Wroclaw University Of Science And Technology


BATIN TOPBAŞOĞLU 276718

CANER OLCAY 276715


# Artificial Intelligence and Computer Vision Project

Final Presentation

# TABLE OF CONTENT

# 1. Introduction:

This program is designed to implement a real-time parking detection system that leverages a YOLOv8 model for object detection and OpenCV for processing video streams. It aims to optimize parking space usage by identifying free and occupied spots in a given parking area. This system is especially relevant in urban environments where parking availability is limited, and efficient management is critical.

# 2. Key Features:

- ## 2.1YOLOv8 for Object Detection:

- The system uses the YOLOv8 model, known for its speed and accuracy in detecting objects in real-time. It detects vehicles (specifically cars) in the video stream.

- ## 2.2 Defined Parking Areas:

- The code defines specific parking areas with coordinates. Each area is represented as a polygon, making it possible to determine whether a detected vehicle is located within a parking spot.

- ## 2.3 Video Processing with OpenCV:

- OpenCV handles the video input, frame resizing, and drawing operations. It overlays information such as free/occupied spots, occupancy rate, and a warning message if the parking lot is nearing full capacity.

- ## 2.4 Dynamic Spot Detection:

- By combining YOLO's car detection with polygon overlap testing, the system dynamically updates the status of each parking spot as vehicles enter or leave.

- ## 2.5 Real-Time Metrics Display:

- The system visually highlights free and occupied spots using color-coded outlines (green for free, red for occupied). It also displays the total number of spots, free spots, occupied spots, and an occupancy warning when utilization exceeds a threshold (like 50%).

- ## 2.6 User Interactivity:

- The video playback can be paused/resumed with a keypress ('p') and stopped entirely with the 'ESC' key.

- ## 2.6 Performance Monitoring:

- Frames-per-second (FPS) calculation ensures real-time performance is monitored, making the system viable for live deployment.

- ## 2.7 Practical Applications:

This system can be deployed in parking lots of shopping malls, airports, or residential complexes to provide live updates on parking availability, reducing congestion and improving user convenience. Its scalable design allows customization for different parking layouts by adjusting the parking area coordinates.

# 3. YOLO Model Overview

YOLO (**You Only Look Once**) is a state-of-the-art object detection algorithm renowned for its exceptional speed, accuracy, and suitability for real-time applications. Unlike traditional region proposal-based methods that first identify potential object regions and then classify them, YOLO treats object detection as a single regression problem. By processing the entire image in one evaluation, YOLO predicts bounding boxes and class probabilities simultaneously, offering a unified and efficient architecture. The algorithm divides the input image into a grid, where each cell predicts objects, their locations, and associated confidence scores. Its ability to achieve real-time detection without compromising accuracy makes it ideal for applications such as traffic monitoring, parking lot surveillance, autonomous vehicles, and other scenarios where swift and reliable object detection is essential.

# 4. Integration of AI and CV

The integration of Artificial Intelligence (AI) models like YOLO with Computer Vision (CV) techniques creates a powerful synergy that significantly enhances the efficiency and accuracy of parking detection systems. YOLO excels in identifying vehicles in real-time with remarkable speed and precision, while CV processes these detections to analyze their spatial relationships with predefined parking areas. This hybrid approach combines the strengths of both technologies: AI provides robust and reliable object detection, capable of handling complex scenes, while CV enables detailed spatial analysis, visualization, and contextual understanding of the parking environment. Together, they ensure an intelligent, real-time solution for managing and monitoring parking spaces effectively.

# 5. Project Goals

The primary objective of this project is to develop a real-time parking detection system that enhances parking management and provides actionable insights. The specific goals include:

### 1. Real-Time Monitoring:

Continuously monitor parking lot usage in real time through video streams to ensure up-to-date information on parking availability.

### 2. Parking Spot Status:

Accurately identify and display the number of free and occupied parking spots, leveraging AI-based object detection and spatial analysis.

### 3. Occupancy Metrics:

Calculate and present the percentage occupancy of the parking lot, providing an overview of utilization rates.

### 4. Threshold-Based Warnings:

Generate warnings when parking lot usage exceeds predefined thresholds (e.g., 50%), allowing proactive management of overcrowding.

## 5. User Convenience:

Improve user experience in parking facilities, such as shopping malls, offices, and public areas, by reducing congestion and ensuring efficient utilization of parking spaces.

## 6. Visual Feedback:

Provide intuitive visual indicators, such as color-coded outlines for free and occupied spots, and real-time metrics displayed directly on the video feed.

## 7. Scalability:

Design the system to be easily adaptable for various parking lot layouts by modifying the parking area coordinates and parameters.

.

# 6. Code Functionality

The project uses Python with the YOLO model and OpenCV library to process and annotate video frames. Below are the major components of the code, explained step by step.

