3D Traffic Scenes Reconstruction for Autonomous Vehicles using Gaussian Process Latent Variable Model (GPLVM)

Bryan Amirul Husna
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
bryanahusna@gmail.com

Dr. Ir. Rinaldi Munir, M.T.
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
rinaldi@staff.stei.itb.ac.id

***Abstract*—Traffic visualization in autonomous vehicles is important to improve the passengers' sense of safety. This paper presents a workflow and implementation that can reconstruct traffic scenes using only a single image from a single monocular camera installed in a vehicle. The reconstruction process is also applied between frames utilizing Simple Online and Realtime Tracking (SORT) framework to improve vehicle movement smoothness. Vehicle shape reconstruction is carried out using gaussian process latent variable model (GPLVM) to embed 3D model shapes to latent variable space. Multisegmented Hough transform is used to detect lane marking resulting in line equation which approximate the lane's shape. Finally, both the vehicle and road shape information are used to visualize the traffic scenes, although real-time performance is not achieved yet.**

***Keywords*—traffic scenes reconstruction; autonomous vehicle; gplvm; SORT; GPLVM;**

I. Introduction

Autonomous vehicle (AV) is a term for vehicle equipped with driving automation system [1]. According to National Highway Traffic Association, most traffic accidents are caused by human error [2]. Therefore, one of potential benefits of autonomous vehicles is reduced number of traffic accidents [3]. However, many people are still afraid of using autonomous vehicles [4] so it will potentially prevent the widespread usage of autonomous vehicles.

Autonomous vehicles need to understand their surroundings to be able to make decisions. That understanding is not only important for themselves, but also for the passengers riding them. Passengers feel safer inside autonomous vehicles that can visualize their surroundings [4]. Therefore, traffic scenes visualization is an important aspect of autonomous vehicles.

In this paper, a three-dimensional traffic scenes reconstruction from single monocular camera workflow and implementation is presented. The contribution of this paper is modular architecture that integrates vehicles and road reconstruction to form a 3D visualization. The implementation is suitable for autonomous vehicles since it only requires single dashboard-view monocular for reconstruction, although real-time processing time is not achieved yet.

II. Literature Review

Tesla car manufacturer is well known for its electric autonomous driving car and its traffic visualization. One of its car models, Tesla Y [5], utilizes six cameras and sensors to sense its surroundings. It then visualizes the result to the passengers, being able to detect other vehicles, traffic lights, pedestrians, etc. However, since it is a proprietary system, the working mechanism is not publicly available.

Another system that can visualize traffic scenes is in [6]. It uses several monocular surveillance cameras from various angles to reconstruct vehicle shapes and the traffic map. The system consists of three subsystems: tracking, reconstruction, and replay. Vehicle shape reconstruction is carried out using shape-from-silhouette and 3D CAD model fitting. However, it is not suitable for autonomous vehicles case since it requires cameras from various angles.

One research that is suitable for autonomous vehicles is in [7]. It only uses a single monocular camera from inside a vehicle. A vehicle in an image is segmented between foreground and background and is estimated its orientation angle using two convolutional neural networks (CNN). The results are then reconstructed using gaussian process latent variable model (GPLVM). However, it only reconstructs the shape, position, and orientation of the vehicles; other aspects of the traffic are not reconstructed. The traffic reconstruction in this paper is inspired by this approach, but with an additional aspect of road reconstruction.

III. Problem and Solution Analysis

Two of many aspects important in traffic scene understanding are surrounding vehicles and surrounding road. By knowing the position of vehicles around ego vehicle, the ego vehicle can avoid potential crash with another vehicle. By knowing the shape of the road, the ego vehicle can detect and react to lane change and know where it can drive (drivable area).

The proposed architecture is shown in Fig. 1. There are two parallel flows that will finally be merged to compose final visualization: surrounding vehicles understanding and surrounding road understanding. First, vehicles are detected using an object detector. Various object detectors can be used, but faster inference time is preferred since time is critical for reconstruction in an autonomous vehicle. In this paper, YOLOv9 [8] is used. Since YOLOv9 detects objects of various classes, only objects with “car” class are processed further.

