SLAM in Lane changing of Data Driven Vehicles

Wayne Tsuei CE student
ht2383@nyu.edu

Samin Ghasem EE student
sg7884@nyu.edu

***Abstract*—Self driving vehicles become significantly popular in today's automation industry. One of the most important challenges of autonomous driving is how these vehicles will optimally interact with each other as well as their surroundings. SLAM (Simultaneous Localization & Mapping). It is a method to generates the map of a vehicle's surroundings and locates the vehicle in that map at the same time. The SLAM system uses the depth sensors such as 3D Cameras to gather a series of views, with approximate position and distance. Then, it stores these 3D views in memory.**

I. Introduction

Autonomous driving is widely regarded as having the potential to revolutionize transportation systems and eliminate traffic accidents almost entirely caused by improper human options [1]. One of the benefits of autonomous driving is that it can create a safe and efficient driving experience. Typically, autonomous vehicles use a variety of sensors and cameras to detect other vehicles, pedestrians, and obstacles around them [2], [3] then to make intelligent decisions regarding lane keeping, and lane change [4]. This paper proposes a data-driven optimal control strategy combining data-driven modeling and reinforcement learning techniques.

The existing lane change decision-making methods can be classified into three categories: trajectory planning (Li et al., 2020; Wang et al., 2019), factory assessment-based methods (Liu et al., 2019), and learning-based methods (Xie et al., 2019; Gu et al., 2020; Li et al., 2022; Tang et al., 2019). The classical approaches to trajectory planning based on continuous optimization formulate to generate a nonlinear program, in which vehicle dynamics and obstacle avoidance requirements as nonlinear equality and inequality constraints. Li et al. (2020) develop a dynamic cooperative planning model for lane changes in autonomous driving integrated into the driving control system of CAVs. The emerging technology of environmental sensing and vehicle-to-vehicle (V2V) communication, vehicle-to-infrastructure (V2I) communication, sharing the real-time velocity, acceleration, jerk, position, direction angle, and lane change demand of vehicles, enable the cooperative trajectory planning of lane changes for connected and automated vehicles (CAVs) to improve the efficiency and stability of traffic [5]. Wang et al. (2019) proposed a Deep Q-Network (DQN) algorithm with rule-based constraints for autonomous driving lane change decision-making tasks. Through the combination of high-level lateral decision-making and low-level rule-based trajectory modification, a safe and efficient lane change behavior can be achieved. With the setting of the state representation and reward function, the trained agent can take appropriate actions in a real-world-like simulator [6]. However, the above-mentioned methods do not fully consider vehicle dynamics so the generated path may be incompatible with the dynamics. Factory assessment-based methods usually adopt the following two steps to guarantee driving safety: 1) assess the benefit, safety, and tolerance of the current driving state, 2) formulate a sequential action strategy based on the factory assessment results to maintain safety. Liu et al. (2019) established an autonomous lane change decision-making model with Adaptive Cruise Control (ACC) for feature selection of support vehicle machine (SVM) model to solve the multi-parametric based on benefits, safety, and tolerance and nonlinear autonomous lane change decision-making model promising the decision-making model fits the driver's habits [7]. A common limitation of the probabilistic approaches is that they only use expert knowledge to generate rule-based decisions, due to the complexity of real traffic and road conditions, the feasibility of the model needs to be further researched. With the development of deep learning technologies, researchers began to address the decision-making problem using reinforcement learning methods. Xie et al. (2019) proposed a data-driven LC model based on deep learning models, combining deep belief network (DBN) and Long short-term memory (LSTM) NN to depict LCD (LC decision) and LCI (LC implementation) process based on the relative position of the preceding vehicle in the target lane, making the LC the proposed model more accurate in describing and predicting the LC process [8]. Liu et al. (2019) established an autonomous lane change decision-making model with Adaptive Cruise Control (ACC) for feature selection of support vehicle machine (SVM) model to solve the multi-parametric based on benefits, safety, and tolerance and nonlinear autonomous lane change decision-making model promising the decision-making model fits the driver's habits [9]. Tang et al. (2019) proposed an advanced lane change intention prediction method with a Relevance Vector Machine controller based on Multi-LSTM (Long Short-Term Memory) neural network. The method uses a real traffic information data set (NGSIM) to train the prediction model, which can predict left-turn intention, right-turn intention, and going-straight intention. The first LSTM network is used to

