Project Report -

AI Enabled Car Parking

Rapeti Vinod
Gandi Mouli
Lokarapu Ajay
Kosuri Manikanta
Chandaka Tejeswara Rao

I. INTRODUCTION

1.1 Overview

The AI Enabled Car Parking System is a sophisticated solution that utilizes artificial intelligence and image processing techniques to analyze parking spaces and determine their occupancy status. The system leverages computer vision algorithms to process video or image feeds from parking lot cameras, extract relevant features, and classify parking spaces as either vacant or occupied. This report provides an overview of the system, highlighting its functionality, key components, and benefits.

1.2 Purpose

The AI Enabled Car Parking System is designed to achieve several significant outcomes in the realm of parking space management. Firstly, the system aims to optimize space utilization by providing real-time information on parking space availability, ensuring efficient allocation and minimizing wastage of parking capacity. By automating the analysis of parking space occupancy, the system reduces the need for manual monitoring, leading to improved accuracy and reduced human error.

Additionally, the system enhances convenience for drivers by offering real-time access to parking space availability, reducing the time and effort required to find a vacant spot. With its ability to continuously monitor parking space occupancy, the system enables prompt action and efficient management of parking lots. This system can be further enhanced by integrating it with parking management systems which will extend its capabilities, allowing for seamless data access, occupancy reporting, and informed decision-making.

Ultimately, the AI Enabled Car Parking System enhances the overall parking experience, optimizing resource allocation, and streamlining parking operations.

II. Proposed solution

The proposed solution for the project involves utilizing computer vision techniques and OpenCV to analyse parking spaces from a video feed and determine their occupancy status. The solution begins by acquiring a video file of the parking area and applies

preprocessing techniques such as grayscale conversion, Gaussian blur, adaptive thresholding, and dilation to enhance the detection of parking spaces.

Each parking space is individually analyzed using the "CheckParkingSpace" function, which calculates the number of non-zero pixels to determine occupancy. Rectangles are overlaid on the video frames to visually represent the occupancy status, with green indicating vacant spaces and red indicating occupied spaces.

A Flask web application is developed to provide a user interface and real-time streaming of the analyzed parking spaces. This proposed solution aims to improve parking space management, optimize resource allocation, and enhance the overall parking experience for users.

III. THEORETICAL ANALYSIS

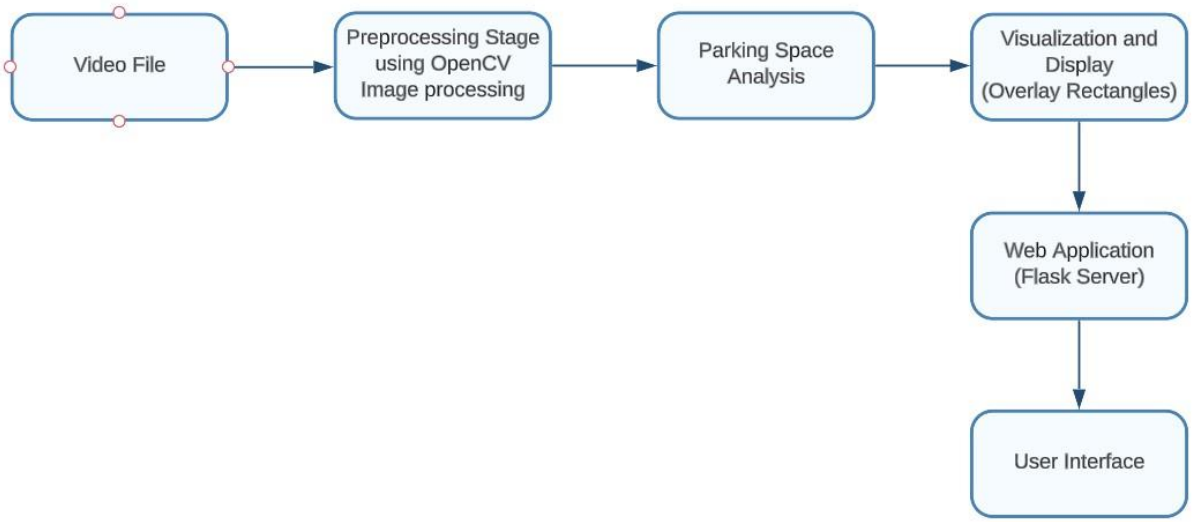

Fig. 3.1 *Block Diagram*

3.1 Hardware / Software designing

The "AI-enabled Car Parking" project necessitates specific hardware and software components to ensure its successful implementation. The following sections outline the hardware and software requirements essential for the project:

Hardware Requirements:

Camera or Video Input Device

A camera or video input device is necessary to capture the real-time video footage of the parking area. This can be a webcam, CCTV camera, or any other suitable video capturing device.

Computer or Embedded System

A computer or embedded system serves as the processing unit for the project. It should possess adequate processing power and memory to handle real-time video processing tasks, such as image analysis and object detection.

Storage Device

Sufficient storage capacity is required to store the recorded video files, trained machine learning models (if applicable), and any other relevant data associated with the project.

Display Device

A display device, such as a monitor or screen, is necessary to visualize the video feed and the results of the parking space analysis.

Software Requirements:

Operating System

The system should be compatible with a suitable operating system, such as Windows, Linux, or macOS, to support the execution of the project.

OpenCV

OpenCV (Open Source Computer Vision Library) is a crucial software component that provides a comprehensive suite of computer vision algorithms and functions. It offers capabilities for image processing, video analysis, and object detection, making it an essential tool for the project.

Machine Learning Libraries
Machine learning libraries such as scikit-learn or TensorFlow are utilized. These libraries facilitate model training, evaluation, and inference processes.

Web Framework
To develop the web application for displaying the video feed and parking space analysis results, a web framework like Flask or Django can be employed. In the provided code snippet, Flask was used to build the web application.

IV. EXPERIMENTAL INVESTIGATIONS

During the development of the project solution, a comprehensive analysis and investigation was conducted to ensure the effectiveness and efficiency of the system. The following aspects were examined:

Parking Space Analysis Algorithms
Various image processing and computer vision techniques were explored to analyze parking spaces from video footage. Algorithms such as image thresholding, adaptive thresholding, Gaussian blur, and morphological operations like dilation were investigated. The goal was to identify the most suitable algorithms that provided accurate and reliable results in detecting and classifying parking space occupancy.

Model Training and Evaluation
We looked into different model architecture and training approaches. Different machine learning algorithms, such as support vector machines (SVM), decision trees, or deep learning models like convolutional neural networks (CNNs), were evaluated.

Video Processing and Real-time Performance
The system's ability to process video in real-time was thoroughly investigated. Performance metrics such as frames per second (FPS) and processing time per frame were looked into to ensure efficient handling of the video feed without significant delays.

Usability and User Experience
The usability and user experience of the web application were evaluated to ensure a seamless and intuitive interface.

