CAR PARKING SPACE DETECTION USING YOLO

A PROJECT REPORT

Submitted by

19104047	**NELSON J**
19104063	**SANTHOSH E**
19104084	**VERJIN V**
19104086	**VIGNESHWARAN A**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

ERODE SENGUNTHAR ENGINEERING COLLEGE

(AUTONOMOUS)

PERUNDURAI, ERODE – 638 057

MARCH 2023

ERODE SENGUNTHAR ENGINEERING COLLEGE
(AUTONOMOUS)
PERUNDURAI, ERODE – 638057

BONAFIDE CERTIFICATE

Certified that this project report "**CAR PARKING SPACE DETECTION USING YOLO**" is the bonafide work of **NELSON J (19104047)**, **SANTHOSH E (19104063)**, **VERJIN V (19104084)** and **VIGNESHWARAN A (19104086)** who carried out the project work under my supervision. Certified further that to the best of my knowledge the reported here in does not from part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Ms.R.SAVITHA, M.E.,
SUPERVISOR
ASSISTANT PROFESSOR,
Department of Artificial Intelligence and Data Science,
Erode Sengunthar Engineering College,
(An Autonomous Institution).
Thudupathi, Erode - 638 057.

SIGNATURE
Dr.G.SIVAKUMAR, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR AND HEAD,
Department of Computer Science and Engineering,
Erode Sengunthar Engineering College,
(An Autonomous Institution)
Thudupathi, Erode - 638 057.

Submitted for End Semester Project Viva-Voce Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

First of all, we would like to convey our heartfelt thanks to our respected Founder, "**UDYOG RATTAN" DEIVATHIRU J. SUDHANANDHEN, B.Sc.,** who always blessed us to give the best.

We are delighted to thank our esteemed President **Thiru.V.ANNADURAI** of Erode Sengunthar Educational Trust & Correspondent **Thiru.G.KAMALAMURUGAN, B.B.M.,** of Erode Sengunthar Engineering College and gave useful support for carrying out our project in adequate time and providing us with facilities.

We have immense pleasure to thank our Secretary **Thiru.S.N.THANGARAJU, B.E., M.B.A.,** of Erode Sengunthar Educational Trust who gave us full support and guide in all the endeavours of our project.

We would like to express our thanks to our Principal **Dr.V.VENKATACHALAM, B.E., M.S., M.Tech., Ph.D.,** for forwarding us to do our project and offering adequate duration in completing our project.

We render our sincere thanks to, **Dr.G. SIVAKUMAR, M.E., Ph.D.,** Professor and Head, Department of Computer Science and Engineering for his anchoring support in doing this project.

We are grateful to our project coordinator **Dr.S.TAMILSELVAN, M.E., Ph.D.,** Assistant Professor for his excellent ideas and encouragement.

We are hearty extend our thanks to our project supervisor **Ms.R.SAVITHA, M.E.,** Assistant Professor for her support, constant supervision and as well as for providing necessary information regarding the project.

ABSTRACT

The project aim is to detect the parking spaces of cars using the YOLO (You Only Look Once) algorithm. The camera footage is utilized to extract images which then undergo object detection to identify cars in parking spaces. The current approach to detecting cars in parking spaces involves the use of sensors as well as Internet of Things (IoT) technology and also uses object detection algorithms such as Support Vector Machine (SVM) and Region-based Convolutional Neural Networks (R-CNN). However, these methods have been found to be inadequate in terms of efficiency and accuracy. One of the main challenges associated with these techniques is their high cost, which can be a significant barrier to widespread adoption. In addition, their installation can be quite complex, requiring skilled technicians to set up and configure the system correctly. Furthermore, these methods are often required regular maintenance to ensure that they continue to function correctly. Despite these challenges, researchers and developers are working to improve these techniques and overcome these limitations to make car parking space detection more efficient and accurate. We proposed a solution that aims to overcome these drawbacks by introducing a deep learning approach that uses the YOLO algorithm. This algorithm is efficient and has proven to be highly accurate in object detection tasks. The proposed solution is expected to outperform existing solutions, providing a cost-effective and accurate solution to detect car parking spaces. The advantages of using YOLO includes improved accuracy, faster processing and reduced computational costs compared to other algorithms.

Keywords: parking spaces, YOLO algorithm, object detection, camera footage, sensors, IoT, SVM, R-CNN, accuracy, efficiency, cost-effective, deep learning, computational costs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	FULL FORM OF ABBREVIATION
AP	Average Precision
AVM	Around View Monitor
CLI	Command-Line Interface
CNN	Convolutional Neural Network
GPS	Global Positioning System
GUI	Graphical User Interface
IoT	Internet of Things
IoU	Intersection over Union
mAP	Mean Average Precision
NMS	Non-Maximum Suppression
OCR	Optical Character Recognition
RANSAC	Random Sample Consensus
RFID	Radio Frequency Identification
RNN	Recurrent Neural Network
SSD	Single-Shot Detector
SVM	Support Vector Machine
YOLO	You Only Look Once

CHAPTER 1

INTRODUCTION

Car parking is a major issue in urban areas where limited parking spaces and high demand lead to congested roads, increased traffic and frustrated drivers. To address this issue, various solutions have been proposed and implemented. However, most of these solutions are still not effective enough to provide real-time information about available parking spaces. To overcome this, the integration of deep learning techniques with computer vision has shown promising results in the development of effective car parking space detection systems. This new approach to car parking space detection leverages the advances in machine learning and computer vision to provide accurate and real-time information about the availability of parking spaces in urban areas.

The use of deep learning algorithms has proven to be a highly effective solution for car parking space detection. These algorithms utilize complex neural networks to analyse large amounts of data and make accurate predictions about the presence of objects in an image. In the case of car parking space detection, deep learning algorithms can quickly and accurately identify the presence of cars in parking spaces, making it possible to determine the availability of parking spaces in real-time.

There are several advantages to using deep learning for car parking space detection. First, deep learning algorithms are highly accurate, providing a more reliable and effective solution than traditional methods. Finally, deep learning algorithms are flexible and can be easily adapted to different parking scenarios, making them a highly versatile solution for car parking space detection.

The purpose of parking space detection is to provide drivers with information about the availability of parking spaces in a given area, allowing them to make informed decisions about where to park their vehicle. This is achieved by using a variety of technologies, including cameras, sensors and computer vision algorithms, to detect and track the presence of vehicles in parking spaces. The result is an efficient and effective system that can quickly and accurately identify available parking spaces, reducing traffic congestion and improving the overall experience for drivers.

In recent years, deep learning algorithms, such as YOLO (You Only Look Once), have become increasingly popular for parking space detection due to their accuracy and efficiency. These algorithms are trained using large datasets to identify and classify objects in images, making them well-suited for parking space detection tasks. By utilizing these cutting-edge technologies, parking space detection systems are becoming increasingly sophisticated and effective, providing drivers with real-time information about the availability of parking spaces in their area.

CHAPTER 2

LITERATURE REVIEW

2.1 IoT based Intelligent Parking Management System

Author: Annirudh D, Arun Kumar D, Adabala Taraka Sai Raghava Kumar & K.R.M. Vijaya Chandraka

Publication: IEEE 2nd International Conference on Control, Measurement and Instrumenation (CMI)

Year of Publication: 2021

The biggest problem for car owners driving in urban areas is finding a parking space for their car. In most cases, people waiting in front of parking spaces are informed at the last moment that the parking space is not available. The purpose of this work is to provide users with real-time information about the number of free parking spaces using the GSM messaging service and to manage parking and calculate fees using Optical Character Recognition (OCR) and time stamps. It's about automating. This significantly reduces customer waiting times and also reduces parking labor costs.

In a country like India, most of the city streets are congested with cars. The majority of vehicles that cause traffic jams are looking for efficient parking spaces. The average vehicle spends 15 minutes looking for an available parking space at a very slow speed. The main reason for this situation is the unavailability of parking spaces in shopping malls and other parking slots.

In addition, drivers waiting for parking are notified at the last moment that the parking space is not available. With so many cars on the street, there isn't much you can do to keep the parking slot full. The least that a parking slot can do is to publish the information on the number of available parking slots so that customers who want to park their car can check this information and check the availability of experienced parking slots, that's it.

So, customers can plan or change their plans accordingly. This significantly reduces traffic congestion outside the parking slot and shortens waiting times for customers. To do this, they need to collect data from all parking spaces under consideration. This includes user notification systems using either GSM messaging or cloud-based systems. Additionally, vehicles receive time-stamped tokens at entry and exit and spend a significant amount of time paying for parking. This entire parking fee collection process can be automated. This allows for seamless movement of vehicles and also reduces the need for manpower in parking slots.

The suggested smart parking system is a dependable and cost-effective parking slot management system. The entire system is designed to provide the user with maximum flexibility by allowing the user to add more modules to the system as required. For example, because they employ Automatic Number Plate Recognition, by adding an additional module, the tariff data might be delivered to the car owner's phone numbers. They could also add an Android app for consumers to view the slot status, pre-book a parking spot and pay the rate, which would cut customer wait time.

If the system is installed in several parking slots across the city, the aggregate data may be evaluated to determine the pattern of individuals visiting a specific spot. The parking slot management can utilise this data to forecast a rise in the number of cars and prepare for capacity expansion. The data may also be used to design the capacity and placement of a new parking slot in the city.

Despite the smart parking system's many advantages, there are few more potential drawbacks to consider such as setting up such a system can be costly, involving hefty investments in infrastructure, technology and more over to run and maintain the system, it may be necessary to have specific expertise and training. Any technological issues or malfunctions could lead to erroneous data, system unavailability and lost sales due to dissatisfied customers. This have some privacy concerns such as the system records information about client vehicles, licence plate numbers and parking behaviours. This poses privacy issues and might need following stringent data protection laws.

2.2 Smart Parking System Based on Optical Character Recognition

Author: Athira A., Lekshmi S., Pooja Vijayan, Boby Kurian

Publication: IEEE 3rd International Conference on Trends in Electronics and Informatics (ICOEI)

Year of Publication: 2019

Parking slot detection and user notification are the two major sections of a smart parking system. The empty parking space detection was initially done by deploying a number of sensors in the parking slot. It is highly expensive and complicated to install. But the advancement of image processing has enabled us to use images of the parking area to find out empty spaces.

When the drivers are in search of a parking space, the possibility for accidents increases as they give less attention to the road. A sophisticated car parking system can only solve these problems. That is why numerous research works are taking place in this area all around the world. Empty parking slot detection is the first phase of any smart parking system. The second phase is sharing this information to the drivers who are in search of parking slots.

Wherever there is a significant amount of traffic some form of car parking system is also needed. Rapid rise of urban population and the increase in traffic congestion has escalated the demand for the researches in the smart parking system domain during the past two decades. By taking the advantage of some modern tools in information and communication engineering, the parking space information can be shared effectively in real time.

Modern smart parking systems can be broadly divided into two based on the technology used for identifying the empty parking spaces in a parking slot, one is Sensor based methods and another is Image processing-based methods.

Here in this paper they make a comparative study about different parking spot detection techniques and they also propose an advanced parking spot identification technique using OCR algorithm of image processing.

The first section of the smart parking system is the parking slot detection. This can be done using sensors placed at different locations of the parking slot. But this is a very costly system with a slot of drawbacks. The image processing based empty parking slot identification has made the system simple as well as cheap. Using suitable image processing algorithms, the accuracy of detection can be improved.

The parking spot detection is the initial part of the smart parking system. This may be accomplished by placing sensors throughout the parking slot. However, this is an extremely expensive device with several downsides. The image processing-based vacant parking slot recognition has simplified and reduced the cost of the system. They merely need to utilise photos recorded by the parking slot surveillance camera to discover empty slots here. Future smart parking systems will use an OCR based approach. The detection accuracy can be increased by using appropriate image processing methods.

This can be done using sensors placed at different locations of the parking lot. But this is a very costly system with a lot of drawbacks. The image processing based empty parking lot identification has made the system simple as well as cheap. Here we need only to use images captured by the surveillance camera in the parking lot for empty slot detection. OCR based solution will be an integral part of future smart parking systems. Using suitable image processing algorithms, the accuracy of detection can be improved.

Some of the drawbacks including the image processing-based approach is dependent on the parking lot's cameras. The system's accuracy could be jeopardised if the cameras break down or become obscured. The parking lot can only be partially viewed by the cameras with some restrictions. This implies that regions not covered by the cameras may not be able to be detected by

the system as empty parking spaces. The system's accuracy may be impacted by things like the weather, the illumination and obstacles. For instance, glare, reflections, or shadows can obstruct the image processing algorithms and cause erroneous detections. Moreover, the efficiency of the image processing algorithms utilised will determine how accurate the system is.

2.3 Smart Parking Using IoT Technology

Author: Rachapol Lookmuang, Krit Nambut, Sasiporn Usanavasin

Publication: IEEE 5th International Conference on Business and Industrial Research (ICBIR)

Year of Publication: 2018

The efficient and intelligent parking system management with the IoT technology provides wireless access to the system and allows the user to keep this one. Tracking of parking space availability. with an increase in Car Population and Road Congestion in Large Cities That's the main problem they faced. This work aims to solve this problem. Users typically waste time and effort Looking for specific free space availability parking. Email parking information notification. This reduces user latency when searching. Parking spaces are kept to a minimum. Adopt Radio Frequency Identification (RFID) technology to Prevent car theft.

The IoT can switch information thru the community without concerning human interactions. IoT permits a person to apply for a lower-priced wi-fi era and additionally allows the person to switch the information into the cloud. IoT allows the person to hold transparency. The concept of IoT commenced with the identification of factors for connecting numerous gadgets. The Internet permits the records to be sent, get hold of or maybe speak with the gadgets. The parking trouble reasons air pollutants and site visitor congestion. In a state of an art scenario, a parking area is difficult to look in a day after day existence for the people.

According to the latest survey, there might be a fast growth inside the vehicle's populace of over 1.6 billion around 2035. Around 1,000,000 barrels of the world's oil are being burnt every day. Thus, a clever parking gadget is an important thing answer to lessen the waste degree of gasoline. The answer for the trouble is being raised. Clever parking may be an option to minimize a person's time and performance in addition to the general value of the gasoline burnt on the lookout for the parking area. In this, the information is accrued from the sensor and through analyzing and processing, the output is obtained.

They created this smart parking system with hardware and software based on the IoT concept, as well as a mobile application that allows the driver to easily check parking information and pay

the parking fee via mobile payment. Their study's goal is to improve the parking process by shortening the time it takes to park a car.

