Project 3: Einstein Vision

Ankit Mittal
Department of Robotics Engineering
Worcester Polytechnic Institute
Email: amittal@wpi.edu

Rutwik Kulkarni
Department of Robotics Engineering
Worcester Polytechnic Institute
Email: rkulkarni1@wpi.edu

***Abstract*—(utilizing 1 late day) This project focuses on enhancing Human-Robot Interaction (HRI) through improved visualization techniques, inspired by Tesla's dashboard technology. By processing videos from a 2023 Tesla Model S, we aim to create an intuitive visualization interface that displays both the vehicle's immediate view and its surroundings. This interface will offer users insights into the vehicle's perception and decision-making processes, improving user experience and trust in autonomous vehicles.**

I. Introduction

The importance of visualizations in technology, particularly for autonomous machines, cannot be overstated. The first interaction between a human and a machine is crucial for building trust. As smartphone UIs have evolved to become more user-friendly, so must the interfaces of autonomous vehicles and robotics. Visualization tools like rviz have been essential for developing robotic systems by graphically representing sensory data. However, they often fail to effectively address Human-Robot Interaction (HRI) for the average user, focusing too much on technical details.

Tesla's recent dashboard designs have marked a significant step forward by offering intuitive and informative visualizations, enhancing the communication between humans and machines. Yet, there is still room for improvement in making these visualizations more accessible to all users. This project aims to advance Tesla's innovations by creating a visualization interface that not only displays the car's immediate view but also provides a comprehensive overview of its environment. This approach seeks to improve user understanding of how the vehicle interacts with its surroundings, addressing key HRI challenges.

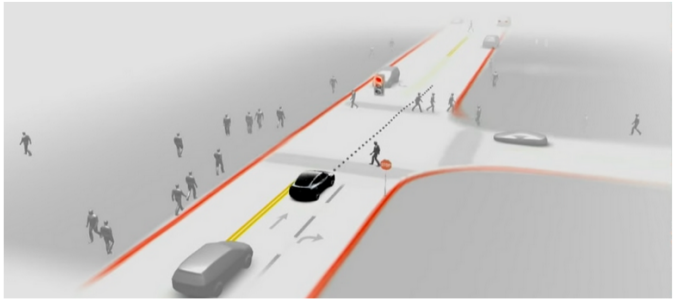

Fig. 1: Tesla's Dashboard

II. Outline of the Paper

1) **Obtaining Depth Maps:**
2) **Object Detection and Projection in 3D space:**
3) **Orientation of Vehicles w.r.t. Camera:**
4) **Lane Detection and Road Signs:**
5) **Human Pose Detection:**
6) **Limitations of the Approach:**
7) **Results:**

III. Obtaining Depth Maps

A pivotal element in constructing an intuitive and informative visualization for autonomous vehicle systems is the accurate rendering of the vehicle's surroundings in three dimensions. This necessitates a reliable method of obtaining depth information from the vehicle's surroundings. To accomplish this, we utilized the ZoeDepth repository (https://github.com/isl-org/ZoeDepth), which offers significant advancements in depth perception technology.

ZoeDepth is notable for its ability to produce metric depth maps in a single shot, a substantial improvement over its predecessor, MiDAS[1], which only provided relative depth information. This advancement is crucial for our project as metric depth maps allow for a more accurate and spatially correct rendering of the environment around the vehicle. The repository achieves this by building upon the foundational MiDAS technology, integrating additional layers of sophistication to produce metric depth values.

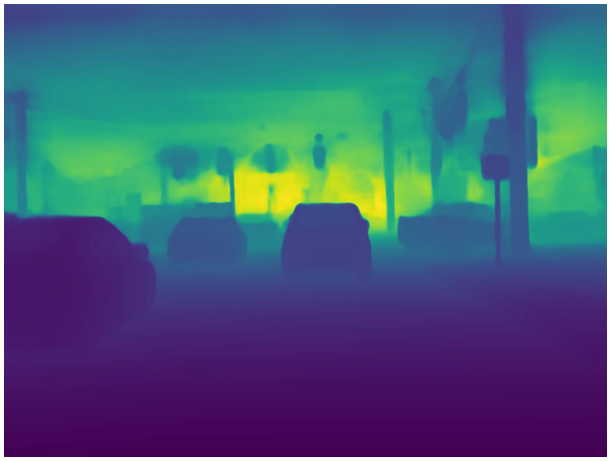

Fig. 2: Depth Map

Despite the advancements ZoeDepth[2] provides, it is important to acknowledge the inherent inaccuracies in the metric depth values it generates. These inaccuracies stem from the complex nature of depth estimation algorithms and the diverse environments and conditions under which they operate. To address this challenge, we adopted a strategy of manually scaling the depth values through a process of trial and error. This approach was taken to ensure that the depth maps accurately represent the real-world distances for each specific sequence of video data, thereby enhancing the fidelity of our visualizations.

