

AI ENABLED CAR PARKING USING OPEN CV

A Project Report

Submitted by

VENUGOPAL V

SHYJIN A

RAMESH S

SHIJO M

*In partial fulfillment for the award of the degree
Of*

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING**



ST. XAVIER'S CATHOLIC COLLEGE OF ENGINEERING

(An Autonomous Institution Affiliated to Anna University)

NAGERCOIL – 629 003

May 202

LIST OF CONTENTS

S.NO	CONTENTS	PAGE NO
1	INTRODUCTION	3
2	IDEATION & PROPOSED SOLUTION	5
3	REQUIREMENT ANALYSIS	15
4	PROJECT DESIGN	19
5	CODING & SOLUTIONING	24
6	RESULTS	25
7	ADVANTAGES & DISADVANTAGES	30
8	CONCLUSION	32
9	FUTURE SCOPE	33
10	APPENDIX	36

1. INTRODUCTION

AI-enabled car parking systems leveraging OpenCV utilize computer vision techniques to revolutionize traditional parking management. These systems employ cameras and sensors to monitor and analyze parking areas in real time. OpenCV's image processing capabilities enable accurate vehicle detection, tracking, and differentiation. By optimizing parking space utilization and guiding drivers to vacant spots, these systems improve efficiency and reduce congestion. Additionally, OpenCV's integration with intelligent surveillance features enhances security by detecting and alerting suspicious activities. AI-enabled car parking systems are poised to become an integral part of smart cities and transportation infrastructure, offering a seamless parking experience and contributing to a more connected and sustainable future.

1.1 Project Overview

The AI-enabled car parking project utilizes OpenCV and artificial intelligence to develop an intelligent parking system. The project aims to address challenges faced by traditional systems, such as limited space and human errors. Key objectives include real-time monitoring and analysis using cameras and sensors, vehicle recognition and differentiation, occupancy detection and guidance, security and surveillance using computer vision algorithms, and a user-friendly interface. The system will be scalable, easily integrated with existing infrastructure, and measured based on accuracy, optimization, efficiency, and security. The project aims to enhance the parking experience, contribute to smart cities, and promote sustainability.

1.2 Purpose

The purpose of AI-enabled car parking using OpenCV is to optimize parking space utilization, enhance efficiency and convenience, improve security and surveillance, integrate with smart city infrastructure, and provide data-driven insights. By leveraging artificial intelligence and computer vision, the system aims to revolutionize traditional parking management systems. It optimizes parking space allocation, reduces the search time for drivers, enhances security through real-time monitoring, seamlessly integrates with smart city infrastructure, and generates valuable data for informed decision-making. The purpose is to transform the parking experience, alleviate urban parking challenges, and contribute to the development of smarter and more sustainable cities.

2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition

The problem addressed by the implementation of AI-enabled car parking using OpenCV is the inefficiency, inconvenience, and security concerns associated with traditional parking management systems. The key problems that this technology-driven solution aims to solve include:

Inefficient parking space utilization: Traditional parking systems often suffer from inefficient utilization of available parking spaces. Vehicles are often parked haphazardly, leading to wasted space and overcrowding. This inefficiency poses challenges in accommodating a growing number of vehicles in urban areas.

Time-consuming search for parking: Drivers frequently encounter the frustrating task of searching for available parking spaces. Inefficient parking guidance systems and lack of real-time information result in wasted time and increased traffic congestion in and around parking areas.

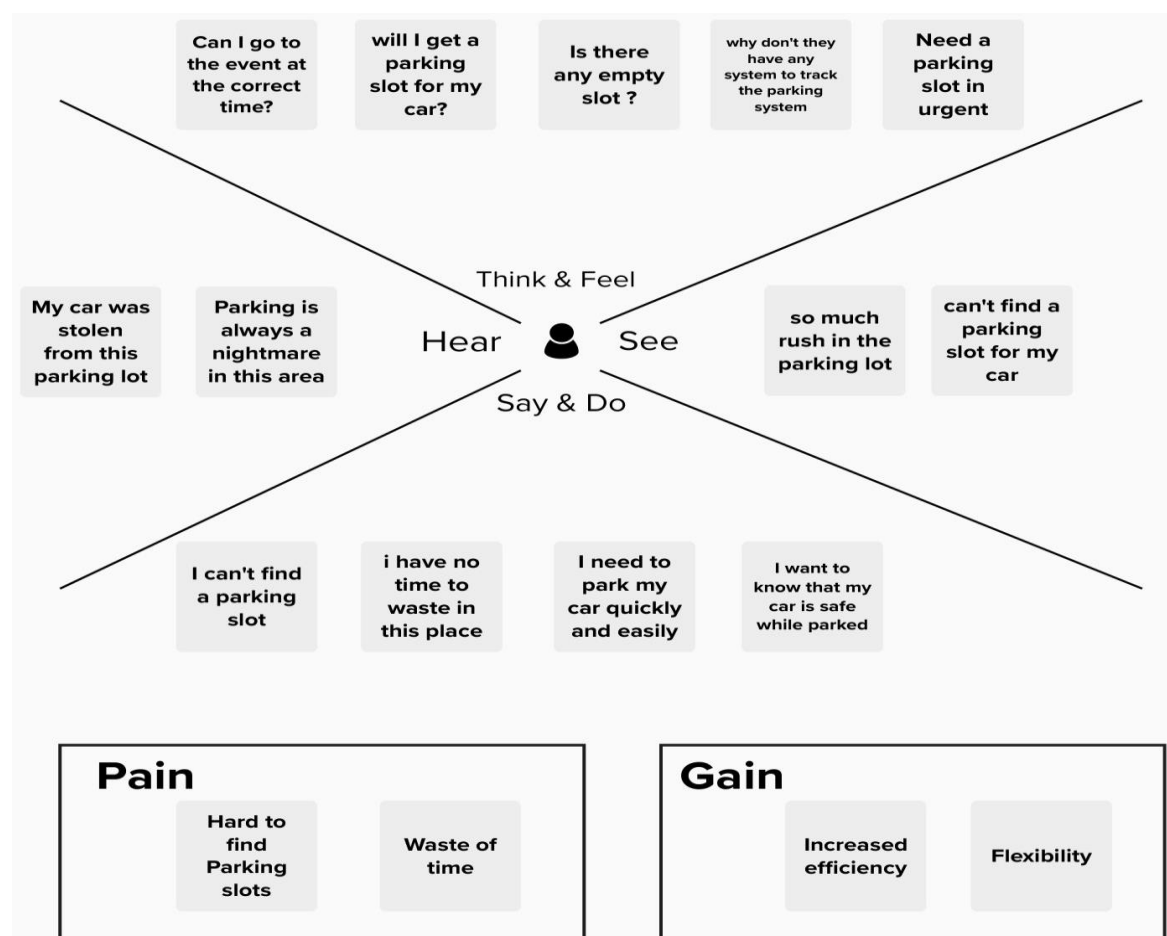
Human errors and inconsistencies: Manual operations and human errors in guiding vehicles often lead to inconsistent and unreliable parking experiences. Inaccurate vehicle counting, improper allocation of spaces, and lack of surveillance contribute to operational inefficiencies and inconvenience for drivers.

Security vulnerabilities: Traditional parking systems are susceptible to security vulnerabilities, including unauthorized access, vehicle theft, and vandalism. Insufficient surveillance and limited security measures pose risks to parked vehicles and compromise the safety of drivers and their belongings.

