ISSN:2790-1688 Volume-15-(2025)

An Integrated Prediction and Trajectory Planning Method for Mixed Traffic Environments

Dongting Ge

Beijing University of Posts and Telecommunications,Beijing,100876,China;

Abstract. Among the countless challenges confronting autonomous driving, a primary impediment to its real-world deployment lies in achieving safe and efficient decision-making and planning within dynamic and mixed traffic environments. The key to addressing this issue lies in the development of a sophisticated framework for trajectory prediction and planning that can effectively cope with the challenges posed by mixed traffic environments. This paper proposes an integrated research method that unifies trajectory prediction and trajectory planning to address this challenge. Firstly, the proposed method employs a deep learning model, based on the standard encoder-decoder architecture, to predict the future trajectories of surrounding vehicles. Secondly, based on these prediction results, A*-based search algorithm is employed to plan a collision-free, optimal trajectory for the ego-vehicle on the given map. Finally, the proposed method was comprehensively validated through simulations on the large-scale, high-fidelity highD real-world dataset. The experimental result corroborates the accuracy of our approach.

Keywords: Autonomous driving; trajectory prediction; trajectory planning; encoder-decoder framework.

1. Introduction

1.1 Background

Autonomous Driving stands at the forefront of the application of AI technologies. While significant progress has been made in this field, research into mixed traffic environments—characterized by uncertainty, stochasticity, and individual heterogeneity—remains incomplete. To get self-driving cars out of controlled test environments and onto real public roads, we have to solve a key problem: how to predict what other drivers will do and plan a safe path through busy, mixed traffic. Studying this is essential for making autonomous technology a part of our everyday world.

1.2 Related Work

For a self-driving car to navigate busy roads, it has to master two key skills. First, it needs to predict what other vehicles are going to do. Second, based on those predictions, it must plan its own safe and efficient trajectory. Both challenges have become major areas of research.

The prediction of vehicle trajectories is a fundamental component for autonomous driving systems operating in mixed traffic environments. The Social-LSTM model proposed by Alahi et al. (2016) [1] is considered a pioneering work in this field. Building on this foundation, Deo et al. (2018) [2] proposed Convolutional Social Pooling, which leverages convolutional operations to capture more spatially structured interaction features, thereby further enhancing prediction accuracy in crowded vehicle scenarios. The Trajformer model, proposed by Gu et al. (2021) [3], leverages the self-attention mechanism to process long-term temporal information in parallel, effectively overcoming the challenge of handling long-range dependencies inherent in RNN models. The Target-driven Trajectory Prediction model, proposed by Cui et al. (2019) [4], effectively covers behavioral multimodality by first predicting a set of potential target endpoints and then generating multiple candidate trajectories leading to these goals. CoverNet, proposed by Phan-Minh & Ourique (2020) [5], also performs multimodal prediction by generating a set of trajectories designed to cover the full spectrum of plausible future possibilities. Rhinehart et al. (2019) [6] proposed the PRECOG model, which is capable of generating trajectories consistent with a given target point. Trajectory planning for the ego-vehicle, informed by accurate predictions, is a critical step for autonomous driving

systems operating in mixed traffic environments. The review articles by Paden et al. (2016) [7] and Schwarting et al. (2019) [12] provide a comprehensive survey of motion planning techniques for autonomous driving. Ziegler et al. (2014) [8] provided a detailed account of their long-distance autonomous driving project with the automated vehicle 'Bertha'. Li et al. (2023) [15] specifically consider the interaction uncertainties with other vehicles. Shalev-Shwartz et al. (2016) [9] pioneered a safe multi-agent reinforcement learning framework for autonomous driving decision-making. The survey by Wang et al. (2021) [14] provides a systematic summary of the application of DRL-based decision-making and planning methods for autonomous driving in mixed traffic environments. Kuefler et al. (2017)[10] utilized Generative Adversarial Networks (GANs) to mimic human driving behavior. In a different approach, Li et al. (2020) [13]proposed Conditional Imitation Learning. Hoel et al. (2019) [11] explored a framework that combines a planner with a learned policy to cope with dense traffic scenarios.

