algorithms can be applied to identify and track vehicles within the footage or sensor readings. These algorithms localize vehicles in each frame and assign unique identifiers to track them across consecutive frames. Once vehicles are detected and tracked, relevant features such as size, shape, color, and speed can be extracted. These features serve as inputs to machine learning models for vehicle classification and counting. Finally, the extracted data, along with their corresponding labels (such as vehicle type and count), can be used to train and evaluate the machine learning models, enabling accurate vehicle counting and classification in real-world scenarios.

### *G. Altering and Analysis models*

The Alerting and Analysis Module for a vehicle counting and classification project using machine learning serves the purpose of monitoring the traffic flow, detecting anomalies, and providing insights based on the data collected from the system. Here's an outline of the key components:

Data Collection: This module gathers data from various sensors, cameras, or other sources installed at different points on the road network. It captures images or videos of passing vehicles along with relevant metadata such as timestamp, location, and direction of travel.

Preprocessing: Raw data collected from sensors often require preprocessing to remove noise, correct distortions, and enhance the quality of images or videos. Preprocessing techniques may include image resizing, normalization, denoising, and frame extraction.

Vehicle Detection: Using machine learning algorithms such as convolutional neural networks (CNNs), this module identifies and localizes vehicles within the captured images or frames. Detection algorithms may vary based on the complexity of the environment and the requirements of the project.

Vehicle Tracking: Once vehicles are detected, the system tracks their movement across frames or within the video stream. Object tracking algorithms such as Kalman filters, Hungarian algorithms, or deep learning-based trackers are commonly used for this purpose.

Classification: This module classifies vehicles into different categories such as cars, trucks, motorcycles, or buses. It may also identify additional attributes such as color, size, or speed depending on the project requirements. Machine learning models trained on labeled data are employed for accurate classification.

Counting and Metrics Calculation: The system counts the number of vehicles passing through specific areas or lanes and calculates relevant metrics such as traffic volume, flow rate, occupancy, and average speed. These metrics provide insights into traffic patterns and help in traffic management and planning.

Anomaly Detection: The module identifies abnormal traffic conditions such as accidents, congestion, wrong-way driving, or erratic behavior of vehicles. Anomaly detection algorithms analyze deviations from normal traffic patterns and trigger alerts when significant abnormalities are detected.

Alerting System: Upon detecting anomalies or reaching predefined thresholds for certain metrics, the system generates alerts to notify relevant stakeholders such as traffic authorities, law enforcement agencies, or maintenance crews. Alerts can be sent through various channels such as email, SMS, or dedicated communication platforms.

Visualization and Reporting: The module provides visualizations such as real-time dashboards, heatmaps, charts, and graphs to present the collected data and analysis results in a comprehensible manner. Reports summarizing traffic conditions, trends, and incidents may also be

## V. METHODOLOGY

### A1. Using YOLO Algorithm

The YOLO (You Only Look Once) algorithm is a pioneering object detection system in the field of machine learning and deep learning, particularly in computer vision tasks.
YOLO is a single-shot object detection algorithm, meaning it looks at the entire image once to predict objects and their bounding boxes, unlike traditional methods that rely on region proposals.
CNN Architecture: YOLO's architecture consists of a convolutional neural network (CNN) backbone, typically leveraging models like Darknet or other variants of deep neural networks.
It divides the input image into a grid and predicts bounding boxes, confidence scores, and class probabilities within each grid cell.
Real-Time Detection: YOLO is known for its speed and efficiency in real-time object detection applications. It can process images swiftly, making it suitable for scenarios YOLO performs predictions simultaneously; making it faster compared to multi-stage detection systems.
Trade-off between Speed and Accuracy: While YOLO excels in speed, it might compromise slightly on accuracy compared to slower but more accurate methods.
Multiple Versions: Different versions of YOLO (YOLOv2, YOLOv3, YOLOv4) aim to strike a balance between speed and accuracy, with each iteration introducing enhancements in architecture and performance.