Lack of integration with smart city infrastructure: Traditional parking systems often lack integration with broader smart city infrastructure. This results in missed opportunities for leveraging data-driven insights, seamless connectivity with transportation systems, and coordinated urban planning and traffic management efforts.

2.2 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.




2.3 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration, and Select the Problem Statement

Template



Brainstorm & idea prioritization

AI enabled car parking using open CV

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-8 people recommended

Share template feedback

➔

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

A

Team gathering

The Team consist of 4 member(VenuGopal V. Shyjin A., Ramesh S, Shijo M).Brainstroming ideasabout the project are discussed among the team members

B

Set the goal

The goal is to improve the accuracy of finding the best parking slot for the vehicle

C

Techniques used

Image processing
Deep learning algorithms
Open CV
Open article ➔

1

Define your problem statement

This project is to design and implement a smart parking system that uses AI and computer vision technologies to detect and monitor available parking spots in real-time.

🕒 5 Minutes

PROBLEM

The problem is to develop a deep learning model that can accurately identify a parking slot

Key rules of brainstorming

To run an smooth and productive session

➔

Stay in topic.

💡

Encourage wild ideas.

➔

Defer judgment.

👂

Listen to others.

🗣️

Go for volume.

👁️

If possible, be visual.

Step-2: Brainstorm, Idea Listing, and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

VenuGopal V

- Install a CCTV camera in the car parking area
- By using CCTV camera and open CV to detect the empty slot in parking area
- Open CV detect the empty parking slot
- If the parking slot is empty it will indicate as green color else it will indicate as red color
- To manage the parking slot. We have to develop one parking application

Ramesh S

- Is there any alternative model
- Is it hard to use
- will it work in all platforms
- Can we improve its accuracy
- Will it take more time to identify the parking slot

Shijo M

- It help the car drivers to park the car without any human intervention
- Install camera and sensor to guide the drivers to park the car to the empty parking slot
- This system uses camera ,sensor and gps to locate the car drivers to the empty slot in the car parking ground
- The system can also retrieve the car when the driver is ready to leave.
- Is it useful for public

Shyjin A

- Will it take more storage space
- can this application used for drivers
- By using this application we can book parking slots
- The system can then drivers to the nearest available parking spot
- will it work accurately

3

Group ideas

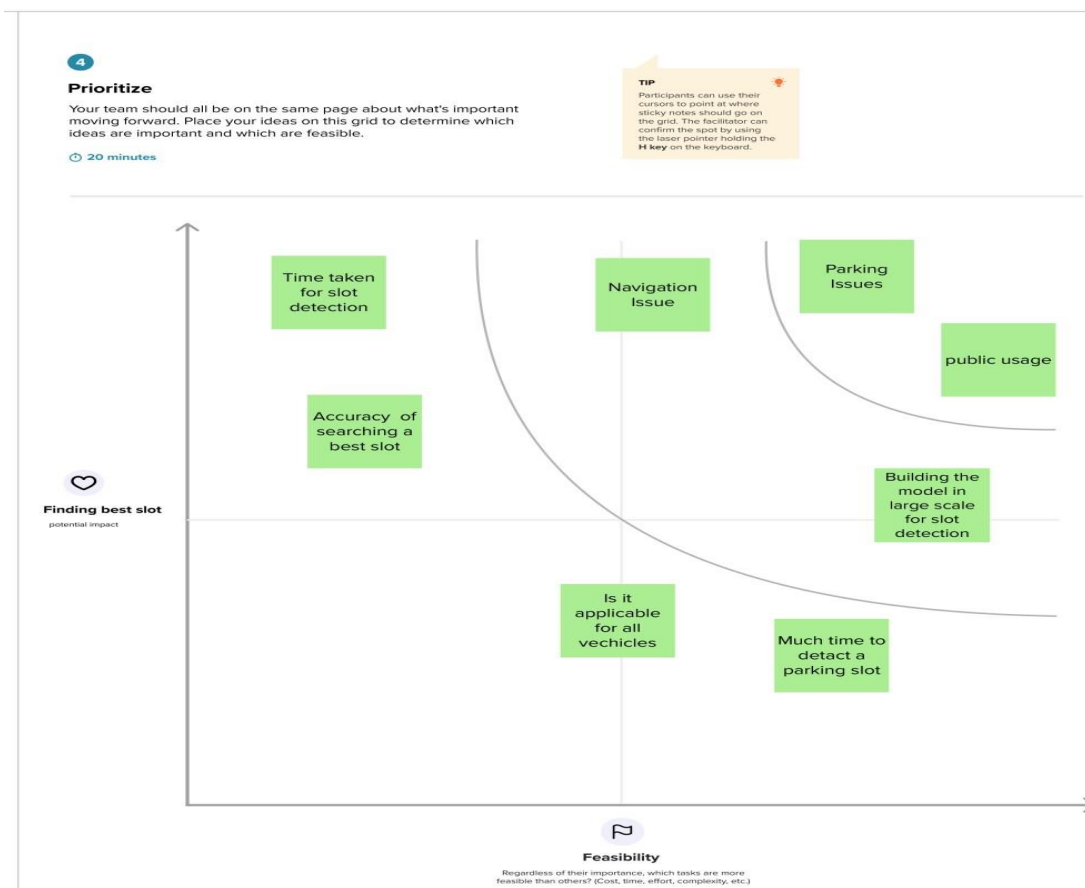
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

Grouped Ideas:

- Classify and identify the parking lot
- Find an empty slot and mark it as a green colored box.
- Find an occupied parking space and mark it in red.
- This system uses camera ,sensor and gps to locate the car drivers to the empty slot in the car parking ground
- By using CCTV camera and open CV to detect the empty slot in parking area
- Open CV detect the empty parking slot.
- To manage the parking slot. We have to develop one parking application.
- By using this application we can book parking slots

Step-3: Idea Prioritization



2.4 Proposed Solution

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem statement for AI-enabled car parking using OpenCV would be to develop a system that can efficiently manage car parking spaces in a given area. This system should be able to detect available parking spots and guide drivers to these spots using real-time image processing with OpenCV. The system should also be capable of tracking vehicles within the parking lot and providing relevant information to the parking management team to ensure smooth parking operations.
2.	Idea / Solution description	An AI-enabled car parking system using OpenCV can be designed to provide a real-time parking management solution for large parking areas, such as shopping malls, airports, and stadiums. The system can use a combination of cameras, sensors, and machine learning algorithms to detect available parking spaces and guide drivers to these spaces.