To the best of their knowledge, they are the first to build this type of device architecture and mobile application to identify vehicle/registration plate objects. They evaluated the system to discover the optimal range sensor detection for the camera to snap a decent picture of the view. The findings reveal that our different ranges have an influence on each element of the extraction text, allowing our application to apply the suggested technique to give appropriate function services to the user.

While the adoption of IoT technology in parking management has various advantages, there are also possible disadvantages to consider. To begin, building an IoT-based parking system might be costly, demanding major investments in hardware, software and infrastructure. This cost may be exorbitant for certain towns or groups with limited funds, making large-scale implementation difficult.

IoT-based parking solutions rely significantly on technology, which might be faulty at times. Technological concerns, such as network connectivity issues or sensor breakdowns, might result in erroneous data, causing consumers confusion and annoyance and implementing an IoT-based parking system necessitates extensive planning and cooperation with a wide range of stakeholders, including city authorities, parking management businesses and technology providers.

2.4 A Grid Projection Method Based on Ultrasonic Sensor for Parking Space Detection

Author: Yunfeng Shaoa, Pengzhen Chena, Tongtong Caoa

Publication: IEEE International Geoscience and Remote Sensing Symposium (IGARSS)

Year of Publication: 2018

In this paper a parking space detection method is proposed. It is based on the grid map projection and utilizes the ultrasonic sensors. Firstly, it applies a virtual grid map to quantify the observing target space and build a coordinate system. Then the probable outline of the targets can be deduced from the ultrasonic wave echo signal and projected into the grid map.

The boundary of the object can be obtained by comparing the overlap number of the outline in the grid with a threshold. In this way, the size of the target space can be calculated and determined whether it is proper for parking a vehicle. The method is simple while remaining effective.

Various sensors have been applied in the detection of parking space, including the vision sensor, short range radar and laser radar to name a few of them. For instance, vision sensor such as the camera is excellent in the scene recognition, the parking line detection.

For instance, when the parking area is located indoor with a dark surrounding environment or the artificial light are too dazzling, the performance of the sensor will decrease. Additionally, it is difficult detect the parking slot when the parking line is damaged or shaded from other obstacles. It can be easily disturbed by the external condition.

In this paper, they have proposed a grid projection method to calculate the size of the parking space. It can detect the boundary of the obstacles by counting the total overlap numbers of the deduced outlines according to the call back ultrasonic signal. It is founded that the edge of the obstacle, overlap the number of echo signal which is larger than the non-boundary area. The performance of this method is only influenced by the space size, it can tell the approximate shape of the target and has been verified according to the experimental tests.

For the parallel parking slot case in which the size of the parking space is very small, the measurement error of the space detection is 0.2m. Moreover, for the perpendicular parking slot situation in which the size is larger, it can precisely measure the size of the parking slot. This method can be applied in multiple scenes. Furthermore, the proposed method has a significant advantage regarding the calculation and implementation.

It can detect the boundary of the obstacles by counting the total overlap numbers of the deduced outlines according to the call back ultrasonic signal. It is founded that the edge of the obstacle, the overlap number of the echo signal is larger than the non-boundary area. The performance of this method is only influenced by the space size, it can tell the approximate shape of the target and has been verified according to the experimental tests. For the parallel parking slot case in which the size of the parking space is very small. Moreover, for the perpendicular parking slot situation in which the size is larger, it can precisely measure the size of the parking slot.

The Ultrasonic sensor used in the test works at a frequency of 20Hz and the wave beam angle is 60 degree. The position of the vehicle where the sensor is installed can be obtained through Global Positioning System (GPS). Considering the fact that the smaller size of the parking space is, the bigger challenge will be detecting it accurately. Parallel parking is to park a vehicle parallel to the road and keep in line with other parked vehicles. Therefore, the size of the parking space needed should be larger than the length of the vehicle.

Thus, it is difficult to get the accurate edge of the target. In other words, it is easy to recognize the target, but it is hard to tell the shape information of it using the traditional detecting method.

2.5 A Review on Automatic Parking Space Occupancy Detection

Author: Twinkle Singh, Safdar Sardar Khan, Surendra Chadokar

Publication: International Conference on Advanced Computation and Telecommunication (ICACAT)

Year of Publication: 2018

The problem of parking is also increasing due to the increase in the number of vehicles. From the last decade, there are various researches took place with an objective to develop an ideal automatic parking slot occupancy detection. There is an auto mechanism that can park vehicle automatically but it is required to detect which parking slot is vacant and which one is occupied.

The systems which have been designed so far are intended to build parking slot detection using some sensors or mechanism that can indicate whether the vehicle has been parked correctly or not. This paper has been designed for reviewing various techniques which have been used for automated parking slot detection till now.

There are different methods related to parking system have been employed till now such as user interface-based approach, free space-based approach, parking slot marking-based approach, infrastructure-based approach is the best example for parking slot marking based approach. Around View Monitor (AVM) provides 360 degrees around the system nearby visual images.

Hierarchical tree structure is used for the marking of parking slots of various types. AVM system is used for the assistance to the driver while parking. System consists of four cameras which is capable to show the footage of all direction. Visual assistance on the monitor while parking helps the driver to accurately park his vehicle.

Ultrasonic sensors, RFID and some other sensing technique were also used to develop an Automatic Parking System. The method which has been used in this paper employs simple detector to find the slots and merges sequentially obtained slots. The technique applied in this paper uses four different types of parking slot marking which are rectangular, slanted rectangular, diamond and open rectangular types of Transformation of current and previously detected images of parking slot took place to predict the current position of earlier detected slot. Thus, obtained outcomes of slots are clustered according to their assortment and orientation.

The method in which parking assistance system is developed to find the vacant slot for parking and reserve, it. Fragmentation of parking area into blocks is done by using calibration process. Every block is organized in a manner through which vehicles can be identified and process the information with reference to the vacant and occupied slot to the driver.

As per the survey of many approaches most of the systems are using sensors like ultra-sonic, AVM, proximity and fish cameras which increase the cost of the system since these components needs to install in every vehicle to execute the system. AVM itself indicates whether the vehicle has been parked correctly or not along with occupancy detection but vehicles can be parked manually.

Instead of using AVM or other monitoring sensors a visual based approach that can be installed at parking slots at particular elevation which is useful and that can intelligently monitor several slots along with vehicles transition. It would be best for vehicle security views that indicate vehicle transitions and pedestrian occurrence. The real time parking slot detection can be used for automatic parking management. But accuracy is often important which is required for developing an ideal system. OpenCV based approaches can build a cost-effective system.

The proposed system is not reliable for practical execution as the sensors needs to install in every vehicle which is not cost effective. Meanwhile, sensors are able to detect the free space only. System is not trained in a way to detect the adjacent and slanted slots.

2.6 Mobile Outdoor Parking Space Detection Application

Author: Chin-Kit Ng, Soon-Nyean Cheong, Erfan Hajimohammadhosseinmemar, Wen-Jiun Yap

Publication: IEEE 8th Control and System Graduate Research Colloquium (ICSGRC)

Year of Publication: 2017

Finding a vacant parking space in outdoor parking slots is a daily concern of most vehicle drivers during rush hours, especially in the urban context. In this paper, an outdoor parking space vacancy detection system is proposed, using mobile devices to improve parking space searching experience for vehicle drivers by providing them with the location and occupancy information of parking spaces. The system uses state-of-the-art image recognition algorithm, Convolutional Neural Network (CNN) with a Raspberry Pi to identify vacant parking spaces from a parking slot image retrieved in real time via an IP camera.

Due to the ever-increasing parking demand in contrast with the shortage of parking facilities available, it has become an everyday norm for vehicle drivers to bear with the frustration of ceaselessly circling the parking slots for prolonged time just to look for a vacant parking space. The time-consuming process of cruising for parking is not only non-environmentally friendly due to carbon emissions from vehicles, but also constitutes as one of the major causes of traffic congestion. Some of these parking slots are also installed with additional display panels in their interior to navigate drivers towards those available parking spaces by showing the number of vacant parking

spaces along with the corresponding direction. The time-consuming process of cruising for parking is not only non-environmentally friendly due to carbon emissions from vehicles, but also constitutes as one of the major causes of traffic congestion.

The proposed system has its detection mechanism realized using state-of-the-art image recognition algorithm, namely CNN to identify vacant parking spaces within an outdoor parking slot monitored using a stationary IP camera on real time basis. The capability of CNN classifier is to provide high detection accuracy on vacant parking spaces serves as the basis for assuring the reliability and practicality of the Driver App to generate smart parking service for the benefits of vehicle drives.

As for computer vision-based system, existing methods for vacant parking space detection can be categorized as car-driven or space-driven as proposed by Huang and Wang. The former method uses car features in vehicle detection algorithms to determine vacant parking spaces based on the results of vehicle detection. For instance, Wang et al. presented a vision based vacant parking space detection system which employed a feature-based background model and foreground feature extraction module to identify the occupancy status of outdoor parking spaces.

With this feature, vehicle drivers can easily find a vacant parking space without having to circle around the outdoor parking slot, thus improving their parking space searching performance.

An outdoor parking space vacancy detection system was designed and implemented for real-time detection of parking space vacancy using CNN classifier at an outdoor parking slot within a university campus.

While the suggested outdoor parking spot vacancy monitoring system offers several benefits, there are some possible drawbacks to consider the system's implementation can be costly because it necessitates the installation of IP cameras, Raspberry Pi devices and software development. While the CNN algorithm is cutting-edge image recognition technology, it is not flawless and may occasionally misidentify parking spots as available or occupied, providing drivers with inaccurate information. The usage of IP cameras creates privacy issues since it entails recording photos of people and cars in public places. To solve these issues, the system would need to be built and deployed with privacy in mind. The system is reliant on technology and any technological malfunctions or failures can cause serious issues.

2.7 Automatic Parking Space Detection System

Author: Nazia Bibi, Muhammad Nadeem Majid, Hassan Dawood & Ping guo

Publication: IEEE 2nd International Conference on Multimedia and Image Processing (ICMIP)

Year of Publication: 2017

In this paper, a system is proposed that will detect the total number of available parking spaces and displays the information to the drivers so that they can easily parked their cars. A web camera is used to get the images of the parking area and image processing techniques are used to detect the presence or absence of cars to count and locate the available parking spaces. The status of the parking slot is updated whenever a car enters or leaves the parking slot.

Various methods and techniques have been proposed to overcome the problem of parking in the congested areas. A method for counting the vehicles at the checkpoint from which the number of available parking spaces can be counted. The counting is performed by installation of the induction loop sensors under the road surface. Although the usage of sensors was less costly, not easily affected by environmental conditions and it detects accurately however, its installation was difficult and cause damage to roads. It was also difficult to maintain it in case of malfunction. Moreover, the exact locations of free parking area cannot be determined because the counting method is not able to give the detail information, it just records the number of vehicles passing the checkpoints.

The other kinds of detection methods are presented based on vision-based methods. Through vision-based methods, the whole parking area available for parking can be examined though the camera, the data is than processed and the result generated will determine the exact number and location of the free parking spaces. A proposal that vision-based parking space detection methods are very easy to install, low in cost and the detector can be easily adjusted according to requirements. Moreover, the data obtained from images is very rich. However, the defects in the vision method are that the accuracy is highly dependent upon the position of the camera.

An image-based method for detection of free slots in the outdoor parking area. A low-resolution web camera is utilized for acquiring images of the parking slot that reduces the cost greatly. The images acquired are preprocessed and then a pair of Region of Interests is applied on every division of the parking slot, which increases the reliability of detecting vehicles.

To strengthen the recognition capacity of system video data was captured at different environmental conditions and temporal shifts. Video is segmented into frames. Then from each segment a key frame is extracted and further processing is applied on this key frame, to reduce

computational complexity. When radio-controlled toy car enter or leave the parking lot from parking arena, motion of car is estimated by key frame subtraction.

The main contribution of this study is to optimize the identification of available parking slots to possibly reduce the congestion in parking arena. Due to advancement in machine learning and vision base technology cost effective automatic parking systems facilitate the drivers to locate available spaces at parking arena. Future researchers can focus on allocation specific location to customers already registered from online parking management system.

The disadvantage of this method was that the sensors are easily affected by weather conditions like rain, temperature, snow and fast air breeze. Another method was presented based on wireless sensor nodes. This method was less costly and it uses the wireless sensor nodes implemented at the critical places like the lane turns, entrance and exit positions of the parking slot. The total number of cars in the parking area can be determined by the difference of incoming and outgoing cars.

2.8 Automatic Parking Space Detection and Tracking for Underground and Indoor Environments

Author: Jae Kyu Suhr and Ho Gi Jung

Publication: IEEE Transactions on Industrial Electronics

Year of Publication: 2016

Although many public parking slots in heavily populated countries are located underground or indoors, most existing methods have difficulty handling such scenarios. These parking slots also consist mostly of narrow perpendicular parking slots along with numerous pillars, which causes the most commonly used ultrasonic sensor-based parking systems to frequently fail to detect available parking spaces due to inaccurate range data. Also, dim lighting and reflections cause imaging sensor-based methods to work improperly. It can be seen that they contain various severe situations such as amplified noise, road surface reflections, low contrast markings and pillars. In addition, the reflections on the road surfaces and the existence of pillars partially hide parking slot markings in the AVM images.

The proposed method detects vacant parking spaces by combining two complementary approaches- free space-based and parking slot marking-based. However, its performance depends on the existence and positions of adjacent vehicles. The parking slot marking-based approach finds parking spaces by recognizing slot markings on road surfaces. While its performance is independent of adjacent vehicles, it can be degraded under severe illumination and slot marking conditions.The proposed method takes advantage of these two approaches in order to achieve robustness against illumination, slot marking and adjacent vehicle conditions. It presents a method that reliably

recognizes parking spaces in underground and indoor environments based on a high-level fusion of free space-based and parking slot marking-based approaches. It proposes a method that efficiently detects pillars via a low-level fusion of AVM images and ultrasonic sensor data and utilizes them to increase both parking space detection and tracking performances. It suggests a method that robustly recognizes parking slot markings under severe illumination conditions using Random Sample Consensus (RANSAC) and chamfer matching.

An automatic parking system consists of target position designation, path planning and path tracking by active steering. This paper only deals with target position designation, as both path planning and path tracking have already been adopted by mass produced vehicles.