1.3 Contributions

This study introduces a two-step method to help self-driving cars make smart decisions. First, we use an Encoder-Decoder model to predict how nearby cars will move and interact over the next few seconds. This model is great at understanding complex traffic situations and guessing future paths. Once we have those predictions, we use an A* algorithm to plan the safest and most efficient route for our own car. By combining this prediction step with a planning step, we've created a system that can successfully simulate autonomous driving in busy, mixed traffic.

2. Methodology

This study creates a complete decision-making system for a self-driving car. Firstly, we teach a prediction model (based on an Encoder-Decoder design) how to forecast vehicle trajectories using a prepared dataset. Then, for our self-driving car (the ego-vehicle), this trained model predicts the future paths of all surrounding vehicles. Finally, these predictions are used to create rules for avoiding moving obstacles, which helps the car plan the safest and most optimal route. The proposed method primarily consists of two core modules: 2.1) a Transformer-based trajectory prediction model, and 2.2) an A*-based optimal trajectory planner.

2.1 Encoder-Decoder-based Trajectory Prediction Model

To accurately capture the complex interactions among vehicles in mixed traffic environments, we have developed a trajectory prediction model based on the standard Encoder-Decoder architecture. Due to their sequential processing mechanism, Recurrent Neural Networks (RNNs/LSTMs) are prone to issues such as vanishing gradients. In contrast, the self-attention mechanism employed in our model can directly establish global dependencies between any two points in the sequence, thereby better capturing long-term dynamics.

2.1.1 Input Feature and Embedding

The model's input data is divided into two categories: dynamic features and static features. The dynamic features are represented as a 13-dimensional vector, encompassing time-varying states. The static features are represented by a 5-dimensional vector, including properties.

2.1.2 Model Architecture

The model follows the Encoder-Decoder architecture, comprising an encoder and a decoder. The encoder is composed of a stack of N (N=6 in our implementation) identical layers, where each layer contains a multi-head self-attention module and a feed-forward neural network. Through its self-attention mechanism, it captures the dependencies among all time steps within the sequence, ultimately generating a memory tensor that contains rich contextual information. This memory tensor can be interpreted as an understanding of the vehicle's historical behavior. The decoder is similarly composed of a stack of N (N=6) identical layers. Unlike the encoder, however, each layer in the

decoder contains two multi-head attention modules. The first attention module performs masked self-attention on the already generated target sequence. This ensures that for the prediction at time step t, the model can only attend to positions up to and including t-1. The second attention module then performs cross-attention on the encoder's output ('memory') and the decoder's own input.

2.1.3 Model Training and Inference

During the training phase, the complete ground-truth future trajectory is fed as input to the decoder, but it is shifted to the right by one time step. The task of the decoder is to predict the position at time step t, given the historical information provided by the encoder and the ground-truth position at time step t-1.

During the prediction phase, the model generates trajectories in an autoregressive manner. Firstly, the encoder performs a full computation on the historical trajectory to generate the memory tensor. Then, the last point of the historical trajectory is used as the initial input to the decoder to predict the trajectory point for the first future time step. Finally, this newly generated point is appended to the decoder's input sequence to predict the trajectory point for the second future time step. This process is repeated iteratively until a complete future trajectory of the desired length has been generated.

2.2 A*-based Path Planning for Trajectory Generation

After obtaining the future trajectory predictions for neighboring vehicles, the objective of the planning module is to find a safe and optimal path for the ego-vehicle from a start to a goal position. The A*-based algorithm was selected because it can efficiently find the optimal path in a discrete space while guaranteeing optimality. This path then serves as the foundation for generating a smooth, drivable trajectory.