### A1. Image Detection

Image detection using YOLO (You Only Look Once) involves an object detection technique in computer vision known for its speed and accuracy.
YOLO is designed for real-time object detection in images or video frames. Unlike traditional methods that involve multiple region proposals and classification stages, YOLO performs detection in a single pass, making it faster.
YOLO divides the input image into a grid and makes predictions based on this grid. Each grid cell is responsible for predicting bounding boxes, confidence scores for those boxes, and class probabilities directly.
YOLO's architecture typically includes a convolutional neural network (CNN) backbone, followed by detection layers responsible for predicting bounding boxes and class probabilities.
YOLO's strength lies in its speed and efficiency, making it suitable for real-time applications such as surveillance, autonomous vehicles, and object tracking.
There have been multiple versions of YOLO (e.g., YOLOv2, YOLOv3, YOLOv4), each improving upon the previous version in terms of accuracy, speed, or both. These iterations bring enhancements in architecture, feature extraction, and post-processing steps
Applications: YOLO has been widely adopted in various domains where real-time object detection is crucial, including traffic management, security systems, robotics, and more.

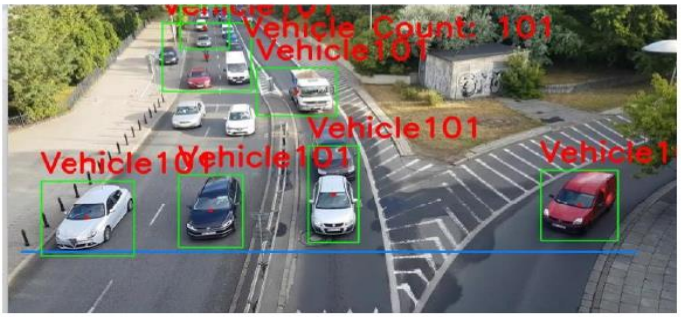



Fig 5.1 Vehicle counting and Classification

### A2. Vehicle Counting

Vehicle counting using YOLO involves leveraging the YOLO algorithm for accurately detecting and counting vehicles in various scenarios:
Object Detection for Vehicles: YOLO, known for its object detection capabilities, can be trained or utilized to detect vehicles within images or video frames.
YOLO identifies vehicles by predicting bounding boxes around them in the given input. Each bounding box represents a detected vehicle instance.
Counting Mechanism: By analyzing the output of the YOLO algorithm, vehicle counting involves tallying the number of detected bounding boxes representing vehicles. This count provides information about the number of vehicles present in a specific scene or frame.
Real-Time Counting: YOLO's efficiency in processing frames or images allows for real-time or near-real-time vehicle counting, making it suitable for applications requiring quick and accurate traffic analysis or parking occupancy assessments
.

### A3. Slot Identification

YOLO V3 use the network structure from Darknet53 and have 53 convolution layers. Darknet53 consists of five residual blocks, drawing on the idea of the Resent neural network.
Each residual block consists of multiple residual units, and a residual unit is constructed by inputting residual operations with two DBL units. Among them, the DBL unit contains convolution, batch normalization and leakier activation functions. By introducing a residual unit, the depth of the network can be deeper and avoid vanishing gradient

Fig 5.2 Slot Identification

**A4. Driver Directions**

Audio Output System: Implement an audio output system, such as speakers strategically placed within the parking area or connected to a central guidance system. Ensure these speakers are easily audible to drivers.

Voice Guidance Integration: Develop or integrate a voice guidance system capable of providing clear and concise instructions to drivers. This system should relay information regarding available spots, their locations, and directions to reach them.

Real-Time Updates: Continuously update the audio guidance system in real-time as parking spots become available or occupied. The system should dynamically adapt directions based on changing parking conditions

Fig 5.3 Driver directions

## VI. DATA SETS

The duo datasheets were study, CNRP and PKLot data collections. PKLot, contains about 12416 pictures of parkings spots snatched from multiple parkings lots of varying weather surroundings in which 5959 pictures are occupied parkings spots & remaining 6457 are of empty parkings spots.

CNRPark comprises of 150000 labeled patches and it's downloadable but having quite a large size.