3.	Novelty / Uniqueness	<ol style="list-style-type: none"> 1. Real-time image processing: The system uses real-time image processing with OpenCV to detect available parking spaces and guide drivers to these spaces. 2. User-friendly interface: The system provides a user-friendly interface that allows drivers to easily locate available parking spaces and navigate the parking area 3. Data analytics: The system generates valuable data on parking utilization, such as the number of vehicles parked, the average duration of parking, and the busiest times of the day. 4. Flexibility: The system can be customized to fit the specific needs of different parking areas, such as shopping malls, airports, and stadiums. This allows for a more flexible and adaptable parking management solution.
----	----------------------	--

4.	Social Impact / Customer Satisfaction	<ol style="list-style-type: none"> 1. Reduced congestion: The system can help reduce congestion in parking areas by efficiently guiding drivers to available parking spaces. 2. Improved safety: The system can improve safety in parking areas by reducing the likelihood of accidents caused by drivers circling the parkinglot in search of a spot 3. Environmental benefits: The system can help reduce carbon emissions by reducing the amount of time drivers spend circling parking areas in search of a spot. 4. Enhanced customer experience: The system provides a user-friendly interface that allows drivers to easily locate available parking spaces and navigate the parking area <p>Improved accessibility: The system can be designed to include features that improve</p>
----	---------------------------------------	---

		accessibility for drivers with disabilities, such as reserved parking spots and audio guidance for visually impaired drivers.
5.	Business Model (Revenue Model)	<ol style="list-style-type: none"> 1. Subscription model: The parking area can charge a monthly or yearly subscription fee to drivers who use the system to locate parking spaces. 2. Pay-per-use model: The parking area can charge drivers a fee for each parking session, based on the duration of the session and the type of parking spot accessed by the driver. 3. Advertising model: The parking area can generate revenue by displaying targeted advertisements within the user interface of the system. 4. Data analytics model: The system can generate valuable data on parking utilization,

		<p>which can be sold to third-party organizations, such as city planners or real estate developers.</p> <p>5. Value-added services model: The parking area can offer value-added services, such as car wash or car detailing services, to drivers who use the system to locate parking spaces.</p>
6.	Scalability of the Solution	<p>1. Open-source technology: OpenCV is an open-source computer vision library, which means it can be easily customized and integrated with other technologies.</p> <p>2. Cloud-based infrastructure: The system can be hosted on cloud-based infrastructure, which allows for easy scaling and management.</p> <p>3. Machine learning algorithms: The machine learning algorithms used in the system can be trained and fine-tuned based on the</p>

		<p>specific parking area and environment.</p> <p>4. Customizable user interface: The user interface of the system can be customized to meet the specific needs of different parking areas and drivers.</p> <p>5. Integration with other systems: The system can be integrated with other systems, such as payment gateways and access control systems, to create a seamless and integrated parking management solution.</p>
--	--	---

3. REQUIREMENT ANALYSIS

3.1 Functional requirement

Vehicle Detection and Tracking: The system should utilize OpenCV's computer vision capabilities to accurately detect and track vehicles entering and exiting the parking area in real time. This requirement ensures that the system can monitor the occupancy status of parking spaces.

Vehicle Classification: The AI-enabled system should be able to differentiate between different types of vehicles, such as cars, motorcycles, and trucks. This functionality helps optimize parking space allocation based on the specific requirements of each vehicle type.

Occupancy Monitoring: The system should continuously monitor the occupancy status of individual parking spaces. It should accurately determine whether a parking spot is vacant or occupied in real time.

Parking Space Guidance: The AI-enabled system should provide real-time guidance to drivers, directing them to available parking spaces. This requirement ensures efficient utilization of parking areas and reduces the time spent searching for parking.

Security and Surveillance: The system should incorporate intelligent surveillance features using OpenCV. It should detect and raise alerts for suspicious activities, such as unauthorized access, vehicle theft, or vandalism. This requirement enhances the security of the parking area and ensures the safety of parked vehicles.

User Interface: The system should provide a user-friendly interface, accessible through mobile applications or digital displays. The interface should display real-

time information on available parking spaces, guidance for parking, and any security alerts.

Integration with Smart City Infrastructure: The AI-enabled car parking system should support seamless integration with smart city infrastructure. It should offer APIs or integration options to connect with transportation systems, parking management platforms, and other smart city applications. This requirement enables data exchange and coordinated efforts in urban planning and traffic management.

Scalability and Flexibility: The system should be scalable to accommodate parking lots of varying sizes, from small lots to large multi-level parking structures. It should also be flexible enough to adapt to different camera and sensor configurations.

Performance and Accuracy: The AI-enabled system should demonstrate high performance and accuracy in vehicle detection, tracking, and classification. It should minimize false positives and negatives to ensure reliable parking occupancy information.

Data Analytics and Reporting: The system should capture and analyze parking occupancy data to generate insights for decision-making. It should provide reporting capabilities, including statistics on occupancy rates, peak hours, and trends, to support efficient resource allocation and planning.

Compliance with Privacy Regulations: The AI-enabled car parking system should adhere to privacy regulations and guidelines. It should ensure the responsible handling of personal data collected during the parking process.

3.2 Non-Functional requirements

Performance: The system should exhibit high performance, ensuring real-time processing and response for vehicle detection, tracking, and classification. It should minimize latency and provide quick and accurate results to ensure a smooth parking experience for drivers.

Accuracy: The AI-enabled car parking system should demonstrate a high level of accuracy in vehicle detection, tracking, and classification. It should minimize false positives and negatives to ensure reliable occupancy information and guidance for drivers.

Reliability: The system should be reliable, operating consistently and without disruptions. It should be able to handle high volumes of vehicles and effectively handle potential issues, such as system failures or network outages, to ensure uninterrupted parking management.

Scalability: The system should be scalable, capable of accommodating varying parking lot sizes and increasing volumes of vehicles. It should efficiently handle a growing number of parking spaces and vehicles without compromising performance or accuracy.

Security: The AI-enabled car parking system should have robust security measures in place to protect the integrity and confidentiality of data collected and stored. It should implement encryption protocols, access controls, and authentication mechanisms to prevent unauthorized access or data breaches.

Usability: The system should have a user-friendly interface that is intuitive and easy to navigate for both administrators and drivers. It should provide clear and concise instructions, guidance, and feedback to ensure a seamless and convenient parking experience.

Maintainability: The system should be designed and developed with maintainability in mind. It should be modular and well-documented, allowing for easy maintenance, updates, and enhancements. The codebase should follow best practices and coding standards to facilitate future modifications or additions.