Target position designation methods can be categorized into four approaches: free space-based, parking slot marking-based, user interface-based and infrastructure-based. Since this paper is concerned with free space-based and parking slot marking-based approaches, this section focuses on these two approaches.

The free space-based approach finds vacant parking spaces by recognizing adjacent vehicles. This is the most popular approach as it can be implemented using various range-finding sensors. However, this approach has a fundamental drawback in that it cannot find free spaces when there is no adjacent vehicle and its accuracy depends on the positions of adjacent vehicles. Among a variety of range-finding sensors, an ultrasonic sensor is most widely used as it is easy to attach to vehicles at low cost.

The parking slot marking-based approach finds parking spaces by recognizing markings on road surfaces. Unlike the free space-based approach, performance of this approach does not depend on the existence and positions of adjacent vehicles. However, it cannot be used in cases where parking slot markings are not present or are severely damaged. All methods in this approach utilize imaging sensors and can be categorized into semi-automatic and full-automatic methods. The methods in semi-automatically detect parking slot markings.

Parking slots are detected by combining two separating lines. To reduce the number of false detections, the proposed method utilizes two constraints: one is that one of the two separating lines should contain both positive and negative lines; the other is that a parking slot width should be between 200 cm and 400cm.

In addition, it is hard to separate a pillar and a parked vehicle if they are located next to each other, thus deteriorating the obstacle position estimation accuracies of the ultrasonic sensor-based method. To overcome this problem, this paper proposes a method that detects pillars based on a low-level fusion of AVM images and ultrasonic sensor data and generates free spaces according to

the positions of the detected pillars. It is notable that the pillar detection results are not only used to enhance the free space detection performance but also improve the parking slot tracking accuracy.

This paper has proposed a parking space detection and tracking method for underground and indoor parking slots. It has shown that are parking spaces can reliably be detected by taking advantage of two complementary approaches: free space-based and parking slot marking-based, pillars can effectively be detected by fusing AVM images and ultrasonic sensor data and are useful to enhance both parking space detection and tracking performance and parking slot markings can robustly be recognized by RANSAC and chamber matching-based parallel line detection.

Nevertheless, a few potential issues or restrictions that could surface are dependence on certain sensors. The suggested technique could largely rely on particular kinds of sensors, including image or ultrasonic sensors, which might have limits in particular situations or circumstances. For instance, noisy or reflective environments may make ultrasonic sensors less effective, while dim lighting may have an impact on image sensors. Depending on how complicated the suggested solution is, it can need a lot of processing power or hardware, which would raise the implementation cost. This could make it less useful in specific circumstances or for particular users. How effectively the suggested strategy will work in practice cannot be predicted without extensive testing in real-world settings.

2.9 A Convenient Vision-Based System for Automatic Detection of Parking Spaces in Indoor Parking Lots Using Wide-Angle Cameras

Author: Shen-En Shih and Wen-Hsiang Tsai

Publication: IEEE Transactions on Vehicular Technology

Year of Publication: 2014

In this paper, they investigate the use of cameras for parking slot management. Most cameras used in parking slot systems are perspective cameras; wide-angle cameras, such as fisheye-lens or catadioptric cameras, are not commonly adopted yet. No matter what type of camera systems is used, one problem in using them is the complicated system setup procedure, including camera calibration, environment learning, object modelling. A convenient indoor vision-based parking lot system has been proposed in this study, which is easy to set up by a typical user with no technical background and can detect vacant parking spaces automatically. The system uses wide-angle cameras, such as fisheye-lens or catadioptric cameras and analyses parking space boundary lines based on a new camera model proposed in this study. In addition, finding vacant parking spaces is often a trouble for people, particularly in large indoor parking slots, where one has to spend much

time with concentration to drive through the entire parking area to visually search for a vacant space if the parking slot is nearly fully occupied.

This study aims to develop an intelligent vision-based system using wide-angle cameras for parking slot management, which has merits:

- The camera system can be set up easily by a common user with no technical background, Parking spaces can be detected precisely,
- Vacant parking spaces can be identified automatically for convenient car parking.
- In these methods, the regions of the parking spaces were segmented out manually

On the contrary, the method proposed in this study aims to conduct this major step automatically to construct an easy-to-set-up system. The indoor parking slot system proposed in this study utilizes multiple wide-angle cameras affixed on the ceiling and looking downward vertically.

In this paper, a simplified camera model is proposed for this aim. To design an easy-to-set-up vision-based parking slot system, the camera calibration process must be easy to carry out by users with no technical knowledge.

The proposed vision-based parking slot system has at least the following merits are wide-angle cameras are used so that less cameras are needed to cover the area of a given parking slot, the camera system can be calibrated using only one line in the environment, so that the system can be set up easily by a user with no technical background, the proposed Hough transform with the new cell accumulation scheme generates equal-width curves.

So that the proposed system is capable of directly dealing with distorted images captured by wide-angle cameras and the space line detection results are more precise than those yielded by conventional methods, unlike many previous studies, which specify parking spaces manually, the proposed method detects them automatically for a convenient system setup as well.

While the suggested vision-based parking lot management system has numerous advantages, there are also possible drawbacks to consider. To begin, the suggested system is strongly reliant on the precision of the camera calibration procedure. If the camera is not correctly calibrated, it might result in erroneous space recognition, causing confusion for cars seeking for parking places.

The suggested system employs wide-angle cameras, which can result in visual distortion, particularly around the image's edges. While the suggested Hough transform with the new cell accumulation strategy is intended to cope with distorted pictures, its usefulness may be limited in extreme situations of distortion. As a result, it is critical to test the system under various scenarios to guarantee that it can function properly.

2.10 Parking Space Detection with Hierarchical Dynamic Occupancy Grids

Author: Matthias R. Schmid, S. Ates, J. Dickmann, F. von Hundelshausen and H.- J. Wuensche

Publication: IEEE Intelligent Vehicles Symposium (IV)

Year of Publication: 2011

An automatic parking system relies on precise estimation of parking space geometry. Today, most parking aid systems use active sensors like ultrasonic or radar sensors to determine the parking space. The field of view of these sensors is limited and quite often they have noticeable measurement noise. For those sensors, the accumulation of measurements over time is helpful to increase accuracy. To gain a comprehensive map of the environment, one thus needs to accumulate sensor data.

A dense representation of three-dimensional space would be costly, so a hierarchical data structure is used. As a secondary effect of the hierarchical data structure, the applications can control the level of detail of the environment representation dynamically. While it is easy to detect free parking space in a two-dimensional dense grid, it is challenging to detect free space in a hierarchical three-dimensional data structure.

The approach proposed in this paper is able to detect spots for parallel parking as well as perpendicular parking. This paper describes two contributions. First, the extraction of parking spaces from a hierarchical three-dimensional occupancy grid and secondly the control of the grid resolution based on the detected parking space.

The hierarchical data structure allows dynamic control of the level of detail in the grid based on the requirements of the applications. Further, the data structure makes three-dimensional grids feasible because of the sparse representation of occupied cells. Areas with a homogenous occupancy probability are grouped together into larger cells. This reduces memory consumption and computing power requirements.

For the estimation of a free parking space, the contour of the parking space has to be modelled accurately. However, it is not practicable to represent the whole surrounding in a high level of detail. Therefore, a hierarchical grid structure is used. This approach increases the level of detail only at the borders of the parking space, whereas all other objects are represented in a low level of detail for the parking function.

For a full parking aid system, other components such as trajectory calculation and execution are necessary. A set of parameters of the detected parking space is therefore defined as an abstract

interface to other components. A big improvement in the proposed approach can be achieved using even smaller grid cells in future evaluations and by using radar sensors with a higher angular resolution. Future work should focus on sensors with real three-dimensional data such as a multi-layer laser scanner or a stereo camera.

Here the full potential of the three-dimensional grid will unfold. For high reliability, a fusion of sensors with different measuring principles should be considered. Possible sensors could be short range radar sensors and cameras or short-range radar sensors and a laser scanner.

While the suggested solution for automatic parking systems based on a hierarchical three-dimensional occupancy grid data structure offers some advantages, it also has certain drawbacks that should be considered. To store and analyses the hierarchical data structure, the technique needs a large amount of computer power and memory. While the sparse representation of occupied cells saves memory usage, the system still requires a substantial quantity of data to be processed, especially when employing smaller grid cells. This might be difficult for some embedded systems or low-power devices. The proposed method may not be appropriate for all parking circumstances. It may not operate effectively, for example, in busy parking lots with numerous parked cars or on small roadways with little moving space. The method assumes that the parking place is available.

2.11 Summary

In an overview of this Literature research, we learned through that IoT and Sensors play a significant part in the present System. Furthermore, we discovered that challenge in the approaches. As a result, it becomes sense to consider Deep Learning Strategies. So that the laborious effort and severe workload of the prior analysis may be eased.

In furthermore, we were inspired by the technique used in wide angle camera views with embedded systems to detect planes for space identification, extract features from image or video frames from the trained model using YOLO and remove conflicts using Non-Maximum Suppression (NMS) strategy to select the single entity (Bounding boxes) out of overlapping entities. The above literature analysis leads us to the conclusion that we should construct a deep learning model utilizes an image processing technique to improve the recognition and correct the faults in the problem to solve it swiftly.

In addition, we compared the research findings to a variety of methodologies in order to determine the best answer for a dynamic approach to determining car space occupancy.

CHAPTER 3

SYSTEM ANALYSIS

This chapter gives the complete outlook of what the existing system will do and how the proposed system will work. This gives the advantage and importance of using this new system.

3.1 Existing System

- **Ultrasonic sensors:** These sensors use high-frequency sound waves to detect the presence of a vehicle in a parking space.
- **Infrared sensors:** These sensors use infrared radiation to detect the presence of a vehicle in a parking space.
- **License plate recognition (LPR) systems:** These systems use cameras and OCR technology to identify and track vehicles as they enter and exit a parking space.
- **Magnetic field-based systems:** These systems use magnetic sensors to detect the presence of a vehicle in a parking space.
- **Wireless communication systems:** These systems use wireless communication technology, such as Bluetooth or Wi-Fi, to communicate information about the availability of parking spaces.
- **RFID systems:** These systems use RFID tags attached to vehicles to track their location and determine the availability of parking spaces.
- **Pressure-sensitive systems:** These systems use pressure sensors embedded in the pavement to detect the weight of a vehicle and determine the availability of parking spaces.

Moreover, each of these systems has its own advantages and disadvantages and the choice of system will depend on the specific requirements and constraints of the parking environment.

3.1.1 Disadvantages

- **Ultrasonic sensors:** These sensors can be affected by obstacles such as walls or other vehicles, reducing their accuracy.
- **Infrared sensors:** These sensors may not work well in direct sunlight, as the intense light can interfere with the sensor's ability to detect vehicles.
- **License plate recognition (LPR) systems:** These systems can be affected by low-light conditions, making it difficult to accurately read license plate numbers. They may also raise privacy concerns, as they capture images of vehicles and their license plates.

- **Magnetic field-based systems:** These systems can be affected by metal objects in the surrounding environment, which can interfere with the magnetic field and reduce accuracy.
- **Wireless communication systems:** These systems may have limitations in terms of range and reliability, as they depend on the availability of wireless networks.
- **RFID systems:** These systems require RFID tags to be attached to each vehicle, which can be inconvenient for users and may not be feasible for all types of vehicles.
- **Pressure-sensitive systems:** These systems can be affected by environmental factors such as rain or snow, which can change the pressure readings and reduce accuracy. Additionally, they can be damaged by heavy vehicles, reducing their lifespan.

However, the system comes with several disadvantages includes high cost of installation, maintenance costs, technical difficulties, privacy concerns, dependence on technology, vulnerability to hacking, inaccurate results and resistance to change. It is important to consider the pros and cons of each system before deciding on the best approach for a particular parking environment.

3.2 Proposed System

- The proposed system uses the YOLO object detection algorithm to detect and classify parking spaces in real-time video frames.
- The system was trained on a dataset of parking scenes to improve its accuracy in detecting parking spaces.
- The performance of the system is evaluated and compared with other existing systems and the results showed high accuracy in detecting and classifying parking spaces.
- It can detect and classify parking spaces in various lighting and weather conditions, making it a versatile solution for different parking scenarios.
- It can also reduce the need for manual inspection of parking lots, saving time and resources.
- It has potential applications in intelligent transportation systems, as it can provide real-time information about parking availability and optimize parking space utilization.
- It also has potential for additional features, such as license plate recognition and payment processing, to further improve the efficiency and convenience of car parking.
- It has the potential for further improvement and integration with other technologies, such as machine learning and artificial intelligence, to enhance its performance and capabilities.

Overall, the proposed system has several advantages including its accuracy, speed and ease of integration. It is a highly accurate and efficient algorithm for object detection, making it an ideal choice for detecting parking space occupancy. The system can be easily integrated into existing car parking systems, allowing for a seamless and cost-effective upgrade to the system.

3.2.1 Advantages

- **Increased Efficiency:** By providing real-time parking space detection, the system can reduce the time and effort required for drivers to find available parking spaces, ultimately increasing the efficiency of the parking facility.
- **Cost-Effective:** The integration of the YOLO model into existing parking systems is a cost-effective way to upgrade the system, as it does not require significant changes to the infrastructure.
- **Accurate and Reliable:** The YOLO algorithm is known for its high accuracy in object detection, ensuring that the system can accurately detect parking space occupancy and reduce the likelihood of errors.
- **Fast Processing:** YOLO is also known for its speed in processing images, making it possible for the system to detect parking space occupancy quickly and efficiently.
- **Easy Integration:** The YOLO model can be easily integrated into existing car parking systems, allowing for a seamless and cost-effective upgrade to the system.
- **User-Friendly:** The system is user-friendly, with clear indicators showing available parking spaces and reducing the need for drivers to search for available spots.
- **Customizable:** The system is highly customizable, with the ability to adjust the sensitivity of the system to fit the specific needs of different parking environments.
- **Potential for Additional Features:** The system has potential for additional features, such as license plate recognition and payment processing, which can further improve the efficiency and convenience of car parking.

Overall, the proposed system offers several advantages that can greatly benefit parking facilities and their customers, ultimately improving the efficiency, convenience and overall experience of car parking.

CHAPTER 4

SYSTEM REQUIREMENTS

This chapter exhibits the hardware and software requirements for the proper running of the project. Understanding the system requirements is important as it ensures that the software will run smoothly on the user's device and meet their expectations for performance and functionality.

