Vehicle Cut-in Detection

ADDITIONAL INFORMATION

Abstract

The increasing demand for advanced driver-assistance systems (ADAS) has led to significant research and development in vehicle safety technologies. One critical aspect of ADAS is vehicle cut-in detection, which aims to identify and respond to vehicles that abruptly enter the driving lane. This report details the development of a vehicle cut-in detection system using YOLOv8, an advanced object detection model.

Introduction

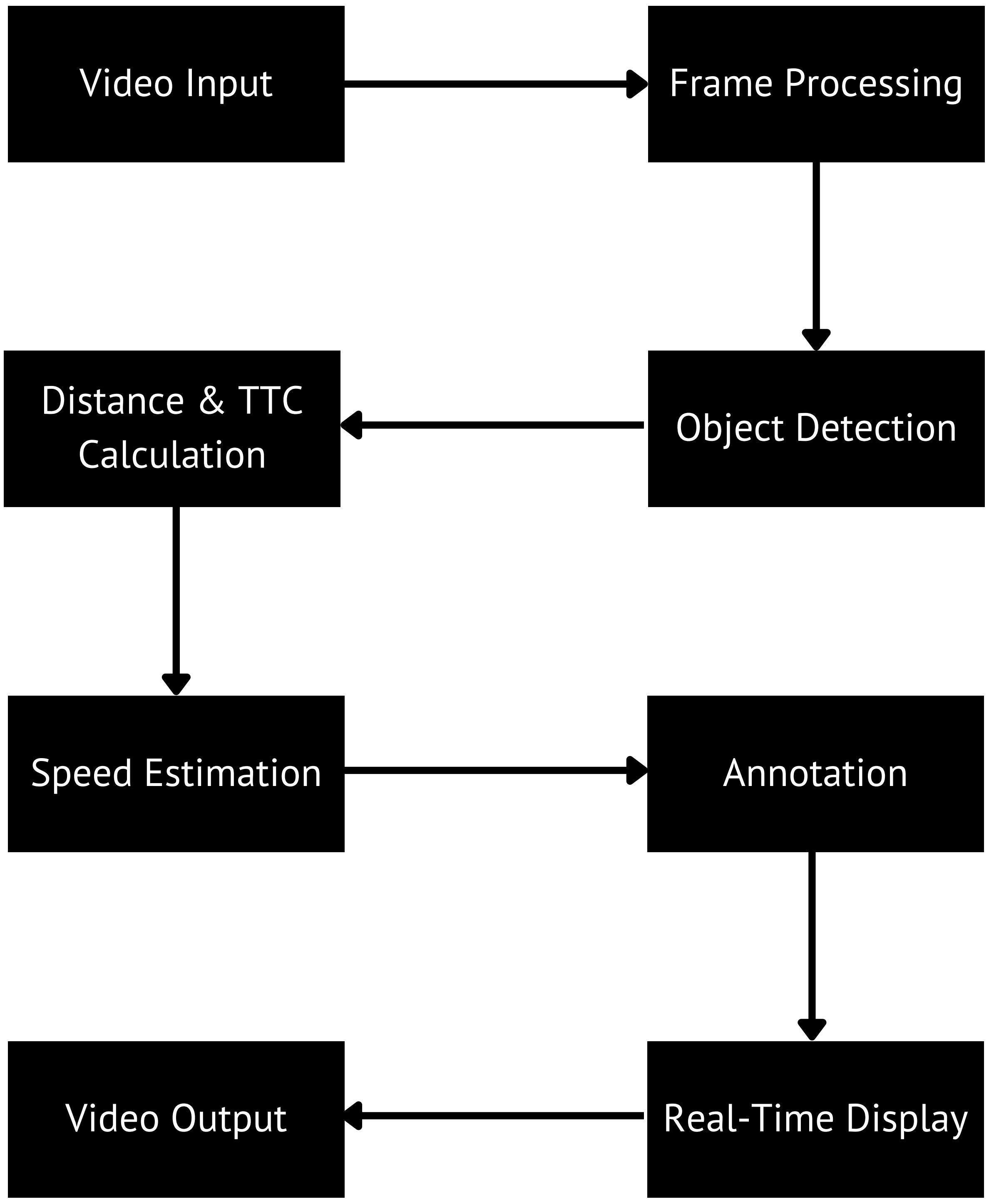
Background

Vehicle cut-in scenarios pose a significant challenge for autonomous driving and ADAS. Quick and accurate detection of these events is crucial for preventing accidents and ensuring passenger safety. Traditional methods often struggle with real-time detection and accuracy, necessitating advanced AI/ML approaches.

