Motion Prediction for Autonomous Vehicles Based on Feature Extraction

Minghao Chen
Jilin University
paulliantchen@gmail.com

***Abstract*—High-precision motion prediction is essential for safe and efficient autonomous driving, enabling vehicles to anticipate the future positions of surrounding agents and make informed decisions. Existing motion prediction algorithms often rely on deep learning models like LSTMs and Transformers, which, despite their strong performance, lack interpretability and require substantial computational resources, limiting their applicability in real-time scenarios. To address these challenges, we propose a motion prediction algorithm based on feature extraction of kinematic and interaction properties from observed trajectories. By focusing on interpretable features such as velocity, acceleration, angular momentum, our method reduces computational overhead while maintaining competitive accuracy. Additionally, we demonstrate that our model accelerates the convergence of existing models when integrated as a sub-module. Experimental results on the Argoverse Motion Forecasting Dataset show that our model achieves comparable performance to state-of-the-art methods without relying on complex architectures or map information.**

***Index Terms*—motion prediction, autonomous driving, feature extraction, kinematic features, interaction features**

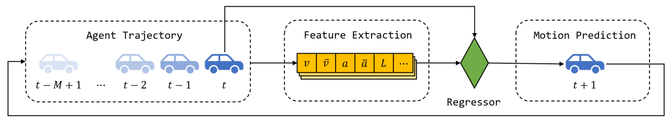

Figure 1. Model Overview

I. Introduction

With the rapid advancement of autonomous driving technology, high-precision motion prediction has become one of the core technologies enabling safe navigation and efficient driving of vehicles. Motion prediction, especially trajectory prediction, aims to forecast the future positions of various agents in the autonomous driving environment—such as other vehicles and pedestrians—to provide essential data support for decision-making and path planning in autonomous driving systems. Accurate and real-time prediction of surrounding agents' future trajectories is crucial for a safe and smooth autonomous driving system, as it allows the motion planner to compute trajectories that satisfy kinematic and dynamic constraints while ensuring passenger comfort [1]. Along with the widespread application of artificial intelligence and machine learning technologies, trajectory prediction techniques have made significant progress, particularly in data-driven algorithms and models.

Existing motion prediction algorithms are predominantly based on Encoder-Decoder architectures, such as Long Short-Term Memory networks (LSTM) [2, 3] and Transformers [4, 5]. While these methods exhibit strong generality and impressive performance, the features they extract often lack interpretability. Deep neural networks inherently function as black boxes, making it challenging to understand the decision-making process behind their predictions. Furthermore, these models typically require substantial computational resources and processing time, rendering them less suitable for real-time motion prediction in autonomous driving scenarios where swift responses are crucial.

To address these challenges, this paper proposes a motion prediction algorithm based on feature extraction. By extracting kinematic and interaction features to supplement the prediction algorithm, we enhanced interpretability while achieving a decent accuracy. This approach reduces computational overhead and allows the algorithm to function effectively as an embedded sub-module within larger systems. In summary, our contributions are:

- We develop a motion prediction algorithm based on extracted kinematic and interaction features, enhancing interpretability compared to deep learning models.
- By relying on a smaller set of features, our approach lowers computational overhead, making it suitable for real-time motion prediction in autonomous driving scenarios.
- The proposed feature extraction model works effectively as an embedded sub-module, accelerating the convergence speed of existing models.

II. Related Work

Motion prediction in autonomous driving. Motion prediction in autonomous driving has been a significant area of research, focusing on accurately forecasting the trajectories of agents, such as vehicles and pedestrians, in dynamic environments. Early approaches predominantly utilized sequence-to-sequence models based on Long Short-Term Memory (LSTM) networks to capture temporal dependencies in motion patterns. For instance, Park et al. [2] proposed an LSTM encoder-decoder architecture for vehicle trajectory prediction, demonstrating the effectiveness of recurrent neural networks in modeling sequential data for motion forecasting. Similarly, Alahi et al. [3] introduced the Social LSTM model, which accounts for human-human interactions in crowded spaces by incorporating social pooling mechanisms, thereby improving the prediction accuracy of human trajectories.

In recent years, there has been a shift towards Transformer-based architectures to better handle long-range dependencies and parallel computations. Lan et al. [4] presented SEPT, an efficient scene representation learning framework that leverages Transformers for motion prediction, achieving improved computational efficiency without sacrificing accuracy. Additionally Nayakanti et al. [5] introduced Wayformer, a motion forecasting model that utilizes simple and efficient attention networks to capture complex interactions in driving scenarios, leading to enhanced prediction performance.