## 6.1. Loading the YOLO Model

The YOLO model (`yolov8s.pt`) is loaded using the `ultralytics` library. This pre-trained model is tailored for detecting various objects, including cars.

```
model = YOLO('yolov8s.pt')
```

## 6.2. Parking Area Definition

Parking spots are defined as polygons using coordinate points. These predefined areas serve as reference zones for checking occupancy.

```
parking_areas = {
    1: [(52, 364), (30, 417), (73, 412), (88, 369)],
...
    12: [(674, 311), (730, 360), (764, 355), (707, 308)],
}
```

## 6.3. Video Input Handling

The system reads video frames from a specified path. Each frame is resized for consistency and efficient processing.

```
if not cap.isOpened():
    print("Video file could not be opened. Please check
the path.")
    exit()

paused = False  # Pause state
fps_start_time = datetime.datetime.now()
frame_count = 0

while cap.isOpened():
    if not paused:
        ret, frame = cap.read()
        if not ret:
            break

        # Resize frame (optional)
        frame = cv2.resize(frame, (1020, 500))
```

## 6.4. Vehicle Detection

YOLO processes each frame to detect objects. Detected vehicles are checked against the parking area polygons to determine their status (free or occupied).

```
results = model.predict(frame)

            detections =
results[0].boxes.data.cpu().numpy()  # Convert tensor to
numpy
```

## 6.5. Parking Spot Evaluation

Each vehicle's center point is calculated and checked against the predefined parking areas. If the point lies within a polygon, the spot is marked as occupied.

```
for spot_id, area in parking_areas.items():
                        result =
cv2.pointPolygonTest(np.array(area, np.int32), (cx, cy),
False)
                        if result >= 0:  # Is the vehicle
in the parking area?
                            free_spots -= 1
                            occupied_spots.append(spot_id)
```

## 6.6. Visualization

Free spots are highlighted in green, while occupied spots are marked in red for better visual clarity.

```
if spot_id not in occupied_spots:

                cv2.polylines(frame, [np.array(area,
np.int32)], True, (0, 255, 0), 2)  # Green for free
```

## 6.7. Occupancy Percentage and Warning System

The system calculates the percentage of occupied spots and issues a warning if it exceeds a specified threshold (like, 50%).

```
if occupied_ratio >= 0.5:  # Warning if occupancy rate is
50% or higher
            warning_message = f"Warning: Parking is
{int(occupied_ratio * 100)}% full!"
            cv2.putText(frame, warning_message, (20, 200),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
            print(warning_message
```

# 7. Results

The parking detection system successfully achieves its intended objectives, offering reliable and actionable outcomes through the following features:

1. **Real-Time Monitoring:**
    - The system accurately identifies free and occupied parking spots in real time, ensuring users and administrators have up-to-date information about parking lot availability.
    - It utilizes the YOLO model's robust detection capabilities to ensure precise results, even in dynamic environments.

2. **Occupancy Insights:**
    - The system calculates and displays the percentage occupancy of the parking lot, offering clear metrics to evaluate usage patterns.
    - Warnings are triggered when occupancy exceeds predefined thresholds, providing proactive alerts to manage overcrowding effectively.

3. **Historical Data:**
    - The system generates logs of parking lot usage, specifically capturing morning and evening occupancy rates.
    - These logs enable further analysis to identify trends, such as peak usage hours, daily patterns, and seasonal variations, aiding in long-term planning and optimization.

These results demonstrate the effectiveness of combining AI and CV technologies to create an intelligent and scalable parking management solution that addresses both immediate and future operational needs.

# 8. Future Enhancements

To further enhance the capabilities and functionality of the parking detection system, the following advancements are proposed:

1. **Advanced Analytics:**
    - Introduce vehicle type classification to differentiate between cars, trucks, motorcycles, and other vehicle types, providing more detailed usage insights.
    - Integrate license plate recognition to enable advanced tracking and security features, such as identifying unauthorized vehicles or linking parking activity to specific users.

2. **User Notifications:**
    - Develop a mobile app or integrate the system with existing platforms to provide real-time updates on parking availability, enabling users to check for free spaces before arriving at the parking lot.
    - Incorporate push notifications to alert users when free spots become available or when a parking lot reaches capacity.

3. **Cloud Storage:**
    - Implement cloud-based long-term data storage to maintain historical records of parking lot usage.
    - Utilize this data for advanced analytics and reporting, such as identifying peak usage times, trends in parking behavior, and optimizing layout and capacity planning.

# 9. Conclusion

This parking detection system represents a practical and innovative solution for modern parking management challenges by seamlessly integrating Artificial Intelligence (AI) and Computer Vision (CV) techniques. The implementation of the YOLO model ensures precise and reliable real-time vehicle detection, even in dynamic and complex environments. Meanwhile, OpenCV facilitates robust visualization and spatial analysis, enabling the system to interpret and display parking lot usage effectively. By leveraging the unique strengths of both AI and CV, this hybrid approach demonstrates how cutting-edge technology can be applied to address real-world problems with efficiency and scalability.

Furthermore, the system's ability to monitor parking spaces in real time, calculate occupancy rates, and issue warnings when utilization exceeds predefined thresholds provides actionable insights for improved decision-making. Its user-friendly visual feedback, such as color-coded indicators for free and occupied spots, enhances usability and makes it highly adaptable for various applications, including shopping malls, office buildings, and public parking areas. This project exemplifies the transformative potential of combining AI-driven object detection with CV-powered spatial analytics, offering a scalable, efficient, and intelligent solution that can significantly reduce traffic congestion and optimize resource utilization in urban environments.