IV. General Approach and Object Detection

For the task of object detection and instance segmentation within the visual field of the autonomous vehicle, we selected the Detic[3] repository for its comprehensive capabilities. Detic distinguishes itself by its ability to perform object detection across more than 21,000 classes, offering unparalleled versatility for our application.

Our methodology encompassed several key steps:

A. Instance Segmentation

Firstly, we performed instance segmentation on a specific set of object classes that are relevant to the project's context, such as sedans, hatchbacks, trucks, and pickup trucks. This process was essential for accurately identifying and segmenting each object within the vehicle's field of view from the video data.

B. Centroid Calculation

Upon successfully segmenting the objects, the next step involved calculating the centroid of the segmentation mask for each detected object instance across all frames. This calculation was critical for determining the precise location of each object in the visual space of the video data.

C. Depth Value and 3D Projection

After identifying the centroids of object instances within the segmentation masks, our next objective was to compute the average depth value for each segmented object. This computation is integral to our methodology as it involves the use of camera intrinsic parameters (such as focal length and optical center) and the predetermined camera extrinsics, which include its position and orientation relative to a known reference frame. For our setup, the camera is posited to be 1.5 meters above the origin in a coordinate system aligned with that used in Blender, a 3D modeling and rendering software.

1) Depth Calculation: The average depth value for each object's segmentation mask is determined by integrating depth information obtained from the depth maps with the segmented areas. This depth is not absolute but is scaled based on empirical adjustments to match the specific sequence of video data being analyzed, providing a more accurate depiction of each object's distance from the camera.

Fig. 3: Centroid and Depth overlay on the original image

2) Projection Equation: To project the 2D centroids of the objects into 3D space, we utilize the projection equation which relates 3D world coordinates to 2D image coordinates through the camera's intrinsic and extrinsic parameters. The basic form of the projection equation can be expressed as:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{Z} K [R|t] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

Where:

- (u, v) are the coordinates of the point in the image plane.
- K represents the camera intrinsic matrix containing focal lengths and optical center.
- $[R|t]$ denotes the camera extrinsic parameters, with R being the rotation matrix and t the translation vector, defining the camera's orientation and position in the world.
- (X, Y, Z) are the 3D world coordinates of the point.
- Z is the depth of the point in the world coordinate system, acting as a scaling factor to ensure the correct projection into 2D space.

Fig. 4: Original

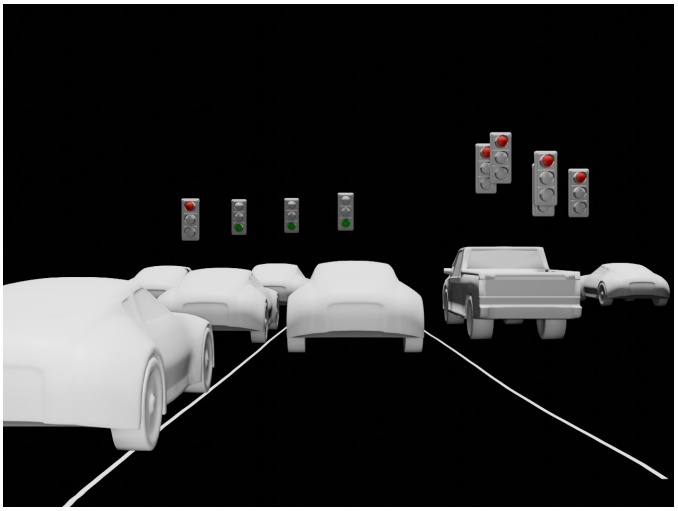

Fig. 5: After Plotting in Blender

Using the depth values obtained from the depth maps and the segmentation masks, we can rearrange this equation to solve for the 3D coordinates (X, Y, Z) of each object's centroid. This transformation allows us to accurately place each object within the 3D space of Blender, based on its observed position in the 2D image plane.