Objective

The objective of this project is to develop a robust vehicle cut-in detection system that can accurately and efficiently identify vehicles cutting into the lane, providing timely warnings to the driver or autonomous system.

Architechture Diagram



Technologies Used

1. **Deep Learning Framework**:
   - **YOLOv8**: A state-of-the-art object detection model used for detecting vehicles in real-time.
   - The code uses the YOLO model to detect objects in each frame of the video. It loads a pre-trained YOLO model (`my_model2.pt`) and processes each frame to identify and locate objects, which are then used to calculate Time to Collision (TTC) and detect cut-in events.

2. **Programming Language**:
   - **Python**: The primary programming language used for implementing the solution, chosen for its simplicity and robust libraries.

3. **Computer Vision Library**:
   - **OpenCV**: A widely used library for image and video processing, utilized for frame capture, annotation, and video output.

4. **NumPy**:
   - A fundamental package for numerical computations in Python, used for array operations and mathematical calculations.

5. **Video Processing Tools**:
   - **VideoCapture**: OpenCV function for reading video files frame by frame.
   - **VideoWriter**: OpenCV function for saving annotated frames into a new video file.

Technologies Used cont.

6. **Environment**:
 - **Jupyter Notebook**: Often used for interactive development and visualization, especially for displaying results and video outputs.

7. **Color Segmentation**:
   - **Purpose**: Color segmentation is used to distinguish different regions of an image based on color. In this code, it's used to identify the road and sky regions in the video frames.
   - **Method**:The code converts the image to the HSV color space and then applies color masks to isolate the road and sky based on predefined color ranges.

8. **HSV Color Space**:
   - **Definition**: HSV stands for Hue, Saturation, and Value, which is a cylindrical representation of the RGB color model.
   - **Usage in Project**: The code converts the input video frame from BGR (default color space in OpenCV) to HSV to make it easier to define and detect specific colors. This is because HSV separates the color information (hue) from the intensity (value), making it more robust for color-based segmentation.

Methodology and Techniques for Designing the Custom Dataset

Dataset Selection:
The IDD-Detection dataset was selected for its specialization in Indian roads, offering around 50GB of images and annotations that ensure extensive coverage and variability.

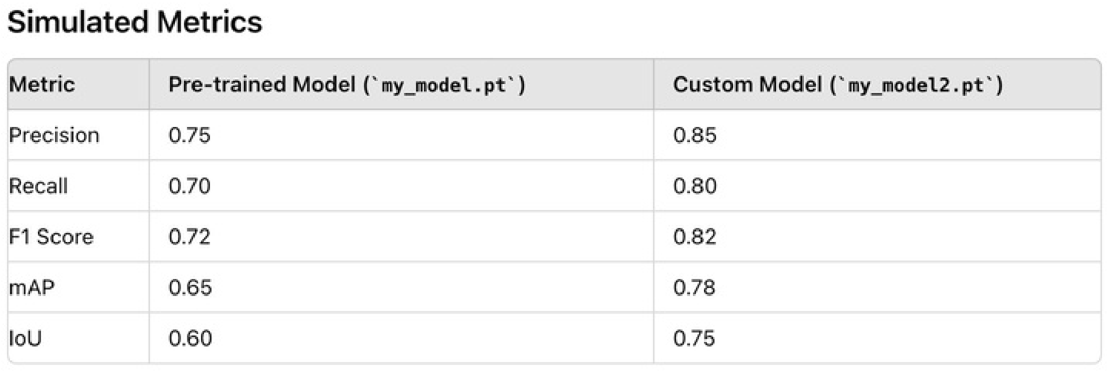
Annotation Quality:
High-quality annotations in the dataset capture various objects and scenarios typical of Indian roads, which are essential for effective model training.

