P3: Einstein Vision(Phase 3) (Using 6 late days)

1st Venkateshkrishna
Masters in Robotics
Worcester Polytechnic Institute
Worcester, MA 01609
vparsuram@wpi.edu

2nd Mayank, Bansal
Masters in Robotics
Worcester Polytechnic Institute
Worcester, MA 01609
mbansal1@wpi.edu

***Abstract*—In this project, we develop 3D visualization of the surroundings of an autonomous car, inspired by the Tesla dashboard. The goal is to use different classical and deep learning methods to identify various objects on the road and render them in Blender to create a real-time 3D visualization.**

I. Introduction

The advancement of autonomous vehicles stands as a monumental leap in technology in recent years. As the presence of these vehicles grows, the necessity for a user-friendly dashboard for both drivers and passengers becomes paramount. Such a dashboard must offer clear and comprehensible insights into the vehicle's environment and its status. This project focuses on Tesla's dashboard HMI (Human-Machine Interface), crafted to ensure a smooth and intuitive user experience. This allows individuals to interact with the car's features and functionalities seamlessly while on the move. Utilizing video footage from a Tesla vehicle, the project's objective was to create visual representations of these sequences using Blender.

II. Dataset

The dataset includes:

1) Thirteen video sequences recorded in diverse environmental settings, provided in both their unaltered raw format and versions that have been corrected for distortion using calibration processes.
2) Videos intended for the calibration of cameras.
3) Blender assets for a range of entities such as vehicles (e.g., sedans, SUVs, pickup trucks, bicycles, motorcycles, and trucks) and roadside infrastructure (traffic signals, stop signs, traffic cones, traffic poles, speed signs, and pedestrian models). Texture images for the stop sign and a blank template for the speed sign are also included.

III. Perception Stack

A. Object Detection

YOLO (You Only Look Once) is a renowned object detection framework, noted for its swift and accurate performance. Originally introduced by Joseph Redmon et al. in 2016, it has progressed through several updates, culminating in the latest version, YOLO v8. Utilizing a single-shot mechanism, YOLO employs an extensive convolutional neural network (CNN) to analyze an image in a singular operation, facilitating

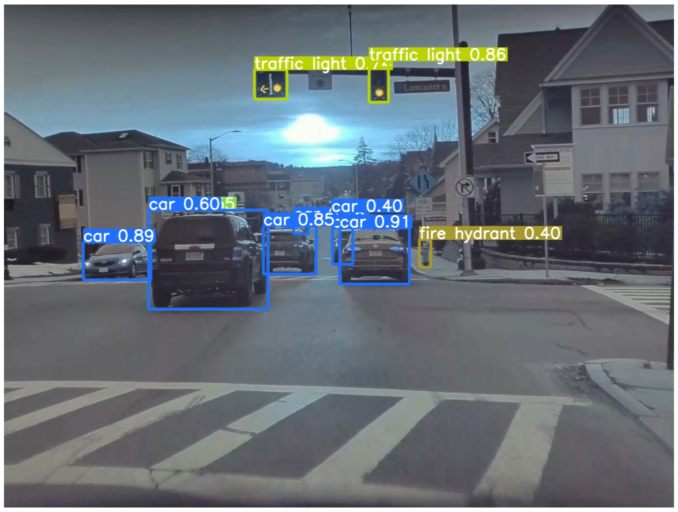

Fig. 1: YOLO v8 object detection

the simultaneous prediction of bounding boxes and class probabilities through an end-to-end network design.

For our project, we employed YOLO v8, which is adept at detecting and classifying objects, in addition to performing semantic segmentation on the identified items. Capable of detecting up to 80 different classes including cars, trucks, bicycles, traffic signals, stop signs, and pedestrians, YOLO v8 outlines these objects with 2D bounding boxes. The coordinates of these boxes were extracted using YOLO v8 and recorded in a '.csv' file. The coordinates obtained were crucial for determining the relative depth of detected objects, and for the instantiation of objects within Blender.