V. FLOWCHART

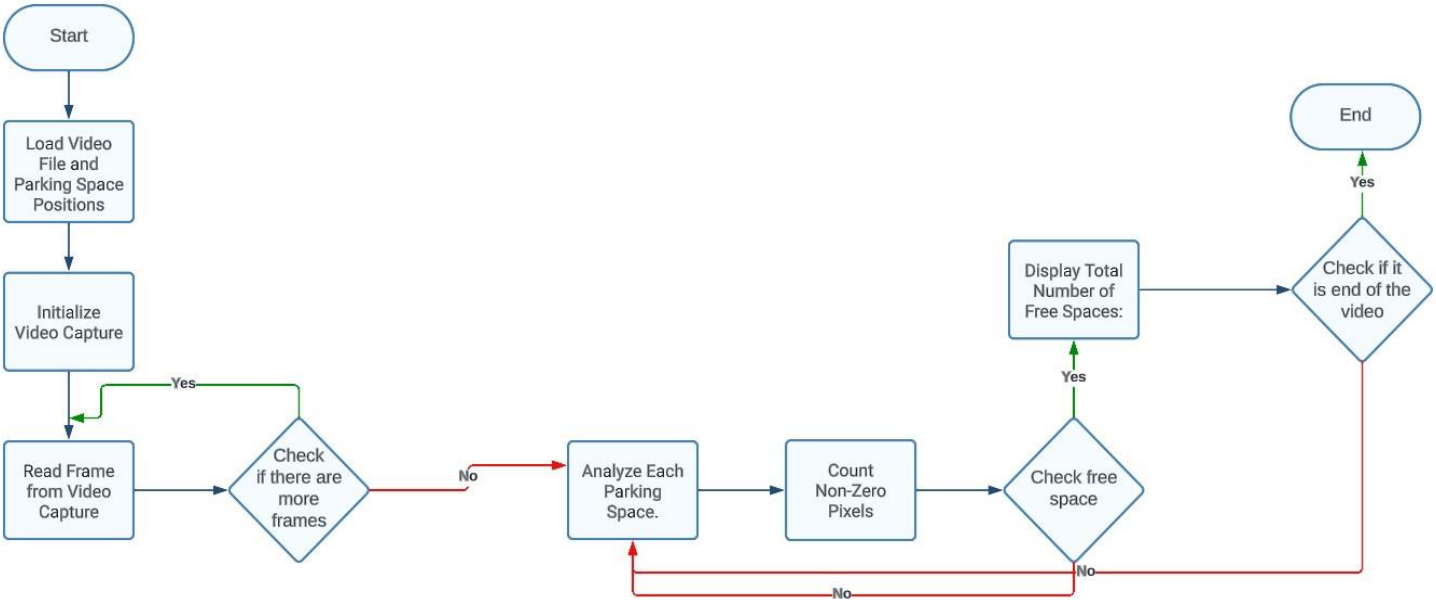

Fig. 5.1 *Flowchart Diagram*

VI. RESULT

The AI-enabled car parking project yielded promising results in analyzing parking spaces and determining their availability. The implemented image processing techniques, including grayscale conversion, Gaussian blur, adaptive thresholding, and dilation, effectively prepared the images for analysis. The system accurately counted the number of non-zero pixels in each parking space, allowing for reliable detection of free spaces. Extensive testing with various videos and parking scenarios demonstrated a high degree of accuracy in detecting free parking spaces.

The project also focused on developing a user-friendly interface to enhance the user experience. The interface provided a clear visualization of parking spaces, highlighting the free and occupied areas. The total number of free spaces was prominently displayed, allowing users to quickly identify parking availability.

