Vehicle RoadSense – Vehicle, Lane and Pedestrian Detection

Dinesh Channabasappa Dhotrad
dept. Computer Science
Case Western Reserve University
dxd539@case.edu

Praneeth Kollati
dept. Computer Science
Case Western Reserve University
pxk505@case.edu

***Abstract*—"Vehicle RoadSense – Vehicle, Lane, and Pedestrian Detection" is an innovative project that integrates advanced digital image processing and deep learning techniques to create a robust system for road safety and intelligent transportation. The project emphasizes the application of sophisticated image processing algorithms for accurate lane detection, which is pivotal for autonomous driving systems. Through meticulous camera calibration, pixel extraction, and lane boundary detection, the system provides reliable lane departure warnings and precise vehicle positioning, thereby enhancing road safety measures.**

The project also incorporates the cutting-edge YOLOv8 pretrained model to achieve real-time detection of vehicles and pedestrians. This deep learning component is crucial for the system's ability to detect and track dynamic objects within the vehicle's vicinity, significantly improving situational awareness and response capabilities. While the core of lane detection relies on time-tested image processing techniques, the inclusion of YOLOv8 extends the system's detection capabilities, offering a comprehensive solution for vehicle and pedestrian recognition.

This technical report delves into the methodologies implemented, with a particular emphasis on the digital image processing techniques that underpin lane detection. It further explores the integration of the YOLOv8 model for detecting vehicles and pedestrians, highlighting its vital role in augmenting the system's overall performance. The results and discussions presented herein demonstrate the efficacy of this hybrid approach in facilitating accurate vehicle positioning, enhancing lane-keeping functions, and providing dependable vehicle and pedestrian detection in various driving conditions.

I. Introduction

As the world moves towards more intelligent and automated transportation systems, the need for advanced safety mechanisms becomes paramount. "Vehicle RoadSense – Vehicle, Lane, and Pedestrian Detection" stands at the forefront of this evolution, aiming to significantly enhance road safety and traffic efficiency. This project is a testament to the synergy between traditional digital image processing techniques and the power of deep learning, tailored to address the critical aspects of modern traffic management.

At the heart of the project lies the lane detection system, a cornerstone of vehicular safety. Employing classical image processing methods such as perspective transformation and polynomial fitting, the system meticulously analyzes camera feeds to delineate lane boundaries with remarkable accuracy. By scrutinizing pixel intensities and exploiting the geometry of road markings, it can discern lane curvatures and assess vehicle positioning, ensuring that drivers receive timely lane departure warnings.

In parallel, the project harnesses the prowess of the YOLOv8 pretrained model for the detection of vehicles and pedestrians in real-time. This deep learning model excels in identifying and tracking various entities within the vehicle's environment, thereby bolstering the system's capability to preemptively react to potential hazards. Despite its minimal reliance on deep learning, YOLOv8's contribution is instrumental in enriching the system's perception of its surroundings.

This report unfolds the methodologies underpinning the Vehicle RoadSense project, with a special focus on the digital image processing techniques that form the backbone of lane detection. It also touches upon the integration of YOLOv8 for vehicle and pedestrian detection, underscoring its pivotal role in reinforcing the system's detection suite. The ensuing sections will delve into the methods deployed, the results obtained, and the discussions therein, culminating in a conclusion that reflects on the project's impact on advancing road safety and traffic management.

II. Methods

Our methodology comprises several key steps to achieve accurate and robust vehicle road sensing, with a primary focus on traditional digital image processing techniques complemented by YOLOv8 object detection for vehicle and pedestrian recognition. The methods section is structured to provide a comprehensive understanding of each step in the pipeline, ensuring replicability and transparency.

Figure 1 below illustrates the architecture of our pipeline, providing a visual representation of the entire process. It details how each component of the system interacts and contributes to the overall functionality, from the input data to the detected and segmented lane and pedestrian. This figure serves as a guide to understanding the flow of data and the sequence of operations that lead to the detection and recognition of lanes, vehicles, and pedestrians.