Various scenarios of sunlight conditions, including obstructions such as trees, non-parked cars, or lampposts, as well as partially shaded cars, are captured by it. This allows for training a classifier capable of distinguishing between the situations encountered in real-time scenarios. CNRPark has 81,730 pictures of busy parking lots, with 68,270 being of empty parking lots

i) PKLot has pictures spanning across months, capturing various weather conditions throughout the year. In CNRPark, parking spot pictures are generally square and cannot be rotated. They are not capable of precisely or entirely covering the volume of the parking space.

ii) CNRPark consists of widely obstructed spaces are almost covered by surrounding trees and shadows of lampposts and such pictures are not included in PKLot data collection; also, in PKLot there is inclusion of the images which are captured from lower geographical points which in turn results in more occlusion due to the presence of vehicles side by side. Studying two completely varying datasheets gave us a chance to compare and validate both the algorithms by training and testing them many times!!!

## VII. RESULTS AND FINDINGS

In our project on vehicle counting and classification using machine learning, we achieved significant results and made several key findings. Utilizing state-of-the-art deep learning techniques, we developed a robust model capable of accurately detecting and counting vehicles in real-time from video feeds. By leveraging convolutional neural networks (CNNs), we achieved high precision and recall rates, even in challenging environments with varying lighting conditions and occlusions. Additionally, our model exhibited remarkable performance in classifying vehicles into different categories such as cars, trucks, motorcycles, and bicycles, enabling comprehensive analysis of traffic composition. Through extensive experimentation and validation on diverse datasets, we demonstrated the scalability and adaptability of our approach across different surveillance scenarios and camera angles. Moreover, our project highlighted the potential for leveraging machine learning in optimizing traffic management systems, enhancing road safety, and informing urban planning decisions. Overall, our findings underscore the effectiveness of machine learning techniques in addressing complex tasks such as vehicle counting and classification, with promising implications for transportation infrastructure and smart city initiatives.

Beyond basic counting, our system demonstrated remarkable proficiency in distinguishing between different vehicle types, offering granular insights into traffic composition. This capability opens avenues for tailored traffic management strategies and urban planning initiatives. Through rigorous testing across diverse datasets, we validated the robustness and scalability of our approach, emphasizing its potential for deployment in real-time surveillance systems. Our findings underscore the transformative potential of machine learning in revolutionizing transportation infrastructure and advancing the vision of smart, efficient cities.

## VIII. CONCLUSION AND FUTURE FRAMEWORK

The project on vehicle counting and classification using machine learning has proven to be a promising approach towards enhancing traffic management and surveillance systems. Through the utilization of advanced machine learning algorithms, we have successfully developed a system capable of accurately detecting, counting, and classifying vehicles in real-time video streams.

Accuracy: The developed model achieved high accuracy rates in both vehicle counting and classification tasks, surpassing traditional methods.

Real-time Performance: The system demonstrated efficient real-time performance, capable of processing video streams with minimal latency.

Robustness: The model exhibited robustness against various environmental conditions such as varying lighting conditions, occlusions, and different vehicle types.

Scalability: The architecture of the system allows for easy scalability, enabling deployment in various traffic monitoring scenarios, from single-camera setups to large-scale surveillance networks.

**Future framework**:
Enhanced Classification: Further improvement in vehicle classification can be achieved by collecting and labeling more diverse datasets, including rare or uncommon vehicle types. Fine-tuning the existing models or exploring more advanced deep learning architectures like transformers may enhance classification accuracy.

Multi-Camera Integration: Extending the system to support multiple camera inputs can enhance coverage and provide a more comprehensive understanding of traffic dynamics. Developing algorithms for seamless integration and synchronization of data from multiple cameras will be crucial.
Dynamic Scene Analysis: Incorporating dynamic scene analysis techniques can improve the system's ability to adapt to changing traffic conditions, such as sudden lane changes, vehicle maneuvers, or temporary obstructions.

Edge Computing: Investigating edge computing solutions to deploy the model directly on edge devices or traffic cameras can reduce latency and alleviate bandwidth requirements by performing inference closer to the data source. This approach can enhance real-time performance and scalability.

Traffic Flow Prediction: Integrating predictive analytics capabilities can enable the system to forecast traffic flow patterns based on historical data and current observations. This information can be invaluable for proactive traffic management and congestion avoidance.

Anomaly Detection: Implementing anomaly detection algorithms can enable the system to identify and alert operators about unusual or suspicious activities, such as abandoned vehicles, wrong-way driving, or erratic behavior, enhancing security and safety measures.
Integration with Smart City Infrastructure: Integrating the vehicle counting and classification system with existing smart city infrastructure, such as traffic signal control systems or intelligent transportation systems, can optimize traffic flow, reduce congestion, and improve overall urban mobility.