extract the lane-changing feature, and the second network can judge the intention [10].
Advanced cars known as Data Driven cars (DDVs) use sensor data and machine learning algorithms to enhance their efficiency, performance, and safety. These cars can gather and interpret massive volumes of data about their surroundings in real-time thanks to a range of sensors, including cameras, lidars, and radars. The behavior of the vehicle is then controlled by machine learning algorithms that are trained and enhanced using this data. Simultaneous Localization and Mapping (SLAM), a computational challenge that includes creating or updating a map of an unfamiliar environment while also keeping track of the location of the vehicle inside that environment, is one of the fundamental technologies utilized in DDVs. For DDVs to be able to independently navigate in complex and dynamic settings, SLAM is crucial

II. Methodology

By simultaneously building a map of the environment and figuring out their position within it. The process begins with using a robot with exceptional odometry performance, which is the measure of how well the robot can estimate its own position based on the position of its wheels..

A. *The requirement of SLAM*

A process used to map an environment while determining the robot's position within it. The process involves the use of a range measurement device, such as a laser scanner, sonar, or imaging device, to observe the environment around the robot. The robot then uses landmarks to determine its location, with the landmarks needing to be stationary, unique, plentiful, and distinguishable from the surrounding environment. The robot also needs a way to identify landmarks, which can be done through algorithms like Spike extraction or scan-matching.

B. *Problem definition: Structure of Probabilistic SLAM*

SLAM technique is using probabilistic approach which applies a probability distribution to predict the robot and landmarks location from the generated map. Given a series of controls and sensor observations over discrete time steps, the SLAM problem is to compute an estimate of the agent's location and a map of the environment. All quantities are usually probabilistic, so the objective is to compute:

$$P(m_{t+1}, x_{t+1} | o_{1:t+1}, u_{1:t})$$

- Applying Bayes' rule gives a framework for sequentially updating the location posteriors $P(x_t|x_{t-1})$,

$$P(x_t|o_{1:t}, u_{1:t}, m_t) = \sum_{m_{t-1}} P(o_t|x_t, m_t, u_{1:t}) \sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}|m_t, o_{1:t-1}$$

- Similarly, the map can be updated sequentially by:

$$P(m_t|x_t, o_{1:t}, u_{1:t}) = \sum_{x_t} \sum_{m_t} P(m_t|x_t, m_{t-1}, o_t, u_{1:t}) P(m_{t-1}, x_t|o_{1:t-1}, m_{t-1},$$

- Like many inference problems, the solutions to inferring the two variables together can be found, to a local optimum solution, by alternating updates of the two beliefs in a form of EM algorithm.

C. *Solution to SLAM problems*

- EKF SLAM

EKF-SLAM (Extended Kalman Filter SLAM) is a specific variant of the SLAM algorithm that uses an Extended Kalman Filter to estimate both the robot's pose and the map of the environment. The EKF-SLAM algorithm works by fusing sensor measurements, such as laser or sonar readings, with odometry data to estimate the robot's current position and orientation. As the robot moves through the environment, it updates the map by adding new landmarks and revising the locations of previously observed landmarks. The EKF-SLAM algorithm is widely used in robotics and autonomous vehicles because it is computationally efficient and provides accurate results. The EKF-SLAM algorithm involves multiple equations for predicting and updating the robot's pose and the map of the environment. Here are some of the key equations

- Prediction:
- Predicted State Estimate:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k)$$

- Predicted Covariance Estimate:

$$\mathbf{P}_{xx,k|k-1} = \nabla\mathbf{f}\, \mathbf{P}_{xx,k-1|k-1} \nabla\mathbf{f}^{\mathbf{T}} + \mathbf{Q}_k,$$

Where ∇f is the Jacobian of **f** evaluated at the estimate xˆ k−1|k−1 . There is generally no need to perform a time update for stationary landmarks.