3) Advantages of Instance Segmentation: Choosing instance segmentation over simpler bounding box detection offers a significant improvement in accuracy for this projection process. Instance segmentation precisely delineates the contours of each object, allowing for a more exact calculation of centroids and depth values. This method enhances the visual quality of the rendered environment and provides users with a more detailed and intuitive understanding of the autonomous vehicle's perception and interaction with its surroundings.

V. Orientation of Vehicles with Respect to Camera

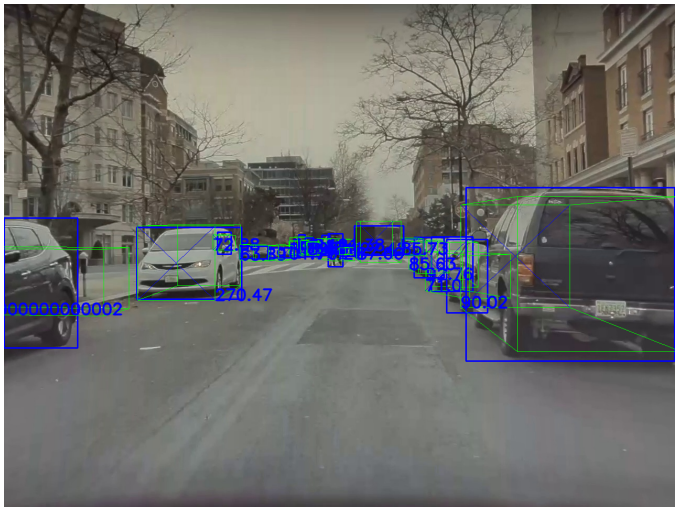

Fig. 6: Orientation Detection

Achieving an accurate representation of vehicles in a 3D visualization requires precise determination of their orientation relative to the camera's perspective. This task is complicated due to the dynamic nature of the scenes captured in the autonomous vehicle's cameras. To address this challenge, we incorporated the 3D Bounding Box repository (https://github.com/skhadem/3D-BoundingBox), an advanced tool built on a PyTorch implementation, improving upon the original YOLO3D methodology. This tool is particularly adept at providing orientation information for a wide range of vehicle types, including cars, trucks, pickup trucks, bicycles, and pedestrians, which are essential for creating a realistic traffic scene in our visualization. Now that the orientation convention in the above image is measured such that the vehicles aligned with our vehicle are 90 degrees and opposite to us are 270 or -90 degrees. Also note that to smoothen our simulations we have done binning and divided into quadrants such that yaw rounds of to the closest axis, i.e 12 degrees will round off to 0 degrees when spawning in blender.

- The repository excels in calculating the yaw angle of vehicles, which is essential for positioning them correctly within the 3D scene. The yaw angle determines the direction the vehicle is facing relative to the camera, allowing for a realistic portrayal of traffic conditions and interactions.
- Despite its effectiveness, the tool has limitations, particularly in generating false positives for objects at intermediate or far distances. This challenge is mitigated by refining the detection parameters and focusing on optimizing the accuracy for objects closer to the camera, where the orientation information is most critical for the visualization's realism.

VI. Human Pose Detection

The representation of human figures and their poses presents another layer of complexity in 3D visualizations. To achieve lifelike human poses, we leveraged the OSX repository[4], which has shown promising results in human pose estimation. This tool outperformed other frameworks we tested, including PyMAF, HybrIK, and NIKI, particularly in scenarios where the human figures were small or ambiguously presented in the camera's view.

- OSX's integration with Grounded sam for human pose detection marks a significant advancement in accurately capturing human postures and movements. This capability is vital for adding a dynamic and realistic element to the human figures within our 3D scenes, ensuring they interact believably with the environment and the vehicles.
- While the OSX model demonstrates improved accuracy and a reduction in false positives compared to other models, it is not without its challenges. The detection system occasionally misinterprets non-human elements in the scene as humans, though at a significantly reduced rate compared to alternatives. Ongoing adjustments and fine-tuning of the model parameters are conducted to further minimize these occurrences.