A. Camera Calibration

Camera calibration is a pivotal procedure in the "Vehicle RoadSense" project, serving as the foundational step that precedes all subsequent image processing tasks. This process

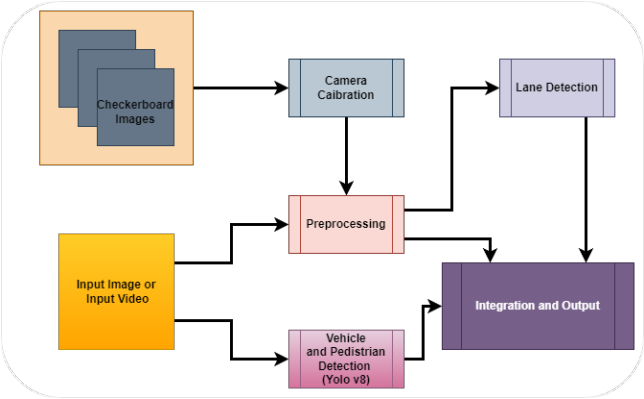


Fig. 1: Pipeline Architecture

is critical for rectifying any optical distortions introduced by the camera's lens, which, if left uncorrected, could lead to significant errors in spatial measurements and object detection.

1) Image Collection: A set of high-resolution images of a standard checkerboard pattern is captured from various angles and distances. This pattern provides a known geometric reference that is essential for calibration.

2) Corner Detection: Utilizing the OpenCV function 'cv2.findChessboardCorners', the precise corners of the checkerboard squares are detected in each image. These corners serve as key points for mapping the 2D image points to 3D real-world space.

3) Point Mapping: The detected 2D points are mapped to a predefined 3D model of the checkerboard pattern. This step establishes the correspondence between the 2D image plane and the 3D world coordinates.

4) Calibration Computation: With the mappings established, the 'cv2.calibrateCamera' function computes the camera matrix, which includes intrinsic parameters like focal length and optical center, and distortion coefficients that account for lens imperfections.

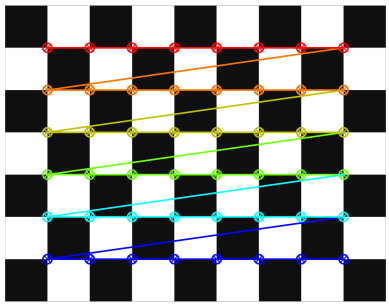


Fig. 2: Corner Detected in Checkerboard

The rationale behind camera calibration is rooted in the principles of photogrammetry and computer vision. Lens distortion, particularly radial distortion, can warp images, causing straight lines to appear curved. This warping effect is detrimental to applications like lane detection, where the accuracy of line curvature and vehicle positioning is paramount. By calibrating the camera, we ensure that the images used for lane detection are as close to the real-world scenario as possible, thereby enhancing the reliability of the entire system.

B. Image Preprocessing

Image preprocessing is a multi-faceted process that prepares the captured images for the lane detection algorithm. This process enhances the relevant features and suppresses noise, ensuring that the lane detection is as accurate as possible.

1) Image Undistortion: The first step in preprocessing is to apply the camera calibration results to correct any distortion in the captured images. Using the camera matrix and distortion coefficients, we employ the cv2.undistort function to rectify the images, ensuring that the geometric integrity of the scene is maintained.

2) Perspective Transformation: A crucial step in preprocessing is the perspective transformation. By transforming the image to a bird's-eye view, we simplify the lane detection problem, making the lanes appear parallel and easier to analyze.

(a) Front View Selction

(b) Processed Top View

Fig. 3: The input image is the figure on top and the perspective-transformed image on the bottom

3) Color Space Conversion: The undistorted image is first converted into HLS (Hue, Lightness, Saturation) and HSV (Hue, Saturation, Value) color spaces. These color spaces are more robust to changes in lighting conditions compared to the traditional RGB space and allow for more effective isolation of lane markings. Below in Figure 4 there is conversions for selection.

Fig. 4: HLS Color Spacing

4) Gradient Thresholding: The next step involves applying Sobel operators to the luminance (lightness) channel to detect edges in the image. This method highlights the structural outline of lane lines by focusing on areas where there is a rapid change in intensity.

5) Color Thresholding: In parallel with gradient thresholding, color thresholding is applied to the saturation channels of both HLS and HSV color spaces. This step is crucial for identifying lane lines based on their color, which is particularly useful in differentiating them from the surrounding pavement. Figure 6 shows the adaptive thresholding done.

