AI System for Autonomous Vehicle Navigation: Methodologies and Rationale

By
LUONG NGUYEN MINH CHANH

1. **Introduction**

Autonomous vehicles (AVs) must operate in complex environments and be capable of perceiving their surroundings accurately to navigate safely. This system focuses on enhancing an AV's ability to detect known objects, identify novel or unexpected objects, and make real-time decisions to avoid collisions. The code uses several advanced computer vision techniques, including object detection with YOLOv8, background subtraction for novel object detection, lane detection, and a collision avoidance strategy that integrates data from both object detection and lane tracking. Each of these methods has been chosen for its specific strengths in addressing the challenges of autonomous navigation in both familiar and unfamiliar environments.

2. **YOLOv8 for Object Detection**

The primary method chosen for detecting known objects is YOLOv8. YOLOv8 is a fast and accurate object detection algorithm, ideal for real-time applications such as autonomous driving. The reason for using YOLOv8 in this system is its capability to perform high-speed detection of multiple objects in a single frame, which is crucial for a moving vehicle that needs to perceive its environment continuously.

YOLOv8 was selected for its balance between speed and precision. It employs a single convolutional neural network (CNN) to predict bounding boxes and class probabilities directly from an image, avoiding the slower region proposal techniques used in older models like R-CNN. By detecting objects in one pass through the network, YOLOv8 minimizes latency, making it suitable for real-time systems. Additionally, YOLOv8 is pre-trained on the COCO dataset, which includes a wide range of objects commonly encountered in road environments, such as cars, pedestrians, bicycles, and traffic lights. This makes it highly effective for identifying road hazards and obstacles that an autonomous vehicle needs to navigate around.

3. **Background Subtraction for Novel Object Detection**

While YOLOv8 is effective for detecting known objects, the system also needs to handle unexpected or novel objects that the model has not been trained to recognize. In rural or unfamiliar environments, objects like wildlife, fallen branches, or unconventional road hazards may appear. For this purpose, the system uses background subtraction.

Background subtraction is a method used to detect foreground objects by distinguishing them from a static or slowly changing background. In the system, OpenCV's **createBackgroundSubtractorMOG2** method is employed to dynamically learn the background over time and identify any objects that differ from this background model. The rationale behind using background subtraction is that it allows the system to detect novel, potentially dangerous objects that appear suddenly or that are not included in YOLO's training set.

This technique is particularly useful in rural areas where new or moving objects might not match the typical categories used in urban environments (e.g., vehicles or pedestrians). By applying background subtraction, the system can dynamically detect any moving or stationary objects in the scene that may pose a threat, even if those objects are not recognized by the pre-trained YOLO model. The method is simple and effective for identifying novel objects in dynamic scenes without requiring additional labeled training data.

4. **Lane Detection with Edge and Color Filtering**

Lane detection is a critical component of autonomous vehicle systems, as it helps the vehicle maintain a safe path by staying within the road boundaries. The chosen approach for lane detection involves a combination of color filtering and edge detection, followed by line detection using the Hough Transform. The reason for using color filtering is that lane lines on roads usually have distinctive colors (typically white or yellow) that can be separated from the rest of the road surface. By converting the image into the HSV (Hue, Saturation, Value) color space, the system can isolate these lane lines by applying a mask that highlights road-like colors while filtering out irrelevant parts of the image. This makes it easier to detect lane boundaries under varying lighting conditions.

Once the lane lines are highlighted, edge detection is performed using the Canny edge detection algorithm, which identifies sharp changes in intensity that correspond to lane markings. This step is crucial for recognizing the edges of lane lines that may otherwise be obscured by noise or variations in the road surface. After edge detection, the system uses the Hough Line Transform to identify straight lines that represent the lanes, mapping points from Cartesian coordinates to polar coordinates (ρ,θ), where:

$$\rho = x\cos\theta + y\sin\theta$$

The system looks for clusters in the (ρ,θ) space to detect straight lines representing the lanes. The reason for using the Hough Transform is that it is robust in detecting lines even when parts of the lane markings are missing or faded. This technique allows the system to predict the location of the lane lines based on the detected segments, providing the vehicle with accurate lane positioning information.

5. **Collision Avoidance Using Relative Object Positioning**

The ultimate goal of the system is to ensure the vehicle avoids collisions with both known and novel objects. The method chosen for collision avoidance involves calculating the relative position and distance of objects in relation to the vehicle and making decisions based on this data.

This approach is effective because it allows the system to make context-aware decisions. For instance, the system estimates the relative distance to objects based on their size in the video frame, with the assumption that larger objects are closer to the vehicle. By analyzing the relative position of detected objects and the vehicle's position within its lane, the system can decide whether to slow down or change lanes to avoid a collision.

