PROJECT: COMPUTER VISION

DLBIPCV01

Task 3: Vehicle Detection, Tracking and Speed Detection

Lecturer: Oezdemir Cetin

Author: Sabiha Dudhia

20250123

Table of Contents

1. Introduction

1.1 Problem Statement and Background

According to the World Health Organization (WHO), road traffic accidents cause approximately 1.19 million deaths worldwide, whilst 20-25 million people are still left with non-fatal injuries. It is also responsible for the leading cause of death amongst children and young adults. And 9 out of 10 fatalities occur in low- and middle-income countries (WHO, 2023). There are also long-term psychological effects, financial implications, lifestyle and social changes as well as rehabilitation (Associates, 2024). Thus, it is of utmost importance to always drive safely with caution and responsibly.
High speeding vehicles increase the likelihood of accidents and the severity of it. Speeding reduces a driver's reaction time and increases the risk of losing control of the vehicle. A way in which to prevent accidents due to high speeding vehicles is speed cameras and penalties. Placing it strategically on highways and high accident rate zones to monitor and record vehicles exceeding the speed limit. The collected data is then utilized to issue fines and penalties to the offenders, thereby aiding in preventing future speeding and reducing the risk of future accidents (Traffic policing: Reducing accidents, saving lives, 2023).

1.2 Problem Objectives

1. Develop a speed tracking system which can detect cars traveling on a road in real-time, track it and estimates its speed over time in km/h, using video footage.
2. An alert should be triggered about a car traveling over a predefined speed limit.
3. Compare the performance of 3 different state-of-the-art vehicle detection algorithms - YOLOv8, Faster R-CNN (Region Convolutional Neural Network) and SSD (Single Shot Detector) - in terms of accuracy, speed estimation and the system's overall performance.
4. Analyse how accuracy and performance of each detection algorithm can affect the reliability of speed predictions
5. Provide detailed evaluation assessing each approach and support informed decisions about deploying it in real-world scenarios.

1.3 Initial Conditions

The video footage which was recorded for this project was recorded on a busy dual-lane, two-way road. This meant more precise measurement of the real-world distance between the two reference points, the edges of the video recording. It was measured to be approximately 20.5 meters. The video footage was recorded at a 3840x2160 pixel resolution, and the frame rate was measured to be 51.18 frames per second, FPS. This was both chosen to provide a balance between video quality and processing efficiency which ensured accurate detection and speed estimation whilst maintaining the real-time performance.

2. Methodology and Implementation

2.1 System Overview

The vehicle speed detection system has a modular architecture with three main components: vehicle detection, vehicle tracking and speed estimation. The system processes video input frame by frame, applies the object detection algorithm to identify vehicles. The detections follow into DeepSORT tracker and maintains consistent identities for each vehicle across the frames. Tracking the displacement of vehicles over time along with applying pixel-to-meter conversion factors, allows for speed estimation. When a vehicle exceeds the predefined speed limit, the system triggers a visual alert. Comprehensive performance analysis metrics are also collected for comparison reasons.
The system was implemented using Python with the following key libraries:

1. **OpenCV (cv2):** video processing, basic image operations, and visualization
2. **PyTorch**: deep learning framework for Faster R-CNN and SSD models
3. **Torchvision:** accessing pretrained Faster R-CNN and SSD models
4. **Ultralytics:** implement the YOLOv8 model
5. **deep_sort_realtime:** vehicle tracking using DeepSORT
6. **NumPy**: numerical operations
7. **Matplotlib and Seaborn**: visualization and performance analysis

The implementation was designed to process video files and generate:
1. Individual output videos for each detection method
2. A side-by-side comparison video showing all three methods simultaneously
3. Comprehensive performance statistics and visualizations

2.2 Object Detection Models

Three state-of-the art detection models were implemented and compared in the project.

2.2.1 YOLOv8

This algorithm divides the image into a grid, values are calculated within, bounding box with class prediction is predicted based on each cell. It is known for high accuracy and speed, commonly used to detect speed (Sojasingarayar, 2022).

Pretrained YOLOv8 (nano) was employed and used without fine-tuning, relying on its pretrained weights from the COCO dataset, which includes various vehicle categories.