B. Monocular Depth Estimation

Human depth perception, informed by cues like perspective and shading, is emulated in computational models by depth maps from the MIDAS model. MIDAS ("Million Depth Annotations Samples") utilizes a ResNeXt-based deep learning architecture, trained on a vast dataset of RGB-D image pairs, to estimate depth from visual features.

MIDAS produces a depth map with pixel shades indicating distance from the camera—darker shades signify closer proximity, lighter shades indicate distance. By analyzing the average depth within an object's area, identified via the YOLO

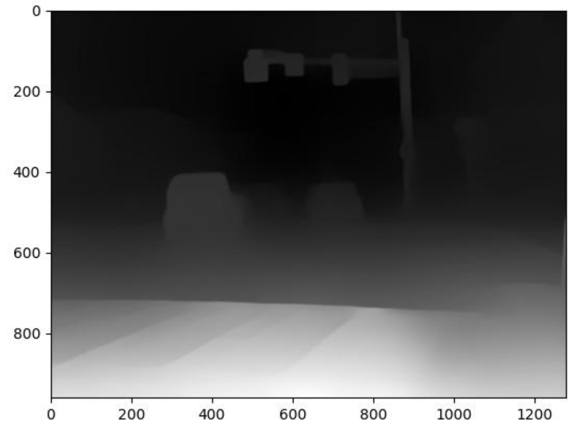

Fig. 2: MiDaS depth estimation

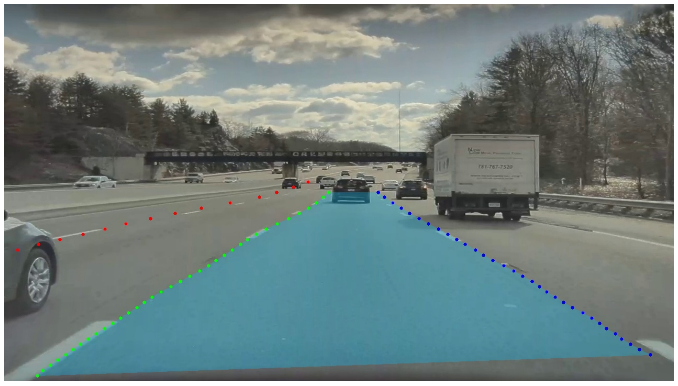

Fig. 3: Ultrafast lane detection

v8 network, against the image's deepest point, we achieve a measure of relative depth.

This approach facilitates the extraction of spatial information from scene objects, thus improving analysis of visual content.

C. Lane Detection

In lane detection tasks, we employ a model called Ultrafast Lane Detection to identify the 2D coordinates of lane boundaries. Following the detection, we conduct post-processing to render these points as continuous lines, enhancing the visual representation.

D. 3D vehicle pose estimation

Determining the precise position and orientation (pose) of vehicles is a crucial task for autonomous driving systems. To achieve accurate 3D pose estimation, we employed the YOLO 3D model. Initially, the results were not optimal, leading us to make several enhancements. The process starts with employing the YOLOv5 model, particularly its YOLOv5x variant, for its high precision in detecting 2D bounding boxes around vehicles, despite its slower performance. This step was critical in improving the quality of 3D pose estimations. Moreover, we

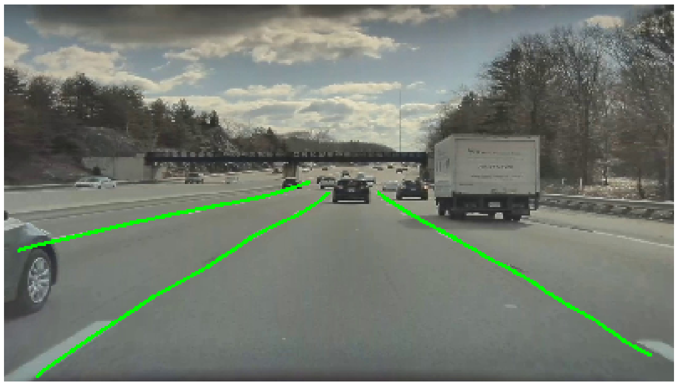

Fig. 4: Post-processed lane output

found that resizing images to 640x640 pixels before processing significantly boosted the model's accuracy. This size matches the YOLOv5's optimal input dimension for 2D detection, ensuring the process is finely tuned for the best results.