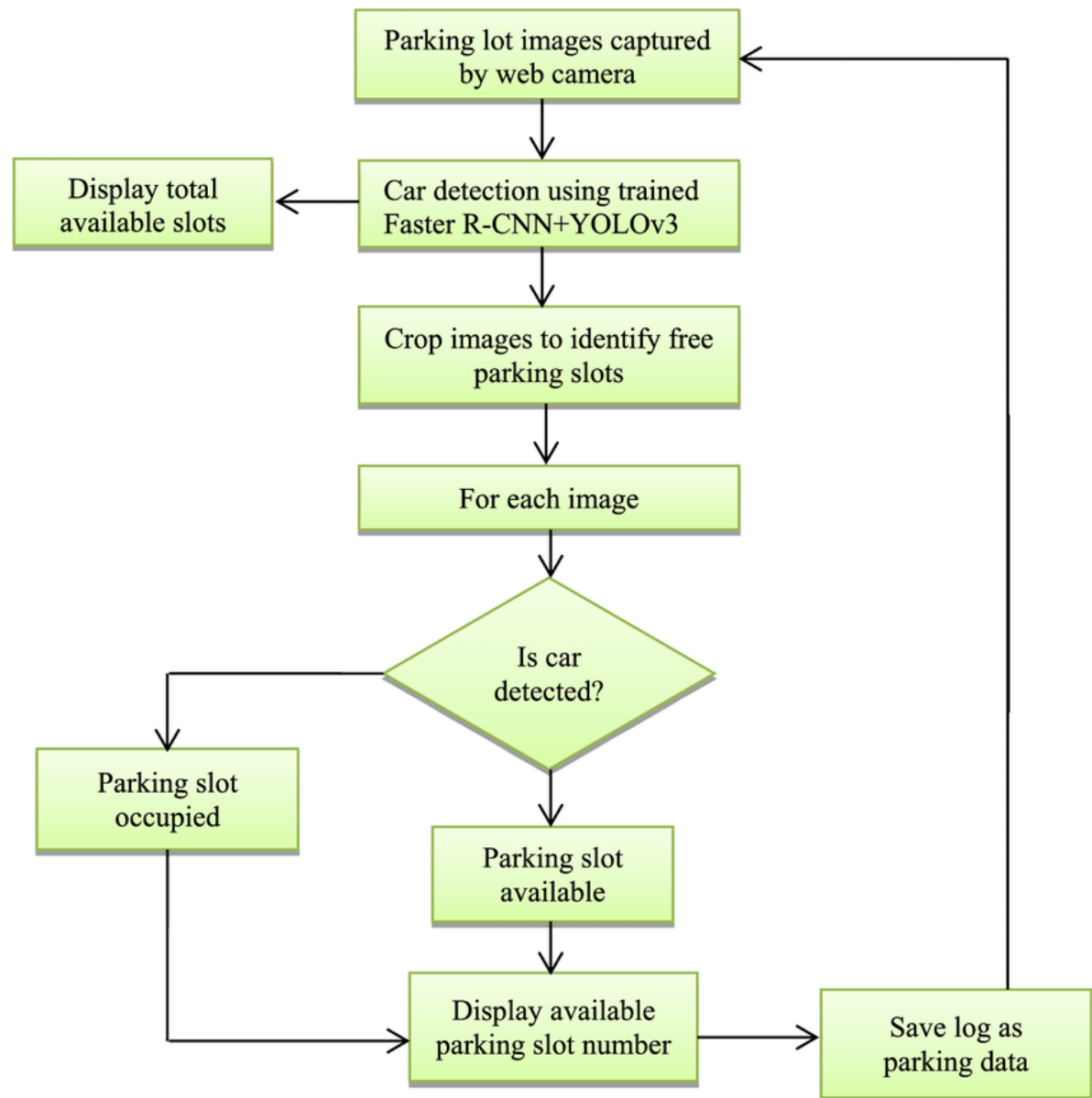
Compatibility: The AI-enabled car parking system should be compatible with different hardware configurations, including cameras, sensors, and network infrastructure. It should also be compatible with various operating systems, ensuring widespread adoption and ease of integration with existing parking infrastructure.

Compliance: The system should comply with relevant regulations, privacy laws, and industry standards pertaining to data handling, storage, and user privacy. It should adhere to ethical guidelines and ensure transparency in data usage and security practices.

Performance Monitoring and Analytics: The system should provide mechanisms for performance monitoring and analytics. It should generate reports and metrics to measure system performance, accuracy, and utilization. This information can be used for system optimization, troubleshooting, and continuous improvement.

4. PROJECT DESIGN

4.1 Data Flow Diagrams



4.2 Solution & Technical Architecture

An AI-enabled car parking system using OpenCV (a computer vision library) would use cameras and sensors to collect and analyze data on parking lot occupancy and traffic flow. It would guide drivers to available parking spots, provide parking assistance, handle payments, and enhance security and surveillance. The system would improve the parking experience for drivers and increase efficiency, safety, and security in parking lots.

Data Collection: The system would use cameras and sensors to collect data on parking lot occupancy, traffic flow, and other relevant information. The data would be processed by OpenCV to extract useful information such as the number of available parking spaces and the location of parked cars.

Data Processing: OpenCV would be used to process the collected data and analyze it to detect empty parking spots, calculate occupancy rates, and identify any issues such as double parking or cars parked in non-designated areas.

Real-time Parking Guidance: Using the processed data, the system would provide real-time guidance to drivers on available parking spots and the most optimal route to reach them. This would improve traffic flow and reduce the time it takes for drivers to find a parking spot.

Parking Assistance: The system would use OpenCV to provide parking assistance to drivers, such as detecting obstacles and guiding them into the parking spot. The system would also provide feedback to the driver on parking accuracy and any potential damage to the car or other objects.

Payment and Access Control: The system would use OpenCV to identify and authenticate the vehicle, and allow access to the parking lot only to authorized vehicles. The system would also handle payment processing and issue parking tickets.

Security and Surveillance: OpenCV would be used to monitor the parking lot and detect any suspicious activity, such as unauthorized access or vandalism. The system would alert security personnel in real-time to take appropriate actions.

Solution Architecture Diagram:

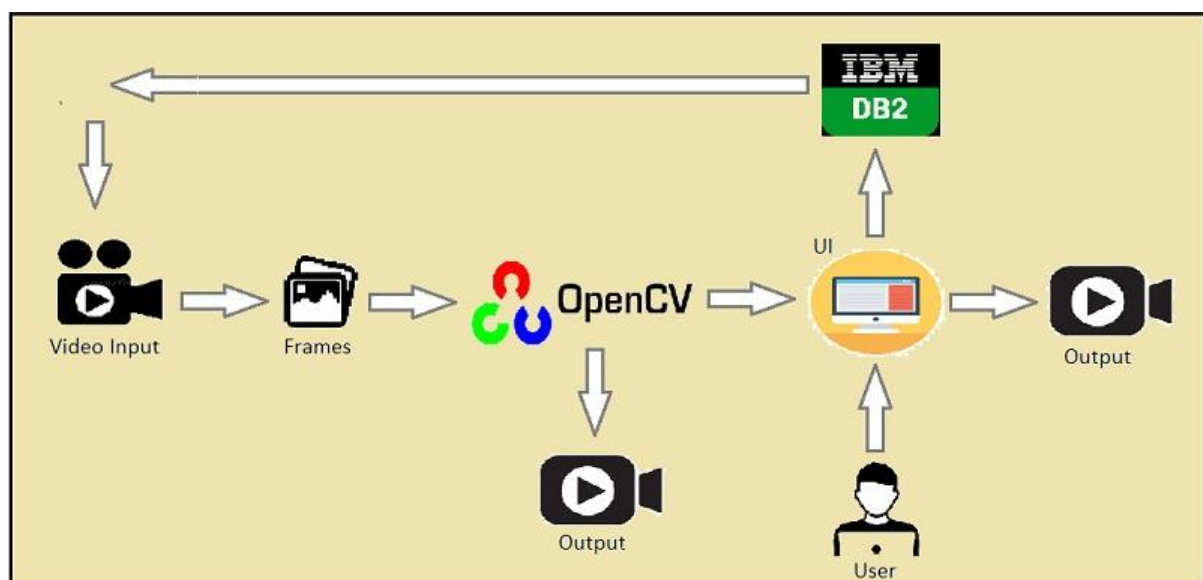


Figure 1: Architecture and data flow of the AI-Enabled Car Parking Using open CV

4.3 User Stories

As a driver, I want to quickly find an available parking spot in the parking lot, so I can save time and reduce the frustration of searching for parking.

As a parking lot administrator, I want to monitor the occupancy status of each parking space in real-time, so I can efficiently manage the parking lot and

allocate spaces effectively.

As a driver, I want to receive real-time guidance and directions to the nearest available parking spot, so I can easily navigate and park my vehicle without any hassle.

As a security personnel, I want to be alerted in real-time about any suspicious activities, such as unauthorized access or vehicle theft, so I can respond promptly and ensure the safety of the parking area.