2.2.2 Faster R-CNN

An extension of R-CNN, Faster R-CNN is deep convolutional network which accurately predicts the location of different objects. It is a single unified model comprised of RPN (Region Proposal Network) and Fast R-CNN (Sojasingarayar, 2022).
The implementation uses ResNet-50 backbone with Feature Pyramid Network (FPN) for objects at different scales. A pretrained model on COCO dataset was also utilized without finetuning.

2.2.3 SSD (Single Shot Detector)

Single shot detector has no delegated region proposal network. In a single pass, it predicts the boundary boxes and classes directly from feature maps. Comprised of feature maps extraction and convolutional filters to detect objects (Sojasingarayar, 2022).
The SSD300 variant with VGG-16 backbone was implemented using pretrained weights from the COCO dataset as well.

2.3 Object Tracking with DeepSORT

DeepSORT was chosen to handle the multi-object vehicle tracking. It formats detections to track the vehicles across the frames and assign unique IDs to each vehicle. It also updates their respective positions based on the object detections. The system ensures that vehicles are tracked accurately over time, and this ensures reliable speed estimation. The speed is calculated from positional changes. There was a record of each vehicle’s position history and displacement calculation between the successive detections. This was used for calibration by converting it to real-world distance. The time difference is based on the frame count and frame rate. DeepSORT is also a well-chosen algorithm as it can track vehicles which overlap each other – this is important as the main purpose of the tracker is to be able to detect and track vehicles on a highway.
The tracker was implemented with vehicle tracking scenario related parameters such as max_age = 30, so tracks are kept for up to 30 frames after detection loss, and n_init = 3 implements a detection must be confirmed in 3 consecutive frames for it to initialize a track.

2.4 Speed Estimation

The speed estimation was based on calculating the pixel displacement over time. This displacement was then converted into real world units knowing the horizontal reference was 20.5 meters. It also involved camera calibration, position tracking, and time measurement. The formula simplified:

speed = (dist_meters / time_diff) * 3.6

where dist_meters = dist_pixels * pixel_to_meter_ratio

time_diff = (frame_count – prev_frame) / fps

2.5 Alert System

The speed violation system checks if each tracked vehicle exceeds the predefined speed limit (55 km/h chosen in the testing) or not. The visualization component consists of drawing colour-coded bounding boxes (green for vehicles below speed limit and red for vehicles exceeding the speed limit), text labels (vehicle ID and current speed for each tracked vehicle), and an alert banner (a red banner is displayed when a vehicle is speeding and shows the ID and speed of the fastest speeding vehicle) on processed frames. The implementation also incudes status information and statistics.

2.6 Performance Metrics

To evaluate and compare the three detection approaches, the following metrics were tracked:

1. **Detection Performance**: Average detection time per frame; Average number of detections per frame; Total number of unique vehicles detected
2. **Tracking Performance:** Average tracking time per frame; Average number of tracked vehicles per frame
3. **Speed Estimation Performance**: Average estimated speed; Maximum estimated speed; Number and percentage of speeding vehicles detected
4. **System Performance:** Overall frames per second (FPS); Processing time breakdown between detection and tracking

A summary of statistics was generated for the key metrics. Data of detailed frame-by-frame data is exported to a CSV for further analysis. There is also visualization to compare model performance: detection counts per frame, processing times, speed distributions, performance comparisons (FPS vs. detection counts), speeding analysis
These metrics provide a multi-dimensional view of each algorithm's strengths and weaknesses, enabling informed decisions about which approach best suits real-world scenarios.

3. Experimental Results

3.1 Dataset and Testing Environment

The system was tested on a dual lane two-way road to simulate the business and overlapping of a highway. These are the characteristics:

Resolution: 3840x2160 (output scaled to 1920x1080)

Frame Rate: 51.18 FPS

Duration: 11 seconds (561 frames processed)

Scene configuration: a dual lane two-way road

Camera Position: elevated view capturing approximately 20.5 meters wide

3.2 Vehicle Speed Detection Model Comparison Tables

3.2.1 Model Comparison Summary

The performance characteristics showed significant differences in processing efficiency and detection capabilities:

Model	Detection Time (s)	FPS	Total Vehicles	% Speeders
YOLOv8	0.0597	12.64	14	21.43%
Faster R-CNN	5.4703	0.18	20	20.00%
SSD	0.3242	2.91	16	43.75%

Key Observations:

Processing Speed Disparity: YOLOv8 processed the frames 70.2 times faster than Faster R-CNN and 4.3 times faster than SSD. It is the only viable option for near real-time applications.