2.2.1 Planning Problem Formulation and Scenario Construction

We randomly select a scene from the test dataset, which contains an ego-vehicle and multiple neighboring vehicles within a certain range (set to 50 meters in this project). Using the prediction model trained in Section 2.1, we predict the future trajectories for each neighboring vehicle in the scene. The predicted results are a series of future position points of the neighboring vehicle in the world coordinate system.

2.2.2 Spatial Representation of the Planning Space

To apply the A* algorithm, a two-dimensional discrete grid map is created with the current position of the ego vehicle as the center. Subsequently, the predicted future trajectory points of all neighboring vehicles are mapped onto this grid map. Any grid cell containing the predicted position of a neighboring vehicle is marked as an obstacle, indicating that the corresponding area will be inaccessible at a future time. In this way, the dynamic behavior of the neighboring vehicles is statically projected onto a spatio-temporal occupancy map.

2.2.3 A*-based Trajectory Planning Algorithm

The A* algorithm identifies the optimal path by evaluating the cost function of nodes, expressed as $f(n) = g(n) + h(n)$. In our study, each node corresponds to a grid cell in the map, and the function $g(n)$ represents the actual movement cost from the start node to the current node. The cost of moving one step horizontally or vertically is 1.0, while the cost of a diagonal move is 1.414. The function $h(n)$ represents the estimated cost from the current node n to the target node. We adopt the square of the Euclidean distance as the heuristic function, which can effectively guide the search towards the target direction.

The algorithm starts from the initial node, explores its adjacent nodes that are not obstacles, and places them into a priority queue. At each iteration, the node with the smallest $f(n)$ value is extracted from the queue for expansion until the target node is found. The output of the A* algorithm consists of a series of grid coordinates connecting the start and end points. After coordinate transformation, this sequence forms the optimal, collision-free path of the ego-vehicle.

2.2.4 Path Smoothing and Trajectory Generation

The raw path from the A* algorithm is often jagged with sharp turns, making it impossible for a real car to follow. Therefore, we must smooth it out to create a usable trajectory. We apply a smoothing algorithm, like Bezier curves, to this rough path. This transforms the sharp, connected points into a single, continuous curve. Next, we plan how fast the car should travel along this new, smooth path. Based on the car's normal driving habits (like its usual speed and acceleration), we assign a comfortable velocity to each point on the curve. This final step ensures the car stays within its physical limits and results in a complete, executable trajectory.

3 Simulation & Discussion

3.1 Experiment Setup

To test our method, we built a simulation that uses real-world data of how cars move. The data comes from the highD dataset[16], a highly accurate collection of natural driving behavior recorded by drones on German highways. The selection of this dataset is primarily based on several advantages: The overhead perspective ensures data completeness. The dataset is very precise, covering a wide range of real-world traffic situations from freely moving cars to heavy traffic jams, and it includes details about the vehicles themselves to ensure our simulations are physically realistic. Getting the raw data ready for our model involved a few steps. First, we chopped up the long traffic recordings into short clips. Each clip showed 3 seconds of a car's past movements and 5 seconds of what it did next. We then tossed out any clips that had errors or missing information. After that, we pulled the key details for each car—like its speed and position—and organized it all into a format our model could work with. Finally, we split the data into three piles: an 80% pile for teaching the model, a 10% pile for fine-tuning it, and a final 10% pile to test how well it learned.

All experiments were run on a laptop with an Intel Core Ultra 9 CPU, 32 GB of RAM, and an NVIDIA GeForce RTX 5070 Laptop GPU. We used Python3.12 and the PyTorch2.7 framework on a Windows operating system to build and train our model.

3.2 Analysis of Trajectory Prediction Results

In this section, we’ll look at some examples to see how well the model actually performs on the test data. By visualizing randomly selected samples, we compared the predicted results with the real trajectories in the same coordinate system to evaluate the model's prediction performance across different driving scenarios. Additionally, we compared our prediction model with the RNN model to validate the performance of our proposed prediction model.