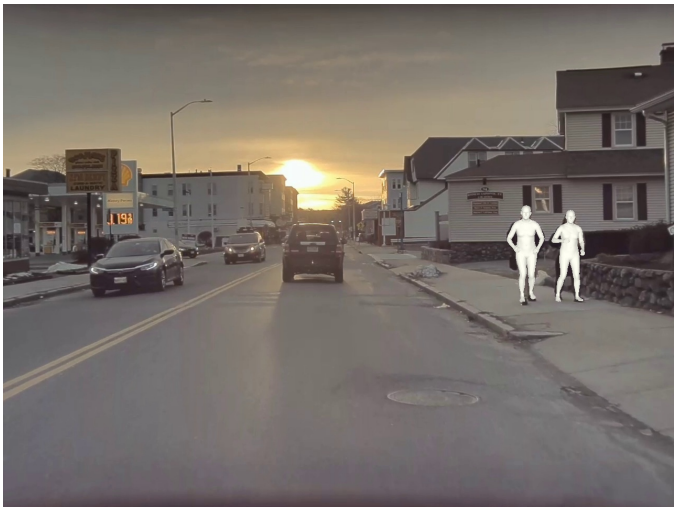

Fig. 7: Human Detection using OSX

VII. Lane Detection and Plotting

Lane Segmentation into different categories was done using https://debuggercafe.com/lane-detection-using-mask-rcnn/. This model provided us with the mask of different lanes on the road, after getting the mask we fitted a curve in each blob to get the lane. for extracting the road same procedure was used. The above model provides a different label for the road marks.

Earlier we also tried the CLRnet model to extract the lane however it cannot classify between the different types of lanes and cannot identify the road, therefore we switched to this model.

