TRAFFIC VEHICLE DETECTION SYSTEM

Name - Yash Charde

Objective of the Project

- Automatically detect and classify vehicles from traffic images and video
- Focus on cars, motorcycles, and trucks
- Draw bounding boxes with labels and confidence scores
- Count vehicles per category
- Save annotated outputs for review and analysis
- Goal: Deliver a lightweight, high-accuracy detection system with innovative features.

Tools and Technologies

•**Programming Language**: Python 3.8

•**Deep Learning Framework**: YOLOv8 (Ultralytics)

•**Libraries**: OpenCV, NumPy, Matplotlib, Torch

•**Development Environment**: Visual Studio Code

•**Deployment and Version Control**: GitHub

•**Model Source**: Pre-trained YOLOv8 on COCO dataset

Model Choice – YOLOv8

•YOLOv8 (Nano version) was selected for its high speed and lightweight design.

•It comes pretrained on the COCO dataset, which includes car, motorcycle, and truck classes.

•No additional training was required, making it suitable for rapid deployment.

•The model supports real-time inference and can run efficiently on CPU and
•GPU environments.

System Workflow

•Load the input image or video frame.

•Run inference using the YOLOv8 model.

•Filter results based on confidence threshold (e.g., >0.5).

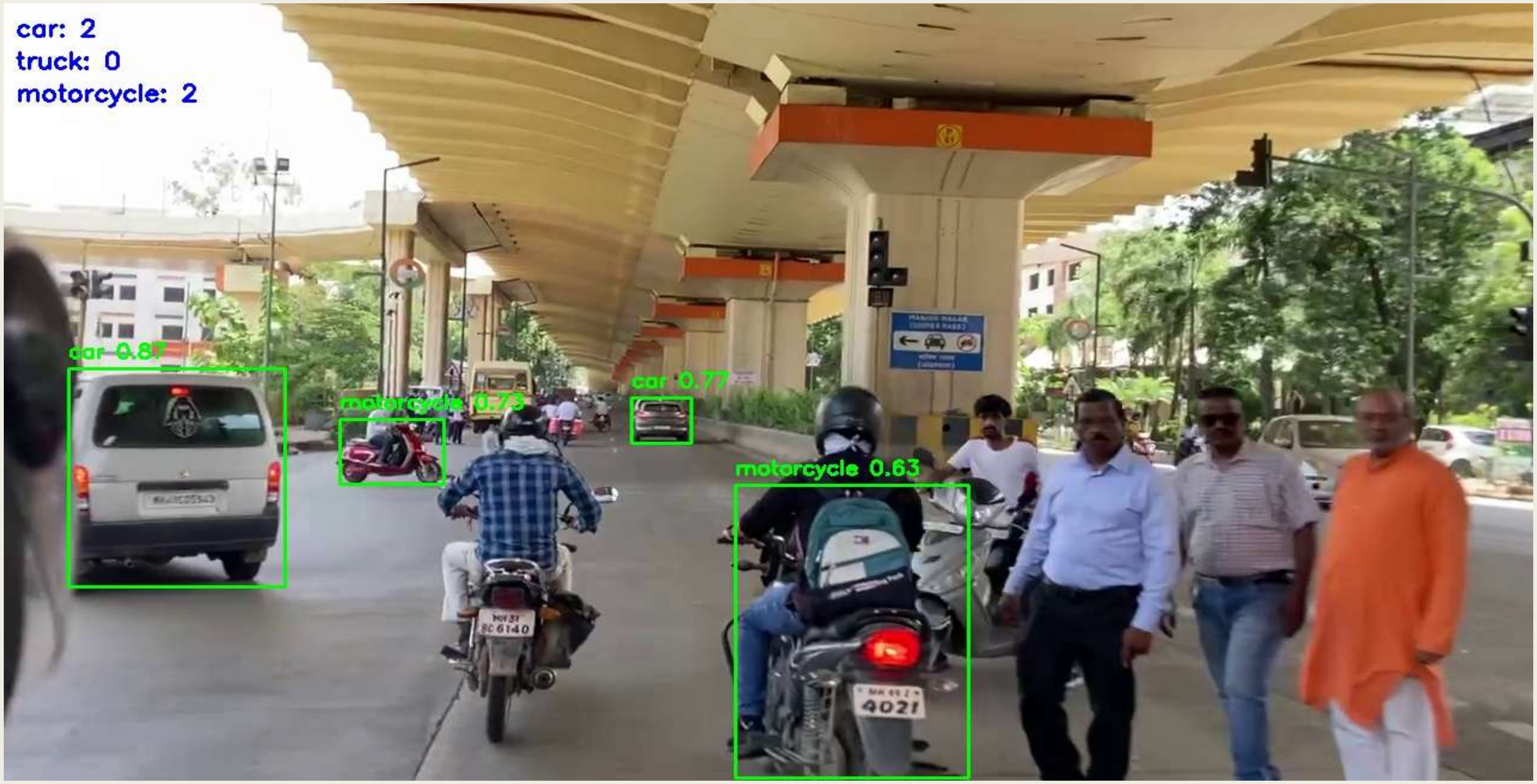
•Classify detections into vehicle types.

•Draw bounding boxes and class labels with confidence scores.