As a system administrator, I want to analyze the parking occupancy data to identify peak hours, usage patterns, and trends, so I can make informed decisions regarding resource allocation and future parking lot expansions.

As a driver, I want to have a user-friendly mobile application that provides me with parking availability, reservation options, and cashless payment methods, so I can have a seamless and convenient parking experience.

As a parking lot owner, I want to integrate the AI-enabled car parking system with other smart city infrastructure, such as transportation systems or payment gateways, so I can provide a more integrated and connected parking solution for drivers.

As a driver, I want to have access to historical parking data, such as past transactions or parking durations, so I can review my parking history and track my expenses.

As a maintenance personnel, I want to easily identify and troubleshoot any technical issues with the AI-enabled car parking system, so I can ensure its smooth operation and minimize downtime.

As a city planner, I want to leverage the data generated by the AI-enabled

car parking system to make informed decisions about parking infrastructure development, traffic management, and urban planning initiatives.

5. CODING & SOLUTIONING

5.1 Feature 1

To develop this project, we used some tools there are OpenCV, and IBM db-2 for database connectivity and PyCharm. If you want to add additional frames in the parking lot then you have to perform a left click on your mouse. If you want to delete unnecessary frames in the parking lot you have to perform, right-click on the mouse.

OpenCV (Open-Source Computer Vision Library) is a popular open-source computer vision and image processing library. It provides a wide range of functionalities for image and video processing, including object detection, image recognition, and camera calibration. OpenCV is widely used in various applications, including robotics, augmented reality, surveillance, and more.

5.2 Feature 2

If the car is parked in the parking area, then CCTV will indicate it as a red rectangle, if the parking lot free means it will indicate to the user as a green color. By using the visual representation in the CCTV user can easy to detect available parking lots in the parking areas.

5.3 Database Schema

To develop this project, we used the IBM-db2 database to store the user's information in a structured manner. IBM offers a range of database products, including IBM Db2, which is a widely used relational database management system (RDBMS). Db2 provides a secure and scalable platform for storing and managing structured data. It offers features such as high availability, robust security mechanisms, and support for SQL queries and transactions.

6. RESULTS

6.1 Performance Metrics

Detection Accuracy: The accuracy of vehicle detection refers to how well the system can identify and locate vehicles in the parking area. This metric measures the percentage of correctly detected vehicles compared to the total number of vehicles present.

Classification Accuracy: If the system includes vehicle classification, this metric measures the accuracy of correctly classifying vehicles into their respective types (cars, motorcycles, trucks, etc.). It assesses the system's ability to differentiate between different vehicle categories accurately.

Occupancy Detection Accuracy: This metric evaluates how accurately the system determines the occupancy status of individual parking spaces. It measures the percentage of correctly classified occupied and vacant parking spaces.

Processing Speed: The processing speed measures the time taken by the system to perform vehicle detection, classification, and occupancy monitoring tasks. It can be evaluated in frames per second (FPS) or milliseconds (ms) to ensure real-time or near-real-time performance.

False Positive Rate: The false positive rate represents the percentage of falsely

detected or classified vehicles or parking space occupancies. It indicates the system's ability to minimize incorrect detections or classifications.

False Negative Rate: The false negative rate measures the percentage of missed detections or classifications. It assesses the system's ability to accurately identify all vehicles and their occupancy status.

System Availability: This metric measures the system's uptime and availability, ensuring that it operates reliably and consistently. It accounts for any potential system failures, maintenance, or downtime that may affect the system's performance.

User Satisfaction: User satisfaction can be assessed through surveys, feedback, or user experience testing. It measures the users' overall satisfaction with the AI-enabled car parking system, including its accuracy, responsiveness, ease of use, and convenience.

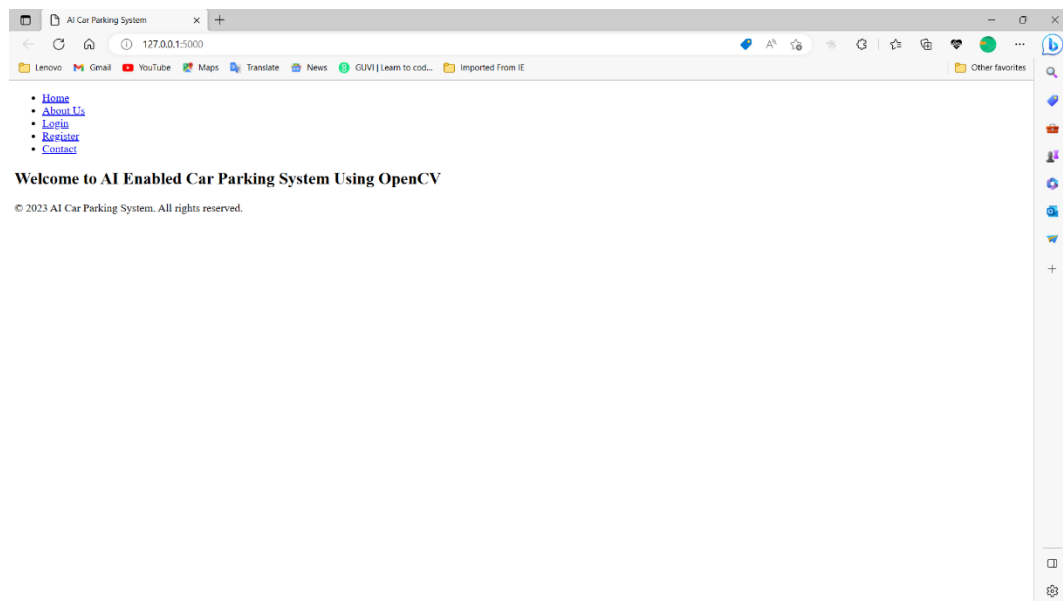
Security and Alert Response Time: If the system includes security features, the response time to security alerts can be measured. This metric evaluates how quickly the system detects and alerts security personnel about suspicious activities or security breaches.

Scalability: Scalability measures the system's ability to handle increasing

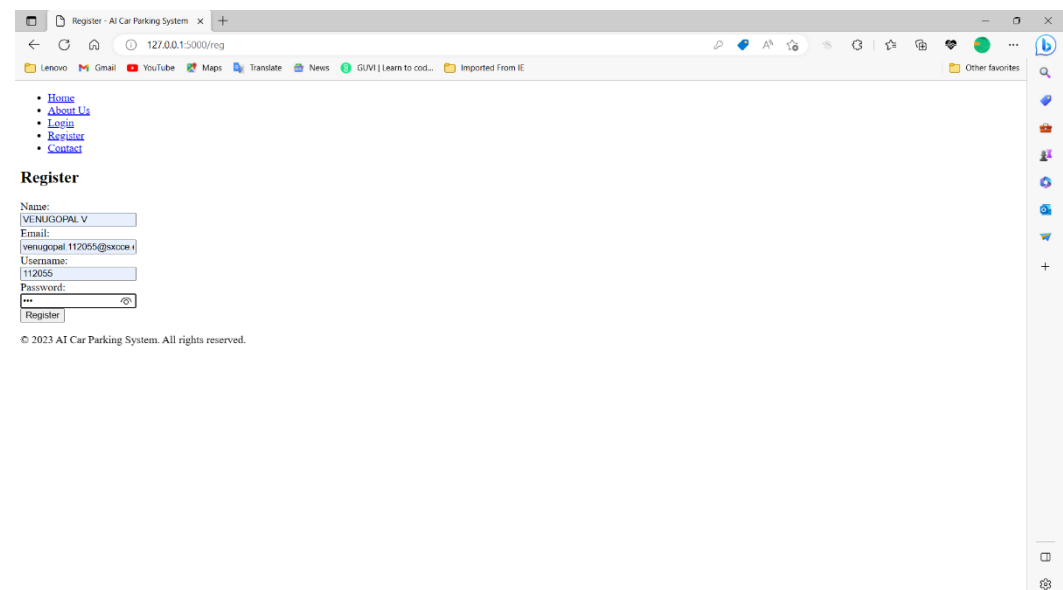
volumes of vehicles, parking spaces, and system users without compromising performance or accuracy. It assesses whether the system can accommodate the growth of the parking area and adapt to changing demands.