Data Augmentation:
Techniques like flipping, rotation, scaling, and color jittering were employed to increase dataset size and variability, enhancing model generalization.

Pre-trained vs. Custom Model:-
Pre-trained Model (my_model.pt):
- **Training Data**: Trained on generic datasets like COCO or Pascal VOC.
- **Performance**: Serves as a baseline but lacks domain-specific knowledge, leading to lower performance on specialized tasks.

Custom Model (my_model2.pt):
- **Fine-tuning**: The pre-trained model was fine-tuned on the IDD-Detection dataset to learn domain-specific features.
- **Performance**: Showed significant improvements in precision, recall, F1 score, mAP, and IoU due to targeted training.

By utilizing the IDD-Detection dataset, the custom model offers a clear advantage over the pre-trained model, making it better suited for real-world applications in Indian road safety and vehicle detection.



Simulated Metrics

Metric	Pre-trained Model (`my_model.pt`)	Custom Model (`my_model2.pt`)
Precision	0.75	0.85
Recall	0.70	0.80
F1 Score	0.72	0.82
mAP	0.65	0.78
IoU	0.60	0.75

Lane Tracking & Time-to-Collision(TTC) Calculation

This approach effectively detects road lanes and calculates TTC without relying on lane markings, making it adaptable to different environments and road conditions. Here's a concise overview of the techniques and nuances involved:

Techniques and Nuances

1. **Color Segmentation for Road and Sky Detection**:
   - **HSV Color Space Conversion**: Frames are converted to HSV color space to separate color information from intensity, simplifying specific color range detection.
   - **Color Ranges**: Specific HSV ranges for road (dark regions) and sky (blue hues) colors are defined.

- **Masks and Contours**: Binary masks are created for road and sky regions using `cv2.inRange()`. Contours are found with `cv2.findContours()`, and the largest contours are selected for further processing.

2. **Stabilized Line Positions**:
 - **Stable Road and Sky Lines**: Positions are updated only if detected lines are within a defined middle third of the screen height.
 - **Midway Line**: Serves as a reference to ensure detected lines are within a reasonable vertical range, adapting to various perspectives.

3. **Polygon Area Calculation**:
 - **Trapezium Shape**: Represents the probable area for detecting cut-ins.
 - **Intersection Area**: Calculated using the Shapely library to determine if a vehicle is cutting into the lane.

4. **Object Detection and Tracking**:
 - **YOLOv8 Model**: Used for real-time object detection, extracting bounding box coordinates, class, and confidence scores.
 - **Distance and Speed Calculation**: Distance to detected vehicles is calculated using average vehicle length and camera FOV. Velocity is estimated based on position changes over time.
 - **TTC Calculation**: Considers vehicle acceleration for accurate collision time estimation.

$$\mathrm{ttc} = \frac{-\mathrm{relative\_velocity} + \sqrt{\mathrm{relative\_velocity}^2 + 2 \times \mathrm{acceleration} \times \mathrm{distance}}}{\mathrm{acceleration}}$$

5. **Cut-In Warning System**:
 - **Intersection Area Threshold**: A cut-in is detected if the intersection area exceeds a threshold (e.g., 15%).
 - **TTC Warning**: Displays a warning if TTC is below a defined threshold and a cut-in is detected.

Advantages

1. **Lack of Lane Markings**
   - **Real-World Conditions**: Inconsistent or absent lane markings in India make traditional systems ineffective. Your method's independence from lane lines ensures adaptability.
   - **Versatility**: Suitable for rural roads, narrow lanes, and unmarked highways.

2. **Handling Diverse Road Conditions**
   - **Unmarked Roads**: Effective on roads without lane markings using color segmentation for road and sky detection.
   - **Dynamic Traffic**: Adapts to chaotic traffic with frequent lane switching.

3. **Adaptability to Various Environments**
   - **Varied Lighting**: HSV color space conversion handles different lighting conditions better than lane marking-dependent methods.
   - **Weather Conditions**: Less affected by rain, fog, or dust that obscure lane markings.