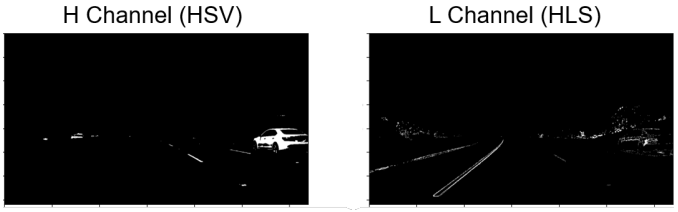


Fig. 5: Adaptive Thresholding

6) Combining Thresholds: The outputs from both gradient and color thresholding are combined to produce a binary image where the pixels of interest, primarily those belonging to lane lines, are represented as white against a black background.

Fig. 6: Combining H and S Threshold

The rationale behind each step of image preprocessing is grounded in the goal of maximizing the visibility of lane lines under varying environmental conditions. Color space conversion is justified by the distinct color properties of lane markings, which can be more easily separated from the rest of the scene in HLS and HSV spaces. Gradient thresholding is employed to detect the physical structure of lanes, which is characterized by edges and lines. Color thresholding complements this by targeting the unique color signatures of lane paint. The combination of these methods ensures that the system remains effective across a range of scenarios, including changes in natural and artificial lighting, weather conditions, and road textures.

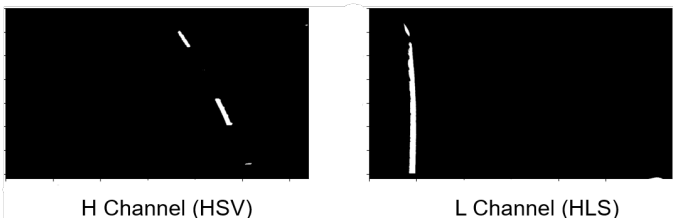
In addition to the core preprocessing steps, we discuss the selection of threshold values and their optimization for different driving conditions. We also explored the use of additional image filters, such as Gaussian blurring, to reduce noise and improve the reliability of edge detection. The choice of color spaces and thresholding techniques is further justified by referencing literature that highlights their effectiveness in lane detection applications.

C. Lane Detection

Lane detection is the centerpiece of the "Vehicle RoadSense" system, where the precise boundaries of the lanes are identified, and the vehicle's position within these lanes is determined. This process is vital for autonomous driving and driver assistance systems, as it directly influences navigation and safety.

1) Perspective Transformation: Once the perspective transformation has been applied to achieve a bird's-eye view, the lane detection algorithm proceeds to the critical task of identifying the pixels that make up the lane markings. This identification is a multi-step process:

a) Binary Thresholding: The transformed image is subjected to binary thresholding, which simplifies the image into a binary format where the lanes are represented by white pixels (value 1) and the rest of the image by black pixels (value 0). This thresholding is not a single-step process but involves several sub-steps, each designed to target specific features associated with lane lines, such as color and gradient.



(a) Adaptive Thresh on Perspective Transformed Img

(b) Combining Thresholds

Fig. 7: The input image is the figure on top and the perspective-transformed image on the bottom

b) Noise Reduction: To ensure that the binary image primarily contains information relevant to the lane lines, noise reduction techniques such as Gaussian blurring are applied. This step helps to mitigate the effects of shadows, road texture, and other extraneous factors that could lead to false detections.

c) Sliding Window Technique: With a cleaner binary image, the sliding window technique is employed to locate the lane pixels. This involves dividing the image into a number of horizontal bands and, within each band, sliding a window across the width of the image to identify the concentration of white pixels. The center of these windows is adjusted dynamically, moving left or right, depending on where the majority of the white pixels are concentrated. This approach is particularly effective in tracking the lanes through curves and variations in the road's width.

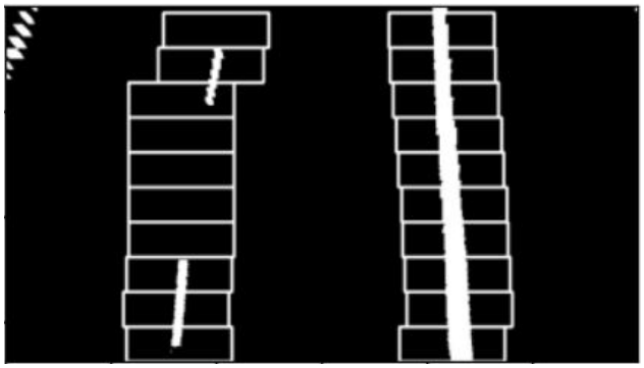


Fig. 8: Example Representation of Sliding Windows

2) Polynomial Fitting: The polynomial fitting process is where the detected lane pixels are used to define the actual path of the lane lines:

a) Least Squares Method:: The least squares method is used to fit a second-degree polynomial to the lane pixels. This statistical method minimizes the sum of the squares of the residuals, the differences between the observed values (the detected lane pixels) and the values predicted by the polynomial function.

b) Polynomial Representation: The second-degree polynomial is represented by the equation

$$y = Ax^2 + Bx + C$$

, where A, B, and C are the polynomial coefficients that define the curvature and orientation of the lane lines. The variable y represents the vertical position in the image, and x represents the horizontal position.