Project Screenshot:

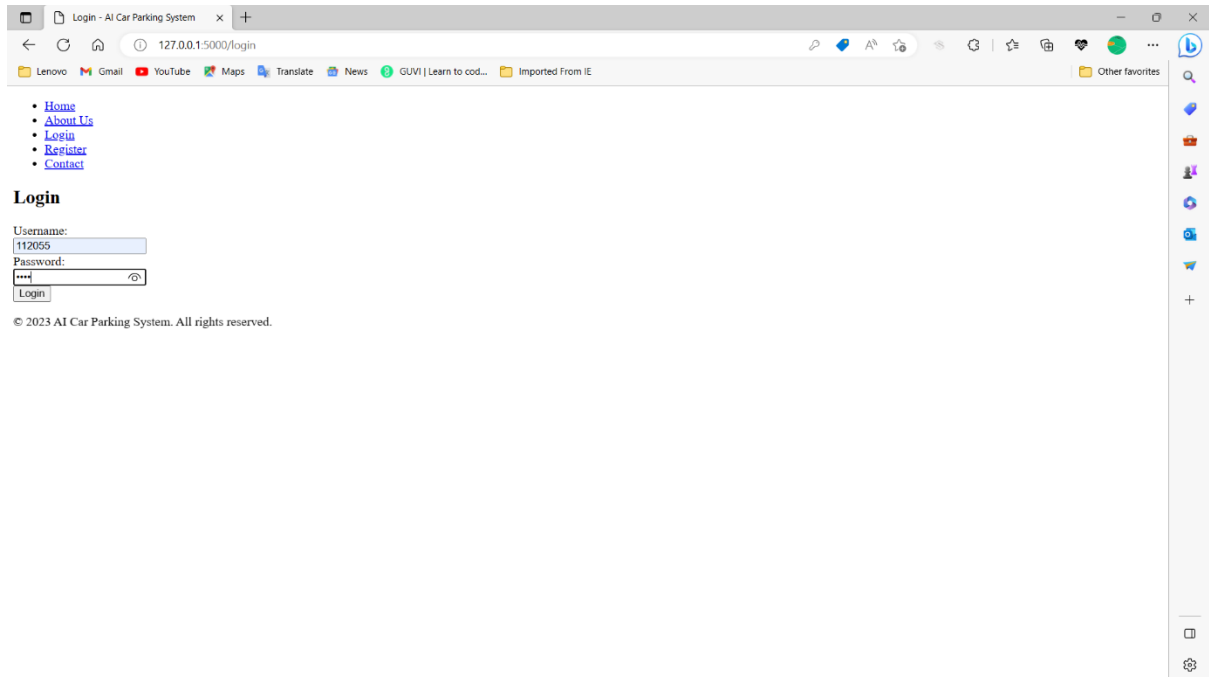
Home Page:



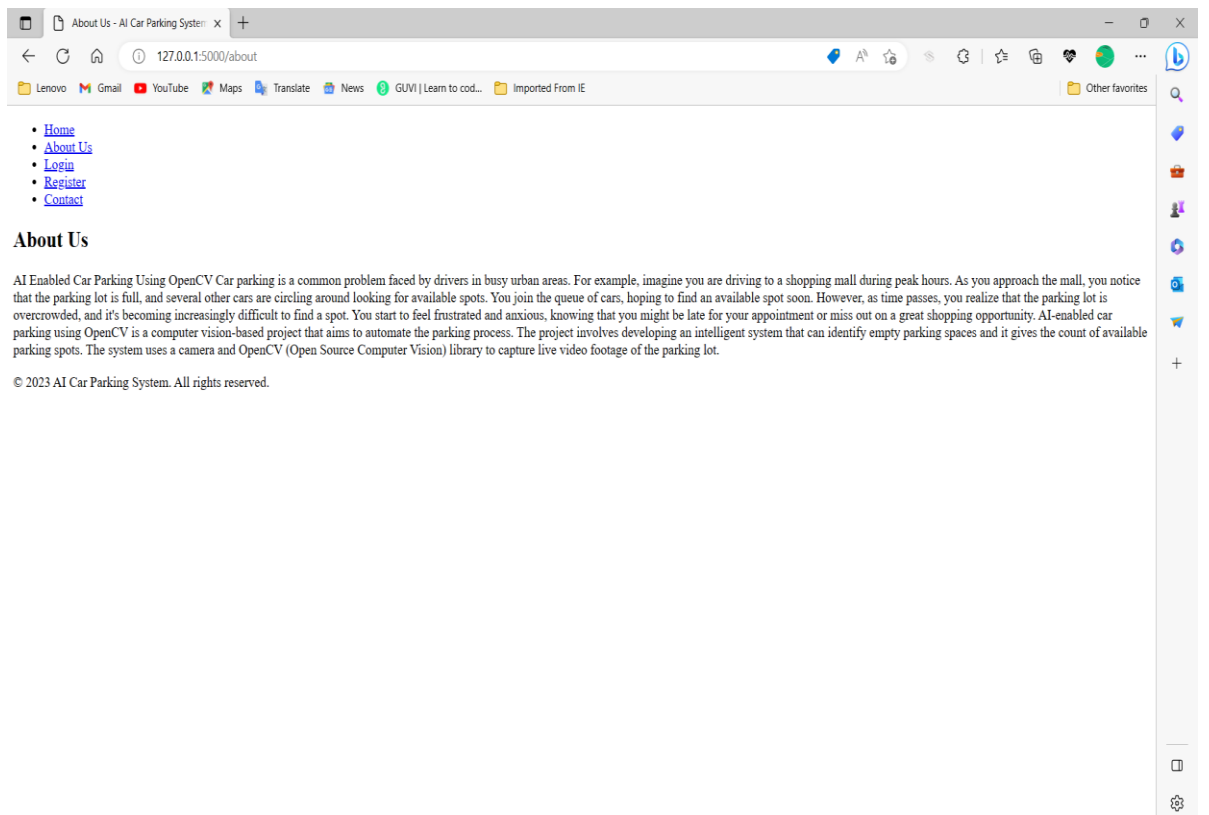
Registration Page:



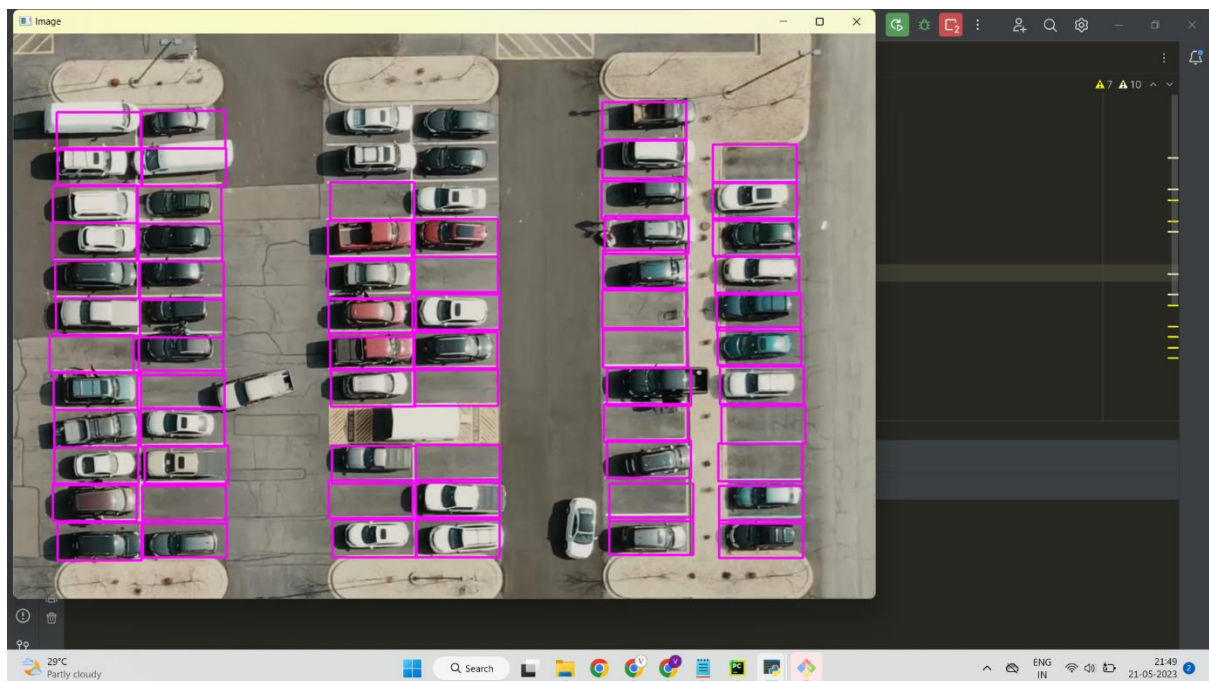
Login Page:



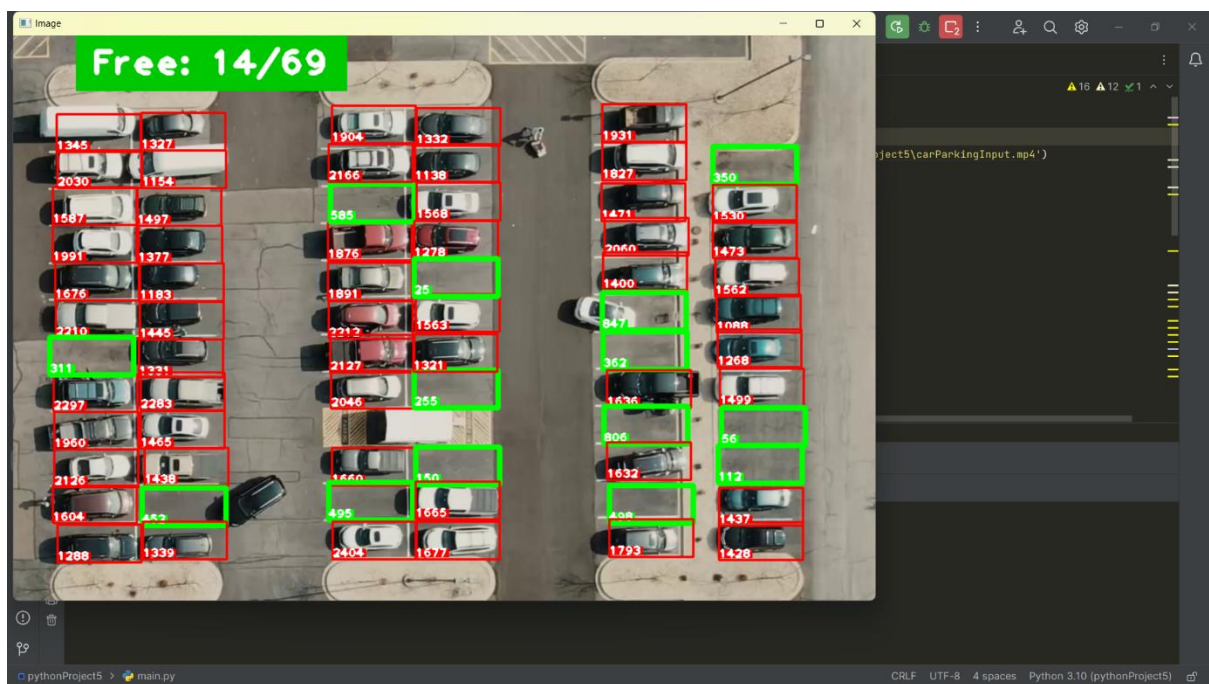
About Page



Object Detection:



Parking Lot Detection:



7. ADVANTAGES & DISADVANTAGES

Advantages

- Enhanced Efficiency
- Real-time Monitoring
- Improved User Experience
- Accurate Vehicle Detection
- Cost Optimization
- Enhanced Security
- Scalability and Flexibility
- Integration with Smart City Infrastructure
- Data-driven Insights
- Sustainability

Disadvantages

While AI-enabled car parking systems using OpenCV offer numerous advantages, it's important to consider potential disadvantages and challenges associated with their implementation.

- Initial Cost
- Technical Complexity
- Reliance on Infrastructure
- Environmental Limitations
- Privacy Concerns
- Limited Compatibility
- Maintenance and Upkeep
- Potential False Alarms
- Limited Adaptability

- Adoption Challenges

8. CONCLUSION

AI-enabled car parking systems using OpenCV offer a promising solution to optimize parking efficiency, enhance user experience, and streamline parking management. By leveraging computer vision algorithms, these systems provide accurate vehicle detection, real-time monitoring, and intelligent guidance to available parking spaces. The advantages of increased efficiency, improved security, scalability, and integration with smart city infrastructure make them a valuable asset for parking lot operators and city planners.

However, it's important to consider the potential challenges and disadvantages associated with the implementation of AI-enabled car parking systems. These include initial costs, technical complexity, reliance on infrastructure, privacy concerns, and compatibility issues. Addressing these challenges through careful planning, proper maintenance, privacy protocols, and user education is crucial to ensure the successful deployment and acceptance of these systems.

Overall, AI-enabled car parking using OpenCV holds significant potential to revolutionize the parking industry, reducing congestion, optimizing resource allocation, and enhancing the overall parking experience for drivers. With continuous advancements in computer vision and AI technologies, coupled with thoughtful design and implementation, these systems can contribute to building smarter, more efficient, and sustainable cities.

9. FUTURE SCOPE

The future scope for AI-enabled car parking systems using OpenCV is vast, and there are several potential advancements and developments that can be explored.

Advanced Vehicle Recognition: Enhancing the capabilities of vehicle recognition algorithms can enable more accurate detection and classification of vehicles, including differentiating between specific vehicle models or identifying license plate numbers. This can aid in implementing advanced parking management features and enhancing security.

Integration with Autonomous Vehicles: As autonomous vehicles become more prevalent, integrating AI-enabled car parking systems with autonomous vehicle technology can enable seamless parking interactions. This includes automated parking space reservation, navigation to available spots, and communication between the parking system and autonomous vehicles.