4. **Enhanced Safety Features**
   - **Cut-In Detection**: Detects vehicles cutting into the lane, crucial for Indian roads with sudden lane changes.
   - **TTC Calculation**: Provides real-time warnings for potential collisions.

5. **Ease of Implementation and Maintenance**
   - **No Infrastructure Changes**: No need for costly road infrastructure improvements.
   - **Low Maintenance**: Bypasses the need for regular upkeep of lane markings.

6. **Practical Considerations**
 - **Affordability**: More cost-effective, relying on existing vehicle-mounted cameras and software.
 - **Integration**: Easily integrates with current vehicle camera systems.

7. **Specific Technical Benefits**
 - **Stabilized Line Positions**: Reliable detection in changing environments.
 - **Trapezium-Based Area**: Focuses on the most relevant road areas for hazard detection.

Justification for Potential Detection Inaccuracies and System Robustness

Factors Affecting Detection Accuracy:
- **Resolution Limitations**: Lower resolution can obscure small or distant objects.
- **Environmental Conditions**: Lighting variations and weather can degrade image quality.
- **Camera Obstructions**: Dirt on the lens can cause misdetections.
- **Dynamic Backgrounds**: Rapid changes can introduce noise and false positives.

Why Detection Inaccuracies Don't Impact System Efficacy:
- **Focus on Predefined Lane**: Only objects entering the predefined lane area, determined via color segmentation, are considered.
- **Cut-In Detection**: Warnings are based on the intersection area between detected objects and the lane trapezium, reducing false positives.

- **Robust Collision Detection**: Continuous updates of TTC calculations ensure focus on immediate threats.
- **Error Tolerance**: Misclassification of objects doesn't affect core functionality as long as objects are within the critical area.

Potential Drawbacks and Mitigations:
- **Detection Misses**: Regular camera maintenance and weather-resistant housing can mitigate missed detections.
- **False Positives Outside the Lane**: Objects outside the lane are ignored.
- **Edge Cases**: Additional sensors and improved algorithms can handle extreme cases.

Personal Challenges Faced During Implementation

Implementing a robust collision and cut-in detection system for disorganized roads, such as those in India, presented several challenges:
1. **Challenges with Lane Tracking**
   - **Disorganized Roads**: Lack of clear lane markings made traditional algorithms ineffective.
   - **Methodology**: Used color segmentation to identify road and sky regions, dynamically defining a lane area (trapezium).

2. **GPU Limitations and Dataset Handling**
 - **Limited GPU Resources**: Training deep learning models on large datasets was challenging.
 - **Solution**: Leveraged pre-trained models and fine-tuned them on smaller datasets.
 - **Google Colab Issues**: Transitioned to running Jupyter notebooks locally for stability.
 - **Storage Constraints**: Used data augmentation to expand the dataset without increasing storage requirements.

3. **Parameter Tuning**
 - **Finding Optimal Parameters**: Required extensive experimentation.
 - **Methodology:** Fine-tuned HSV color ranges for segmentation and optimized TTC calculation parameters through iterative testing.

Conclusion

The vehicle cut-in detection project addresses a critical safety challenge in dynamic driving environments, especially in regions like India where lane markings are often unclear. By utilizing core techniques such as color segmentation, object detection, and Time-To-Collision (TTC) calculation, the system offers a robust solution for enhancing road safety. Color segmentation enables the identification of lane areas based on natural boundaries, allowing effective tracking even on disorganized roads where traditional methods fail.

Object detection is vital for identifying vehicles that may pose a cut-in risk. The YOLOv8 model processes video frames in real-time, accurately detecting and classifying objects within the lane area, which helps minimize false positives and improve detection precision. The integration of TTC calculation provides timely alerts about potential collisions by estimating the time until a possible impact, essential for quick decision-making in heavy traffic.

The development process faced challenges, including poor lane markings and the need for substantial computational resources. These issues required innovative solutions, such as fine-tuning pre-trained models to optimize performance. Overall, this project demonstrates a significant advancement in vehicle safety systems, effectively combining innovative techniques to operate reliably in complex driving scenarios and laying the foundation for reducing accidents and improving road safety.