PS 13: Vehicle Movement Analysis and Insight Generation in a College Campus using Edge AI

Unique Idea Brief

Vehicle Matching:

1. Use YOLOv8 to detect license plate and EasyOCR to determine license plate number.
2. Various image pre-processing techniques like bilateral filtering, image sharpening etc. to get correct readings.
3. Store the result in a CSV file.
    a. If the license plate numbers exists in CSV, then update entry/ exit time.
    b. Else generate an alert and store the new entry with entry time.

Occupancy Monitoring:

1. In any frame of the parking-lot video , the area of interest's(AOI) i.e. parking areas' co-ordinates are specified with respect to any frame size (640 * 360).
2. Then with the help of YOLOv8 pre-trained model we will detect the objects in the frame.
3. If in any frame the detected object is a car and its bounding box's centre lie inside any of the specified parking area, we will mark the area as occupied and decrease the free space number.

Vehicle movement analysis:

1. While monitoring the cars occupancy , we are also storing the time stamp and number of cars at that moment in a csv file.
2. The number of cars is resampled for a specific time duration (20s / 30min/ 1hr) and its mean is taken.
3. The resampled data's mean and standard deviation is used to calculate a range within which parking lot is occupied for most of time.
4. The calculated mean and standard deviations are used to calculate the time when least and most cars are present.

Datasets

Number plate recognition

License Plate Recognition Object Detection Dataset and Pre-Trained Model by Roboflow Universe Projects

10126 open source license-plates images plus a pre-trained License Plate Recognition model and API. Created by Roboflow Universe Projects

https://universe.roboflow.com/roboflow-universe-projects/license-plate-recognition-rxg4e

Occupancy monitoring

used parking-lot video link:

https://mega.nz/file/4w0UCRyJ#TV55n4q-1j4jLCHtT_jmDZjMGdvny0-4Nlj8hAlmd7Y

Methodology

License Plate Validation and Processing

Example 1: 9-character Indian License Plate

OCR Result: "KA05AB1234"

- **Validation Process:**
  1. **State Code:** "KA" (Karnataka)
  2. **District Code:** "05" (Bengaluru)
  3. **Character After District Code:** "A" (Alphabet, valid)
  4. **Next 2 Alphabets:** "B" and "A" (valid)
  5. **Last 4 Digits:** "1234" (valid)
- **Processing Steps:**
  - Remove unwanted characters like "IND" at the beginning.
  - Convert characters like 'A' (which can represent a digit) to '4' if necessary.
  - Validate the format: "KA05AB1234" meets the 9-character format requirements.

Example 2: 10-character Indian License Plate

OCR Result: "MH02CD5678"

- **Validation Process:**
  1. **State Code:** "MH" (Maharashtra)
  2. **District Code:** "02" (Mumbai)
  3. **Next 2 Characters:** "C" and "D" (valid)
  4. **Character After District Code:** "5" (Digit, valid)
  5. **Next 2 Alphabets:** "6" and "7" (valid)
  6. **Last 4 Digits:** "5678" (valid)
- **Processing Steps:**
  - Remove unwanted characters like "IND" at the beginning.
  - Convert characters like 'A' (which can represent a digit) to '4' if necessary.
  - Validate the format: "MH02CD5678" fits the 10-character format as specified.

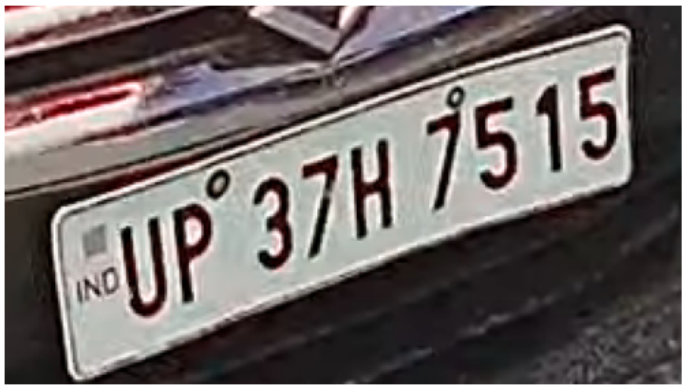
Additional Considerations:

- **Character Mapping:** Ensure correct mapping for characters like 'O' to '0', 'I' to '1', 'A' to '4' to handle OCR errors.
- **Cleanup:** Remove non-alphanumeric characters to clean up OCR artifacts.
- **Validation:** Verify cleaned license plate string adheres to 9-character or 10-character format for Indian plates.

Process Flow

Process Flow for License Plate Detection and OCR:

1. **Read Video Frame by Frame**
   - Use OpenCV to read video frames sequentially.
2. **Detect Cars using YOLOv8 Model**
   - Apply a pre-trained YOLOv8 model to detect cars in each frame.
3. **Crop and Detect License Plate**
   - For each detected car:
     - Use YOLOv8 to identify and crop out the license plate area.