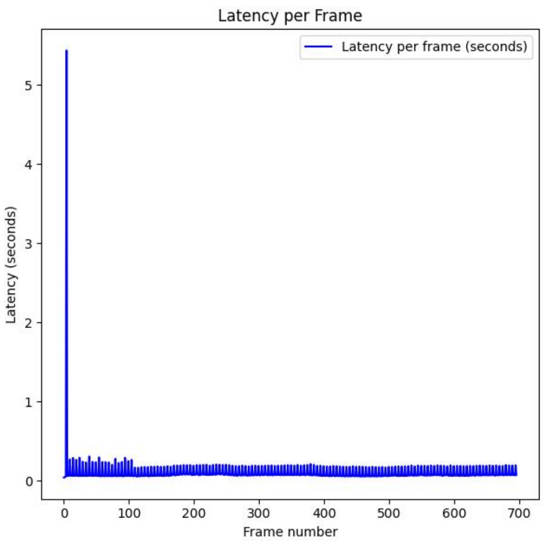
The decision-making process is driven by the position of the object:

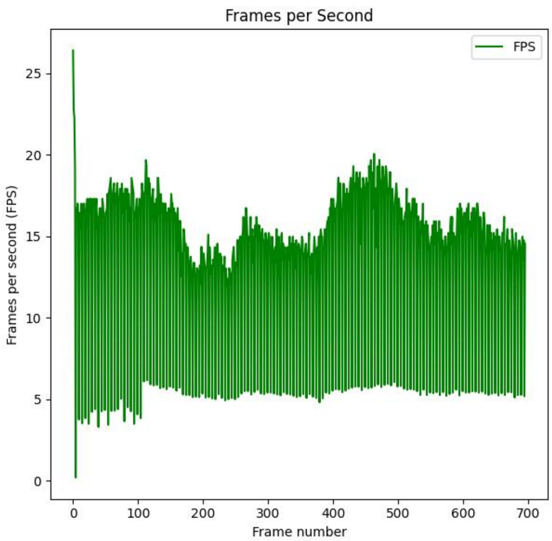
- If an object is detected directly in front of the vehicle and within the same lane, the system checks whether there is enough space in adjacent lanes to perform a safe lane change. If this is possible, the vehicle will attempt to change lanes.
- If a lane change is not possible (for example, if other vehicles are present in the adjacent lanes), the system reduces the vehicle's speed to prevent a collision.

This method is chosen because it is computationally efficient and provides an intuitive solution to the problem of dynamic obstacle avoidance. By focusing on relative positioning and lane data, the system can make real-time decisions that balance safety with the need to maintain traffic flow.

6. **Performance Evaluation**

The effectiveness of the described system relies heavily on its real-time performance, which can be evaluated through metrics such as **latency per frame** and **frames per second (FPS)**. These metrics provide insights into how efficiently the system processes video frames, detects objects, and makes decisions. The graphs accompanying this section depict the system's performance during a video test run, highlighting two important aspects: **latency per frame** and **FPS consistency**.





The **latency per frame** graph shows the time taken to process each individual video frame. The significant initial spike, where the latency reaches over 5 seconds, reflects the computational overhead required for system initialization. This is expected, as loading models such as YOLOv8 and setting up background subtraction takes a moment during the first frame. After this, the latency stabilizes to below 0.1 seconds for subsequent frames. This low and consistent latency is a positive indicator of the system's ability to process frames quickly and efficiently, an essential requirement for real-time applications like autonomous driving.

On the other hand, the **FPS graph** provides a view of the system's frame-processing speed over time. Initially, the FPS spikes to around 25-30, but this quickly drops as the system begins processing more complex scenes that require both object detection and novel object identification. Throughout the video, the FPS fluctuates between 5 and 20, with some noticeable dips to very low values, especially when there is a higher computational load (e.g., multiple objects or complex lane detection). The variability in FPS suggests that while the system can handle real-time processing in general, certain situations cause performance to degrade temporarily, potentially impacting its ability to make fast decisions in those moments.

7. **Conclusion**

The combination of YOLOv8, background subtraction, lane detection, and dynamic collision avoidance provides a comprehensive solution to the challenges of autonomous navigation. The latency and FPS results indicate that the system is capable of processing video data in real time, but the variability in FPS suggests that further optimization is necessary to ensure stability, especially in more computationally demanding environments. By employing these methods, the system can detect both known and novel objects, maintain its position on the road, and make real-time decisions to ensure safe navigation. The reasoning behind the selection of each method lies in their ability to provide reliable performance in dynamic environments, making the system suitable for a wide range of driving scenarios.