4.1 Hardware Requirements

The hardware requirements for using YOLO models depend on the specific architecture and size of the model.

CPU	:	At least 2 cores.
GPU	:	At least 4GB memory for small YOLO models.
		At least 8GB for larger YOLO models.
RAM	:	Minimum 8 GB.
Hard Disk	:	500 GB.
Monitor	:	15" LED.
Input Devices	:	Keyboard, Mouse.

4.2 Software Requirements

The software requirements for using YOLO models include operating systems, necessary platforms and frameworks on which the software requires support to run.

Operating System	:	Windows 10 or Linux.
Programming Language	:	Python 3.5 or higher.
IDE’s	:	Visual Studio, PyCharm, Anaconda.
Tools	:	Jupyter Notebook, OpenCV, NumPy, Keras.
Deep Learning Frameworks	:	TensorFlow.

4.3 Tools and Technology

4.3.1 Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

4.3.2 Pip

Pip is a package-management system written in Python and is used to install and manage software packages. The Python Software Foundation recommends using pip for installing Python applications and its dependencies during deployment.

4.3.3 NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

4.3.4 Keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano and PlaidML.

4.3.5 OpenCV

OpenCV is an open-source computer vision library which consists of programming functions mainly for real-time. It provides a variety of functions and algorithms for image and video processing. It can be used with several programming languages, including C++, Python and Java. Its features include object detection, tracking and recognition, as well as machine learning and deep learning capabilities.

4.3.6 Jupyter Notebook

Jupyter Notebook is an open-source web-based interactive computing environment used to create and share documents that contain live code, equations, visualizations and narrative text. It supports a wide range of programming languages, including Python, R, Julia and many others. It is often used for data exploration, data cleaning, machine learning and scientific computing, but they can also be used for general-purpose programming, teaching and collaboration.

It also provides a rich ecosystem of extensions and plugins that can be used to add new functionality and improve your productivity. It integrates with many popular data science libraries and tools, such as NumPy, Pandas, Scikit-learn and Matplotlib, making it a popular choice for data analysis and visualization.

4.3.7 TensorFlow

TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference

of deep neural networks. It also provides a wide range of pre-built algorithms and models, such as CNN, Recurrent Neural Network (RNN) and more.

It also provides a range of tools for model visualization, including TensorBoard, which enables users to monitor and visualize the progress of their training runs.

4.3.8 PyTorch

PyTorch is a machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing. It provides a platform for building and training deep learning models in a various application including natural language processing, computer vision. It provides several modules and functions that can help developers build and train neural networks, including tensor computation, gradient computation and neural network modules. Additionally, it allows for easy integration with popular Python libraries such as NumPy, SciPy and Pandas.

CHAPTER 5

SYSTEM DESIGN

5.1 System Architecture

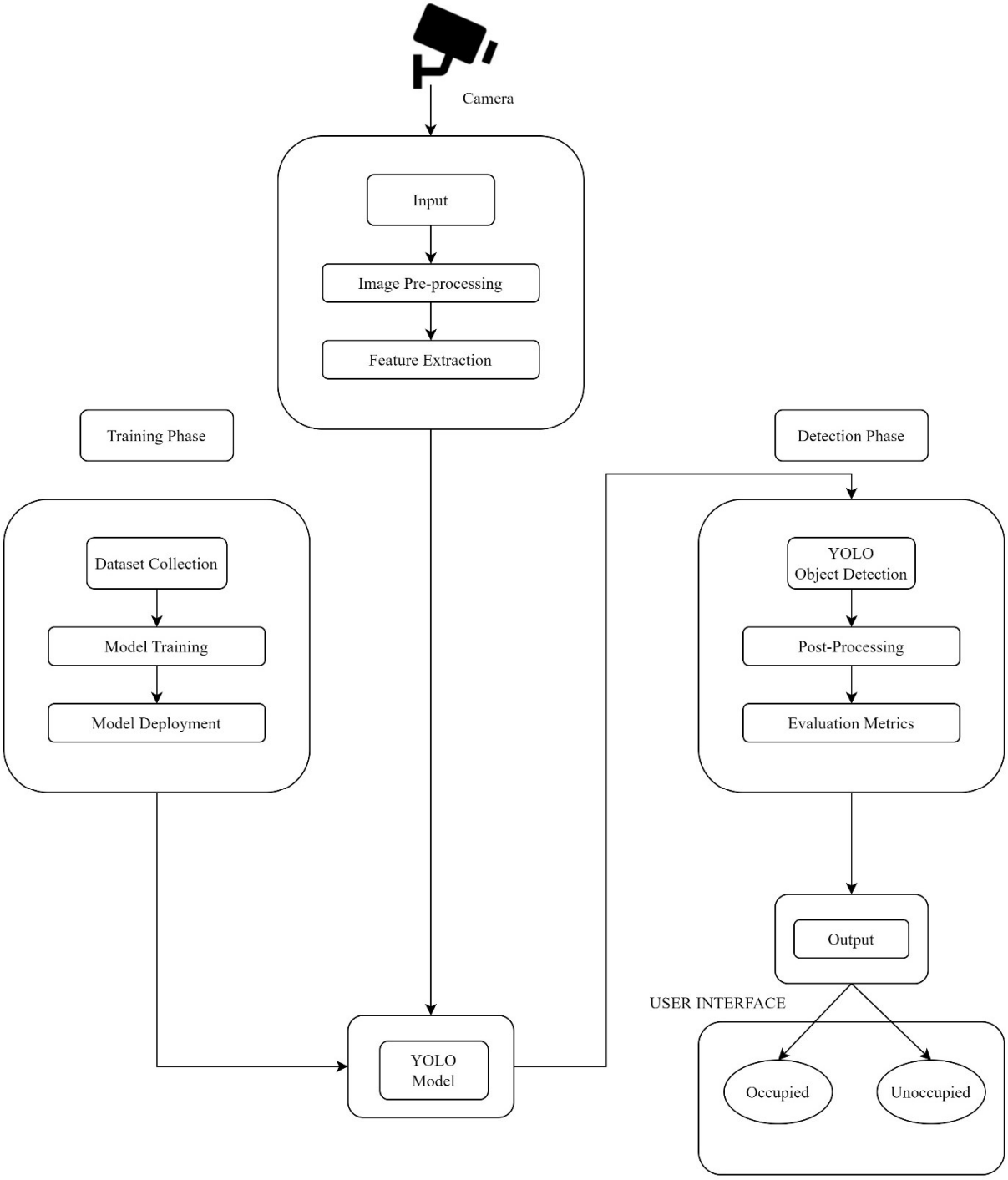


Fig. 5.1 System Architecture

5.2 List of Modules

- Pre-Processing
- Object Detection
- Post-Processing
- User Interface

5.2.1 Pre-Processing

The Image Pre-processing module is responsible for enhancing the input images or video frames to improve the accuracy of object detection. The module may perform operations such as noise removal, image filtering, image enhancement, or resizing of images to suit the requirements of the object detection module. These operations help remove any unwanted information from the images that may interfere with the detection process. Furthermore, the module may also perform data normalization, which makes the input data consistent in terms of image size and format. The aim is to enhance their quality and make them suitable for object detection. The steps may include resizing the image to a standard size, normalizing the pixel values to a certain range, applying colour correction, or removing noise from the image. It is essential to improve the accuracy of the object detection results.

5.2.2 Object Detection

The Object Detection module is the core of the system and is responsible for detecting and classifying parking spaces in the input data. The module uses the YOLO algorithm, which is a deep learning-based object detection algorithm. The algorithm divides the input image into a grid of cells and applies a CNN to each cell to predict the class and bounding box of any object present. In the case of parking space detection, the output of the object detection module will include the location and size of all detected parking spaces. The YOLO algorithm is efficient and fast, allowing real-time object detection.

5.2.3 Post-Processing

The Post-Processing module receives the output from the object detection module and performs additional processing to refine the accuracy of the detection results. The module filters out any false detections or noise in the output and may perform morphological operations such as dilation or erosion to improve object segmentation. Additionally, the module may also perform thresholding to segment the detected parking spaces from the input image accurately.

This module is responsible for refining and improving the accuracy of the output of the object detection module. The output of the object detection module may include false positives or false negatives and the post-processing module aims to remove these errors. The post-processing steps

may include NMS to eliminate overlapping bounding boxes, applying filters to remove small or irrelevant detections, or re-scoring the detected objects to improve their confidence scores.

5.2.4 User Interface

This module provides a user-friendly interface to interact with the system and display the output results. The user interface may include a Graphical User Interface (GUI) or a Command-Line Interface (CLI). The user interface allows the user to input the image or video file, adjust the parameters of the object detection algorithm and view the output results. The user interface module is essential to make the system accessible to users with varying levels of technical expertise.

The User Interface module is responsible for providing a user-friendly interface to interact with the system. It displays the input and output data of the system in a format that is easy to understand for users. The module also enables users to adjust system settings or parameters to fine-tune the performance of the system.

5.3 Proposed Methodology

5.3.1 Data Collection

The first step in the methodology is to collect a dataset of images taken from a multi-level car park. The images were taken at different times of the day and in different weather conditions to ensure that the system could handle variations in lighting and weather. It used for testing YOLO model for car parking space detection. The dataset is used to train YOLO algorithm how to recognize different types of parking spaces in different lighting and weather conditions. The data collection process involved the following steps:

- **Identifying parking lots:** The first step in the data collection process is to identify parking lots in different locations. These locations included public parking lots, private parking lots and parking garages.
- **Capturing images and videos:** Once the parking lots were identified, images and videos were captured using various cameras and recording devices. The images and videos were captured from different angles and distances to ensure that the dataset represented real-world parking lot scenes.
- **Labelling parking spaces:** The captured images and videos were manually labelled to identify the parking spaces within the scenes. This labelling process involved drawing bounding boxes around each parking space and assigning a unique label to each box.
- **Pre-processing the dataset:** The labelled images and videos were pre-processed by resizing them to a standard size and applying data augmentation techniques to increase the size of the dataset. The data augmentation techniques included flipping the images

horizontally, vertically, or both, adding random noise to the images and changing the brightness and contrast levels.

- **Splitting the dataset**: The pre-processed dataset is split into a training set and a testing set. The training set is used to train the YOLO algorithm, while the testing set is used to evaluate the performance of the algorithm.

5.3.2 Image Annotation

The next step is to annotate the images by marking the locations of the parking spaces and vehicles. This is done using a bounding box annotation tool, which allows the user to draw a box around the object of interest. The annotated images were then used to train and test the YOLO algorithm. It is an essential step as it involves the manual labelling of the images with bounding boxes to identify the parking spaces within the images. The process of image annotation involves drawing a bounding box around each parking space in the image and assigning a unique label to the box. The labelled images are then used to train the YOLO algorithm to recognize parking spaces in real-world scenarios. The annotations were used to create ground truth data that the algorithm could use to learn and improve its accuracy. The image annotation process involved the following steps:

- **Selection of annotation tool:** The first step in the image annotation process is to select an appropriate annotation tool. There are several annotation tools available, including LabelImg, RectLabel and CVAT. For this project, LabelImg is used as the annotation tool.
- **Defining the classes:** The next step is to define the classes of objects that need to be annotated. In this project, the classes were defined as "parking space" and "non-parking space."
- **Annotation process:** The annotation process involved opening an image in the annotation tool and drawing a bounding box around each parking space. The bounding box is adjusted to cover the entire parking space and a unique label is assigned to each box.
- **Reviewing the annotations:** After annotating the images, the annotations were reviewed to ensure that all parking spaces were correctly labelled and that there were no false positives or false negatives.
- **Exporting the annotations:** Once the annotations were reviewed and finalized, they were exported in the YOLO format, which includes a text file for each image that contains the coordinates of the bounding boxes and the class labels.

5.3.3 YOLO Architecture

The YOLO algorithm consists of a CNN that is trained to detect objects in an image. The CNN is made up of several layers, including convolutional layers, pooling layers and fully

connected layers. The final layer of the CNN outputs the bounding box coordinates and class probabilities for each object in the image. The YOLO architecture is a popular deep learning algorithm used for object detection in images. In this proposed methodology, the YOLO algorithm is used to detect parking spaces in images. It has the following key features:

- **Single convolutional network:** YOLO is a single neural network that takes the entire image as input and outputs the predicted bounding boxes and class probabilities. This is in contrast to other object detection algorithms that require multiple stages and separate classifiers for each object class.
- **Grid system:** The YOLO algorithm divides the input image into a grid system and predicts bounding boxes and class probabilities for each grid cell. This approach enables the algorithm to detect objects of different sizes and aspect ratios.
- **Bounding box prediction:** The YOLO algorithm predicts bounding boxes that are defined by their centre coordinates, width and height. This approach is different from other algorithms that predict the corners of the bounding box.
- **Class probabilities:** In addition to predicting bounding boxes, the YOLO algorithm also predicts the class probabilities for each bounding box. This enables the algorithm to identify the type of object in each bounding box.

The YOLOv3 architecture uses a series of convolutional layers, followed by a set of detection layers that predict bounding boxes and class probabilities for each grid cell. The YOLOv3 architecture is trained on a dataset of annotated images to learn how to detect parking spaces in real-world scenarios. The trained YOLOv3 model is then used to detect parking spaces in new images by analysing the input image and outputting the predicted bounding boxes and class probabilities.

5.3.4 Training and Testing

The annotated images were used to train the YOLO algorithm. The training process involves adjusting the weights of the CNN so that it can accurately detect the objects in the images. Once the training is complete, the algorithm is tested on a set of images taken from the same car park. The following are the steps involved in training and testing the algorithm:

- **Collection of Data and annotation:** The first step is to collect a dataset of images containing parking spaces and annotate them with bounding boxes around the parking spaces. This annotated dataset is used for both training and testing the algorithm.
- **Training the YOLOv3 model:** The annotated dataset is used to train the YOLOv3 model. The training process involves optimizing the weights of the neural network to minimize the difference between the predicted bounding boxes and the ground-truth bounding boxes.

- **Validation:** The trained model is then validated on a separate validation dataset to ensure that it is not overfitting to the training dataset.
- **Testing the model:** The final step is to test the trained YOLOv3 model on a separate test dataset to evaluate its performance in detecting parking spaces in new images. The performance of the model is evaluated using metrics such as precision, recall and F1 score.

The training and testing process is iterative and the model is retrained and tested multiple times until the desired performance is achieved. The proposed methodology uses of transfer learning to train the YOLOv3 model on a pre-trained model, such as the COCO dataset, to improve the accuracy of the model.