Following the detection of 2D bounding boxes, these details are fed into a subsequent network within the YOLO 3D framework to translate them into 3D space, generating 3D bounding boxes. An essential aspect of this stage is focusing on the vehicle's yaw angle, identified by an angle called alpha, which indicates the vehicle's rotation around the vertical axis. Considering that vehicles on roads typically maintain zero pitch and roll angles simplifies the model without compromising on accuracy. Through these strategic adjustments, we significantly enhanced the YOLO 3D model's ability to estimate vehicle poses, marking a substantial improvement for the navigation and safety mechanisms of autonomous vehicles. We can see in Fig. 5 that the blue surface of the detected box shows the front of the vehicle.

E. Pedestrian pose estimation

For an autonomous vehicle to navigate safely, it's crucial to be aware of the location and actions of pedestrians in its vicinity. This requires understanding the pedestrians' poses to anticipate their next moves. Initially, we utilized the OpenPose model for pedestrian pose detection, which yielded satisfactory results. However, we transitioned to a more advanced and recent solution, the YOLOv8 pose estimation model, which enhanced our ability to accurately estimate pedestrian poses within the scene. Although we successfully integrated this model, we were unable to visualize the pedestrian poses in Blender due to the project's time limitations. Addressing this and improving the visualization of pedestrian poses in Blender remains a key area for future development in our project.

IV. Visualisation in Blender

Blender plays a pivotal role in the domain of 3D creation and animation, serving as an indispensable resource for both personal endeavors and professional assignments. It boasts a comprehensive suite of features and tools, complemented by a vibrant community of enthusiasts and experts. For our project, Blender's robust capabilities and widespread adoption

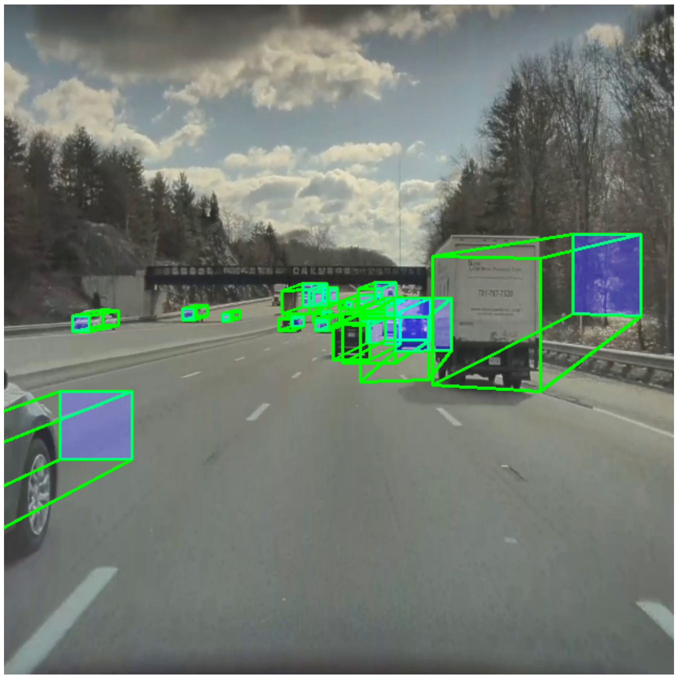
Fig. 5: YOLO 3D bounding box estimation

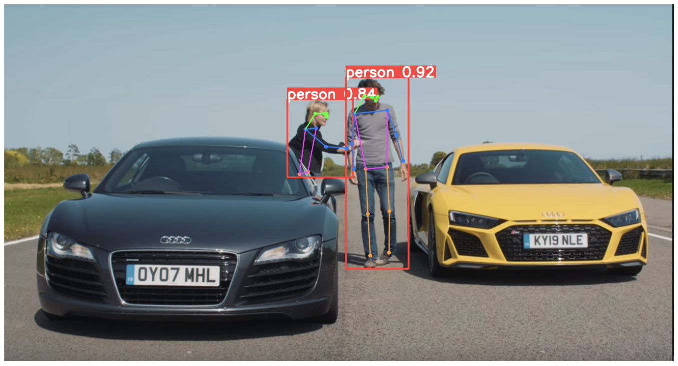

Fig. 6: YOLO v8 human pose estimation

make it an ideal platform for crafting sophisticated 3D content, enabling us to visualize complex scenarios and interactions in autonomous vehicle simulations with precision and creativity.