Below in Fig. 9 is an illustration of the output of the polynomial fit

c) Robustness and Adaptability: To ensure robustness, the fitting process includes mechanisms to adapt to different lane line visibility and road conditions. For instance, if the detected pixels do not form a coherent line, or if there are too few pixels, the algorithm can adjust its parameters or rely on historical data to predict the lane line path.

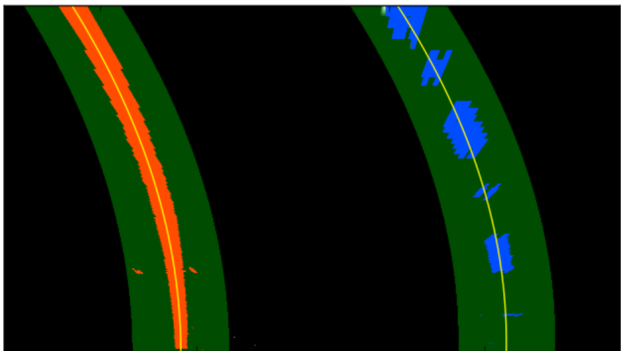


Fig. 9: Example Representation of Polynomial Fit

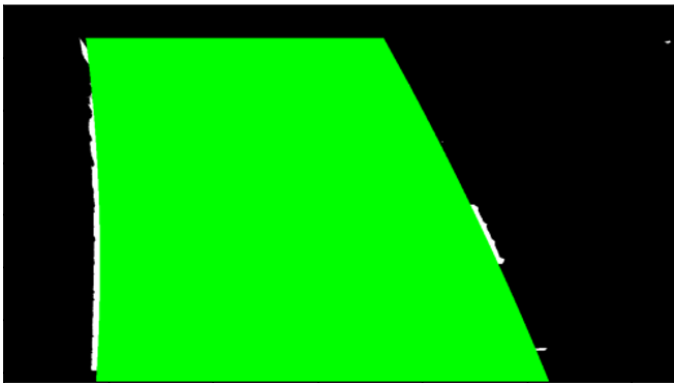


Fig. 10: Detected Lane Segmented

3) Curvature Calculation: The curvature of the lanes is calculated using the polynomial coefficients. This metric provides insight into the road's geometry and is essential for steering control in autonomous vehicles.

The curvature of the lane (C) and is calculated as follows:

$$C = \frac{(1 + (2Ay + B)^2)^{3/2}}{|2A|}$$

where A, B, and y are the coefficients and evaluation point from the polynomial fit of the lane lines.

The perspective transformation is justified by the need to simplify the lane detection problem. By transforming the road view into a plane where lane lines are parallel, we can apply more straightforward computational techniques to detect them. Polynomial fitting is the method of choice for lane boundary representation due to its ability to model the curvature of the road with a high degree of accuracy.

D. Vehicle and Pedestrian Detection

The vehicle and pedestrian detection component of the "Vehicle RoadSense" system is powered by the YOLOv8 pretrained model. This model is a cornerstone of the project due to its exceptional speed and accuracy, which are essential for real-time detection in dynamic traffic environments.

1) Model Selection: The YOLOv8 model is selected for its impressive capabilities in object detection tasks. It is pretrained on a comprehensive dataset, enabling it to recognize a wide array of objects with high precision [7].

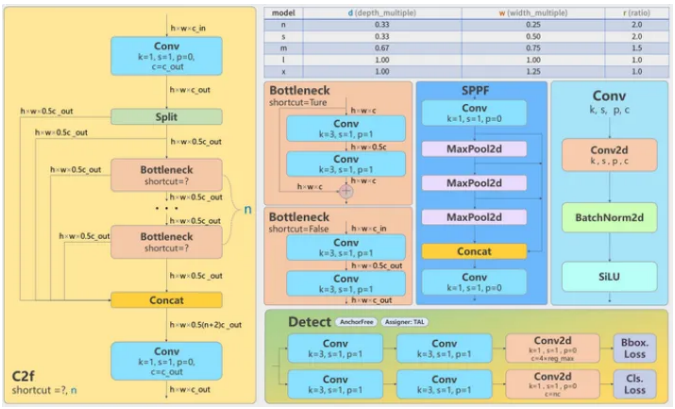


Fig. 11: Yolo V8 Architecture

2) Frame Processing: Each frame from the video feed is passed through the YOLOv8 model. The model analyzes the frame and identifies objects that match the patterns it has learned during training.