5.3.5 Performance Evaluation Metrics

The performance of the YOLO algorithm is evaluated using several metrics. Precision measures number of true positive detections out of all positive detections, recall measures the number of true positive detections out of all the ground truth objects. The following metrics are most commonly used performance evaluation:

- **Precision:** Precision is a measure of the accuracy of the positive predictions made by the model. It is defined as the ratio of true positives (TP) to the total number of positive predictions (TP + false positives (FP)). A high precision value indicates that the model has a low rate of false positives.
- **Recall:** Recall is a measure of the completeness of the positive predictions made by the model. It is defined as the ratio of true positives (TP) to the total number of actual positives (TP + false negatives (FN)). A high recall value indicates that the model has a low rate of false negatives.
- **F1 score:** The F1 score is a harmonic mean of precision and recall and it is used to evaluate the overall performance of the model. It is defined as 2 * (precision * recall) / (precision + recall). A high F1 score indicates that the model has both high precision and high recall.
- **Intersection over Union (IoU):** IoU is a measure of the overlap between the predicted bounding boxes and the ground-truth bounding boxes. It is defined as the ratio of the intersection of the two boxes to the union of the two boxes. A high IoU value indicates that the predicted bounding box is accurately aligned with the ground-truth bounding box.
- **Mean Average Precision (mAP):** mAP is a commonly used metric for evaluating object detection models. It is calculated by averaging the precision values at different recall levels. A high mAP value indicates that the model has high precision across different recall levels.

In the proposed methodology, the performance is evaluated using a combination of these evaluation metrics. The metrics are calculated on a separate test dataset and the model is iteratively refined until the desired performance is achieved.

5.4 Workflow

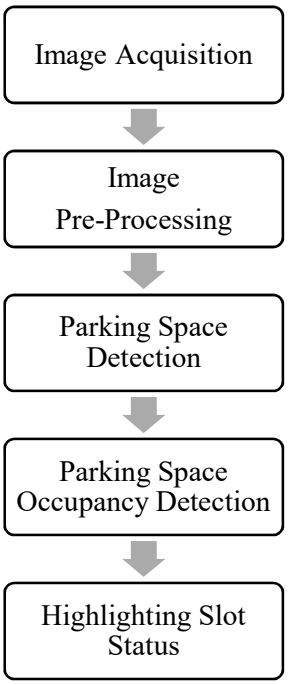


Fig. 5.2 Workflow Diagram

5.5 Data-flow

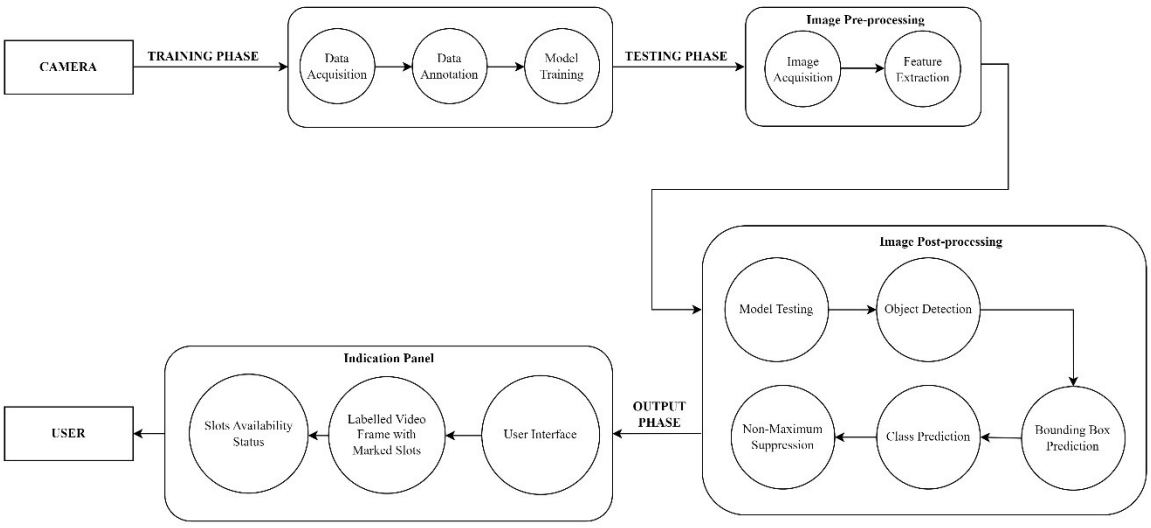


Fig. 5.3 Data-flow Diagram

5.6 Sequence

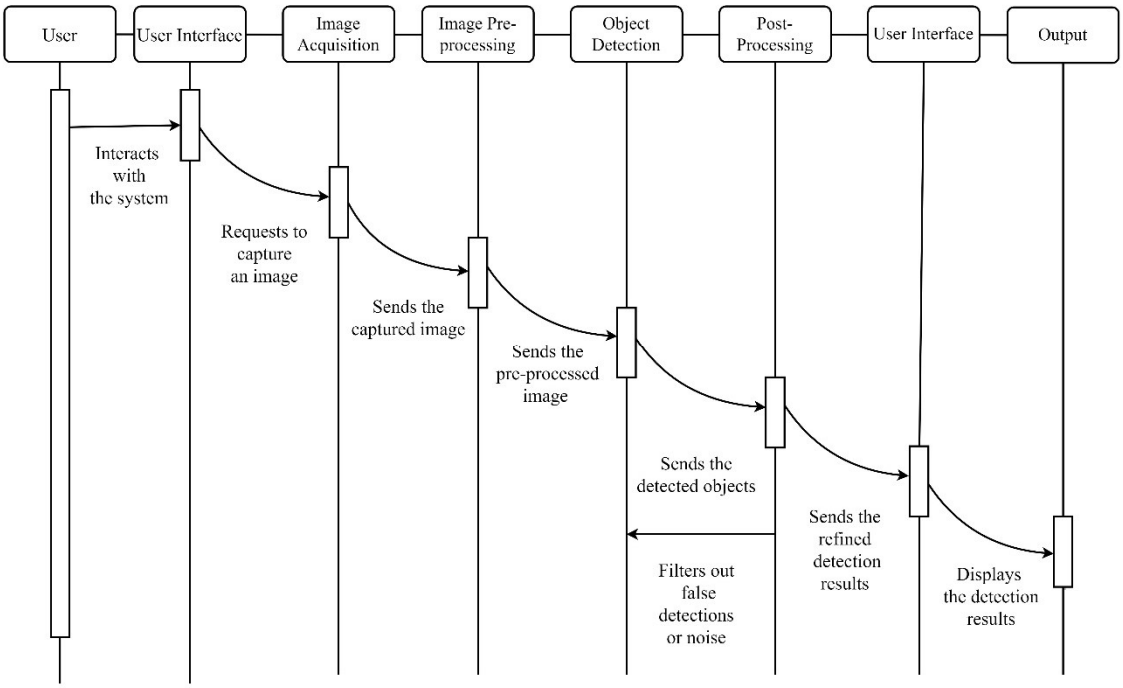


Fig. 5.4 Sequence Diagram

5.7 Use Case

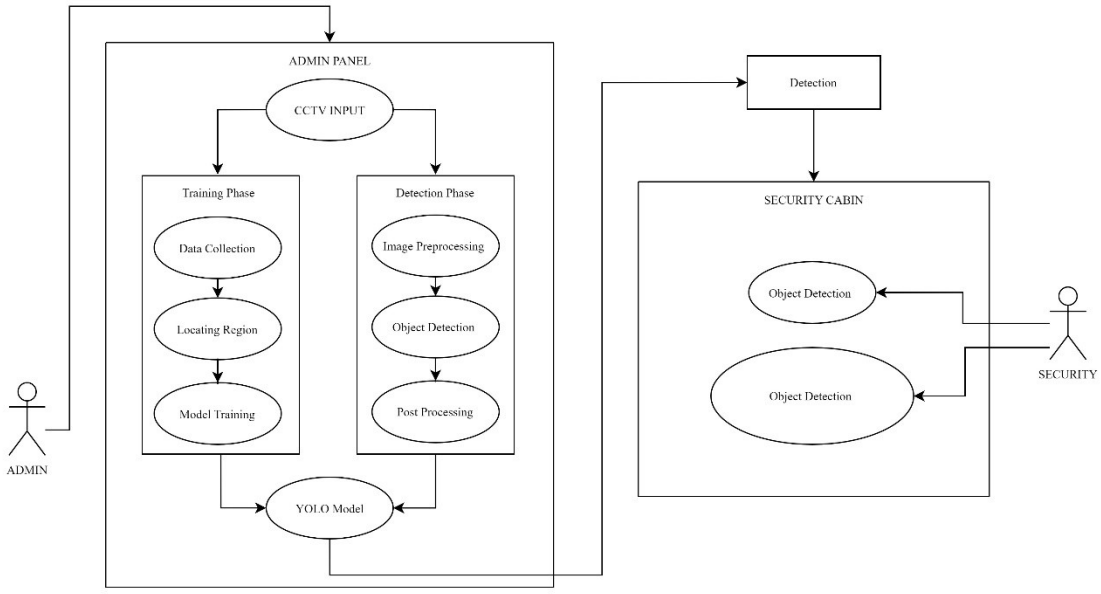


Fig. 5.5 Use Case Diagram

CHAPTER 6

RESULT AND ANALYSIS

The results of the experiments show that the YOLO algorithm is able to accurately detect the parking spaces and vehicles in the images taken from the car park. The Average Precision (AP) for the YOLO algorithm is found to be 0.92, which indicates that the algorithm is able to accurately detect 92% of the parking spaces and vehicles in the images. The precision and recall values were also found to be high, with precision and recall values of 0.926 and 0.92, respectively. These results demonstrate the effectiveness of the YOLO algorithm in detecting parking spaces and vehicles in real-world images.

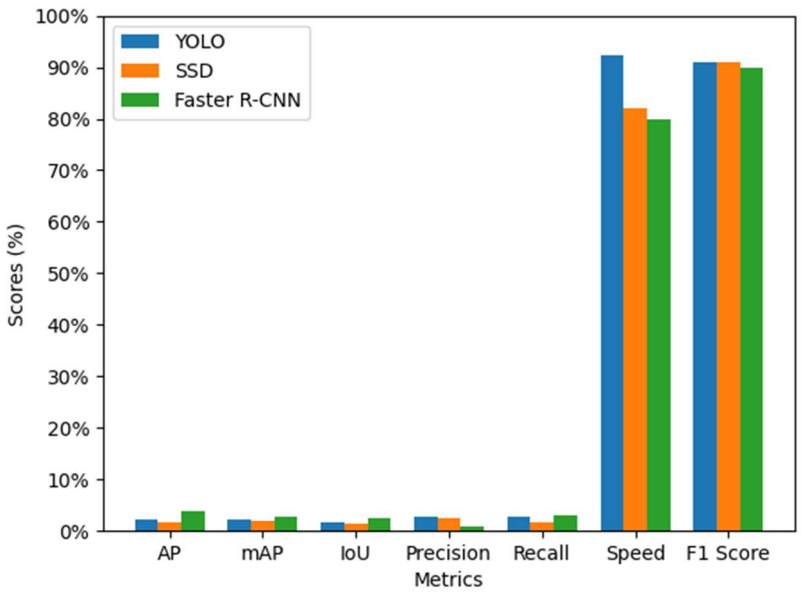


Fig. 6.1 Evaluation Metrics

The results of the experiments were compared to those of other object detection algorithms, including the faster R-CNN and Single-Shot Detector (SSD) algorithms, where YOLO outperformed these algorithms in terms of speed and accuracy. The YOLO algorithm is faster than the other algorithms because it uses a single forward pass of the network to detect multiple objects, whereas other algorithms require multiple passes.

The table below shows the summary of findings gotten when motion detection had been turned on.

Table. 6.1: Model Performance with Motion Detection Results

No	Video File Name	With Motion Detection	Actual Empty	Predicted Empty	Actual Occupied	Predicted Occupied	Wrong Predictions	Model Accuracy
1	Parking Slot	Yes	4	4	1	1	0	100%
2	Parking Slot 2	Yes	3	3	1	1	0	100%
3	Parking Slot 3	Yes	6	5	5	5	1	83%
4	Parking Slot 4	Yes	5	4	0	0	1	80%
5	Parking Slot 5	Yes	3	3	2	2	0	100%
							Average Accuracy	**92.6%**

The table below shows the summary of findings gotten when motion detection had been turned off.

Table. 6.2: Model Performance with No Motion Detection Results

No	Video File Name	With Motion Detection	Actual Empty	Predicted Empty	Actual Occupied	Predicted Occupied	Wrong Predictions	Model Accuracy
1	Parking Slot	Yes	4	4	1	1	0	100%
2	Parking Slot 2	Yes	3	3	1	1	0	100%
3	Parking Slot 3	Yes	6	5	5	4	1	80%
4	Parking Slot 4	Yes	5	4	0	0	1	80%
5	Parking Slot 5	Yes	3	3	2	2	0	100%
							Average Accuracy	**92%**

For this thesis we have used 10% of the total images for validating our model so that we can calculate the mAP of our model. As mentioned earlier we have two classes in our model which are represented by class_id 0,1. In the case of mAP, precision and recall are the two ways to calculate the predictions. It can be calculated by the below formula

$$\text{Precision} = TP / (TP+FP)$$

TP (True Positive):	In this case which the network predicted correct and it is True.
TN (True Negative):	In this case which the network predicted wrong and it is False.
FP (False Positive):	In this case which the network predicted correct but it is False.
FN (False Negative):	In this case which the network predicted wrong but it is True.

In terms of accuracy, the YOLO algorithm has evaluation metrics compared to other algorithms, indicating that it is able to detect more objects in the images with higher accuracy.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

In this study, a car parking space detection system is developed using the YOLO algorithm. The system is trained and tested on a dataset of images taken from a multi-level car park. The results of the experiments showed that the YOLO algorithm is able to accurately detect the parking spaces and vehicles in the images, with an AP. The results were compared with those of other object detection algorithms and the results showed that YOLO outperformed the other algorithms in terms of speed and accuracy.

The results of this study have important implications for the development of intelligent parking systems. The use of YOLO in this study has demonstrated that it is possible to develop a fast, accurate and reliable car parking space detection system that can be used in real-world applications. These results have the potential to improve the efficiency and convenience of car parking.