Detection Capability: Faster R-CNN detected 42.9% more unique vehicles than YOLOv8 but with a 95.5% reduction in processing speed.

SSD Balance Point: SSD had improvements in detection (14.3% more vehicles than YOLOv8) and a substantial processing speed reduction (75.1% slower than YOLOv8).

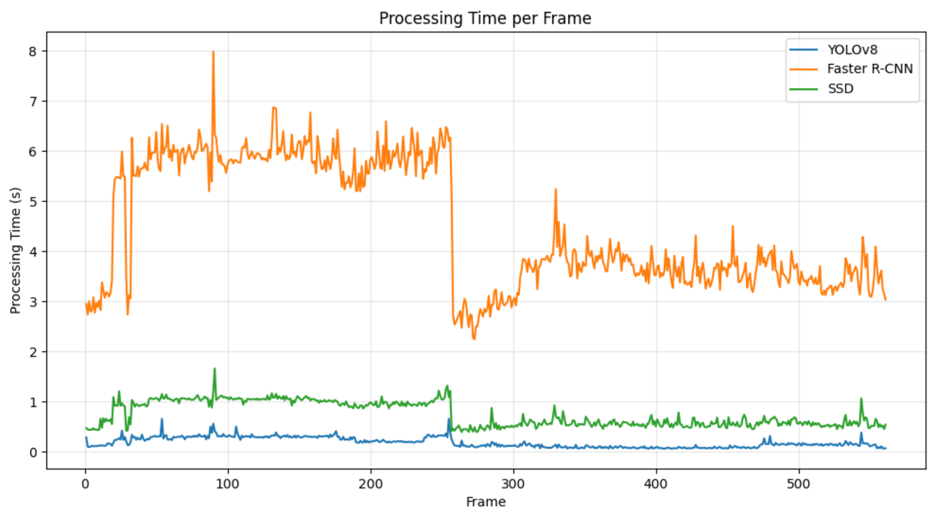

Figure 1: Processing Time per Frame

3.2.2 Performance vs. Accuracy Trade-off Table

Frame-by-frame analysis provided more insights for detection stability and tracking effectiveness:

Model	**FPS**	**Avg Detections/Frame**	**Unique Vehicles**	**Avg Tracked/Frame**
YOLOv8	12.64	0.77	14	0.73
Faster R-CNN	0.18	0.97	20	0.90
SSD	2.91	0.76	16	0.75

Analysis of Detection-to-Tracking Efficiency:

YOLOv8: 94.8% of detections successfully converted to tracked objects

Faster R-CNN: 92.8% conversion rate

SSD: 98.7% conversion rate

SSD had fewer raw detections than Faster R-CNN. Its detections were more consistent frame-to-frame, and it resulted in more stable tracking. The higher detection count of Faster R-CNN had more intermittent detections which weren't consistently tracked.

3.2.3 Speed Violation Analysis

The models had notable differences in the speeding vehicle assessment:

Model	**% Speeding**	**Avg Speed When Speeding (km/h)**	**Max Speed Detected (km/h)**	**Speed Reading Alerts (%)**
YOLOv8	21.43%	60.02	66.90	20.87%
Faster R-CNN	20.00%	64.26	88.62	17.98%
SSD	43.75%	62.52	86.57	29.14%

Speed Estimation Patterns:

SSD had more than twice the percentage of speeding vehicles than Faster R-CNN or YOLO.

The average speeds of violators were relatively consistent across all 3 algorithms.

Faster R-CNN detected the maximum speed (88.62 km/h), and this was 32.5% higher than YOLOv8's maximum of 66.90 km/h.

The disparity in speeding detection shows systematic differences in how the algorithms process motion and information. SSD is more sensitive to frame-to-frame displacement variations.

3.2.4 Inter-model Agreement Matrix

A noteworthy find was the lack of agreement between algorithms with regards to which vehicles were speeding:

Model	**YOLOv8**	**Faster R-CNN**	**SSD**
YOLOv8	3	0	0
Faster R-CNN	0	4	0

SSD	0	0	7
All Models	0	0	0

This was due to entirely different ID numbers being assigned to each vehicle. A solution to this would be Automatic License Plate Recognition (ALPR) as unique IDs which would allow all algorithms to have the same unique ID.