Waymo Open Sim Agents Challenge. The importance of realistic agent feature modeling in autonomous driving has been emphasized by Montali et al. [6] through the introduction of the Waymo Open Sim Agents Challenge. This challenge focuses on developing simulators capable of generating realistic and interactive behaviors of traffic agents, which are essential for training and evaluating autonomous driving systems. Specifically, it addresses the problem of "simulation drift"—the deviation between agent behavior in driving logs and in simulations—by measuring sim agents' compliance with a set of kinematic and interaction features. In our work, we leverage these features proposed in the challenge, incorporating agent kinematics and interactions to improve prediction accuracy.

VectorNet. VectorNet, proposed by Gao et al. [7], is a foundational model in trajectory prediction for autonomous driving and serves as a significant baseline in this domain. Unlike traditional methods that rely on rasterized images of maps and agent trajectories, VectorNet operates directly on vectorized high-definition (HD) maps and agent dynamics. It introduces a hierarchical graph neural network architecture that first captures local spatial relationships within individual polylines—representing roads, lanes, and agent paths—and then models global interactions among these polylines. This approach allows for efficient and precise encoding of map features and agent behaviors without the loss of information inherent in rasterization. By leveraging vectorized representations, VectorNet effectively captures the complex interactions between agents and their environments, leading to improved prediction accuracy. In our work, by adding manually extracted agent features as supplementary local features into the original VectorNet, we achieve faster convergence during training.

III. Method

In this section, we introduce our proposed motion prediction algorithm based on feature extraction. We begin by formally defining the problem of forecasting future agent positions in autonomous driving environments. Next, we describe the extraction of meaningful kinematic and interaction features from observed trajectories. Finally, we explain how these features are utilized in an iterative prediction framework to generate accurate future trajectories.

A. Problem Definition

The motion prediction task aims to forecast the future positions of agents in an autonomous driving environment based solely on their observed past trajectories. Let there be N agents in the scene. For each agent i, we are given its past trajectory as a sequence of positions $\mathbf{X}_i = \{(x_t^i, y_t^i) \mid t = 1, 2, \ldots, T_{\text{obs}}\}$, where (x_t^i, y_t^i) represents the position of agent i at time t while T_{obs} represents the observation length. The goal is to predict the future trajectory of a target agent (without loss of generality, we consider agent $i = 1$ as the target agent), denoted as $\hat{\mathbf{Y}}_1 = \{(\hat{x}_t^1, \hat{y}_t^1) \mid t = T_{\text{obs}} + 1, T_{\text{obs}} + 2, \ldots, T_{\text{obs}} + T_{\text{pred}}\}$, where T_{pred} is the prediction length. The prediction is made without the use of map information or additional contextual data, relying solely on the observed trajectories.

B. Feature Extraction

To enhance the interpretability and efficiency of our motion prediction algorithm, we focus on extracting meaningful kinematic and interaction features from the agents' observed trajectories. Inspired by the kinematic and interaction metrics proposed in the Waymo Open Sim Agents Challenge [6], we derive a set of features that capture essential motion characteristics of the agents without relying on complex deep learning architectures.

Specifically, for each agent, we compute the following features:

- **Velocity Components** (v_x, v_y): The instantaneous velocity components along the x and y axes are calculated as the first-order differences of position over time:

$$v_x(t) = x(t) - x(t-1), \quad v_y(t) = y(t) - y(t-1). \tag{1}$$

- **Mean Velocity Components** $(\bar{v}_x, \bar{v}_y)$: The mean velocity components are obtained by averaging the velocity over the observed trajectory:

$$\bar{v}_x = \frac{1}{T_{\text{obs}}} \sum_{t=1}^{T_{\text{obs}}} v_x(t), \quad \bar{v}_y = \frac{1}{T} \sum_{t=1}^{T_{\text{obs}}} v_y(t). \tag{2}$$

- **Acceleration Components** (a_x, a_y): The instantaneous acceleration components are computed as the first-order differences of velocity over time:

$$a_x(t) = v_x(t) - v_x(t-1), \quad a_y(t) = v_y(t) - v_y(t-1). \tag{3}$$

- **Mean Acceleration Components** $(\bar{a}_x, \bar{a}_y)$: The mean acceleration components are calculated by averaging the acceleration over the observed trajectory:

$$\bar{a}_x = \frac{1}{T_{\text{obs}}} \sum_{t=1}^{T_{\text{obs}}} a_x(t), \quad \bar{a}_y = \frac{1}{T_{\text{obs}}} \sum_{t=1}^{T_{\text{obs}}} a_y(t). \tag{4}$$