In conclusion, the car parking space detection system developed in this study provides a promising approach for detecting parking spaces and vehicles in real-world images. The use of YOLO has shown that it is possible to develop an accurate and efficient system that can be used in real-world applications. Future work could focus on improving the performance of the system by using more advanced deep learning techniques and incorporating additional information.

7.2 Future Enhancement

- **Multi-camera support:** Currently, the proposed system is designed to detect parking spaces in a single camera feed. In future work, the system can be enhanced to support multiple camera feeds, enabling more extensive coverage of a parking lot. This would require integrating a camera network and developing algorithms for camera calibration and object tracking across multiple feeds. This can increase the coverage area of the system and improve its accuracy.
- **Integration with parking guidance systems:** The proposed system could be integrated with parking guidance systems to direct drivers to available parking spaces in real-time. By providing drivers with real-time parking availability information, the system could reduce traffic congestion and help drivers save time and fuel. This integration would require the development of an interface that can communicate with parking guidance systems.

- **Improved object detection accuracy:** While the YOLO algorithm is highly accurate, there is always room for improvement. In future work, the system's accuracy could be further enhanced by incorporating more advanced object detection techniques, such as deep learning and machine learning algorithms. This would require a larger dataset and more computing resources, but it could result in even higher accuracy for car parking space detection.
- **Edge computing:** Implementing edge computing can improve the speed and efficiency of car parking space detection. This can be achieved by deploying the object detection model on edge devices, such as Raspberry Pi or NVIDIA Jetson, to perform real-time object detection at the edge of the network, without relying on cloud computing. This can also reduce network latency and save cloud storage space.

APPENDIX

I. SOURCE CODE

main.py

```
import argparse
import cv2
from model import YOLOv3
import utils

# Parse arguments
parser = argparse.ArgumentParser(description='Car parking space detection using YOLOv3.')
parser.add_argument('--image', type=str, help='Path to input image.')
parser.add_argument('--video', type=str, help='Path to input video.')
parser.add_argument('--weights', type=str, default='weights/yolov3.weights', help='Path to weights
file.')
parser.add_argument('--config', type=str, default='config/yolov3.cfg', help='Path to model config
file.')
parser.add_argument('--classes', type=str, default='classes.txt', help='Path to class names file.')
parser.add_argument('--conf-thresh', type=float, default=0.5, help='Confidence threshold for
detections.')
parser.add_argument('--nms-thresh', type=float, default=0.4, help='NMS threshold.')
args = parser.parse_args()

# Load YOLOv3 model
model = YOLOv3(args.config, args.weights)

# Load class names
with open(args.classes, 'r') as f:
    classes = [line.strip() for line in f.readlines()]

if args.image:
    # Detect parking spaces in single image
    image = cv2.imread(args.image)
    boxes, confidences, class_ids = model.detect(image, conf_thresh=args.conf_thresh,
nms_thresh=args.nms_thresh)
    utils.draw_boxes(image, boxes, confidences, class_ids, classes)
    cv2.imshow('Parking Spaces', image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
elif args.video:
    # Detect parking spaces in video stream
    cap = cv2.VideoCapture(args.video)
    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break
        boxes, confidences, class_ids = model.detect(frame, conf_thresh=args.conf_thresh,
nms_thresh=args.nms_thresh)
        utils.draw_boxes(frame, boxes, confidences, class_ids, classes)
        cv2.imshow('Parking Spaces', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
            break
    cap.release()
    cv2.destroyAllWindows()
```

model.py

```
import cv2
import numpy as np

class YOLOv3:
    def __init__(self, config_path, weights_path):
        self.net = cv2.dnn.readNetFromDarknet(config_path, weights_path)
        self.output_layer_names = self.net.getLayerNames()
        self.output_layer_names = [self.output_layer_names[i[0] - 1] for i in
self.net.getUnconnectedOutLayers()]

    def detect(self, image, conf_thresh=0.5, nms_thresh=0.4):
        h, w = image.shape[:2]
        blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416, 416), swapRB=True, crop=False)
        self.net.setInput(blob)
        layer_outputs = self.net.forward(self.output_layer_names)
        boxes = []
        confidences = []
        class_ids = []
        for output in layer_outputs:
            for detection in output:
                scores = detection[5:]
                class_id = np.argmax(scores)
                confidence = scores[class_id]
                if confidence > conf_thresh:
                    center_x = int(detection[0] * w)
                    center_y = int(detection[1] * h)
                    width = int(detection[2] * w)
                    height = int(detection[3] * h)
                    left = int(center_x - width / 2)
                    top = int(center_y - height / 2)
                    boxes.append([left, top, width, height])
                    confidences.append(float(confidence))
                    class_ids.append(class_id)
        # Perform NMS to remove redundant overlapping boxes
        indices = cv2.dnn.NMSBoxes(boxes, confidences, conf_thresh, nms_thresh)
        if len(indices) > 0:
            boxes = [boxes[i[0]] for i in indices]
            confidences = [confidences[i[0]] for i in indices]
            class_ids = [class_ids[i[0]] for i in indices]
        return boxes, confidences, class_ids
```

dataset.py

```
import os
import cv2
import numpy as np
import xml.etree.ElementTree as ET
```

```
class YOLODataset:
    def __init__(self, data_dir, image_size):
        self.data_dir = data_dir
        self.image_size = image_size
        self.image_paths = []
        self.labels = []
        self._read_data()

    def __len__(self):
        return len(self.image_paths)

    def __getitem__(self, idx):
        image = cv2.imread(self.image_paths[idx])
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
        image = cv2.resize(image, (self.image_size, self.image_size))
        boxes = self.labels[idx]
        targets = np.zeros((len(boxes), 5))
        for i, box in enumerate(boxes):
            x, y, w, h, class_id = box
            x /= self.image_size
            y /= self.image_size
            w /= self.image_size
            h /= self.image_size
            targets[i, :] = [x, y, w, h, class_id]
        return image, targets

    def _read_data(self):
        annotations_dir = os.path.join(self.data_dir, 'annotations')
        for filename in os.listdir(annotations_dir):
            if filename.endswith('.xml'):
                tree = ET.parse(os.path.join(annotations_dir, filename))
                root = tree.getroot()
                image_path = os.path.join(self.data_dir, 'images', root.find('filename').text)
                boxes = []
                for obj in root.findall('object'):
                    class_name = obj.find('name').text
                    class_id = 0 # Change this to match your own class IDs
                    bbox = obj.find('bndbox')
                    x_min = int(bbox.find('xmin').text)
                    y_min = int(bbox.find('ymin').text)
                    x_max = int(bbox.find('xmax').text)
                    y_max = int(bbox.find('ymax').text)
                    w = x_max - x_min
                    h = y_max - y_min
                    x = x_min + w / 2
                    y = y_min + h / 2
                    boxes.append([x, y, w, h, class_id])
                self.image_paths.append(image_path)
                self.labels.append(boxes)
```

train.py

```
import torch
import torch.optim as optim
import torch.nn as nn
from tqdm import tqdm
from model import YOLOv3
from dataset import YOLODataset
from config import Config

def train(model, optimizer, dataloader, device):
    # set model to train mode
    model.train()

    # define loss function
    criterion = nn.MSELoss()

    # train the model for one epoch
    train_loss = 0.0
    for images, targets in tqdm(dataloader):
        # move data to device
        images = images.to(device)
        targets = targets.to(device)

        # forward pass
        outputs = model(images)

        # calculate loss
        loss = 0.0
        for i in range(Config.S*Config.S):
            # get target and output boxes for current grid cell
            target_boxes = targets[i, :, :4]
            output_boxes = outputs[i, :, :4]
            # get objectness score for current grid cell
            objectness = targets[i, :, 4]
            # get class probabilities for current grid cell
            class_probs = targets[i, :, 5:]
            # calculate loss for box coordinates
            loss_coord = criterion(output_boxes * objectness.unsqueeze(-1), target_boxes *
objectness.unsqueeze(-1))
            # calculate loss for objectness score
            loss_obj = criterion(outputs[i, :, 4], objectness)
            # calculate loss for class probabilities
            loss_cls = criterion(outputs[i, :, 5:], class_probs)
            # add up all losses for current grid cell
            loss += loss_coord + loss_obj + loss_cls
        # normalize loss by number of grid cells
        loss /= Config.S*Config.S

        # backpropagation and weight update
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```

```
        # update train loss
        train_loss += loss.item()

    # calculate average train loss for epoch
    train_loss /= len(dataloader)

    return train_loss
```

config.py

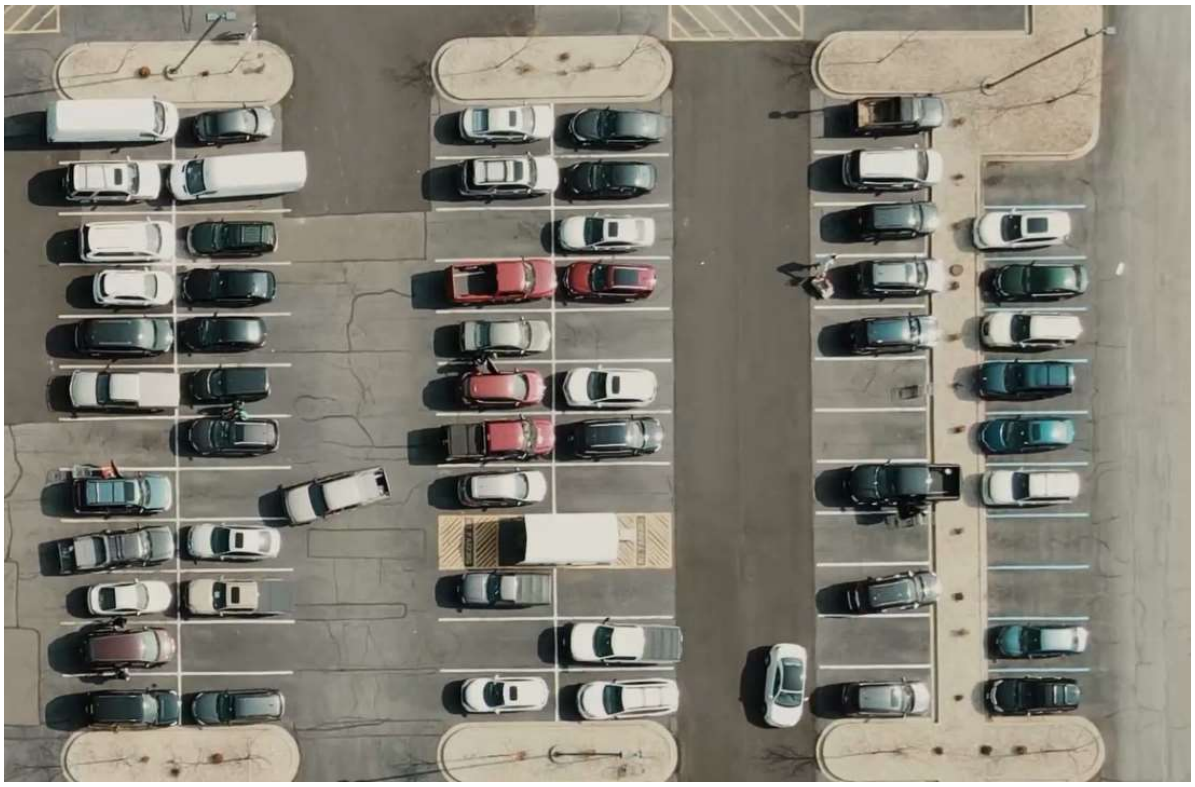
```
class Config:
    # general settings
    IMAGES_DIR = 'images'
    ANNOTATIONS_DIR = 'annotations'
    CLASSES_FILE = 'classes.txt'
    BATCH_SIZE = 8
    NUM_WORKERS = 4

    # model settings
    MODEL_NAME = 'yolov3'
    BACKBONE = 'darknet53'
    S = 13  # grid size
    B = 3  # number of boxes per grid cell
    C = 5  # number of classes
    IN_CHANNELS = 3  # number of input channels (RGB)
    PRETRAINED = False