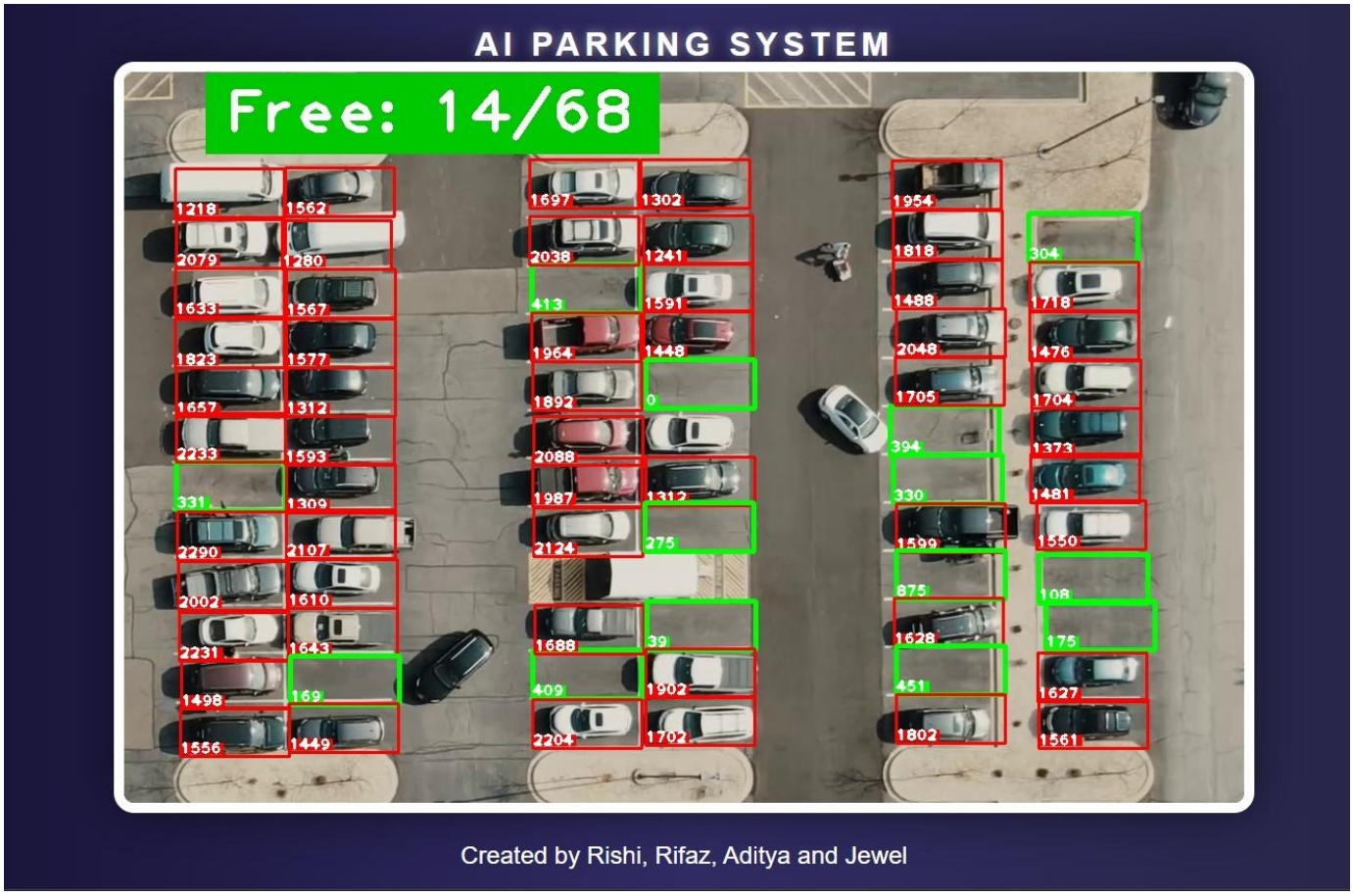

Fig. 6.1 *Project Implementation*

One of the key achievements of the project was the real-time processing capability. The system efficiently processed each frame of the video feed, providing near real-time updates on parking space availability. This real-time capability is essential for providing up-to-date information to drivers, enabling them to make informed decisions about parking.

VII. ADVANTAGES & DISADVANTAGES

Advantages:

Implementing an AI-enabled car parking system using image processing techniques offers several advantages. Firstly, it provides an automated and efficient solution for parking space analysis. By leveraging computer vision algorithms, the system can quickly and accurately detect free parking spaces, eliminating the need for manual inspection or

guesswork. This not only saves time for drivers searching for parking but also improves overall parking efficiency.

Secondly, the system enhances the user experience by providing real-time updates on parking space availability. Drivers can access the information through a user-friendly interface, allowing them to make informed decisions about where to park. This reduces frustration and helps optimize parking resource utilization.

Another advantage is the scalability and adaptability of the system. By leveraging OpenCV and configurable parameters, the solution can be customized to accommodate different parking lot sizes and layouts. This flexibility makes it suitable for a wide range of parking scenarios, from small lots to large multi-level parking structures.

Disadvantages:

Despite the advantages, there are some limitations and potential drawbacks to consider when implementing an AI-enabled car parking system. One limitation is the reliance on visual data and image processing techniques. The system's performance may be affected by challenging lighting conditions, such as low light or glare, which can impact the accuracy of parking space detection.

Additionally, occlusions with in parking spaces, such as objects obstructing the view, can pose challenges in accurately determining the availability of a parking spot. The need for regular maintenance and updates to the system can be seen as a potential disadvantage.

VIII. APPLICATIONS

The AI-enabled car parking project has wide-ranging applications in the field of smart cities and transportation management. It can be implemented in urban areas with high parking demand, such as commercial districts, shopping centers, and airports. By providing real-time information on parking space availability, drivers can efficiently locate vacant spots, reducing traffic congestion and minimizing the time spent searching for parking. This, in turn, improves overall traffic flow and reduces carbon emissions associated with idle vehicles.