- Correction:
- Innovation or Measurement Residual:

y_t = z_t - h(x̂_t, m)

- Innovation Covariance:

S_t = H_t * P_t * H_t^T + R_t

- Kalman Gain:

K_t = P_t * H_t^T * S_t^-1

- Updated State Estimate:

x̂_t = x̂_t + K_t * y_t

- Updated Covariance Estimate:

P_t = (I - K_t * H_t) * P_t

In these equations, x̂_t represents the predicted state estimate at time t, which includes the robot's pose and the map of the environment. f() is the motion model that predicts the robot's pose based on the control inputs u_t and the previous pose x_t-1. P_t represents the predicted covariance estimate, which is a measure of the uncertainty in the predicted state.

z_t is the measurement obtained from the sensors, and h( ) is the observation model that predicts the expected measurement based on the current state estimate. H_t is the Jacobian matrix of the observation model, and R_t is the covariance matrix of the measurement noise. y_t is the innovation or measurement residual, which is the difference between the actual measurement and the expected measurement. S_t is the innovation covariance, which is a measure of the uncertainty in the measurement. K_t is the Kalman gain, which combines the predicted state estimate and the measurement to produce the updated state estimate and covariance estimate.

- FAST SLAM

Fast SLAM is a variant of the SLAM algorithm that combines particle filtering and EKF-SLAM to address some of the limitations of both approaches. In Fast SLAM, the robot's path is represented by a set of particles, each particle representing a different possible trajectory of the robot. As the robot moves, these particles are updated based on the sensor measurements and the robot's motion model. Each particle maintains its own map of the environment, which is updated using the EKF-SLAM algorithm. The Fast SLAM algorithm is able to handle nonlinear motion models and sensor models, and is particularly suited to environments with many features or landmarks, where traditional EKF-SLAM algorithms may struggle to maintain an accurate estimate of the robot's position and the map.

- Initialization:

The algorithm begins by initializing a particle filter with a large number of particles, each representing a possible robot pose.

- Prediction:

Each particle predicts its new pose based on the robot's control inputs.

- Measurement Update:

For each particle, a measurement update is performed by applying a probabilistic weight to the particle based on the likelihood of observing the measurements given the particle's pose.

- Resampling:

Particles with low weights are discarded and new particles are generated from the remaining particles using a resampling technique that ensures high-weight particles have a higher chance of being selected.

- Feature Update:

For each particle, the corresponding map is updated using the feature observations made during the measurement update step.

- Loop:

The above steps (2-5) are repeated for every control input and measurement until the robot has completed its path.

- Estimation:

The final pose estimate is obtained by computing the mean or mode of the particles, while the map estimate is obtained by computing the mean or mode of the features observed by the particles

Note that the Fast SLAM algorithm consists of two main parts: particle filtering and EKF-SLAM. The particle filter predicts the robot's pose and updates the particle weights based on the observed landmarks. The EKF-SLAM estimates the location of the landmarks based on the measurements received by the robot. The algorithm updates the weights of the particles based on the likelihood of the observed landmarks given the estimated landmark positions. The resampling step is then performed to eliminate particles with low weights and duplicate particles with high weights. The updated particles are then used to estimate the robot's pose and the landmarks' positions in the next iteration. The algorithm iterates through these steps for each time step, continuously updating the map and the robot's pose. The mathematical equations and formulas involved in the algorithm are complex and involve probability distributions, Bayes' rule, and more.

In summary, Fast SLAM combines a particle filter for pose estimation with a map representation that is built incrementally by tracking individual features. This allows for efficient and accurate mapping and localization in environments with many features, such as indoor environments or outdoor urban environments.

III. Experimental explanation

Imagine that you have a robot which has been placed in an unknown location. This robot requires map this environment in 3D and localize its position within this.

A) Using a photo camera

Capturing different photos of the scene at different position And angels some commonly used cameras: Panasonic DMCTZ8 digicam.