3) Bounding Box Extraction: For each detected object, the model computes a bounding box that accurately encloses the object. These boxes are crucial for determining the location and size of the objects within the frame.

4) Class Label Assignment: Alongside the bounding boxes, the model assigns class labels to each detected object, such as 'vehicle' or 'pedestrian', providing semantic information about the contents of the frame.

5) Confidence Scoring: Each detection is accompanied by a confidence score, which represents the model's certainty regarding the detection's accuracy. This score is used to filter out less reliable detections and reduce false positives.

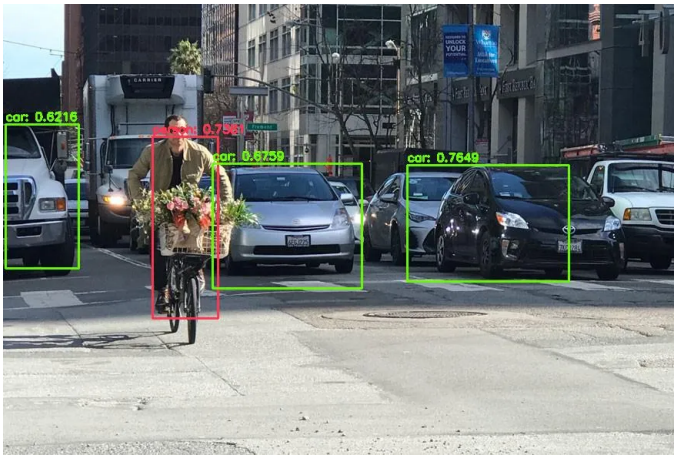


Fig. 12: Yolo V8 Output Example

YOLOv8 is the latest iteration in the YOLO (You Only Look Once) series of models, known for their efficiency and effectiveness in object detection tasks. The model's architecture allows it to process images quickly, making it suitable for applications that require real-time analysis, such as autonomous driving systems [9]. Its ability to detect small and overlapping objects is particularly valuable in cluttered urban traffic scenarios, where pedestrians and vehicles are often in close proximity [7].

In addition to the previously mentioned procedures and justifications for employing the YOLOv8 pretrained model, it is important to note the practical constraints that influenced this decision.

Time Constraints and Model Training: Given the time-bound nature of the project, developing a custom-trained model was not feasible. Training a deep learning model for object detection from scratch requires a substantial amount of time and resources. It involves collecting a large dataset, annotating the images with bounding boxes, and then iteratively training the model, which can take weeks or even months to achieve the desired level of accuracy.

III. Integration and Output - Results

The culmination of the "Vehicle RoadSense" system is the integration of lane detection with vehicle and pedestrian detection, synthesizing the data into a coherent output that can be acted upon by the driver or autonomous driving system.

Fig. 13: Detected segmented Lane Overlay on Input Image

1) Data Synthesis: The system begins by synthesizing the data from both the lane detection and the YOLOv8 object detection modules. This involves overlaying the detected lane boundaries onto the original video frame to provide a visual representation of the lane's path.

2) Bounding Box Overlay: Concurrently, the bounding boxes and class labels generated by the YOLOv8 model for vehicles and pedestrians are overlaid onto the frame. This step is crucial for situational awareness, as it highlights the location and identity of other road users.

3) Information Display: Alongside the visual overlays, the system computes and displays key information such as the curvature of the detected lanes. This information is presented in a clear and concise manner, we chose at the left top edge of the frame or in a designated image window area.

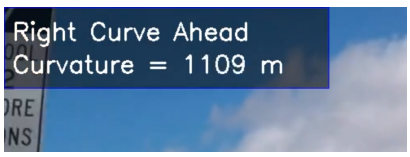


Fig. 14: Dashboard Visual Overlay

4) Real-Time Feedback: The integrated output is designed to provide real-time feedback to the driver or autonomous system. This feedback is critical for immediate decision-making, such as steering adjustments or speed control.