(Figure - 01) Image of license plate

     - Deskew the license plate image to correct any skew.

(Figure - 02) Deskewed image of license plate

4. **Image Processing for OCR**
   - Convert the cropped license plate image to grayscale:

```
gray = cv.cvtColor(corrected_plate, cv.COLOR_BGR2GRAY)
```

(Figure - 03) The image is in BRG format

- Apply noise reduction using a bilateral filter:

```
bfilter = cv.bilateralFilter(gray, 11, 17, 17)
```

(Figure - 04) The image is in BRG format

- Perform edge detection using Canny:

```
edged = cv.Canny(bfilter, 30, 200)
```

(Figure - 05) The image is in BRG format

- Find contours and filter them:

```
keypoints = cv.findContours(edged.copy(), cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
contours = imutils.grab_contours(keypoints)
contours = sorted(contours, key=cv.contourArea, reverse=True)[:10]
```

- Identify and draw the contour of the license plate:

```
location = None
for contour in contours:
    approx = cv.approxPolyDP(contour, 10, True)
    if len(approx) == 4:
        location = approx
        break
mask = np.zeros(gray.shape, np.uint8)
new_image = cv.drawContours(mask, [location], 0, 255, -1)
```

(Figure - 06) The image is in BRG format

- Apply the mask to isolate the license plate:

```
new_image = cv.bitwise_and(corrected_plate, corrected_plate, mask=mask)
```

(Figure - 07) Noise has been reduced as compared to Figure - 02

- Sharpen the license plate image for better OCR:

```
kernel = np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]])
sharpened_image = cv.filter2D(cropped_image, -1, kernel)
```

(Figure - 08) Sharpened image

5. **Perform OCR using EasyOCR**
   - Convert the sharpened image to grayscale:

(Figure - 09) The image is in BRG format

   - Convert the grayscale image to black and white:

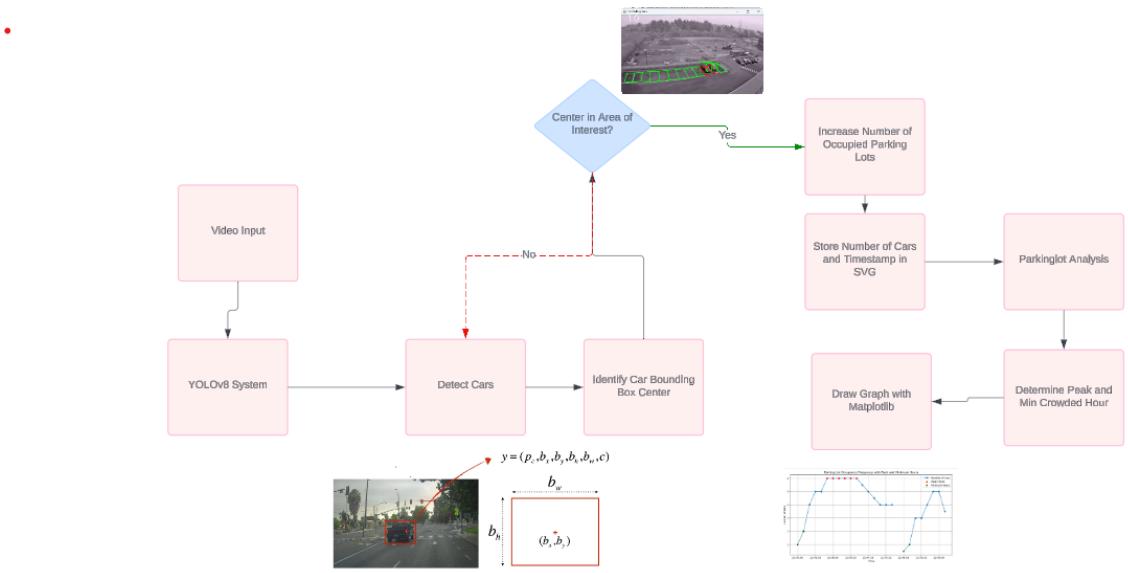
(Figure - 10) The image is in BRG format

   - Feed the processed image to EasyOCR for character recognition.
6. **Filter and Validate License Plate Reading**
   - Filter out non-alphanumeric characters and smaller strings not matching license plate dimensions.
   - Validate if the recognized plate follows the Indian license plate format.
   - Handle OCR errors by mapping characters like 'O' to '0', 'I' to '1', 'A' to '4', etc.
7. **Store Data**
   - If the license plate is new (not in the CSV file), log entry/exit time and trigger alert for unknown vehicles.
   - If the license plate is recognized, update its entry/exit time in the CSV file.

