Fakultät für Informatik
Lehrstuhl für Echtzeitsysteme und Robotik

Scene-aware and Social-aware Motion Prediction for Autonomous Driving

Alfred Nguyen

Practical Course *Motion Planning for Autonomous Vehicles* WS 2023/2024

Advisor:	Dr. Di Liu
Supervisor:	Prof. Dr.-Ing. Matthias Althoff
Submission:	20. February 2024

Scene-aware and Social-aware Motion Prediction for Autonomous Driving

Alfred Nguyen
Technische Universität München
Email: alfred.nguyen@tum.de

***Abstract*—Self-driving technology holds the promise to make travel safer and more efficient. One of the most important steps to improving self-driving cars is to understand how humans behave using vehicles. While some paths and trajectories are physically possible, some are socially unacceptable [1]. People inherently respect social norms such as yielding right-of-way or respecting personal space. This paper builds on a method to enhance the prediction of vehicles using a neural network to introduce social forces between vehicles. These social forces aim to mimick the social interaction between human-driven vehicles. In particular, we focus on improving the discrete integration method used in the neural network, reducing the error between prediction and ground truth acceleration.**

I. Problem Statement

In our previous approach, we used a discrete integration method known as ballistic integration. A challenge arises due to the discretization error while working with discrete datasets. Specifically, when we rearrange the integration formula to assess its fit with our data, we observe errors between the predicted acceleration and the ground truth.

The differences in the accelerations affect the performance of the neural network with which the previous team was working with, as the discrete integration results are fed into the network.

II. Concept Overview

As we are working with a discrete dataset, there is no way to integrate or derive over a continuous function which describes the motion of a car. Thus, a discrete integration method to describe the motion of a car is needed.

In previous attempts, other discrete integration methods were tested such as the ballistic integration model,

$$s(k+1) = s(k) + dt \cdot v(k) + \frac{dt^2}{2} a(k) \tag{1}$$

$$v(k+1) = v(k) + dt \cdot a(k) \tag{2}$$

where *dt* describes the time interval between each data point.

The main problem here was, that this integration model had issues with the accuracy of the acceleration. After rearranging both formulas to the acceleration parameter

$$a(k) = \frac{2}{dt^2}\Big(s(k+1) - s(k) - dt \cdot v(k)\Big) \tag{3}$$

$$a(k) = \frac{1}{dt}\Big(v(k+1) - v(k)\Big) \tag{4}$$

the resulting formulas do not calculate the same acceleration.

Our method revolves around the use of a linear model to estimate the acceleration of vehicles. By equating and rearranging the formula, we specifically ensure that the predicted accelerations remain consistent for both equations. The motivation behind choosing a linear model stems from the hope of better performance compared to alternative models tested during our experimentation. Through rigorous testing, we found that the selected linear model consistently outperformed others in terms of accuracy in the equality of both accelerations from both formulas. Additionally, the simplicity and interpretability of the linear model make it a good choice for motion prediction tasks, as we can comfortably use established methods to solve these systems.

To test our results we will be comparing our model to the previoulsy used ballistic integration method. Additionally, we will evaluate the performance of our model using standard linear regression metrics such as mean square error (MSE), mean absolute error (MAE), and R-squared (R^2) score.

Furthermore, we aim to extend our analysis by rearranging the model to predict velocity and distance, allowing us to assess if our model can be used as a valid discrete integration model. In the end, we will visualize our results using the existing drone-dataset-tool repo, which was provided.

III. Method

In this section, we detail the methodology used to determine the discrete integration model along with the implementation steps to validate our approach.

A. Dataset Description

For our experimentation, we utilized the inD [2], exiD [3], and rounD [4] datasets provided by the *Institut für Kraftfahrzeuge (ika) RWTH Aachen University*. The datasets offer vehicle trajectories recorded at German intersections, highway exits and entries, and roundabouts, respectively. Additionally, the datasets were provided with a tool (drone-dataset-tool) [5] that allowed us to visualize them for better understanding. The datasets include 18 columns, spanning from vehicle identifiers and lifetimes of vehicles to various columns of motion-related data points. We will primarily focus on the following columns: x_{Center}, y_{Center}, x_{Velocity}, y_{Velocity}, $x_{\text{Acceleration}}$, $y_{\text{Acceleration}}$rounDdataset.

The columns x_{Center} and y_{Center} represent the current position of the object's center in the coordinate coordinate system.

Preprocessing was done such that the columns describe the distance to the origin of the vehicle.

Velocity information is captured through columns x_{Velocity} and y_{Velocity} representing velocities in the x-axis and y-axis respectively.

Acceleration data are represented in columns $x_{\text{Acceleration}}$ and $y_{\text{Acceleration}}$ which provide information on the acceleration in the x-axis and y-axis respectively.