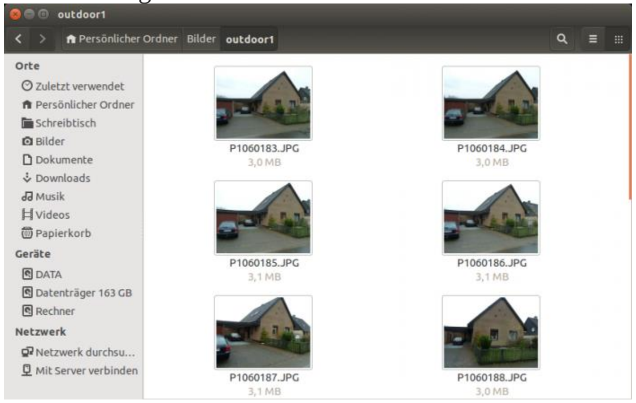

Figure 1 shows Input Images Captured by Camera

B)Using the Capture Images to calculator sparse point cloud, these are the points will be used for matching, this can be seen in figure 2 below:

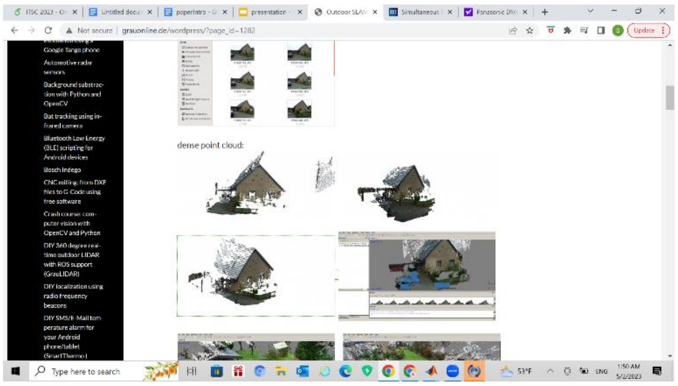

Figure 2 shows matching process in SLAM

C) Reconstructive dense point clouds

D) Optional: make new photo at arbitrary position and use same principles to estimate actual camera position.

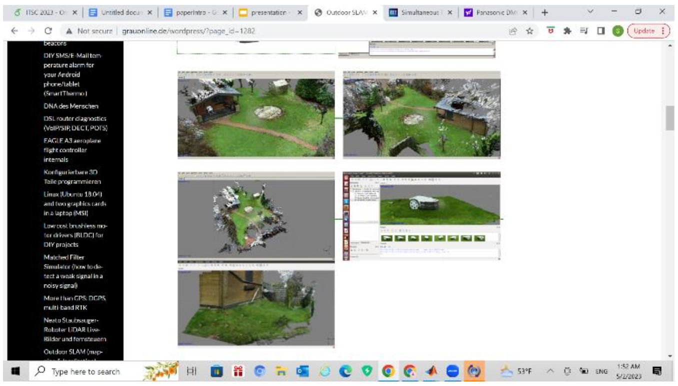

Figure 3 final map created by SLAM

IV. SLAM Implemenation

SLAM technology is now widely used in different section such as autonomous vehicles, road observation and mapping, lane changing and robots navigation

Data acquisition: This involves obtaining data about the environment surrounding the robot. This can be done using sensors such as laser scanners, sonar, or imaging devices.

Feature extraction: In this step, the data acquired in step 1 is processed to extract useful features that can be used to build a map of the environment. These features can include landmarks, corners, and edges.

Data association: This involves associating the extracted features with their corresponding locations in the environment.

State estimation: The robot's pose or position in the environment is estimated using the extracted features and data association.

Mapping: Based on the estimated pose and extracted features, a map of the environment is created.

Loop closure detection: This step involves detecting when the robot has revisited a location it has previously mapped. This helps to improve the accuracy of the map and robot's pose estimation.

Optimization: Finally, the map and the robot's pose estimation are optimized to reduce errors and improve accuracy.