Predictive Analytics and Demand Forecasting: By analyzing historical parking data and incorporating external factors such as events, weather conditions, and traffic patterns, AI-enabled car parking systems can predict parking demand and occupancy trends. This can assist in optimizing parking resource allocation and planning for future parking needs.

Smart Payment and Integration with Mobile Apps: Integrating AI-enabled car parking systems with mobile applications can facilitate convenient and seamless payment processes. Additionally, integrating with digital wallets, cashless payment systems, and parking loyalty programs can enhance the overall user experience.

Smart Parking Guidance Systems: AI-enabled car parking systems can

incorporate dynamic parking guidance systems that guide drivers to available parking spaces in real time. This can involve displaying parking space availability on digital signage, mobile apps, or in-vehicle navigation systems.

Energy Efficiency and Sustainability: Future advancements can focus on optimizing the energy consumption of AI-enabled car parking systems, including intelligent power management, renewable energy integration, and energy-efficient hardware components. This can contribute to sustainable and environmentally friendly parking infrastructure.

Multi-Level Parking Management: AI-enabled systems can be further developed to efficiently manage multi-level or underground parking structures. This includes accurate detection and guidance of parking spaces across different levels and seamless navigation within complex parking environments.

Integration with Smart City Infrastructure: The integration of AI-enabled car parking systems with other smart cities initiatives, such as traffic management systems, public transportation, and urban planning, can create a more cohesive and connected urban environment. This enables better traffic flow, reduced congestion, and improved overall urban mobility.

Edge Computing and Real-time Analytics: Leveraging edge computing capabilities, AI-enabled car parking systems can perform real-time analytics and decision-making at the edge devices. This reduces latency, improves system responsiveness, and enables faster and more efficient parking operations.

Continuous Improvement and Optimization: Ongoing research and development efforts can focus on refining AI algorithms, improving accuracy, and reducing false positives and false negatives. Additionally, regular updates and enhancements can be made to the system's software, ensuring it remains

up-to-date with the latest advancements in computer vision and AI technologies.

10. APPENDIX

Source Code

app.py

```
from flask import Flask, render_template, request, session
import cv2
import pickle
import cvzone
import numpy as np
import ibm_db
import re

app=Flask(__name__)
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=6667d8e9-9d4d-4ccb-ba32-
21da3bb5aafe.clogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30376;SECU
RITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=fsv44141;PWD=Ukz
MgDP1adodPaF8;", "", "")
print("connected")
@app.route("/")
def index():
    return render_template('index.html')
@app.route('/register')
def home():
    return render_template('register.html')
@app.route('/login')
def login():
    return render_template('login.html')
@app.route("/about")
def about():
    return render_template('about.html')
@app.route("/contact")
def contact():
    return render_template('contact.html')

@app.route("/reg", methods=['POST', 'GET'])

def signup():
    msg = ''
    if request.method=='POST':
        NAME=request.form["NAME"]
        EMAIL = request.form["EMAIL"]
        PASSWORD = request.form["PASSWORD"]
        sql="SELECT * FROM REGISTER WHERE NAME=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, NAME)
        ibm_db.execute (stmt,2,EMAIL)
        ibm_db.fetch_assoc(stmt,3,PASSWORD)
        account=ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            return render_template('login.html',error=True)
        elif not re.match(r'[^@]+@[^@]+\.[^@]+',EMAIL):
            msg="Invalid Email Address!"
        else:
            insert_sql="INSERT INTO REGISTER VALUES(?,?,?)"
            prep_stmt=ibm_db.prepare(conn,insert_sql)
            ibm_db.bind_param(prepare_stmt,1,NAME)
            ibm_db.bind_param(prepare_stmt, 2,EMAIL)
```

```

        ibm_db.bind_param(prepare_stmt, 3, PASSWORD)
        ibm_db.execute(prepare_stmt)
        msg="You have successfully registered"
        return render_template('register.html',msg=msg)
@app.route("/log",methods=['POST','GET'])

def login1():
    if request.method=='POST':
        email = request.form["EMAIL"]
        password = request.form["PASSWORD"]
        sql="SELECT * FROM REGISTER WHERE EMAIL=? AND PASSWORD=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, email)
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        account=ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['Loggedin']=True
            session['id'] = account['EMAIL']
            session['email'] = account['EMAIL']
            return render_template('model.html')

        else:
            msg = "Invalid Email Address!"
            return render_template('login.html',msg=msg)
    else:
        return render_template('login.html')
@app.route('/live')
def liv_pred():
    cap = cv2.VideoCapture('C:\\Users\\venug\\Music\\Python\\Flask
Tutorial\\static\\carParkingInput.mp4')
    with open('CarParkPos', 'rb') as f:
        posList = pickle.load(f)
        width, height = 107, 48

    def checkParkingSpace(imgPro):
        spaceCounter = 0
        for pos in posList:
            x, y = pos
            imgCrop = imgPro[y:y + height, x:x + width]
            count = cv2.countNonZero(imgCrop)
            if count < 900:
                color = (0, 255, 0)
                thickness = 5
                spaceCounter += 1
            else:
                color = (0, 0, 255)
                thickness = 2
            cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height),
color, thickness)
            cvzone.putTextRect(img, str(count), (x, y + height - 3),
scale=1,
                                thickness=2, offset=0, colorR=color)

            cvzone.putTextRect(img, f'Free: {spaceCounter}/{len(posList)}',
(100, 50), scale=3,
                                thickness=5, offset=20, colorR=(0, 200, 0))

        while True:
            if cap.get(cv2.CAP_PROP_POS_FRAMES) ==

```

```

cap.get(cv2.CAP_PROP_FRAME_COUNT):
    cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
    success, img = cap.read()
    imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    imgBlur = cv2.GaussianBlur(imgGray, (3, 3), 1)
    imgThreshold = cv2.adaptiveThreshold(imgBlur, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 25,
                                16)

    imgMedian = cv2.medianBlur(imgThreshold, 5)
    kernel = np.ones((3, 3), np.uint8)
    imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)
    checkParkingSpace(imgDilate)
    cv2.imshow("Image", img)
    cv2.waitKey(10)
if __name__ == '__main__':
    app.run(debug=True)

```

main.py

```

import cv2
import pickle
width, height = 107, 48
try:
    with open('CarParkPos', 'rb') as f:
        posList = pickle.load(f)
except:
    posList = []
def mouseClicked(events, x, y, flags, params):
    if events == cv2.EVENT_LBUTTONDOWN:
        posList.append((x, y))
    if events == cv2.EVENT_RBUTTONDOWN:
        for i, pos in enumerate(posList):
            x1, y1 = pos
            if x1 < x < x1 + width and y1 < y < y1 + height:
                posList.pop(i)
        with open('CarParkPos', 'wb') as f:
            pickle.dump(posList, f)
while True:
    img = cv2.imread('carParkImg.png')
    for pos in posList:
        cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), (255, 0,
255), 2)
    cv2.imshow("Image", img)
    cv2.setMouseCallback("Image", mouseClicked)
    cv2.waitKey(1)

```

GitHub & Project Video Demo Link

GitHub:

<https://github.com/naanmudhalvan-SI/PBL-NT-GP--5378-1680779927.git>

Project Video Demo Link:

https://drive.google.com/file/d/1rb4Nn1TffcQG6fZ0KctKa13s6_6VNk68/view?usp=sharing