B. Model Selection Process

Our model selection process involved a systematic trial and error approach with a total of 8 different models. Each model underwent an evaluation process to assess its ability to accurately predict vehicle acceleration. Ultimately, the following two linear models emerged as the most suitable based on their performance metrics

$$a_{dis}(k) = \bar{c}_1\big(s(k) - s(k+1) - v(k)\big) - \bar{c}_2 a(k-1) \quad (5)$$

$$a_{vel}(k) = \bar{c}_3\big(v(k) - v(k+1)\big) - \bar{c}_4 a(k-1) \quad (6)$$

These linear models are then solved by linear regression. After training the model on the acceleration set from our dataset, we can rearrange the formulas to determine the distance and velocity formulas, similar to the ballistic integration

$$s(k+1) = s(k) + v(k) + c_1 a_{dis}(k) + c_2 a(k-1) \quad (7)$$

$$v(k+1) = v(k) + c_3 a_{vel}(k) + c_4 a(k-1) \quad (8)$$

The constants can then be determined through these calculations:

$$c_1 = \frac{1}{\bar{c}_1} \quad (9)$$

$$c_2 = \bar{c}_2 \cdot c_1 \quad (10)$$

$$c_3 = \frac{1}{\bar{c}_3} \quad (11)$$

$$c_4 = \bar{c}_4 \cdot c_3 \quad (12)$$

By rearranging the model as such, we specifically ensure that for both $s(k)$ and $v(k)$ we receive the same acceleration, as we are training both models on the same acceleration set. Thus, we receive a proper integration method for the data set on which we trained the model on, which solves the problems of the mismatched acceleration in previous attempts. As seen in the section III-A, we can see, that the columns for distance, velocity and acceleration are split into their x- and y-components. Our linear models will look like this:

Distance Model (Acceleration from distance formula):

$$\begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix}_{\text{dis}} = \begin{bmatrix} s_x(k) - s_x(k+1) - v_x(k) & -a_x(k-1) \\ s_y(k) - s_y(k+1) - v_y(k) & -a_y(k-1) \end{bmatrix} \begin{bmatrix} \bar{c}_1 \\ \bar{c}_2 \end{bmatrix} \quad (13)$$

Velocity Model (Acceleration from velocity formula):

$$\begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix}_{\text{vel}} = \begin{bmatrix} v_x(k) - v_x(k+1) & -a_x(k+1) \\ v_y(k) - v_y(k+1) & -a_y(k+1) \end{bmatrix} \begin{bmatrix} \bar{c}_3 \\ \bar{c}_4 \end{bmatrix} \quad (14)$$

Afterward, we will compare the accelerations derived from displacement, denoted as $\begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix}_{dis}$, with those derived from velocity, denoted as $\begin{bmatrix} a_x(k) \\ a_y(k) \end{bmatrix}_{\text{vel}}$, to assess the effectiveness of the distance model's acceleration estimation relative to that of the velocity model.

Using the coefficients determined from linear regression from the linear models (13) and (14), we can determine the distance and velocity using the following matrices:

Distance formula

$$\begin{bmatrix} s_x(k+1) \\ s_y(k+1) \end{bmatrix} = \begin{bmatrix} a_x(k) & a_x(k-1) \\ a_y(k) & a_y(k-1) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + \begin{bmatrix} s_x(k) + v_x(k) \\ s_y(k) + v_y(k) \end{bmatrix} \quad (15)$$

Velocity formula

$$\begin{bmatrix} v_x(k+1) \\ v_y(k+1) \end{bmatrix} = \begin{bmatrix} a_x(k) & a_x(k-1) \\ a_y(k) & a_y(k-1) \end{bmatrix} \begin{bmatrix} c_3 \\ c_4 \end{bmatrix} + \begin{bmatrix} v_x(k) \\ v_y(k) \end{bmatrix} \quad (16)$$

Hence, we obtain a discrete integration model with matching accelerations for the distance and velocity formula.

C. Evaluation Metrics and Results

The evaluation of our linear model's performance relies on the following metrics for linear regression: Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2) score.

Mean Squared Error (MSE) measures the average squared difference between the actual and predicted values. It penalizes large errors more heavily than smaller ones, making it sensitive to outliers. A lower MSE indicates better model performance.

Mean Absolute Error (MAE) calculates the average absolute difference between the actual and predicted values and isused not affected by the scale of the data. Like MSE, lower MAE values indicate better model performance.

R-squared (R^2) score quantifies the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges from 0 to 1, where a value closer to 1 indicates a better fit of the model to the data.

IV. Results

A. Linear Model

The linear models that predict the acceleration derived from the distance formula (13) and the model derived from the velocity formula (14) solved using linear regression have the following test results:

TABLE I
COMPARISON RESULTS: DISTANCE MODEL AND VELOCITY MODEL

	MSE	MAE	R^2 Score
Distance Model	3.5392×10^{-3}	1.1892×10^{-2}	9.8918×10^{-1}
Velocity Model	3.5370×10^{-3}	1.1877×10^{-2}	9.8918×10^{-1}

We can see that the predictions for the test set for both models are quite accurate having a very small MSE and MAE. We can also plot our results into graphs to have a better visual understanding of our results in figure 1 and 2.