V. Conclusion

In summary, Data Driven Vehicles and SLAM are advanced technologies that are essential for enabling vehicles to navigate autonomously in complex and dynamic environments. These technologies are based on sensor data and machine learning algorithms, and they are critical for improving the safety, performance, and efficiency of modern vehicles.

References

[1] Y. Ma, Z. Wang, H. Yang, and L. Yang, “Artificial intelligence applications in the development of autonomous vehicles: A survey,” IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 2, pp. 315–329, 2020.
[2] J.-F. Bonnefon, The car that knew too much: can a Machine be moral? MIT Press, 2021.
[3] Y. Dou, F. Yan, and D. Feng, “Lane changing prediction at highway lane drops using support vector machine and artificial neural network classifiers,” in 2016 IEEE international conference on advanced intel- ligent mechatronics (AIM). IEEE, 2016, pp. 901–906.
[4] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, “Sensor

[1] Pengxiang Zhao, Yang Xing, and Jiaqi Ma, “Data-Driven Adaptive Optimal Control of Connected Vehicles Cooperative Adaptive Cruise Control for Mixed Traffic Flow: Design, Implementation, and Evaluation,”

[2] HUGH DURRANT-WHYTE AND TIM BAILEY, “Simultaneous Localization and Mapping: Part 1,”

[3] Alif Ridzuan Khairuddin, Mohamad Shukor Talib, Habibollah Haron

, “Review on Simultaneous Localization and Mapping and
sensor fusion technology in autonomous vehicles: A review,”
Sensors, vol. 21, no. 6, p. 2140, 2021.
[5] H. Min, X. Wu, C. Cheng, and X. Zhao, “Kinematic and dynamic vehicle model-assisted global positioning method for autonomous ve- hicles with low-cost gps/camera/in-vehicle sensors,” Sensors, vol. 19, no. 24, p. 5430, 2019.
[6] M. Huang, M. Zhao, P. Parikh, Y. Wang, K. Ozbay, and Z.-P. Jiang, “Reinforcement learning for vision-based lateral control of a self- driving car,” in 2019 IEEE 15th International Conference on Control and Automation (ICCA). IEEE, 2019, pp. 1126–1131.
[7] T. Li, J. Wu, C.-Y. Chan, M. Liu, C. Zhu, W. Lu, and K. Hu, “A cooperative lane change model for connected and automated vehicles,”
IEEE Access, vol. 8, pp. 54 940–54 951, 2020.
[8] J. Wang, Q. Zhang, D. Zhao, and Y. Chen, “Lane change decision- making through deep reinforcement learning with rule-based con- straints,” in 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, 2019, pp. 1–6.
[9] Y. Liu, X. Wang, L. Li, S. Cheng, and Z. Chen, “A novel lane change
decision-making model of autonomous vehicle based on support vector
machine,” IEEE access, vol. 7, pp. 26 543–26 550, 2019.
[10] D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He, “A data-driven lane-changing
model based on deep learning,” Transportation research part C: emerging technologies, vol. 106, pp. 41–60, 2019.

[11] X. Gu, Y. Han, and J. Yu, “A novel lane-changing decision model for autonomous vehicles based on deep autoencoder network and xgboost,” IEEE Access, vol. 8, pp. 9846–9863, 2020.
[12] L. Tang, H. Wang, W. Zhang, Z. Mei, and L. Li, “Driver lane change
intention recognition of intelligent vehicle based on long short-term memory network,” IEEE Access, vol. 8, pp. 136 898–136 905, 2020.
[13] C. Wei, F. Hui, and A. J. Khattak, “Driver lane-changing behavior pre-
diction based on deep learning,” Journal of advanced transportation, vol. 2021, pp. 1–15, 2021.
[14] G. Hewer, “An iterative technique for the computation of the steady
state gains for the discrete optimal regulator,” IEEE Transactions on Automatic Control, vol. 16, no. 4, pp. 382–384, 1971.
[15] L. Traxxas, “The new traxxas summit 16.8 v electric extreme terrain
monster truck,” 2008.