3.2.1 Visualization Methodology

To eliminate the interference of global position and heading changes on motion pattern analysis, all trajectories were plotted in a normalized local coordinate system centered around the target vehicle. The construction of this coordinate system follows the standard: the origin (0, 0) is defined as the position of the last time step of the historical trajectory; the positive direction of the X-axis aligns with the instantaneous velocity vector of the vehicle at that point; the Y-axis is perpendicular to the X-axis, forming a right-handed coordinate system. In the plotting implementation, we enforced an equal aspect ratio for the coordinate axes to ensure that the geometric shape of the trajectories (such as curvature) is presented without distortion.

Each visualization includes three different trajectory sequences: 1) Historical trajectory (History): the observed data used as model input, represented by a blue solid line; 2) Ground truth future trajectory (Ground Truth Future): the actual path driven by the vehicle within the prediction time horizon, represented by a green solid line; 3) Predicted future trajectory (Predicted Future): the output sequence generated by the model, represented by a red dashed line.

3.2.2 Analysis of Sample Prediction Results

We randomly selected five scenes from the test results for visualization analysis, where the prediction results for test samples #3 and #5 are obtained from the proposed prediction model, while the prediction results for test samples #4 and #10 are from the RNN model, as shown in Fig.1.

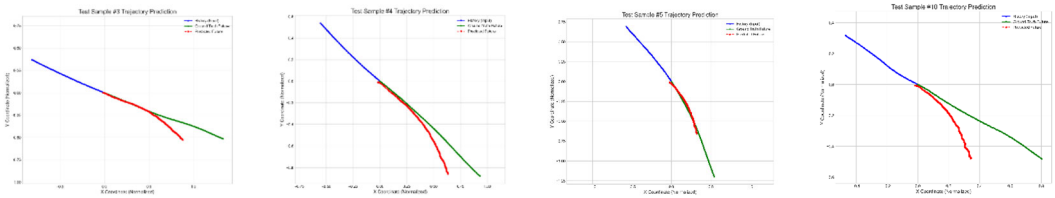

Fig.1 Test samples of trajectory prediction based on encoder-decoder model and RNN model

In all samples, the model is able to identify the vehicle's macroscopic intent. However, the model exhibits variations in the accuracy of predicting the geometric shape of the trajectories. In the scene of test sample #5, the predicted trajectory almost perfectly overlaps with the ground truth trajectory. In contrast, for test samples #3, the model slightly overestimates the sharpness of the turns. In test samples #4 and #10, the model's overestimation of the turn curvature results in significant deviations between the predicted trajectory and the smoother ground truth path. Overall, both the encoder-decoder based model and the RNN model are capable of accurately identifying the vehicle's intent. However, the RNN model still has limitations in long-term path prediction in certain scenarios, whereas the encoder-decoder based model performs more accurately in long-term trajectory prediction.

3.3 Analysis of Planning Results Based on Trajectory Prediction

This section provides a qualitative evaluation of the integrated method of combining trajectory prediction and planning. The predicted trajectory points are transformed from the normalized local coordinate system in Section 4.2 to the global coordinate system and projected onto a discrete grid map centered around the ego-vehicle, forming a dynamic and time-varying obstacle region. Subsequently, the A* planning algorithm is employed to plan an optimal trajectory for the ego-vehicle from its current position to the predefined target point on the grid map. I will now validate the effectiveness and rationality of the method by analyzing three typical scenes randomly selected from the test set.