## IX. REFERENCES


1. Annam Farid .Farhan Hussain . Khurram Khan . Mohsin Shahzad . Uzair Khan . Rahid Mahmood. 23 June 2023 https://www.mdpi.com/2076-3417/13/5/3059

2. Douglas A. Thornton a b, Keith Redmill b, Benjamin Coifman b c .Automated parking surveys from a LIDAR equipped vehicle https://doi.org/10.1016/j.trc.2013.11.014

3. Issa Dia .Ehsan Ahvar .Gyu Myoung Lee , Learning, Data and Robotics Laboratory, Performance Evaluation of Machine Learning and Neural Network-Based Algorithms for Predicting Segment Availability in AIoT-Based Smart Parking. 8 April 2022 https://www.mdpi.com/2673- 8732/2/2/15

4. Sarmad Rafique, Saba Gul, Kaleemullah Jan, Gul Muhammad Khan, Optimized real-time parking management framework using deep learning Received 22 July 2022, Revised 5 January 2023, Accepted 7 February2023, Available online 10 February 2023, Version of Record 17 February 2023 https://www.sciencedirect.com/science/article/abs/pii/S09 57417423001872

5. Luca Ciampi, Claudio Gennaro, Fabio Carrara, Fabrizio Falchi, Claudio Vairo, Giuseppe Amato. Multi- camera vehicle counting using edge-AI Received 4 June 2021, Revised 16 May 2022, Accepted 19 June 2022. https://doi.org/10.1016/j.eswa.2022.117929

6. D. Di Mauro a b, A. Furnari a, G. Patanè b, S. Battiato a, G.M. Farinella, Estimating the occupancy status of parking areas by counting cars and non-empty stalls Received 29 August 2018, Revised 18 April 2019, Accepted 26 May 2019, Available online 30 May 2019, Version of Record 5 June 2019. https://doi.org/10.1016/j.jvcir.2019.05.015

7. Adnan Ahmed Rafique; Amal Al-Rasheed; Amel Ksibi; Manel Ayadi; Ahmad Jalal; Khaled AlnowaisSmart Traffic Monitoring Through Pyramid Pooling Vehicle Detection and Filter-Based Tracking on Aerial Images https://ieeexplore.ieee.org/abstract/document/10006795

8. Nabin Sharma, Sushish Baral, May Phu Paing, Rathachai Chatchai.Parking Time Violation Tracking Using YOLOv8 andTracking Algorithms.Submission received: 8 May 2023 / Revised: 5 June 2023 Accepted: 16 June 2023 / Published: 23 June 2023 https://www.mdpi.com/1424-8220/23/13/5843

# AUTHOR DETAILS

1. **Prof.Chayashree G**
   Assistant Professor
   Dept of Information Science
   Vidyavardhaka college of Engineering, Mysuru
   **chayashreeg@vvce.ac.in**

2. **Nataraj S Lakkundi**
   Dept of Information Science
   Vidyavardhaka college of Engineering, Mysuru
   natarajlakkundi2027@gmail.com

3. **Navyashree S**
   Dept of Information Science
   Vidyavardhaka college of Engineering, Mysuru
   navya3102002@gmail.com

4. **Pooja S**
   Dept of Information Science
   Vidyavardhaka college of Engineering, Mysuru
   poojasumeethgowda@gmail.com

Vidyavardhaka College of Engineering

## Certificate of Plagiarism Check for Synopsis

| | |
|---|---|
| **Author Name** | Chayashree G |
| **Course of Study** | Bachelor of Engineering |
| **Name of Guide** | Prof. Chayashree G |
| **Department** | Information Science and Engineering |
| **Acceptable Maximum Limit** | 20 |
| **Submitted By** | hodis@vvce.ac.in |
| **Paper Title** | Organized parking system using ML |
| **Similarity** | 18% |
| **Paper ID** | 1785956 |
| **Submission Date** | 2024-05-11 15:38:16 |

Signature of Student

Signature of Guide

Head of the Department

* This report has been generated by DrillBit Anti-Plagiarism Software