Sample output

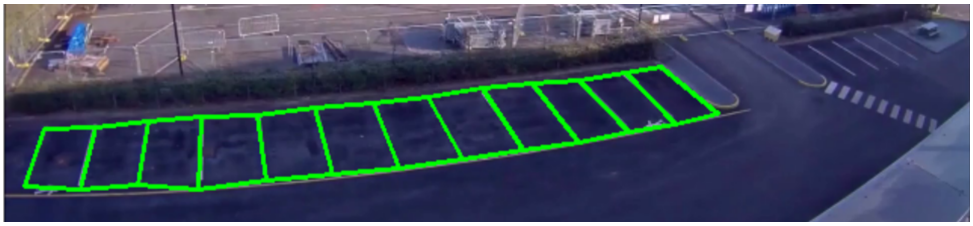
https://youtu.be/B0dHeate1kU

Process Flow for Occupancy monitoring and Vehicle movement analysis:



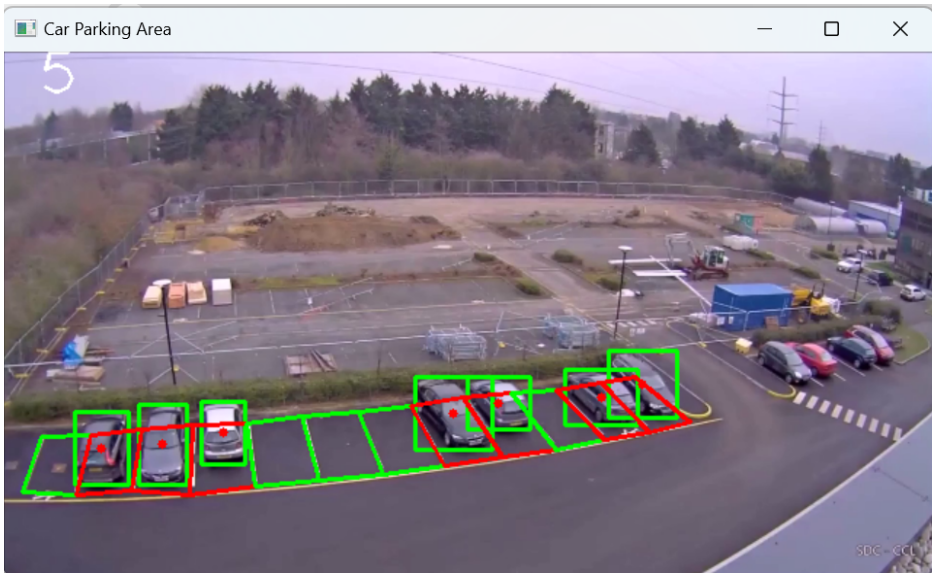
(Figure - 11 : Process flow diagram for monitoring and analysis)

1. **Area of Interest (AOI):**
   - Specify the area of interest for the frame of same size as the screen in which video is going to play.



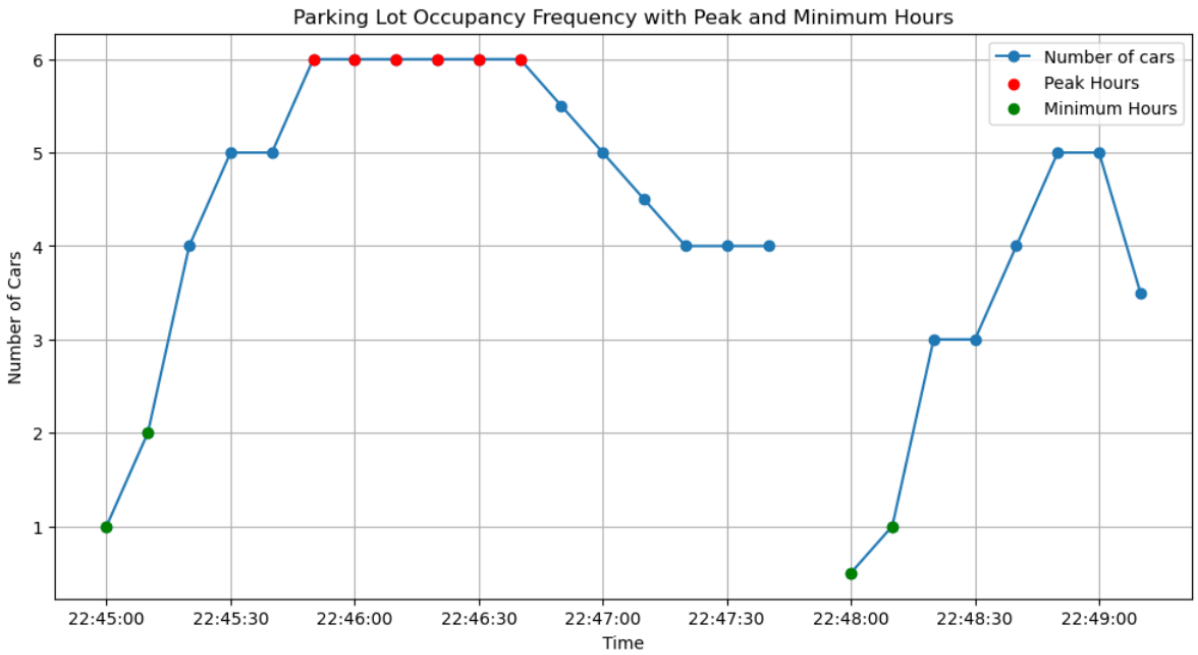
(Figure - 12 : Area of Interest is specified by providing co-ordinates in the frame)