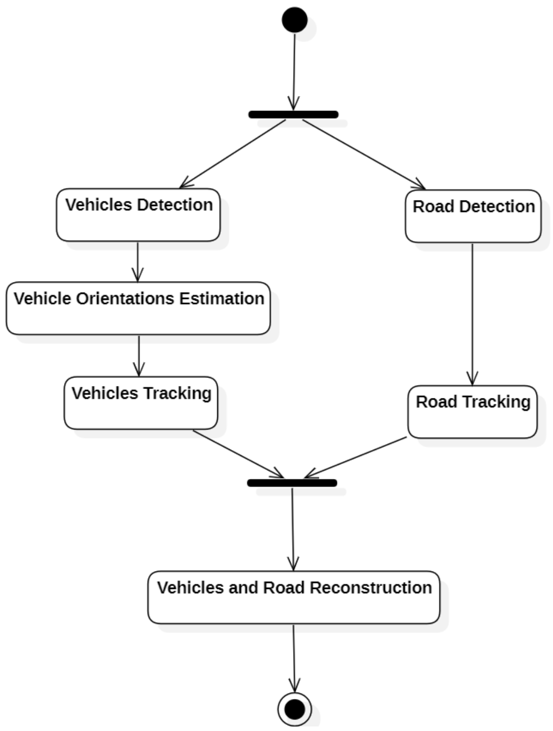
Vehicles Detection
Road Detection
Vehicle Orientations Estimation
Vehicles Tracking
Road Tracking
Vehicles and Road Reconstruction

Fig. 1. Traffic scenes reconstruction activity diagram

Then, each detected vehicle’s orientation angle is estimated using a deep learning model. 3D Deepbox [9] is used as vehicle orientation estimator. Since generally vehicles on a road will have approximately zero roll and pitch, only yaw angle (rotation around y axis) is estimated. This stage adds additional *yaw* data to the bounding box of each vehicle produced by YOLOv9, from *(x, y, w, h)* to *(x, y, w, h yaw)*, where *x* and *y* is the horizontal and vertical position of the vehicle from top left corner of an image, and *w* and *h* is the width and height of the vehicle’s bounding box.

Bounding boxes are in 2D, so depth information needs to be estimated for 3D reconstruction to work. The depth is estimated using a pinhole camera model. A vehicle with height *s* and at distance *d* will be detected in image as a bounding box with *h* height. Given a camera constant z_c, a vehicle’s distance from camera can be estimated using equation (1). Note that *s* can be estimated as the average height of cars, for example 1.6 meter.

$$d = s \times z_c \times \frac{1}{h} \qquad (1)$$

Vehicle detections between frames are not associated yet, so a tracking algorithm is applied. For tracking, SORT [10] tracking framework is used, which is based on Kalman Filter and Hungarian algorithm. Each bounding box detected in a frame is associated with a detection in previous frame, or a new tracked vehicle is added if no association exists for that detection. Constant velocity model is adopted and the state vector for 3D tracking is in equation (2), where *scale* and *ratio* is the area and aspect ratio of the bounding box respectively, and *v* is the speed of each component.

$$\boldsymbol{x} = [x \;\; y \;\; z \;\; yaw \;\; scale \;\; ratio \;\; v_x \, v_y \;\; v_z \;\; v_{yaw} \;\; v_{scale}] \qquad (2)$$

Road understanding consists of three steps: preprocessing, line detection using multi-segment Hough transform, and tracking. Road is detected based on its marking. In the preprocessing step, the traffic image is cropped so that only the region of interest (ROI) remains. The image is then transformed to bird’s eye view (BEV) using projective transformation (homography) to remove the perspective effect (i.e. farther an object, smaller the size). The image is further converted to HSV to extract the road marking based on its color (yellow or white). Finally, the edges of the image are detected using Canny edge detection operator, resulting in binary image (1 if a pixel is part of edge, 0 if not).

After the image has been processed, straight lines in the processed binary image are detected using multi-segment Hough transform to detect road lanes. Since the shape of the road may curve (not perfectly straight), ordinary line detection is not sufficient. Therefore, the image is divided into horizontal segments (Fig. 2) to approximate curved road. A Hough transform is applied to each of these segments to detect lines. Finally, lines located in proximity are grouped together to remove line noises and to connect lines between two adjacent segments. Lanes are represented by their endpoint position on each segment.

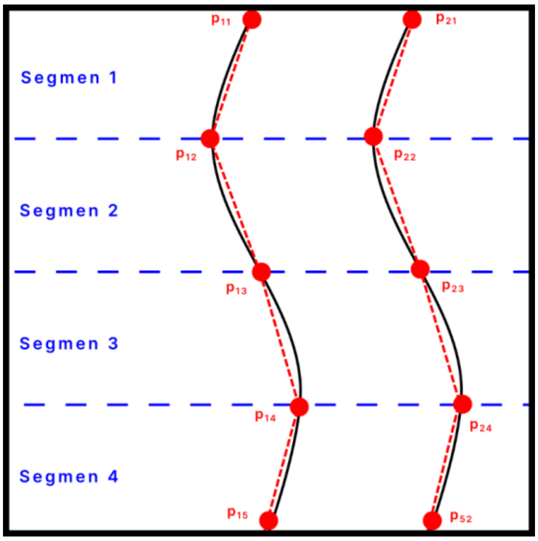

Fig. 2. Multi-segment hough transform

The road lanes are then tracked using SORT [10] tracking framework. Just like vehicle tracking, a constant velocity model is used. The state vector is defined in equation (3). Each p_i denote the endpoint position on segment *i* and v_{pi} denote the corresponding velocity. Lanes between frames are associated based on their endpoints’ proximity.

$$\boldsymbol{x} = [p_1 \quad \cdots \quad p_n \quad v_{p1} \quad \cdots \quad v_{pn}] \tag{3}$$

Road lanes can be simply reconstructed based on the coordinates of its endpoints, while vehicle shapes need a more sophisticated reconstruction method. A good shape reconstruction is one which combines both visual information and shape priors. Therefore, gaussian process latent variable model (GPLVM) based shape reconstruction is used in this paper.