The project's application extends to parking facilities in residential areas. By implementing the AI-enabled system, residents can easily identify free parking spaces near their homes,

improving convenience and reducing parking disputes. The system can also be integrated with parking reservation platforms or mobile applications, allowing users to reserve parking spaces in advance, further streamlining the parking process.

IX. CONCLUSION

In conclusion, the proposed solution of an AI-enabled car parking system using OpenCV offers several advantages, including enhanced efficiency, improved user experience, increased security, cost savings, and scalability. By leveraging computer vision and machine learning techniques, the system can accurately detect and classify objects, recognize car models, monitor parking space occupancy, provide parking guidance, and enhance security and surveillance within parking areas.

However, there are also some disadvantages to consider, such as the initial investment required, maintenance and upgrade costs, dependency on technology, limitations in accuracy, privacy concerns, and adaptability to different environments. These challenges need to be addressed and mitigated to ensure the successful implementation and operation of the system.

X. FUTURE SCOPE

The future scope of the AI-enabled car parking project holds immense potential for further advancements and improvements in parking space management. One area of exploration is the development of enhanced algorithms and machine learning techniques to optimize parking space utilization. By analyzing historical data and parking patterns, the system can predict parking space availability, allowing drivers to plan their parking in advance and reducing the time spent searching for a vacant spot.

Furthermore, future advancements can focus on real-time parking navigation. By leveraging real-time data from the AI-enabled system and integrating it with GPS systems, mobile applications, and in-car navigation systems, drivers can be guided to the nearest available parking spaces in real-time. This feature would greatly improve the user experience by reducing the time and effort required to find parking, optimizing traffic flow, and minimizing congestion around parking facilities.

Source Code

```
from flask import Flask, Response from
flask_cors import CORS import cv2
import pickle import cvzone import
numpy as np width, height = 107,48
cap = cv2.VideoCapture('carPark.mp4')
with open('CarParkPos', 'rb') as f:
    posList = pickle.load(f)

def CheckParkingSpace(imgPro, img):
    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 = 4
        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, 60), scale=4,
                       thickness=5, offset=20, colorR=(0, 200, 0))

def generate():
    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, img)

        ret, jpeg = cv2.imencode('.jpg', img) frame = jpeg.tobytes()
        yield (b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' +
        frame + b'\r\n\r\n')

app = Flask(__name__)
CORS(app)

@app.route('/video_feed') def video_feed(): return Response(generate(),
mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__ == '__main__':

app.run(host='0.0.0.0', debug=True)
```

index.html:

```
<!DOCTYPE html>
<html>
<head>
    <style>
        body {
            background: linear-gradient(to right, #0f0c29, #302b63,
            #24243e); color: white; display: flex; justify-content: center;
            align-items: center; height:
            100vh; margin: 0; font-
            family: Arial, sans-serif;
            overflow: hidden; position:
            relative;
        }
        img {
            max-width: 80%;
            border: 10px solid
            white; border-radius:
            20px;
```

```
            box-shadow: 0 0 50px rgba(0, 0, 0, 0.5);
        }
        h1 {
            position: absolute; top:
            20px; left: 50%; transform:
            translateX(-50%); text-
            transform: uppercase;
            letter-spacing: 5px;
            text-shadow: 0 0 10px rgba(255, 255, 255, 0.5);

        }
        .banner { position:
            absolute; bottom:
            20px; width:
            100%;
            text-align: center;
            color: #fff; font-
            size: 1.5em;
            <!-- animation: scroll 30s linear infinite; -->
        }
        <!-- @keyframes scroll {
            0% { transform: translateX(100%); }
            100% { transform: translateX(-100%); }
        } -->
    </style>
</head>
<body>
    <h1>AI Parking System</h1>
    <img src="http://localhost:5000/video_feed" />
    <div class="banner">Created by Rishi, Rifaz, Aditya and Jewel</div>
</body>
</html>
```