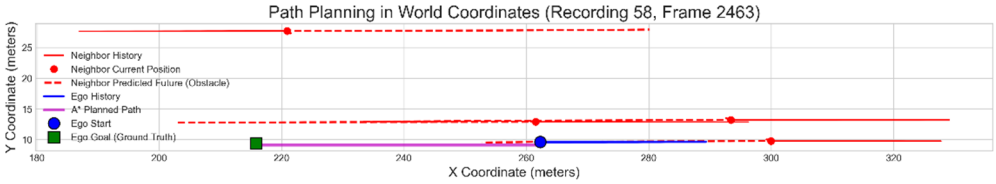

Fig.2 First test sample of the integration of trajectory prediction and planning

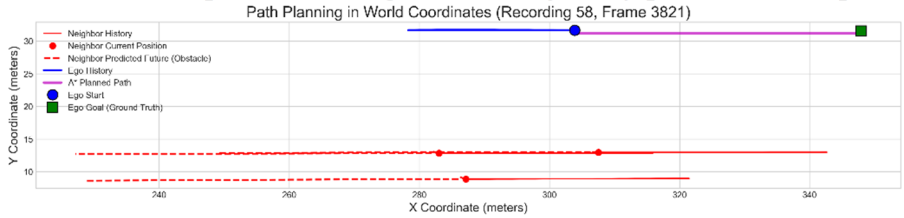

Fig.3 Second test sample of the integration of trajectory prediction and planning

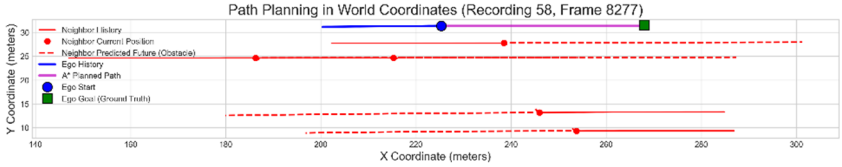

Fig.4 Third test sample of the integration of trajectory prediction and planning

As shown in Fig.2 and Fig.3, in the highway car-following and cruising scenarios, the motion patterns of the neighboring vehicles are stable. Our prediction model is able to accurately forecast that they will continue driving in a straight line within their lane, thus reserving a clear passage area for the ego-vehicle on the spatiotemporal occupancy map. As a result, the planner successfully generates a smooth and direct trajectory to the target point, demonstrating the framework's stable decision-making capability under conventional conditions. A more complex scenario is shown in Fig.3, where the ego-vehicle needs to navigate in a dynamic environment formed by multiple neighboring vehicles. Despite the increased environmental constraints, the trajectory prediction model still accurately predicts the trajectories of neighboring vehicles, allowing the planner to safely generate a path through the available space between the neighboring vehicles. Overall, the planner is able to generate a safe and reasonable driving trajectory upon receiving the predicted trajectories of neighboring vehicles, thereby confirming the effectiveness and robustness of the proposed method.

3.4 Analysis of the Loss Curve and Test Metrics of the Prediction Model

This section will analyze the convergence behavior during the training process and the final accuracy metrics on the independent test set. The training objective of the model is to minimize the Mean Squared Error (MSE) between the predicted and ground truth trajectory coordinates. To prevent overfitting, we adopt a strategy of saving the optimal model checkpoint based on the lowest Final Displacement Error (FDE) on the validation set, rather than solely relying on the training loss.

3.4.1 Convergence Analysis of the Training Process

Fig.5 Loss curve

As shown in Fig.5, the training process of the model exhibits efficient and stable convergence characteristics. In the early stages of training, the loss value experienced a sharp decrease, indicating that the model quickly learned the fundamental motion patterns and physical constraints within the data. After processing approximately 500 batches of data, the loss curve quickly entered a stable convergence phase and gradually approached zero. After processing approximately 500 batches of data, the loss curve quickly entered a stable convergence phase and gradually approached zero. This characteristic learning curve morphology demonstrates the stability of the training process, indicating that the model has achieved effective and sufficient fitting of the training data without exhibiting oscillations, divergence, or other anomalous behaviors.

3.4.2 Analysis of Validation and Testing Performance Metrics

The model achieved an Average Displacement Error (ADE) of 0.1842 meters and a Final Displacement Error (FDE) of 0.4487 meters on the validation set. To comprehensively evaluate the model's generalization capability, we conducted a rigorous assessment of this optimal model on the independent test set. The results show our model performs well. On the test set, the Average Displacement Error (ADE) was 0.1796 meters, meaning the predicted path was, on average, less than 18 cm off from the actual path. The Final Displacement Error (FDE) was 0.4373 meters, which is the error at the very end of the prediction. Interestingly, the model performed slightly better on the final

test data than it did during its tuning phase (the validation set). This is a strong confirmation that the model learned general driving behaviors effectively and wasn't just "memorizing" the data it was trained on. This outcome shows our training strategy was successful in creating a reliable and robust model.