GPLVM is a dimensionality reduction with nonlinear property. The idea behind reconstruction using it is to embed high dimensional 3D shapes to low dimensional latent space. Therefore, to find the shape of a vehicle, only search in low dimensional latent space is needed. In [11], 3D shape reconstruction is carried out by converting training 3D shapes to signed distance function (SDF) first. Then, 3D DCT (discrete cosine transform) to compress the SDF is applied before GPLVM training is carried out. The original shape can be approximately reconstructed by doing the inverse of these transformations from latent variable, i.e. applying inverse of GPLVM to a latent coordinate, then inverse of 3D DCT, and finally marching cube algorithm to form the mesh. This approach is used in this paper.

Shape reconstruction is an optimization process to find the best latent variable that matches the input image. First, the input image is segmented between foreground vehicle and background. Then, grid search is carried out in this paper to find the best latent variable which when reconstructed, the resulting projection best matches the segmented input image. After the vehicles' shapes are known based on this optimization process, the final piece of 3D traffic visualization is solved. Based on the position, orientation, and shape of the vehicles and road, they can be arranged in a frame to form three-dimensional traffic scenes reconstruction.

IV. Implementation

The program is implemented using Python 3.10. Vehicle detection is carried out using the YOLOv9c pretrained model in Ultralytics library. Vehicle orientation estimation is carried out using 3D Deepbox pretrained model [9]. Tracking is implemented using SORT [10] tracking framework. SORT is designed to track 2D bounding box, so it is modified the matrices and vectors to suit the required 3D tracking (see section III).

The lane detection is implemented using OpenCV. The GPLVM training is implemented using fmin_cg from scipy, with RBF (radial basis function) kernel. Since GPLVM is sensitive to local optima, PCA (principal component analysis) is utilized to initialize the latent variables, using scikit-learn library.

Vehicle models for GPLVM training used in this paper consist of five 3D shapes: jeep, sedan, pickup, SUV, and hatchback. The 3D shapes and resulting latent space are shown in Fig. 3. To form smooth reconstructed shape from a latent variable, marching cube algorithm in skimage library is used.

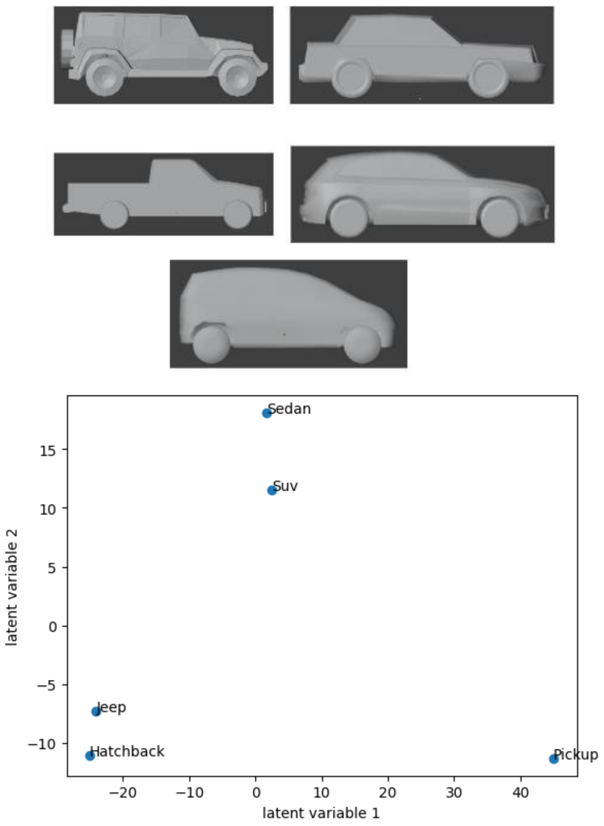

Fig. 3. Training 3D shapes (above) and resulting latent space (below**)**

V. Testing

Testing is carried out using several dashcam videos from the internet. The reconstructed traffic frames can be examined in Fig. 4. Generally, traffic scenes can be reconstructed well from the input image. However, for vehicles located far from the camera, the reconstructed shape and orientation is not accurate due to its small size in the image hence not enough information to accurately reconstruct it. For roads with bright/white color, the lane detection is confused since the road color is like the lane marking color, resulting in many false positive. In addition, the road detection does not work on roads without marking since it is based on the lane marking.