Fig. 15: Integration of Lane and Vehicle Detection

The integration of spatial and object detection data is justified by the need for a comprehensive understanding of the vehicle's environment. By combining these data streams, the system can provide actionable insights that are greater than the sum of their parts. For instance, knowing the position of the vehicle within the lane and the location of nearby pedestrians can inform advanced safety features like automatic emergency braking or evasive maneuvers.

IV. Conclusion

The "Vehicle RoadSense" initiative is a technical endeavor that integrates established image processing methodologies with advanced machine learning techniques to address the challenges of road safety and vehicular automation. The project's success hinges on the precise detection of lane boundaries, the identification of vehicles and pedestrians, and the synthesis of these data streams into a coherent output.

The system's lane detection algorithm is underpinned by a series of technical steps: camera calibration corrects lens distortion; image preprocessing enhances lane visibility; perspective transformation facilitates a top-down view for easier lane identification; and polynomial fitting mathematically models the lane boundaries. These steps are critical for the system's ability to accurately determine the vehicle's position within the lane and to provide essential navigational guidance.

Incorporating the YOLOv8 pretrained model addresses the need for real-time, accurate detection of vehicles and pedestrians. The model's utilization is a strategic decision, circumventing the extensive process of data annotation and custom model training, which is often constrained by time and resources. YOLOv8's proficiency in detecting small and overlapping objects in complex scenes is a key factor in its selection.

The final integration stage combines the lane detection results with the object detection outputs, overlaying this information onto the original video feed. This integrated view is crucial for the system's real-time response capabilities, enabling prompt and informed decision-making critical for autonomous driving applications.

The "Vehicle RoadSense" project exemplifies the practical application of computer vision and deep learning in creating advanced driver-assistance systems. It demonstrates the potential of such technologies to improve road safety and paves the way for future innovations in the field of intelligent transportation systems.

Fig. 16: Detection Example from the Video

V. Roles

A. Dinesh Dhotrad

Developed the utils/cameraCalib.py module, which is responsible for camera calibration. This module corrects for any distortion in the images, ensuring that the subsequent image processing is based on accurate spatial information.

Authored the performThreshold.py script, which includes functions for applying relative and absolute thresholding to images. This is a crucial step in isolating relevant pixels for lane detection.

Contributed to the detectionPipeline/laneDetection.py script, which includes the core algorithms for detecting lane pixels and fitting a polynomial to the lane lines.

Also contributed to the detectionPipeline/utils.py script, which provides utility functions such as calculating the curvature of the lane lines and annotating the lane information on the output frame.

B. Praneeth Kollati

Developed the utils/imagePerspective.py module, which handles the perspective transformation of the images. This transformation is key to converting the camera view into a top-down view, simplifying the lane detection problem. Authored the detectionPipeline/utils.py script alongside Dinesh Dhotrad, contributing to functions that calculate the curvature of the lanes and annotate the output images with lane information.

C. Together

Contributed to the main.py script, which serves as the entry point for the lane detection process. This script integrates various components of the system and orchestrates the flow from image capture to lane annotation.

REFERENCES

1) A curated list of awesome lane detection resources and projects.
   https://github.com/amusi/awesome-lane-detection
2) An article discussing advanced techniques in computer vision for lane detection.
   https://link.springer.com/article/10.1007/s42835-020-00570-y
3) A comprehensive study on the application of deep learning for real-time lane detection.
   https://arxiv.org/pdf/2404.14671
4) Research on object and lane detection techniques for autonomous vehicles using machine learning.
   https://www.researchgate.net/publication/356420839_Object_and_Lane_Detection_Technique_for_Autonomous_Car_Using_Machine_Learning_Approach
5) A chapter from a book that explores lane and object detection for autonomous vehicles.
   https://link.springer.com/chapter/10.1007/978-981-19-3015-7_17
6) A paper presenting a model for lane and object detection in self-driving cars.
   https://www.ijtsrd.com/papers/ijtsrd39952.pdf
7) The official documentation for Ultralytics YOLOv8, detailing its usage and features.
   https://docs.ultralytics.com/
8) The GitHub repository for Ultralytics, providing access to the YOLOv8 source code and resources.
   https://github.com/ultralytics/ultralytics
9) YOLOv8 : Comprehensive Guide to State Of The Art Object Detection
   https://learnopencv.com/ultralytics-yolov8/