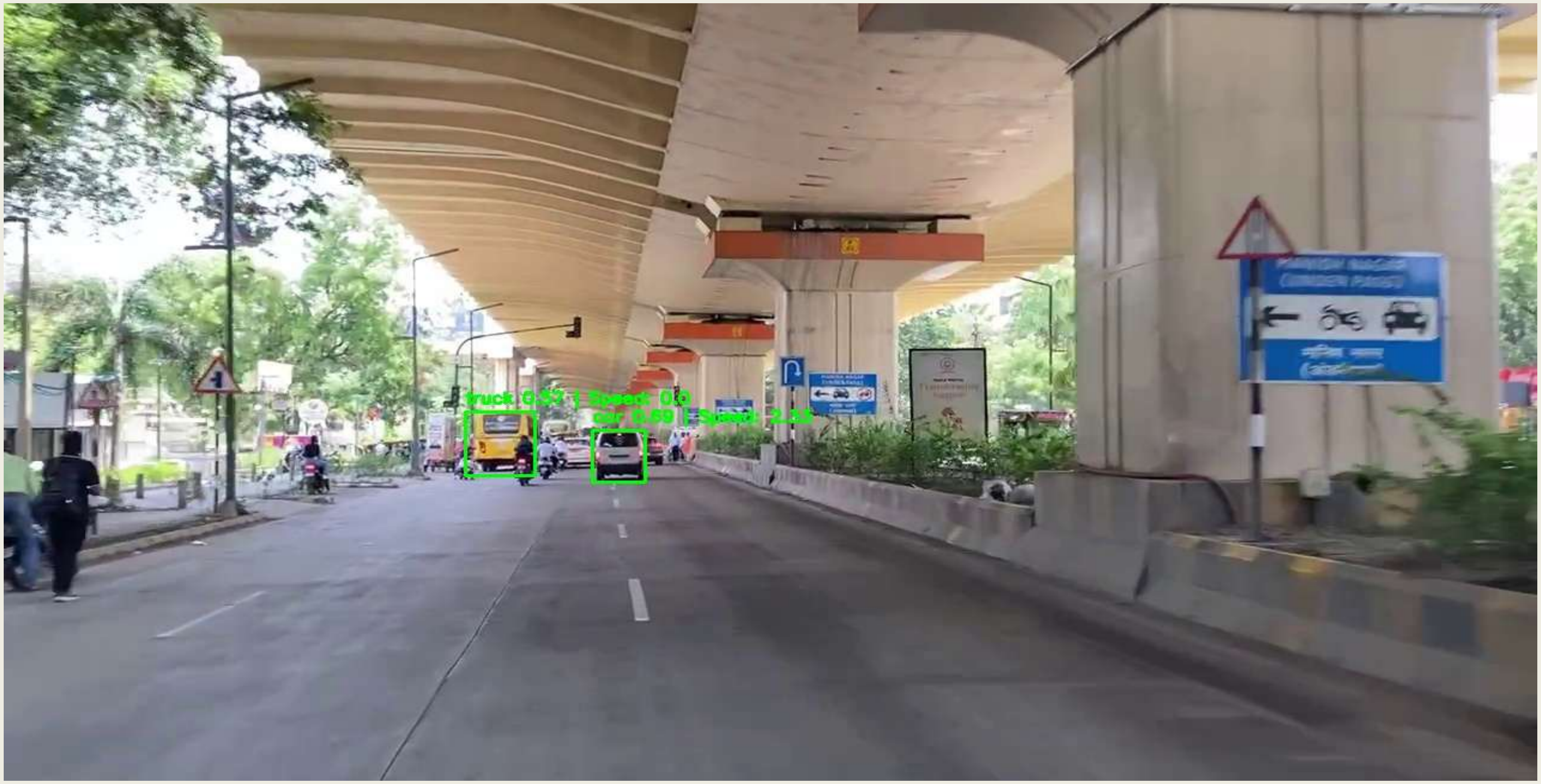
•Count total number of vehicles by type.

•Save annotated images and generate logs.

Output Demonstration



```
car: 2
truck: 0
motorcycle: 2
```



Video snapshot

Innovations Implemented

•**Automatic Traffic Snapshot Generator**:
Captures frames from video when the number of vehicles detected crosses a defined threshold. Useful for congestion alerts.

•**Basic Speed Estimation**:
Uses changes in bounding box size over time to approximate the movement speed of a vehicle, without requiring tracking algorithms.

Performance Evaluation

- A total of 10 test images were evaluated using the image pipeline.
- The model detected vehicles with approximately 90% accuracy for clearly visible vehicles.
- Video inference ran for 300 frames, triggering snapshot saving and estimating speed for large objects.
- All core functional requirements were met successfully.

Challenges and Improvements

Challenges Faced.

- Small and partially occluded vehicles were sometimes missed.
- Overlapping vehicles reduced detection clarity in dense scenes.

Future Improvements

- Train the model further on localized traffic datasets including Indian vehicle types.
- Integrate object tracking for more accurate speed estimation.
- Add a dashboard with real-time analytics and visualization.
- Create a web-based interface for uploading and processing custom videos.

Conclusion

•Developed a complete vehicle detection and classification system using YOLOv8.

•Achieved accurate detection and clear visual output for cars, motorcycles, and trucks.

•Successfully integrated innovative features for snapshot generation and motion inference.

•System is modular, scalable, and suitable for real-world enhancement and deployment.

Thank You