Execution time is calculated to measure its real-time performance and the results are shown in **Error! Reference source not found.**. The execution time measurement is performed on machine with CPU Intel i5-1035G1 and GPU NVIDIA GeForce MX330. According to the measurements, the implementation has not reached real-time processing time yet. Particularly, the shape reconstruction takes a long time to process. Therefore, more optimization is needed so that it can be suitable for autonomous vehicles which need real-time processing time.

Table 1. Execution time per step

Step	Execution time
Vehicle detection and orientation estimation	120 ms per frame
Vehicle tracking	20 ms per frame
Road detection and tracking	220 ms per frame
Vehicle shape reconstruction	3,1 second per frame

VI. Conclusion

A workflow and implementation of 3D traffic scenes reconstruction is developed in this paper. The program can reconstruct the position, orientation, and shape of the vehicles in vicinity of ego vehicle. The road and its lanes are also detected based on its road marking and then reconstructed along with the vehicles. However, the program has not achieved real-time processing time yet, so it is not suitable for autonomous vehicle applications yet.

The architecture of the reconstruction program is modular so various of its components can be replaced with better methods or models in the future. An additional model for detection is needed since currently road detection is based on road marking and not all roads have marking. The program can be further improved by combining it with GPS and map to prevent failed road detection in case of occlusion due to traffic.

References

[1] SAE International, “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles,” 2021.

[2] National Highway Traffic Association, “Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey,” 2018.

[3] U. of M. Center for Sustainable Systems, “Autonomous Vehicles Factsheet,” 2023.

[4] R. Häuslschmid, M. von Bülow, B. Pfleging, and A. Butz, “SupportingTrust in Autonomous Driving,” in *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, New York, NY, USA: ACM, Mar. 2017, pp. 319–329. doi: 10.1145/3025171.3025198.

[5] Inc. Tesla, “Model Y Owner’s Manual,” 2023.

[6] D. Lu, V. C. Jammula, S. Como, J. Wishart, Y. Chen, and Y. Yang, “CAROM - Vehicle Localization and Traffic Scene Reconstruction from Monocular Cameras on Road Infrastructures,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2021, pp. 11725–11731. doi: 10.1109/ICRA48506.2021.9561190.

[7] Q. Rao and S. Chakraborty, “In-Vehicle Object-Level 3D Reconstruction of Traffic Scenes,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 12, pp. 7747–7759, Dec. 2021, doi: 10.1109/TITS.2020.3008080.

[8] C.-Y. Wang and H.-Y. M. Liao, “YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information,” *arXiv preprint arXiv:2402.13616*, 2024.

[9] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, “3D Bounding Box Estimation Using Deep Learning and Geometry,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jul. 2017, pp. 5632–5640. doi: 10.1109/CVPR.2017.597.

[10] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *2016 IEEE International Conference on Image Processing (ICIP)*, IEEE, Sep. 2016, pp. 3464–3468. doi: 10.1109/ICIP.2016.7533003.

[11] V. A. Prisacariu, A. V. Segal, and I. Reid, “Simultaneous Monocular 2D Segmentation, 3D Pose Recovery and 3D Reconstruction,” 2013, pp. 593–606. doi: 10.1007/978-3-642-37331-2_45.

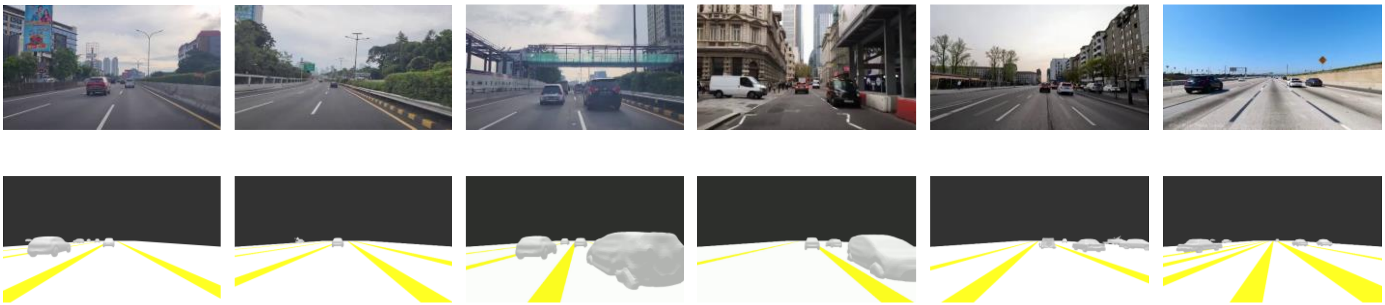

Fig. 4. Some examples of reconstructed traffic scene