- **Angular Momentum** $L(t)$: The instantaneous angular momentum is defined based on the cross product of the velocity and acceleration vectors:

$$L(t) = v_x(t) \cdot a_y(t) - v_y(t) \cdot a_x(t). \tag{5}$$

- **Mean Angular Momentum** $\bar{L}$: The mean angular momentum is obtained by averaging the angular momentum over the observed trajectory:

$$\bar{L} = \frac{1}{T_{\text{obs}}} \sum_{t=1}^{T_{\text{obs}}} L(t). \tag{6}$$

- **Nearest Obstacle Distance** $d_{\min}(t)$: For each agent at time t, we compute the minimum Euclidean distance to all other agents present at the same time step:

$$d_{\min}(t) = \min_{j \neq i} \sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2}. \tag{7}$$

By extracting these features, we aim to capture both the kinematic properties and interaction dynamics of the agents. The mean values serve as aggregate descriptors of the agents' overall motion tendencies, while the instantaneous values provide detailed temporal information.

C. Iterative Prediction

Building upon the extracted features, we employ an iterative approach to predict the agent's future trajectory. At each time step, we use the recent history of features to forecast the next relative displacement. Specifically, we utilize regression models trained to predict the next velocity components based on the features from the last M time steps.

Let $\mathbf{f}_{t-M+1}, \mathbf{f}_{t-M+2}, \ldots, \mathbf{f}_t$ denote the feature vectors at the most recent M time steps, where each $\mathbf{f}_t$ includes the kinematic and interaction features described earlier. Our regression model $\mathcal{M}$ takes the concatenated features as input and outputs the predicted relative displacement for the next time step:

$$(\Delta x(t+1), \Delta y(t+1)) = \mathcal{M}\left(\mathbf{f}_{t-M+1}, \mathbf{f}_{t-M+2}, \ldots, \mathbf{f}_t\right). \tag{8}$$

Subsequently, we compute the next position by adding the predicted relative displacement to the current position:

$$x(t+1) = x(t) + \Delta x(t+1), \quad y(t+1) = y(t) + \Delta y(t+1). \tag{9}$$

This process is repeated iteratively for each subsequent time step. After obtaining the new position, we update the feature vector $\mathbf{f}_{t+1}$ using the same feature extraction methods, including the newly predicted position. This updated feature vector will be used in the next iteration to predict the following time step. By iteratively applying the regression model and updating the features, we generate the agent's future trajectory over the prediction horizon. An overview of our approach is shown in Figure 1.

IV. Experiments

In this section, we evaluate our motion prediction algorithm. We outline the dataset and metrics used, conduct an ablation study to understand the impact of different features and parameters, compare our method with existing approaches to demonstrate its effectiveness, and examine the integration of our extracted features into VectorNet to validate their utility.

A. Experimental Setup

1) Dataset: We evaluate our motion prediction algorithm on the Argoverse Motion Forecasting Dataset [8], a large-scale collection of vehicle trajectories recorded by Argo AI's autonomous vehicle fleet. The dataset consists of 333,000 5-second sequences, each sampled at 10 Hz, resulting in trajectories with 50 time steps. For each sequence, the first 2 seconds (20 time steps) are used as the observation period, and the remaining 3 seconds (30 time steps) are designated for future trajectory prediction. The dataset is divided into 211,000 training sequences, 41,000 validation sequences, and 81,000 testing sequences. Each sequence focuses on a target agent whose future trajectory is to be predicted. While the dataset provides rich map information, including lane centerlines and traffic signals, our experiments rely solely on the agents' trajectories without utilizing map data.

2) Metrics: To assess the performance of our motion prediction algorithm, we employ two commonly used metrics in trajectory forecasting: the Minimum Average Displacement Error (minADE) and the Minimum Final Displacement Error (minFDE). The minADE measures the average Euclidean distance between the predicted trajectories and the ground truth trajectory over all time steps The minFDE focuses on the final position, calculating the Euclidean distance between the predicted endpoint and the ground truth endpoint.

B. Ablation Study

We conducte an ablation study to evaluate the impact of different parameters and features on the performance of our motion prediction model, as summarized in Table 1. We experiment with varying the history length M and find that $M = 4$ yield the best minADE of 1.56 and minFDE of 5.90. Comparatively, when selecting various regressors, the SVR model outperforms Random Forest and Gradient Boosting models. Additionally, we assess the importance of different features by excluding them individually; each exclusion resulted in a decline in minADE and minFDE. Notably, excluding velocity components $(v, \bar{v})$ leads to a significant performance drop, highlighting their critical role in prediction accuracy.