4 Conclusion

This study establishes a complete perception-to-execution pipeline by integrating an encoder-decoder based trajectory prediction module with an A*-search driven motion planner, thereby enabling seamless environment-to-action mapping. Through extensive simulation experiments conducted on the highD real-world trajectory dataset, we rigorously validate both the effectiveness and robustness of the proposed framework under realistic driving conditions.

Our study yields the following key conclusions: Firstly, the encoder-decoder-based trajectory prediction model demonstrates excellent performance in processing long-term vehicle motion data. The training process shows stable convergence, achieving an Average Displacement Error (ADE) of 0.1796 m and a Final Displacement Error (FDE) of 0.4373 m on the independent test set. Secondly, when provided with neighboring vehicle behavior predictions, the A*-based planning algorithm effectively generates safe driving trajectories in dynamic, multi-agent environments. Simulation results validate the framework's decision-making capability in typical scenarios.

Despite its successful result, this study's limitations highlight future research directions. Firstly, the deterministic prediction model cannot capture the uncertainty and multimodality of driving behavior; future work could employ generative models like VAEs or GANs for probabilistic prediction. Secondly, the discrete grid-based A* planner could be improved by adopting advanced planners such as Hybrid A* or MPC. Finally, the prediction and planning modules lack feedback. Exploring more tightly coupled or even end-to-end learning architectures is crucial for enhancing system performance in highly interactive scenarios.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 961-971.

[2] N. Deo and M. M. Trivedi, "Convolutional Social Pooling for Vehicle Trajectory Prediction," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 2018, pp. 1549-15498.

[3] Gu, T., Chen, C., & Ren, M. (2021). Trajformer: A Transformer-based trajectory prediction model.

[4] H. Cui et al., "Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 2090-2096.

[5] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom and E. M. Wolff, "CoverNet: Multimodal Behavior Prediction Using Trajectory Sets," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 14062-14071.

[6] N. Rhinehart, R. Mcallister, K. Kitani and S. Levine, "PRECOG: PREdiction Conditioned on Goals in Visual Multi-Agent Settings," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 2821-2830.

[7] B. Paden, M. Čáp, S. Z. Yong, D. Yershov and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," in IEEE Transactions on Intelligent Vehicles, vol. 1, no. 1, pp. 33-55, March 2016.

[8] J. Ziegler et al., "Making Bertha Drive—An Autonomous Journey on a Historic Route," in IEEE Intelligent Transportation Systems Magazine, vol. 6, no. 2, pp. 8-20, Summer 2014.

[9] Shalev-Shwartz, S., Shammah, S., & Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving.

[10] A. Kuefler, J. Morton, T. Wheeler and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 2017, pp. 204-211.

[11] Hoel, C. J., Driggs-Campbell, K., & Kochenderfer, M. J. (2019). Combining planning and learning for driving in dense traffic.

[12] Schwarting, W., Alonso-Mora, J., & Rus, D. (2019). Planning and decision-making for autonomous vehicles. Annual Review of Control, Robotics, and Autonomous Systems.

[13] Li, L., Ouyang, W., Sheng, L., Wang, X., & Zhou, B. (2020). End-to-end driving via conditional imitation learning. ICRA.

[14] Wang, K., et al. (2021). A review of autonomous vehicle decision-making and planning based on deep reinforcement learning in mixed traffic environments. Control and Decision. (CNKI).

[15] Li, J., et al. (2023). Motion planning method for autonomous vehicles considering interaction uncertainty. Journal of Mechanical Engineering. (CNKI).

[16] R. Krajewski, J. Bock, L. Kloeker and L. Eckstein, "The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems," 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 2018, pp. 2118-2125.