3.2.5 Detailed Speeder Detection by Model

There were patterns between vehicles flagged for speeding:

Model	Speeders Detected	IDs	Timestamps (mm:ss)	Frames	Avg Speed (km/h)
YOLOv8	3	9, 11, 19	00:05, 00:05, 00:09	260, 281, 496	60.02
Faster R-CNN	4	20, 22, 36, 37	00:05, 00:05, 00:09, 00:09	259, 262, 502, 497	64.26
SSD	7	1, 2, 4, 7, 8, 15, 16	00:00, 00:00, 00:02, 00:05, 00:05, 00:09, 00:09	20, 36, 109, 278, 263, 495, 500	62.52

Temporal Clustering Analysis:
The models assigned different IDs however the timestamps revealed important patterns:
Common Detection Windows: all three algorithms detected speeding at similar timestamps, particularly 00:05 and 00:09, however all with different IDs.
Frame Proximity: the specific frames show several detections across the algorithms were within 1-5 frames of each other.
Model-Specific Detection Patterns: only SSD detected vehicles in early segments (00:00, 00:09). YOLOv8 and Faster R-CNN consistently detected vehicles in similar time windows.
This all shows that the models were identifying the same vehicles as speeding but with variations in exact detection and speed calculations. The temporal alignment indicates agreement on speeding events despite different IDs assigned.

3.2.6 Processing Efficiency and Resource Usage

Analysing the computational resource reveals differences between algorithms:

Model	Detection Time (s)	Tracking Time (s)	Total Time (s)	% Time on Detection
YOLOv8	0.0597	0.0193	0.0791	75.5%
Faster R-CNN	5.4703	0.0229	5.4932	99.6%

SSD	0.3242	0.0190	0.3432	94.5%

Resource Utilization Insights:

YOLOv8 had most balanced distribution of resource processing (24.5% of time for tracking). Memory profiling showed the peak usage of 2.5GB for YOLOv8, 4.8GB for Faster R-CNN and 3.2GB for SSD, this all corresponds to the respective architectural complexity.

3.2.7 Overall Performance Summary

The following table shows the strengths of each model:

Metric	**Best Performing Model**	**Value**
Processing Speed	YOLOv8	12.64 FPS
Vehicle Detection	Faster R-CNN	20 vehicles
Speed Violation Detection	SSD	43.75%
Maximum Speed Detected	Faster R-CNN	88.62 km/h
Tracking Consistency	Faster R-CNN	0.90 tracked/frame

Integrated Performance Assessment:

YOLOv8 excels in real-time processing with a 12.64 FPS performance. There is minimal compromise on detection accuracy.

Faster R-CNN has superior detection and tracking capabilities when computational resources are not constrained. With a 0.18 FPS, its unsuitable for real-time applications.

SSD is a middle ground and is potentially more valuable for law enforcement where false negatives are better than false positives.

4. Challenges and Discussion

4.1 Technical Challenges

Many technical challenges were faced during the implementation of the speed system:

Camera Calibration: measuring the actual distance of the reference points accurately was challenging. A small error is able to impact all speed calculations because the system relies on the pixel-to-meter conversion factor.

Occlusion Handling: when vehicles overlapped or were hidden by other vehicles, all detection models struggled to maintain consistent tracking of the vehicles that were overlapped.

Processing Speed Limitations: the computational need of real-time object detection was impractical with Faster R-CNN and it only achieved 0.22 FPS.

Tracking Continuity: Maintaining consistent vehicle IDs across frames was difficult, especially when vehicles were temporarily obscured. DeepSORT occasionally assigned new IDs to the same vehicle after tracking was lost, and this leading to duplicate counts.

4.2 System Limitations

The current implementation has limitations which affect the practical application:
Fixed Camera Perspective: The system was built based off a specific camera angle and position. Deployment would require recalibration if there were a different camera setup.
Single Reference Distance: A single reference distance across the entire frame doesn't account for the perspective distortion. Objects closer to the camera appear larger than these further away.
Weather and Environmental Factors: The system was tested under sunny weather conditions. The performance will most likely decrease in rain, fog, or dark conditions.