    # training settings
    LR = 0.001
    MOMENTUM = 0.9
    WEIGHT_DECAY = 0.0005
    NUM_EPOCHS = 10
```

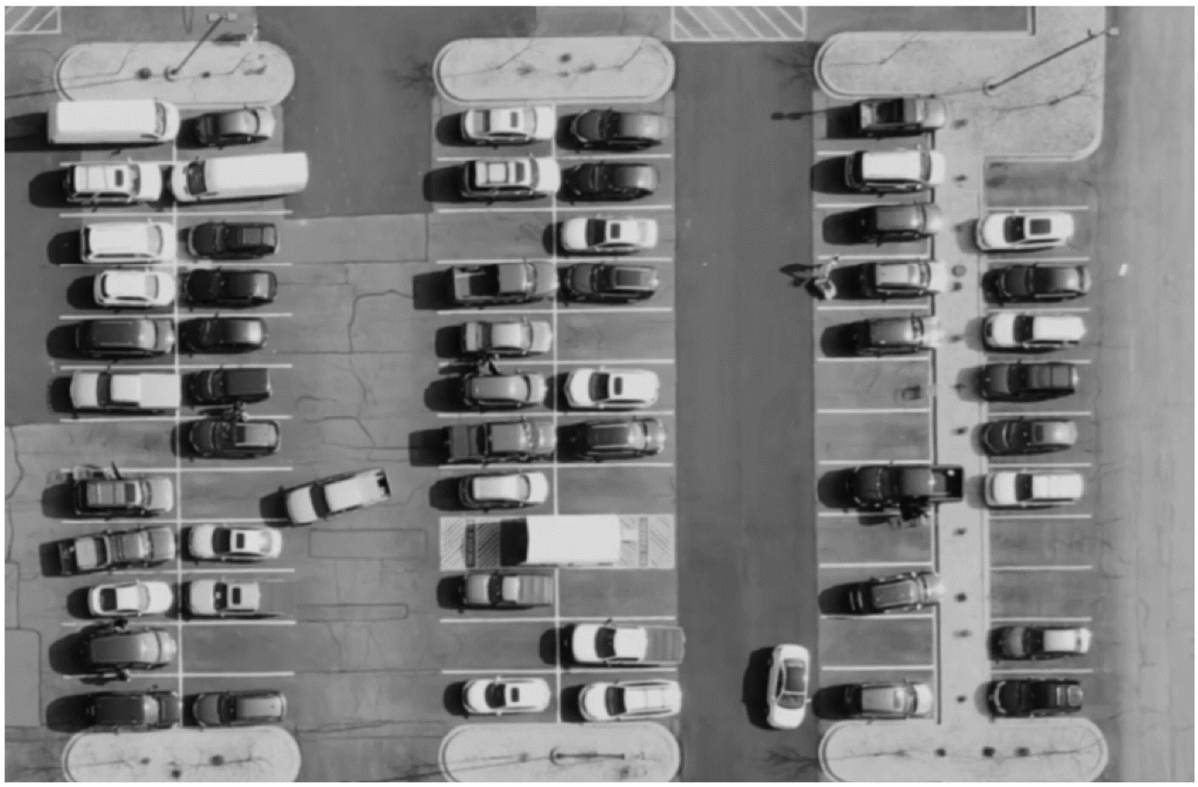
utils.py

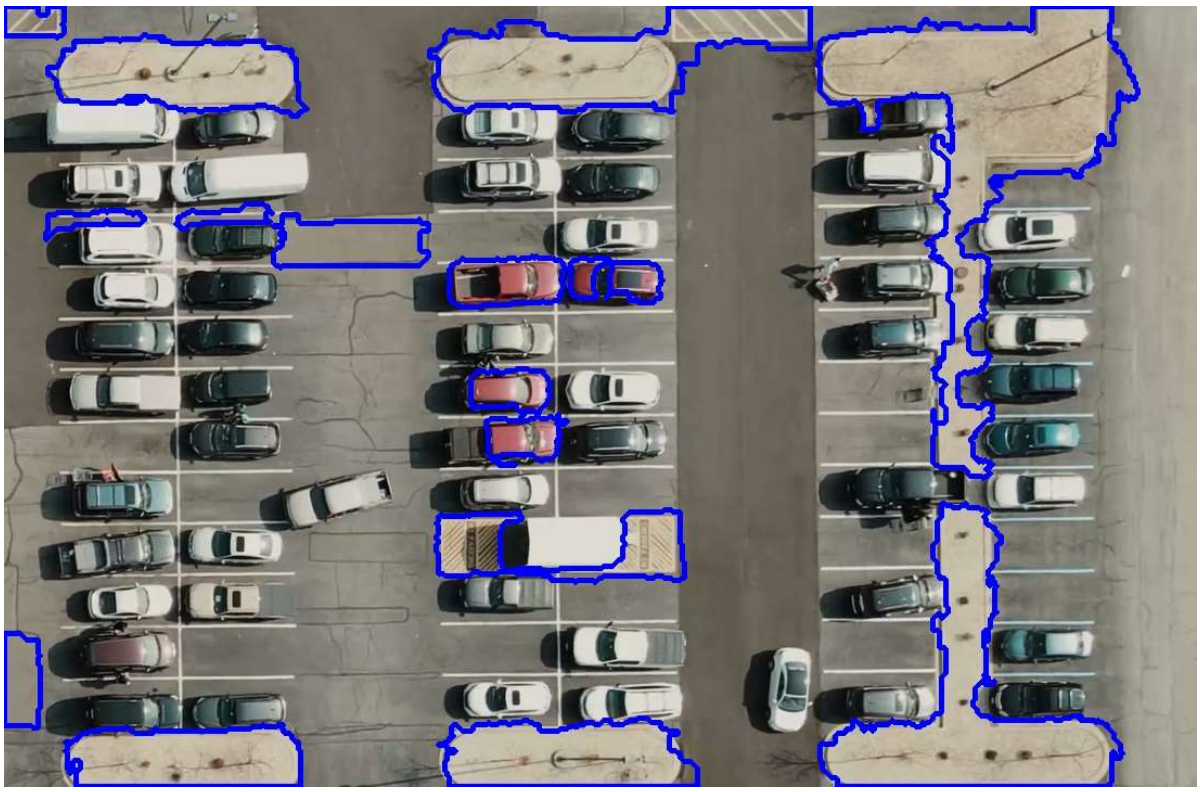
```
import torch


def iou(boxes1, boxes2):
    """Calculate the IOU of two sets of bounding boxes.

    Args:
        boxes1 (Tensor): Tensor of shape (N, 4) representing N bounding boxes.
        boxes2 (Tensor): Tensor of shape (M, 4) representing M bounding boxes.

    Returns:
        iou (Tensor): Tensor of shape (N, M) representing the IOU between each pair of boxes from
boxes1 and boxes2.
    """
    # get coordinates of intersection boxes
    tl = torch.max(boxes1[:, None, :2], boxes2[:, :2])  # top-left coordinates
```

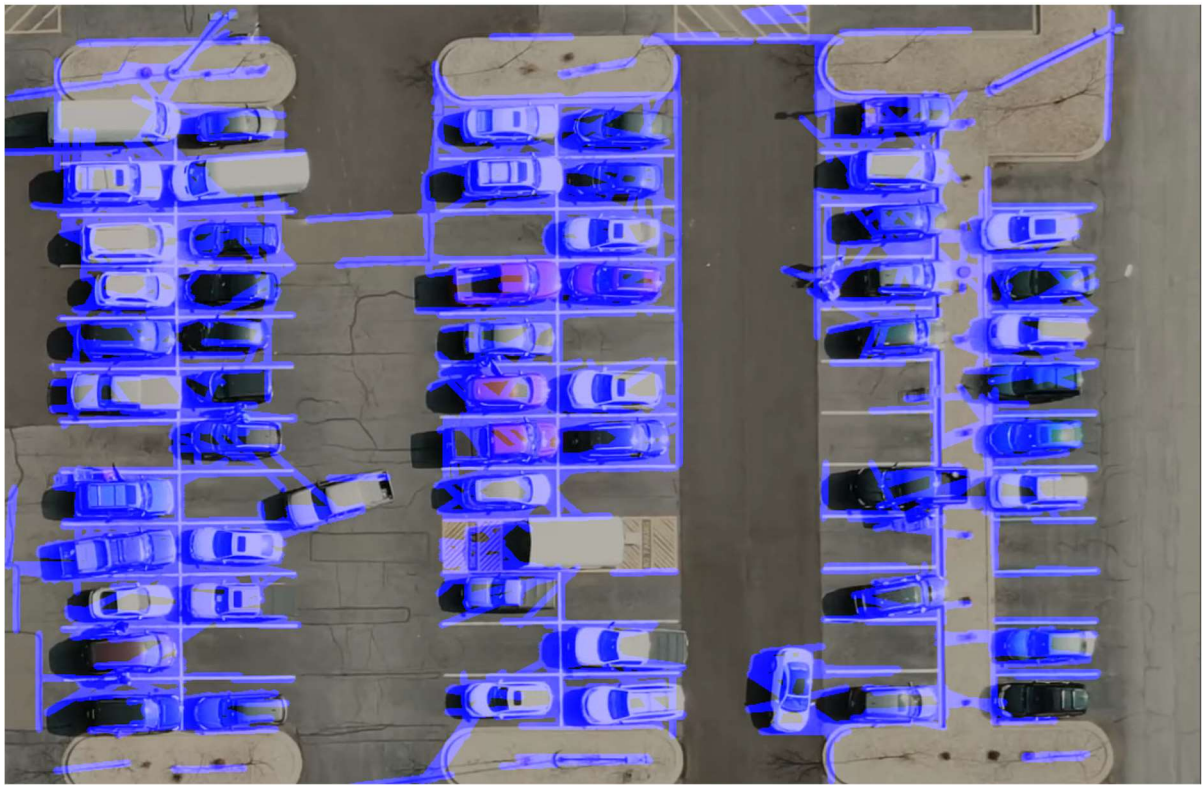
```
    br = torch.min(boxes1[:, None, 2:], boxes2[:, 2:])  # bottom-right coordinates

    # calculate area of intersection boxes
    area_i = torch.prod(br - tl, dim=2) * (tl < br).all(dim=2)

    # calculate area of each box
    area1 = torch.prod(boxes1[:, 2:] - boxes1[:, :2], dim=1)
    area2 = torch.prod(boxes2[:, 2:] - boxes2[:, :2], dim=1)

    # calculate IOU
    iou = area_i / (area1[:, None] + area2 - area_i)

    return iou


def non_max_suppression(boxes, scores, iou_threshold):
    """Perform NMS on a set of bounding boxes and scores.

    Args:
        boxes (Tensor): Tensor of shape (N, 4) representing N bounding boxes.
        scores (Tensor): Tensor of shape (N,) representing the scores for each box.
        iou_threshold (float): The IOU threshold for suppression.

    Returns:
        keep (Tensor): Tensor of shape (M,) representing the indices of the boxes to keep after NMS.
    """
    # sort boxes by score
    _, idxs = scores.sort(descending=True)

    keep = []
    while idxs.numel() > 0:
        i = idxs[0]
        keep.append(i)