The graph seems to fit our low MSE and MAE values, as all predicted points are lying close to the red line which

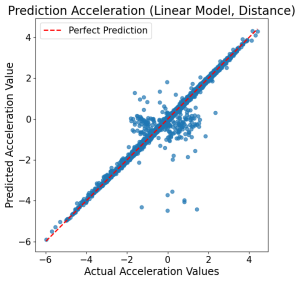

Fig. 1. Prediction results Distance Model

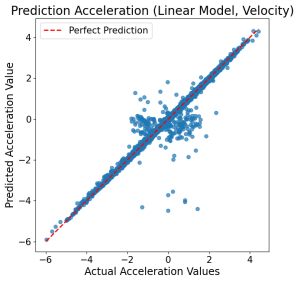

Fig. 2. Prediction results Velocity Model

indicates a perfect prediction. Interestingly we can see that most inaccuracies lie in the middle where the acceleration is close to zero. Our hypothesis is, that if our acceleration is close to zero the model has the least amount of information for inference, leading to a more scattered prediction in this region.

B. Equivalence of accelerations

Note that both results of the models look quite similar. This is due to the fact, that we specifically trained both models to fit the same dataset. The similarity can be seen in the following figure. The left shows how the predicted accelerations of both our linear models match up, while on the right we can see how the accelerations match up for the previously used ballistic integration method

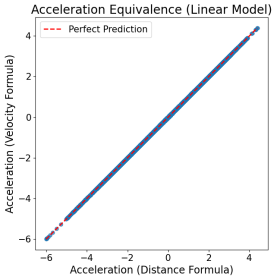

Fig. 3. Acceleration equivalence using linear model

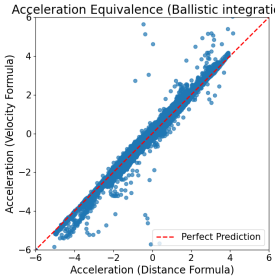

Fig. 4. Acceleration equivalence using ballistic model

Fig. 5. Caption for the entire figure

In the following, we can see the metrics of our results for the equivalence of the accelerations

TABLE II
MODEL PERFORMANCE COMPARISON

Model	MSE	MAE
Linear Model	1.8141e-06	7.5061e-04
Ballistic Integration	2.0698e+02	5.0158e-01

We can see that the accelerations from both our linear models match up almost perfectly, while the results from our previously used model are more scattered.

C. Prediction of the movement

Using the coefficients determined by the linear regression we can rearrange the model such that the model predicts the distance (15) and the velocity (16). The following shows metrics for the prediction of the distance and the velocity using the coefficients.

TABLE III
REGRESSION METRICS

	MSE	MAE	R^2 Score
Distance Formula	1.3867×10^{1}	3.7239×10^{0}	9.8926×10^{-1}
Velocity Formula	2.2635×10^{0}	1.5045×10^{0}	9.3378×10^{-1}

As we can see, all metrics are quite low for both formulas. Though, our prediction for the distance and velocity is not perfect. For a better understanding, we also visualized our findings

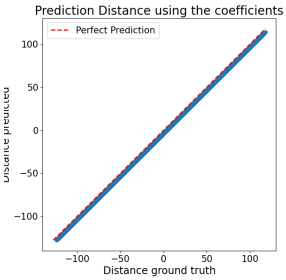

Fig. 6. Prediction of the distance using coefficients

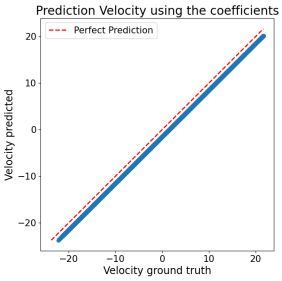

Fig. 7. Prediction of the velocity using coefficients

Again, we can see that the prediction matches up with the ground truth well. Although we can see a small offset to the red line, especially on the velocity formula, our results might be sufficient for our use case. These results can then be visualized using the drone-dataset-tool provided with the dataset. Results comparing the prediction of the moving cars to the ground truth can be found here.

V. CONCLUSION AND FUTURE WORK

A. Conclusion

We demonstrated that our linear model provided us with more equal results for both the distance model and the acceleration model. This highlights the effectiveness of using a linear model as a discrete integration system. A comparison with the previously used ballistic integration method revealed significantly better performance in terms of accuracy. Finally, using the coefficients obtained from linear regression, we were able to predict both distance and velocity with high accuracy Although some small error was observed, these results might be sufficient for our neural network. Visualizations of these predictions showcased their alignment with the ground truth, further validating the effectiveness of our integration model.

B. Future Work

In future work, we plan to continue refining our integration model to achieve even better results by further fine-tuning its parameters and training procedures.

Additionally, we aim to explore the integration of our discrete integration model with the neural network developed by the previous team. By testing the combined model and comparing its effectiveness against the old method, we can gain insights into its actual performance.

REFERENCES

[1] H. Cui, V. Radosavljevic *et al.*, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[2] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1929–1934.

[3] T. Moers, L. Vater, R. Krajewski, J. Bock, A. Zlocki, and L. Eckstein, "The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 958–964.

[4] R. Krajewski, T. Moers, J. Bock, L. Vater, and L. Eckstein, "The round dataset: A drone dataset of road user trajectories at roundabouts in germany," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[5] ika-rwth aachen, "drone-dataset-tool," https://github.com/ika-rwth-aachen/drone-dataset-tools, 2022.