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*—The abstract goes here.**

I. Problem Statement

In our previous approach, we relied on a method known as ballistic integration. However, a significant challenge arises due to the inherent discretization error with working with discrete data. 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. These errors will furthermore be explored in the results section.

The differences in the accelerations affect the performance of the neural network, as the 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. Detailed results will be shown in the results section.

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 validate the effectiveness of our approach, we plan to compare the results obtained with our linear model against the previous Ballistic Integration method, particularly focusing on the accuracy and consistency of the predicted accelerations. Additionally, we will evaluate the performance of our model using standard linear regression metrics such as R-value and MSE. Furthermore, we aim to extend our analysis by rearranging the model to predict velocity and distance, allowing us to assess its predictive capabilities. 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 employed to determine the optimized integration model along with the implementation steps undertaken to validate our approach.

A. Dataset Description

For our experimentation, we utilized the inD, exiD, and rounD datasets provided by the ika of 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 that allowed us to visualize them for better understanding. The datasets includes 18 columns ranging from the ids of a vehicle, lifetime to all various motion related information. We will primarily focus on the following columns: x_{Center}, y_{Center}, x_{Velocity}, y_{Velocity}, $x_{\text{Acceleration}}$, $y_{\text{Acceleration}}$.

The columns x_{Center} and y_{Center} represent the current position of the object's center in the coordinate coordinate system. Preprocessing can be 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 across various scenarios. Ultimately, the following two linear models emerged as the most suitable choice based on their superior performance metrics.

$$a_{dis}(k) = \bar{c}_1\big(s(k) - s(k+1) - v(k)\big) - \bar{c}_2 a(k-1) \quad (5)$$
$$a_{vel}(k) = \bar{c}_3\big(v(k) - v(k+1)\big) - \bar{c}_4 a(k-1) \quad (6)$$

This linear model is 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, similarly 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, which solves the problems of the mismatched acceleration in previous attempts.

As seen in the section Dataset Description, we can see, that the column for distance, velocity and acceleration are split into their x- and y-components. Thus our linear models will look like this:
Distance Model (Acceleration from distance formula),

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

Velocity Model (Acceleration from velocity formula),

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

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 (5) and (6) we can determine the distance and the velocity using the following matrices incorporating the x and y components.

Distance matrix

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

Velocity matrix

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

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

C. Training and Testing Procedure

For training the model, we used the $LinearRegression()$ function, provided by the sci-kit-learn Python library to train our linear model. The $train\_test\_split()$ function was utilized, with a test size of 0.3, to ensure a sufficient amount of data for evaluation.

D. Evaluation Metrics and Results

The performance of our linear model was assessed using common metrics for linear regression, including R-squared (r2) and Mean Squared Error (MSE). Detailed evaluation results, including comparisons with the old method, will be provided later in the report, offering insights into the model's predictive capabilities.

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 are quite accurate having small MSE and MAE. We can also plot our results into graphs to have a better visual understanding of our results

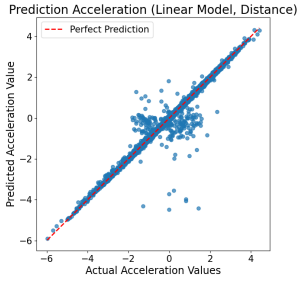

Fig. 1. Test results Distance Model

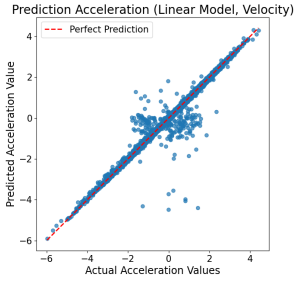

Fig. 2. Test results Velocity Model

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

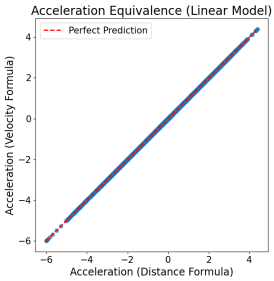

Fig. 3. Caption for Figure 1

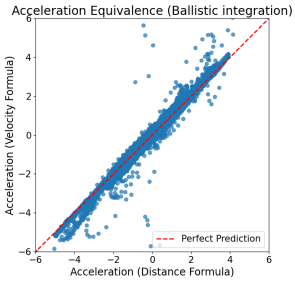

Fig. 4. Caption for Figure 2

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 clearly see that the accelerations from both our linear models match up better than the previously used ballistic integration method.

C. Predicion of the movement

Using the coefficients determine by the linear regression we can rearrange the model such that the model predicts the distance (III-B) and the velocity (16). The following shows metrics for the prediciton of the distance and the velocity using the coefficients

As we can see all metrics are conditioned well. Though we can see that the MSE and MAE values the error is quite some bit zero. This means that our prediction for the distance and velocity is not perfect. For a better understanding, we also visualized our findings

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}

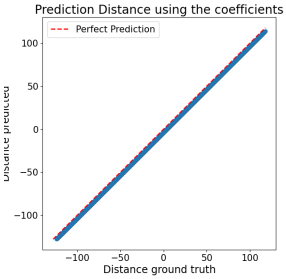

Fig. 6. Caption for Figure 1

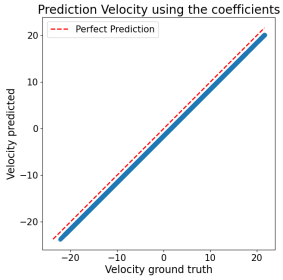

Fig. 7. Caption for Figure 2

Fig. 8. Caption for the entire figure

Again, we can see that the prediction matches up with the ground truth very well. 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. **Insert url here**

V. CONCLUSION AND FUTURE WORK

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.

the equivalence of accelerations derived from both linear models was demonstrated, highlighting the effectiveness of our approach in matching acceleration estimates A comparison with the previously used ballistic integration method revealed significantly better performance in terms of MSE and MAE.

Finally, using the coefficients obtained from linear regression, we were able to predict both distance and velocity with high accuracy, although some error was observed. Visualizations of these predictions showcased their alignment with the ground truth, further validating the effectiveness of our integration model.

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

VI. INTRODUCTION

This demo file is intended to serve as a "starter file" for IEEE conference papers produced under LaTeX using IEEEtran.cls version 1.8 and later. I wish you the best of success.

mds

December 27, 2012

A. Subsection Heading Here

Subsection text here.

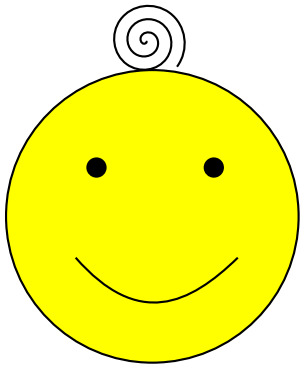

Fig. 9. A vector graphic loaded from a PDF file

Fig. 10. A bitmap graphic loaded from a PNG file

1) Subsubsection Heading Here: Subsubsection text here.

VII. Conclusion

The conclusion goes here.