Fig. 8: Different types of lanes

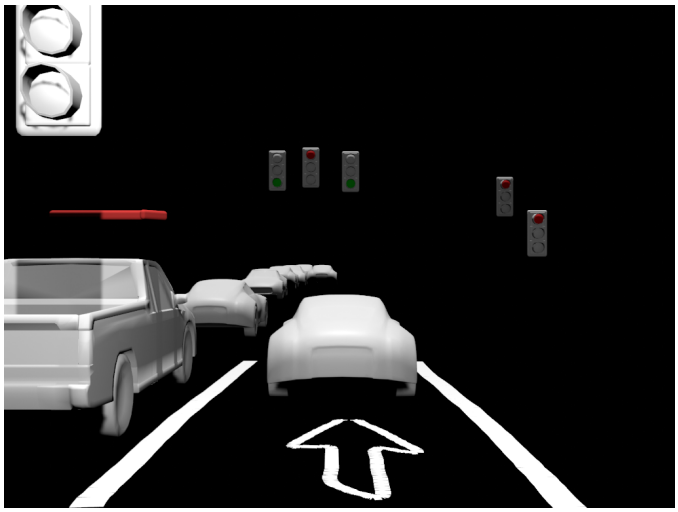

Fig. 9: Road Marks

VIII. Distinction between parked and moving vehicles

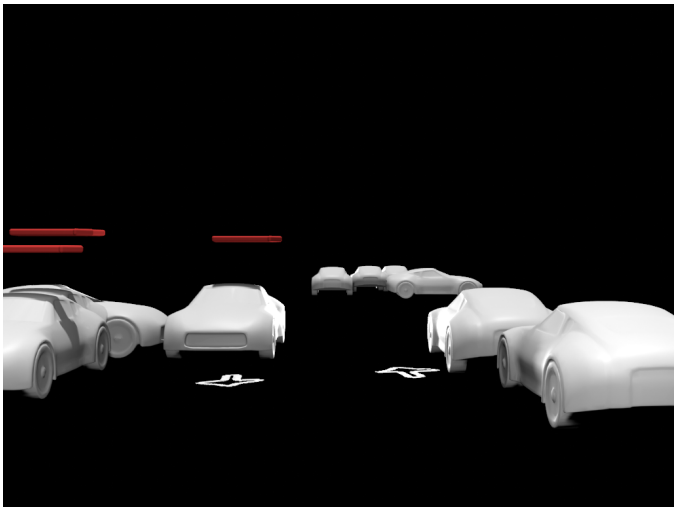

Fig. 10: orientation with arrows

IX. Results

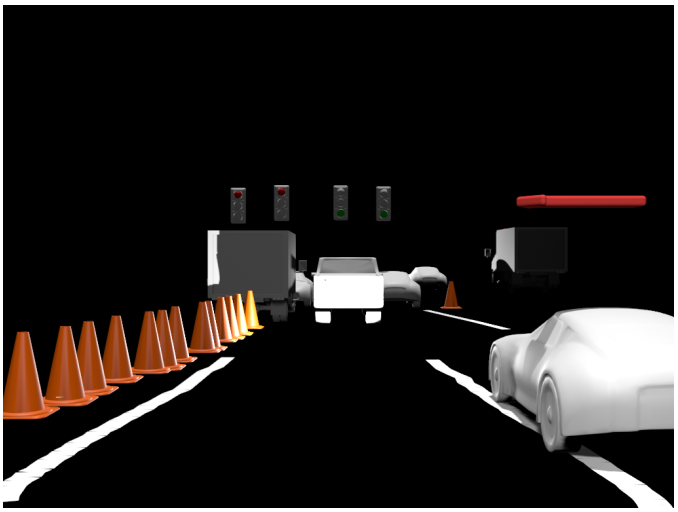

Fig. 11: result1

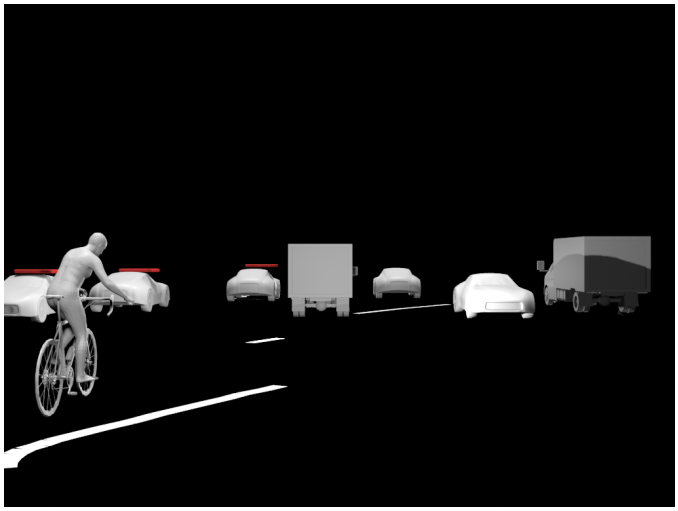

Fig. 12: result2

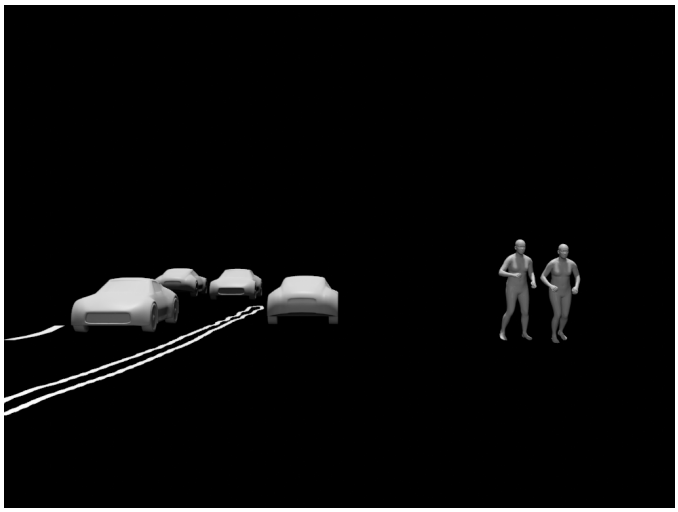

Fig. 13: result3

X. Acknowledgment

The author would like to thank Prof. Nitin Sanket, Teaching Assistant, and Grader of this course RBE549- Computer Vision.

References

[1] R. Birkl, D. Wofk, and M. Müller, "Midas v3.1 – a model zoo for robust monocular relative depth estimation," *arXiv preprint arXiv:2307.14460*, 2023.

[2] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," 2023. [Online]. Available: https://arxiv.org/abs/2302.12288

[3] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra, "Detecting twenty-thousand classes using image-level supervision," in *ECCV*, 2022.

[4] J. Lin, A. Zeng, H. Wang, L. Zhang, and Y. Li, "One-stage 3d whole-body mesh recovery with component aware transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 159–21 168.