A. *Rendering objects*

The YOLO v8 model outputs the coordinates of bounding boxes, identifying the outermost x and y points. We utilize these coordinates in Blender to determine the central position of each detected object by averaging these extremal points. In Blender's spatial framework, the X and Z axes are used to map horizontal and vertical positions, respectively. The coordinates from the YOLOv8 model are translated into Blender's world coordinates through the application of camera intrinsic and extrinsic parameters. Additionally, depth information sourced from the MiDaS model assists in accurately positioning objects closer or further from the camera within the Blender environment.

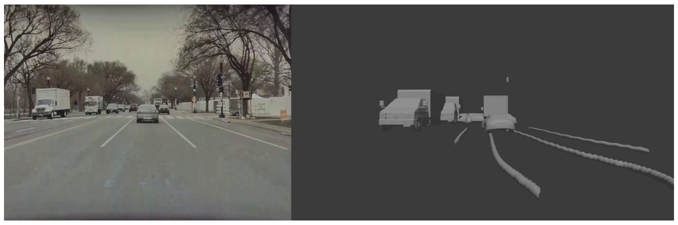
Fig. 8: Comparison of real image and its render in Blender

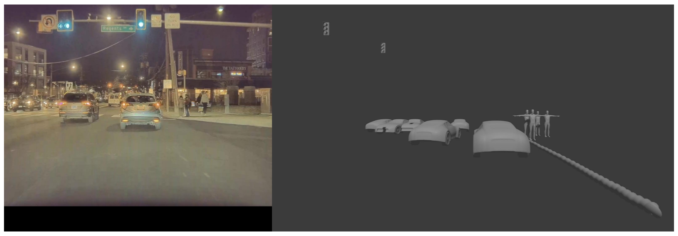
Fig. 9: Comparison of real image and its render in Blender

B. *Lanes*

The 2D coordinates derived from the Ultrafast Lane Detection model are transformed into 3D world coordinates, which are then used to fit a Bezier curve to these points. This process allows us to accurately depict the lanes in Blender with smooth, continuous curves that reflect real-world lane geometry.

C. *3D pose of vehicles*

The orientation of vehicles, specifically their yaw angles, is derived from the YOLO3D model. These angles are then applied to adjust the positioning of the vehicles within the rendered scene.

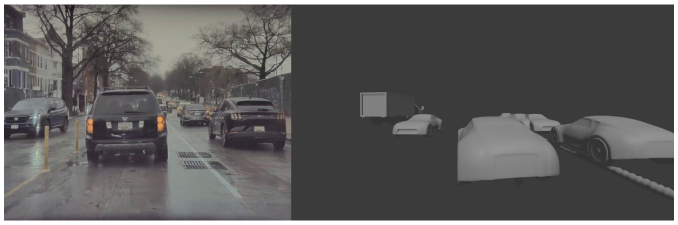
Fig. 7: Comparison of real image and its render in Blender

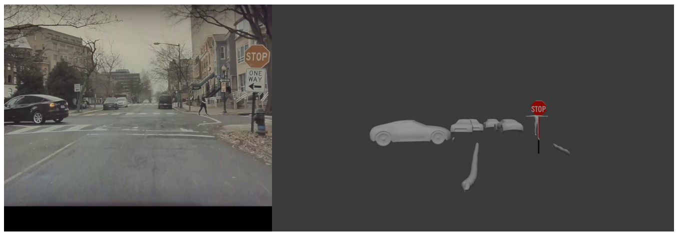
Fig. 10: Comparison of real image and its render in Blender

References

[1] Einstein Vision home page: link
[2] YOLOv8 human pose estimation: link
[3] YOLOv8 object detection: link
[4] MiDaS depth estimation: link
[5] YOLO3D: link
[6] Ultrafast lane detection: link