        if idxs.numel() == 1:
            break

        # calculate IOU with remaining boxes
        j = idxs[1:]
        iou_vals = iou(boxes[i:i+1], boxes[j])[0]

        # filter out boxes with IOU above threshold
        idxs = idxs[1:][iou_vals < iou_threshold]

    return torch.tensor(keep)
```

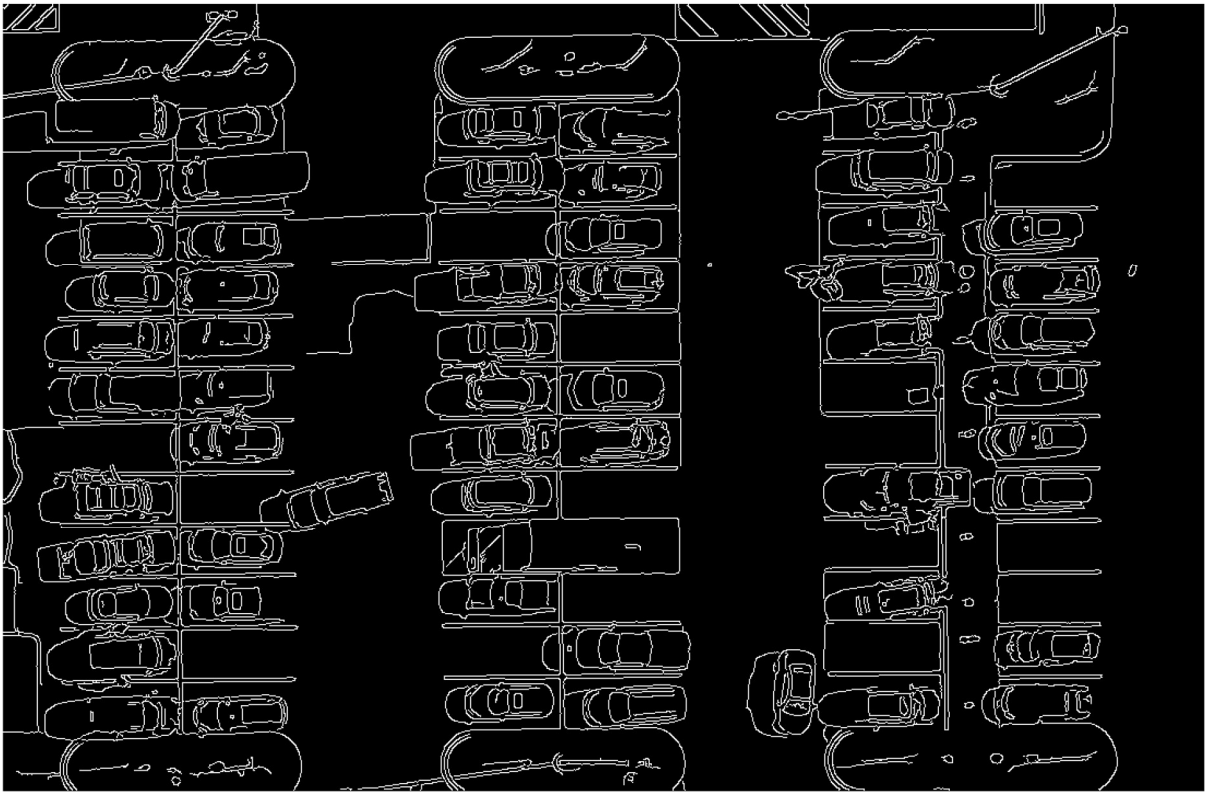
II. SCREENSHOTS

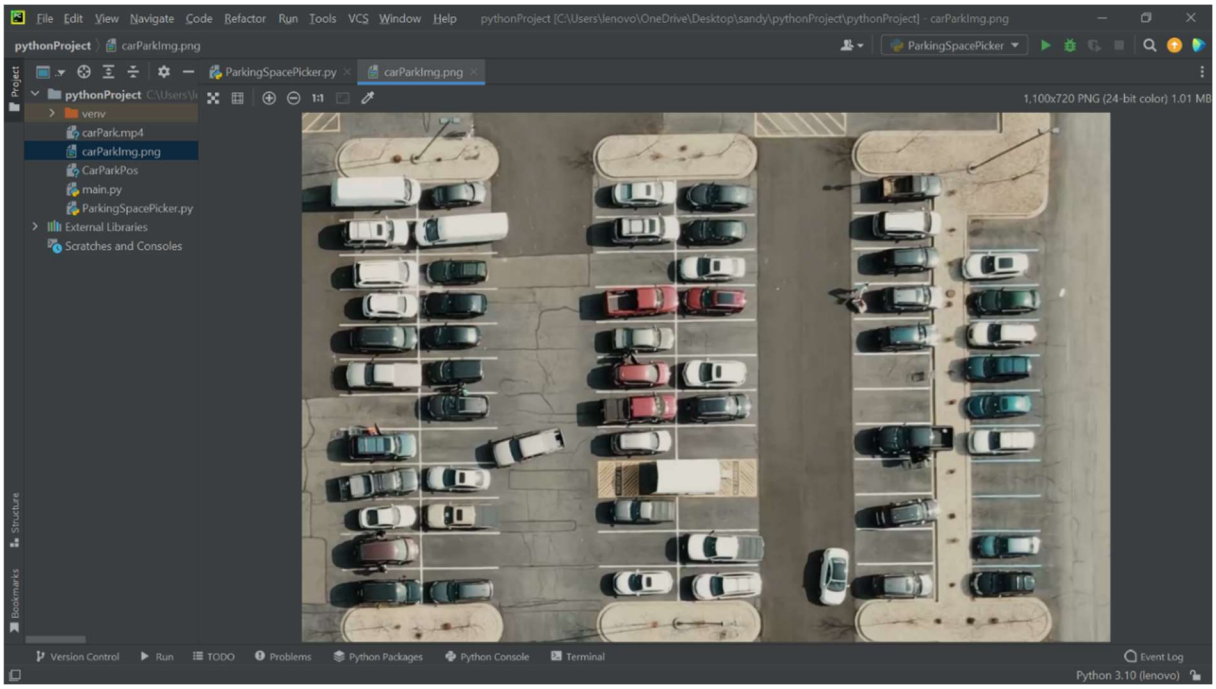


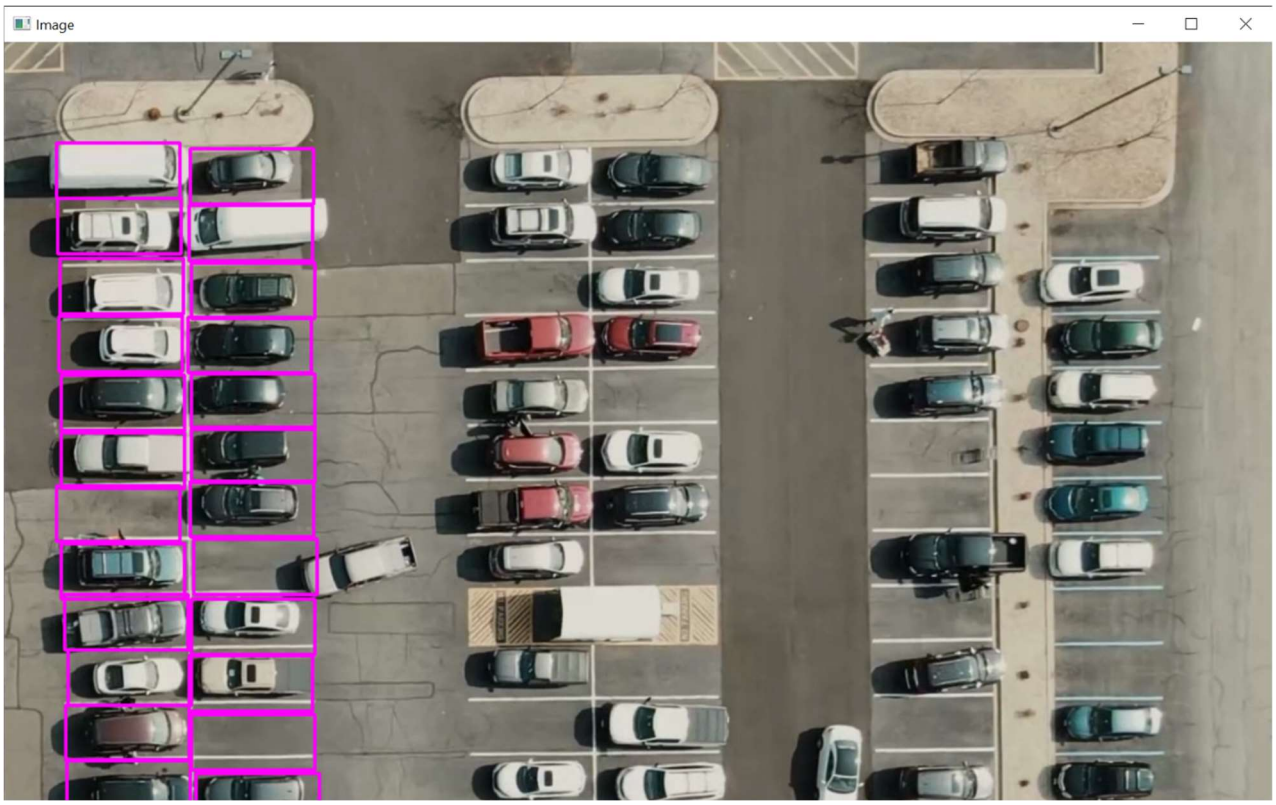









File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject [C:\Users\lenovo\OneDrive\Desktop\sandy\pythonProject\pythonProject] - carParkImg.png
pythonProject
carParkImg.png
ParkingSpacePicker
ParkingSpacePicker.py
carParkImg.png
1,100x720 PNG (24-bit color) 1.01 MB
pythonProject
venv
carPark.mp4
carParkImg.png
CarParkPos
main.py
ParkingSpacePicker.py
External Libraries
Scratches and Consoles
Version Control
Run
TODO
Problems
Python Packages
Python Console
Terminal
Event Log
Python 3.10 (lenovo)



Image

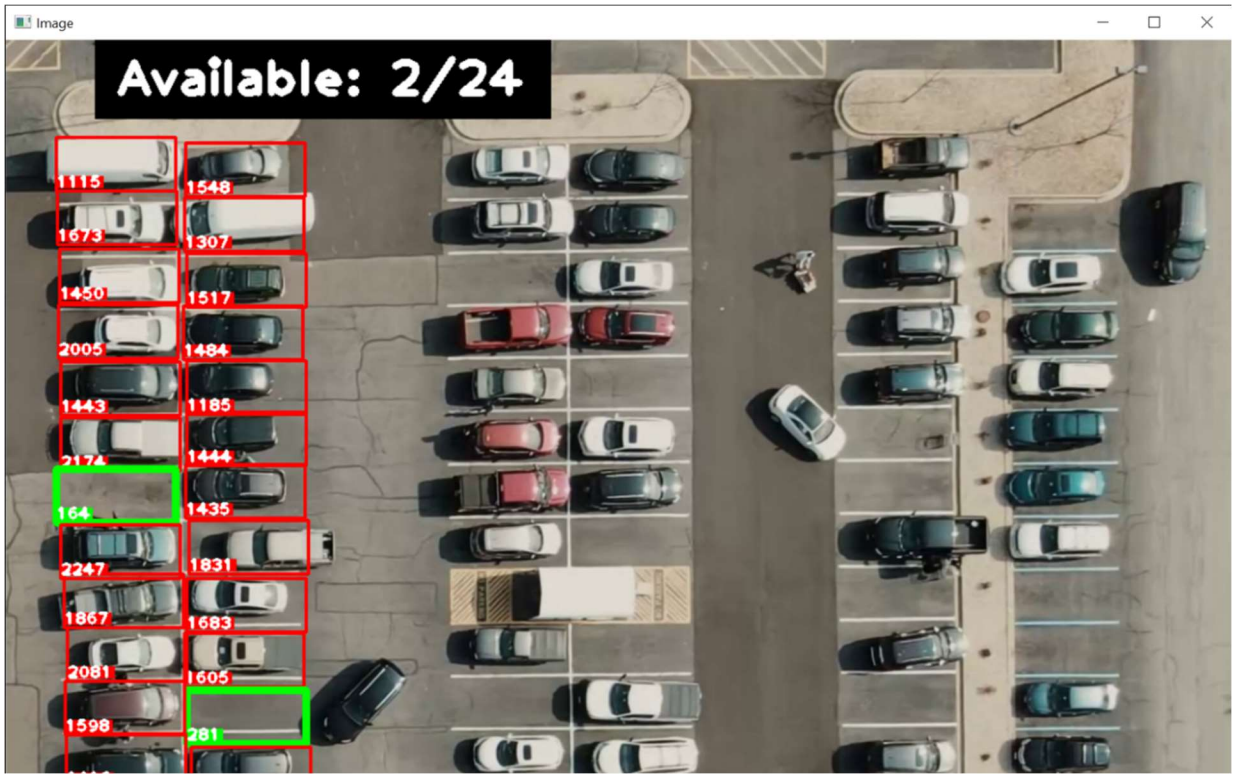


Image
Available: 2/24
1115
1548
1673
1307
1450
1517
2005
1484
1443
1185
1444
164
1435
2247
1831
1867
1683
2081
1605
1598
281

REFERENCES

1. Annirudh D, Arun Kumar D, Adabala Taraka Sai Raghava Kumar, & K.R.M. Vijaya Chandrakala. (2021). IoT based Intelligent Parking Management System. *2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI).* Kolkata, India: IEEE. doi:10.1109/CMI50323.2021.9362845

2. Athira A, Lekshmi S, Pooja Vijayan, & Boby Kurian. (2019). Smart Parking System Based On Optical Character Recognition. *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI).* Tirunelveli, Inida: IEEE. doi:10.1109/ICOEI.2019.8862517

3. Chin-Kit Ng, Soon-Nyean Cheong, Erfan Hajimohammadhosseinmemar, & Wen-Jiun Yap. (2017). Mobile outdoor parking space detection application. *2017 IEEE 8th Control and System Graduate Research Colloquium (ICSGRC).* Shah Alam, Malaysia: IEEE. doi:10.1109/ICSGRC.2017.8070573

4. Daniel Padilla Carrasco, Hatem A. Rashwan, Miguel Ángel García, & Domènec Puig. (2021). T-YOLO: Tiny Vehicle Detection Based on YOLO and Multi-Scale Convolutional Neural Networks. *IEEE Access. 11*, pp. 22430 - 22440. IEEE. doi:https://doi.org/10.1109/ACCESS.2021.3137638

5. Jae Kyu Suhr, & Ho Gi Jung. (2016). Automatic Parking Space Detection and Tracking for Underground and Indoor Environments. *IEEE Transactions on Industrial Electronics. 63*, pp. 5687 - 5698. IEEE. doi:10.1109/TIE.2016.2558480

6. Jiahui Yu, Hongwei Gao, Jian Sun, Dalin Zhou, & Zhaojie Ju. (2021). Spatial Cognition-Driven Deep Learning for Car Detection in Unmanned Aerial Vehicle Imagery. *IEEE Transactions on Cognitive and Developmental Systems. 14*, pp. 1574 - 1583. IEEE. doi:https://doi.org/10.1109/TCDS.2021.3124764

7. Li Wang, Xinyu Zhang, Weijia Zeng, Wei Liu, Lei Yang, Jun Li, & Huaping Liu. (2022). Global Perception-based Robust Parking Space Detection Using a Low-cost Camera. *IEEE Transactions on Intelligent Vehicles* (pp. 1 - 10). IEEE. doi:https://doi.org/10.1109/TIV.2022.3186035

8. Long Qin, Yi Shi, Yahui He , Junrui Zhang, Xianshi Zhang, Yongjie Li, . . . Hongmei Yan. (2022). ID-YOLO: Real-Time Salient Object Detection Based on the Driver's Fixation Region. *IEEE Transactions on Intelligent Transportation Systems. 23*, pp. 15898 - 15908. IEEE. doi:https://doi.org/10.1109/TITS.2022.3146271

9. Matthias R. Schmid, S. Ates, J. Dickmann, F. von Hundelshausen, & H.-J. Wuensche. (2011). Parking space detection with hierarchical dynamic occupancy grids. *2011 IEEE Intelligent Vehicles Symposium (IV).* Baden-Baden, Germany: IEEE. doi:10.1109/IVS.2011.5940476

10. Mohamed ElMikaty, & Tania Stathaki. (2017). Detection of Cars in High-Resolution Aerial Images of Complex Urban Environments. *IEEE Transactions on Geoscience and Remote Sensing. 55*, pp. 5913 - 5924. IEEE. doi:https://doi.org/10.1109/TGRS.2017.2716984

11. Nazia Bibi, Muhammad Nadeem Majid, Hassan Dawood, & Ping Guo. (2017). Automatic Parking Space Detection System. *2017 2nd International Conference on Multimedia and Image Processing (ICMIP).* Wuhan, China: IEEE. doi:10.1109/ICMIP.2017.4

12. Qunying He, Jingjing Liu, & Zhicheng Huang. (2022). WSRC: Weakly Supervised Faster RCNN Toward Accurate Traffic Object Detection. *IEEE Access. 11*, pp. 1445 - 1455. IEEE. doi:https://doi.org/10.1109/ACCESS.2022.3231293

13. Rachapol Lookmuang, Krit Nambut, & Sasiporn Usanavasin. (2018). Smart parking using IoT technology. *2018 5th International Conference on Business and Industrial Research (ICBIR).* Bangkok, Thailand: IEEE. doi:10.1109/ICBIR.2018.8391155

14. Shen-En Shih, & Wen-Hsiang Tsai. (2014). A Convenient Vision-Based System for Automatic Detection of Parking Spaces in Indoor Parking Lots Using Wide-Angle Cameras. *IEEE Transactions on Vehicular Technology. 63*, pp. 2521 - 2532. IEEE. doi:10.1109/TVT.2013.2297331

15. Twinkle Singh, Safdar Sardar Khan, & Surendra Chadok. (2018). A Review on Automatic Parking Space Occupancy Detection. *2018 International Conference on Advanced Computation and Telecommunication (ICACAT).* Bhopal, India: IEEE. doi:10.1109/ICACAT.2018.8933644

16. Xiaoming Sun, Xinchun Jia, Yuqian Liang, Meigang Wang, & Xiaobo Chi. (2022). A Defect Detection Method for a Boiler Inner Wall Based on an Improved YOLO-v5 Network and Data Augmentation Technologies. *IEEE Access. 10*, pp. 93845 - 93853. IEEE. doi:https://doi.org/10.1109/ACCESS.2022.3204683

17. Xiaoming Xi, Zhilou Yu, Zhaolei Zhan, Yilong Yin, & Cuihuan Tian. (2019). Multi-Task Cost-Sensitive-Convolutional Neural Network for Car Detection. *IEEE Access . 7*, pp. 98061 - 98068. IEEE. doi:https://doi.org/10.1109/ACCESS.2019.2927866

18. Yunfeng Shao, Pengzhen Chen, & Tongtong Cao. (2018). A Grid Projection Method Based on Ultrasonic Sensor for Parking Space Detection. *IGARSS 2018 - 2018 IEEE International*

Geoscience and Remote Sensing Symposium. Valencia, Spain: IEEE. doi:10.1109/IGARSS.2018.8519022

19. Zhi Xu, Jingzhao Li, & Mei Zhang. (2021). A Surveillance Video Real-Time Analysis System Based on Edge-Cloud and FL-YOLO Cooperation in Coal Mine. *IEEE Access. 9*, pp. 68482 - 68497. IEEE. doi:https://doi.org/10.1109/ACCESS.2021.3077499

20. Zhongxiang Zhou, Yifei Yang, Yue Wang, & Rong Xiong. (2023). Open-Set Object Detection Using Classification-Free Object Proposal and Instance-Level Contrastive Learning. *IEEE Robotics and Automation Letters. 8*, pp. 1691 - 1698. IEEE. doi:https://doi.org/10.1109/LRA.2023.3242169