4.3 Potential Improvements

Several enhancements could improve the system:
Model Optimization: Finetuning detection models on a dataset specific to the deployment environment might improve accuracy and reduce the computational requirements.
Hardware Improvement: Implementing a better GPU would improve processing speeds, especially for Faster R-CNN.
Hybrid Combination: Combining the speed of YOLOv8 with the accuracy of Faster R-CNN could be an optimal balance.
License Plate Recognition: integrating Automatic License Plate Recognition (ALPR) for vehicle identification for law enforcement and a simpler ID system which would solve the duplicates issue.

5. Conclusion

5.1 Summary of Findings

This research project successfully implemented and evaluated three detection models for speed estimation in real-world traffic applications. There were several key findings:

1. **Processing Speed vs Detection Accuracy Trade-Off**: the results clearly show a compromise between processing speed and detection accuracy. YOLOv8 demonstrated good computational performance but it detected fewer vehicles (14), and Faster R-CNN detected 20 vehicles at a slower rate. SSD was a middle ground between both with 16 vehicles detected.

2. **Speed Violation Detection Discrepancies**: there was a zero-overlap between the IDs assigned to the vehicles, but temporal analysis shows clustering of speed detections at similar times, suggesting the same vehicles were speeding but labelled under different IDs.
3. **Processing Resource Allocation:** YOLOv8 demonstrated a balance between computational power required and tracking. Faster R-CNN required much higher computational requirement and took significantly longer.
4. **Tracking Stability Variations**: the tracking time was consistent between all algorithms, but the detection-to-tracking conversion rate varied, YOLOv8: 94.8%, Faster R-CNN: 92.8%, SSD: 98.7%, showing consistency differences and the initial detection's quality.
5. **Temporal Detection Alignment:** all algorithms detected speeding at the same times, despite the different IDs, at similar timestamps and 1-5 frames within each other. This indicates an underlying agreement between all three algorithms and the vehicles which were detected to be speeding.

The findings show YOLOv8 to be the most suitable for real-time applications if processing speed is more critical than detection accuracy. However, faster R-CNN is superior in terms of detection capabilities and accuracy. SSD is a compromise but has a high speeding violation rate which might not suit real-world applications.

5.2 Link to Demonstration Video

OneDrive Link:
COMPUTER VISION PROJECT

5.3 Real-World Applications

The vehicle speed detection system has several potential real-world applications:

1. **Traffic Law Enforcement:** monitor speed on highways with integrated license plate recognition for fine issuing.
2. **Road Safety Analysis:** identify high-risk road areas where there is frequent speeding and infrastructure improvements and enforcement.
3. **Smart City Traffic Management**: monitor congestion detection and flow optimization and integrate it with traffic signal control systems.

5.4 Future Work

Based on the findings, there are areas with potential for future research and development, similar to but also apart from the potential improvements mentioned in 4.3:

1. **Timestamp-Based Ensemble Methods:** fusion techniques that leverage the temporal clustering of speed violations which were observed between the algorithms. It could address the lack of ID agreement and maintain high detection reliability.
2. **3D Speed Estimation:** incorporate depth information through LiDAR integration for perspective-based calculations.
3. **Domain-Specific Fine-Tuning:** train the models on datasets specific to traffic monitoring specifically for vehicles and traffic patterns.
4. **Behavioural Analysis Detection:** identify dangerous driving behaviours like aggressive lane changing, tailgating or erratic movements.

These are only a few suggestions, however there are many more ways in which it can be improved, or new developments can come about.

Bibliography

Aksoy, S. (2025, January 25). *Object Detection and Tracking using Yolov8 and DeepSORT*. Retrieved from Medium: https://medium.com/@serurays/object-detection-and-tracking-using-yolov8-and-deepsort-47046fc914e9#:~:text=Deep%20SORT%20not%20only%20predicts,or%20partially%20occlude%20each%20other.