2. **Object detection using YOLOv8:**
   - Whenever a car is detected in a frame check the position of co-ordinate of the centre of the bounding box of the car.
   - Use “pointPolygonTest” to check if the centre is inside or outside the AOI (i.e. parking space).
   - If the point is inside the AOI , then mark the parking-space as occupied.



(Figure - 13 : Free space count is specified on the top-left corner)

3. **Store Data**
    - While doing it in each frame store the number of cars present at that moment and the timestamp in a csv file.
    - So that the data from that csv file can be used later for analysis purpose.
4. **Analysis**
    - The number of cars is resampled for a specific time duration (20s / 30min/ 1hr) and its mean is taken. Resampled data's mean and standard deviation are found.
    - Mean and standard deviation are used to calculate the range of [minCarCount, maxCarCount]. Values above and below of these values are considered as least crowded and most crowded time.



(Figure - 14 : Parking lot occupancy frequency analysis (Number of cars vs Time) plot)

Sample Output

https://youtu.be/dM8fXOYfBFs

Data Post-Processing

License plate readings CSV

1. Among all the readings of the license plate from different frames, choose the one with highest confidence score.
2. Interpolate missing frames in the reading.
   Example :- We have readings for car A at frame 98 and 100, so we will fill same data for frame 99. So, when we will show the readings in the video, it will be smooth.

Team members and contribution

1. AYUSH RAJ

Worked on Vehicle Matching through license plate detection.

2. AMIT KUMAR MOHAPATRA

Worked on parking lot occupancy and vehicle movement analysis.

Conclusion

Use-cases

Vehicle matching

Security and Access Control:

- **Unauthorized Vehicle Detection:** Alert security personnel or automatically deny access to vehicles not registered in the system.
- **Visitor Management:** Track and manage visitor vehicle entry and exit times, enhancing campus or facility security.

Law Enforcement:

- **Wanted Vehicle Identification:** Aid law enforcement agencies in identifying vehicles associated with criminal activities or suspects.
- **Stolen Vehicle Recovery:** Quickly identify stolen vehicles through automated checks against databases.

Logistics and Fleet Management:

- **Fleet Tracking:** Monitor the movement of company vehicles for logistics planning and optimization.
- **Delivery Verification:** Confirm deliveries by matching license plates against expected schedules and routes.

Border Control and Immigration:

- **Border Security:** Monitor vehicles crossing borders and verify identities against watchlists or databases.
- **Immigration Control:** Track and manage vehicle entry and exit at border checkpoints.

Public Safety and Emergency Response:

- **Emergency Response Coordination:** Quickly identify and respond to emergencies by identifying vehicles involved or witnesses.
- **Public Event Security:** Monitor vehicles entering event venues for security and safety purposes.

Occupancy and movement analysis

Provide real time occupancy:

- Provide real time data on the number of available space.
- Reduces the time used by drivers to find parking space.

Parking guidance system:

- This system can guide drivers to available space using mobile apps.
- Increases the user experience.

Use cases in smart city:

- It decreases the time spent for searching parking space, which leads to decrease in carbon dioxide emission.

Dynamic pricing system:

- Adjust parking space price based on demand and occupancy of the parking space.

Show-stoppers

Vehicle matching

1. In India, people often use custom license plates with varying fonts making it difficult for EasyOCR to determine correct readings.
2. Any damage or discoloration on the license plate may lead to incorrect reading.
3. Lack of infrastructure for deployment.

Occupancy and movement analysis

1. High initial-cost: Significant investment required for infrastructure and technology integration.
2. Data privacy concern: Collecting and storing data on vehicle movements and parking raises privacy issues.
3. The co-ordinates of AOI(Area of Interest) need to be provided for different parking-lots.