Model	**minADE**	**minFDE**
SVR (M=2)	1.64	6.14
SVR (M=3)	**1.56**	5.91
SVR (M=4)	**1.56**	**5.90**
SVR (M=5)	1.57	5.94
SVR (M=6)	1.59	6.05
RandomForest (M=4)	1.73	6.66
GradientBoosting (M=4)	1.88	7.11
SVR (M=4) w/o $v, \bar{v}$	9.08	30.62
SVR (M=4) w/o $a, \bar{a}$	1.73	6.53
SVR (M=4) w/o $L, \bar{L}$	1.60	**5.90**
SVR (M=4) w/o d	1.61	5.95

Table 1. Ablation Study of History Length, Regressor Choice, and Feature Effectiveness on Model Performance

C. Comparison with Other Methods

We compare our motion prediction model with several baseline and state-of-the-art methods, as presented in Table 2. Our model achieves a minADE of 1.56, which is competitive with methods that utilize map information, such as DenseTNT and SEPT. This demonstrates the effectiveness of our feature extraction approach in capturing the average trajectory behavior without relying on map data. However, our minFDE remains at 5.90, which is higher than those achieved by map-based methods. While our model performs well in predicting the overall path (as indicated by minADE), it is less precise in forecasting the final destination of the agents, suggesting that incorporating additional context or map information might further enhance the accuracy of the endpoint predictions.

Model	minADE	minFDE
Constant Velocity [8]	3.53	7.89
LSTM [8]	2.15	4.97
LSTM+social [8]	2.15	4.95
DenseTNT w/ Map [4]	1.67	3.63
SEPT w/ Map [4]	**1.44**	**3.17**
Our Method	1.56	5.90

Table 2. Comparison of Motion Prediction Performance between Our Model and Baseline/State-of-the-Art Methods

D. Embbeding into VectorNet

To further evaluate the effectiveness of our extracted features, we integrated them into the VectorNet model [7] as supplementary local features. Specifically, we incorporated the mean properties ($\bar{v}$, $\bar{a}$, $\bar{L}$), referred to as "Mean Prop", and the instantaneous properties (v, a, L, d), referred to as "t Prop", into the original VectorNet architecture. As illustrated in Figure 2, which shows the training loss over the number of iterations, the addition of these features accelerates the model's convergence. This result demonstrates that our extracted features effectively capture essential motion characteristics, thereby enhancing the learning efficiency of the model.

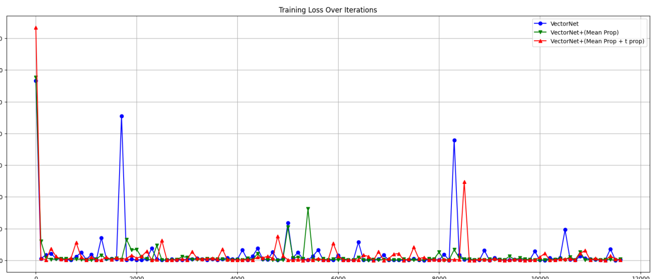

Figure 2. Training Loss Convergence of VectorNet with Supplementary Motion Features

V. Conclusion and Future Work

In this paper, we proposed a motion prediction algorithm based on feature extraction, focusing on kinematic and interaction features derived from agents' observed trajectories. By utilizing interpretable features such as velocity, acceleration, angular momentum, and nearest obstacle distance, our model achieves competitive performance without relying on complex deep learning architectures or map information. The experimental results demonstrate that our method not only reduces computational overhead but also accelerates the convergence of existing models when integrated as a sub-module, highlighting its practicality for real-time autonomous driving applications.

For future work, we plan to incorporate map-related features into our model to further enhance prediction accuracy, particularly in terms of the final destination of agents (minFDE). By integrating high-definition map information and proposing additional features that capture road topology and traffic regulations, we aim to improve the model's ability to predict more accurate and realistic trajectories, ultimately contributing to safer and more efficient autonomous driving systems.

References

[1] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. M. Paixao, F. Mutz *et al.*, "Self-driving cars: A survey," *Expert systems with applications*, vol. 165, p. 113816, 2021.

[2] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture," in *2018 IEEE intelligent vehicles symposium (IV)*. IEEE, 2018, pp. 1672–1678.

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[4] Z. Lan, Y. Jiang, Y. Mu, C. Chen, and S. E. Li, "SEPT: Towards efficient scene representation learning for motion prediction," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=efeBC1sQj9

[5] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple & efficient attention networks," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2980–2987.

[6] N. Montali, J. Lambert, P. Mougin, A. Kuefler, N. Rhinehart, M. Li, C. Gulino, T. Emrich, Z. Yang, S. Whiteson *et al.*, "The waymo open sim agents challenge," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[7] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 525–11 533.

[8] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 8748–8757.