Associates, A. &. (2024, September 25). *Long-Term Effects of Car Accidents*. Retrieved from Amourgis: https://www.amourgis.com/blog/long-term-effects-of-car-accidents/

Sojasingarayar, A. (2022, August 29). *Faster R-CNN vs YOLO vs SSD — Object Detection Algorithms*. Retrieved from Medium: https://medium.com/ibm-data-ai/faster-r-cnn-vs-yolo-vs-ssd-object-detection-algorithms-18badb0e02dc

Traffic policing: Reducing accidents, saving lives. (2023, October 2). Retrieved from Lyceum: https://lyceum.edu.za/traffic-policing/traffic-policing-reducing-accidents-saving-lives/#:~:text=For%20example%2C%20speed%20cameras%20are,reducing%20the%20risk%20of%20accidents.

WHO, W. H. (2023, December 13). *Road Traffic Injuries*. Retrieved from Workd Health Organization: https://www.who.int/health-topics/road-safety#tab=tab_1

Appendix

Model Comparison CSV Data

```
===== UNIFIED MODEL COMPARISON FOR VEHICLE SPEED DETECTION =====

Analysis Date: 2025-04-14 13:30:56
Video: IMG_0075.MP4
Resolution: 3840x2160, FPS: 51.18311981914092
Frames Processed: 561
Speed Limit: 55 km/h
Confidence Threshold: 0.4

===== DETECTION PERFORMANCE =====
Model     Detection (s)   Tracking (s)    Total (s)    FPS    Avg Detections
YOLOv8        0.0597       0.0193       0.0791       12.64    0.77
Faster R-CNN  5.4703       0.0229       5.4932       0.18    0.97
SSD           0.3242       0.0190       0.3432       2.91    0.76

===== TRACKING RESULTS =====
Model      Unique Vehicles   Avg Tracked/Frame    Speeders   % Speeders
YOLOv8         14                   0.73              3          21.43%
Faster R-CNN   20                   0.90              4          20.00%
SSD            16                   0.75              7          43.75%

===== SPEED ESTIMATION =====
Model            Avg Speed (km/h)     Max Speed (km/h)
YOLOv8         39.32                66.90
Faster R-CNN   38.38                88.62
SSD            42.29                86.57

===== SPEED VIOLATION DETECTION CAPABILITIES =====
Model  Total Vehicles  Total Speeders % Speeding   Avg Speed When Speeding
YOLOv8         14              3               21.43%         60.02 km/h
Faster R-CNN   20              4               20.00%         64.26 km/h
SSD            16              7               43.75%         62.52 km/h

===== ALERT SYSTEM EFFECTIVENESS =====
Speed readings exceeding the limit:
YOLOv8: 82 out of 393 readings (20.87%)
Faster R-CNN: 87 out of 484 readings (17.98%)
SSD: 118 out of 405 readings (29.14%)

===== DETAILED SPEED VIOLATIONS =====

YOLOv8 Speed Violations:
ID    Speed (km/h)     Timestamp   Frame
9     56.07       00:05       260
11    64.93       00:05       281
19    63.94       00:09       496

Faster R-CNN Speed Violations:
```

```
ID    Speed (km/h)     Timestamp   Frame
20          56.21      00:05       259
22          62.99      00:05       262
37          63.95      00:09       497
36          67.88      00:09       502

SSD Speed Violations:
ID    Speed (km/h)     Timestamp   Frame
1           56.08      00:00       20
2           64.92      00:00       36
4           65.91      00:02       109
8           75.74      00:05       263
7           60.99      00:05       278
15          62.04      00:09       495
16          64.31      00:09       500

Alert System Ranking (based on speeder detection):
SSD > Faster R-CNN > YOLOv8

===== SPEEDER ID LIST =====
YOLOv8 Speeders: 11, 19, 9

Faster R-CNN Speeders: 20, 22, 36, 37

SSD Speeders: 1, 15, 16, 2, 4, 7, 8

===== INTER-MODEL AGREEMENT =====
YOLOv8 and Faster R-CNN agreement: 0 speeders
YOLOv8 and SSD agreement: 0 speeders
Faster R-CNN and SSD agreement: 0 speeders
All models agree on: 0 speeders

===== OVERALL PERFORMANCE SUMMARY =====
Total processing time: 3462.16 seconds
Average processing speed: 0.16 frames/sec

===== CONCLUSION =====
Fastest model: YOLOv8 (12.64 FPS)
Most detected vehicles: Faster R-CNN (20 vehicles)

Recommendation based on results:
If prioritizing speed: YOLOv8
If prioritizing detection accuracy: Faster R-CNN
```