2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Trajectory Prediction of Turning Vehicles based on Intersection Geometry and Observed Velocities

Atsushi Kawasaki[1] and Tsuyoshi Tasaki[2]

Abstract— **Urban intersections are known to be a hotspot for traffic accidents, and understanding of the dynamic traffic situations at intersections can help to prevent injuries. When an ego-vehicle crosses an intersection, predicting the trajectory of oncoming other vehicles is a requirement for Advanced Driver Assistant Systems. In this paper, we propose a method of trajectory prediction of turning vehicles at urban intersections. Trajectory prediction of turn maneuver vehicles is more difficult than straight-maneuver vehicle because a turning vehicle slows down as it approaches the intersections and speeds up as it leaves the intersections. Furthermore, the variation in velocities depends on factors such as an intersection angle, a corner radius. Our method generates a novel desired velocity model for trajectory prediction that takes into account intersection geometry and observed velocities of other vehicles. Specifically, we assume that the velocity becomes minimum at around the crosswalk, and calculate the velocity model by fitting past sequential velocities and estimated minimum velocity to the cubic function. Our method has the advantage of being able to predict the trajectory at any intersection and from any position. The prediction performance of our method in the real traffic scenarios is better than one of other methods.**

I. INTRODUCTION

Recently, numerous active safety driving systems and self-driving vehicles have been developed. These driving systems need to guarantee the driver's safety in a dynamically changing traffic environment, and need to understand what will happen to the environment in the future. Urban intersections are known to be a hotspot for traffic accidents, and understanding dynamic traffic situations at urban intersections can help to prevent injuries for passengers. Furthermore, long-term trajectory predictions of traffic participants at intersections are needed for path planning algorithms in self-driving vehicles. When a self-driving vehicle crosses the intersection, the predictions of oncoming vehicles are a requirement. The target maneuvers of trajectory predictions of oncoming vehicles are straight and turn-left in left-hand drive countries such as Japan and the UK. However, predicting the trajectories of turning vehicles is difficult because the vehicle undergoes a large change in velocity. A turning vehicle slows down as it approaches the intersection and speeds up as it leaves the intersection. Furthermore, the variation in velocity depends on an approach speed and an intersection geometry, including factors such as an intersection angle, a road width, and a corner radius.

Wolfermann et. al. [1] analyzed and modeled the behavior of vehicle turning at any intersection for a driving simulation. They constructed a velocity model which is represented by two cubic functions that express the deceleration and acceleration. The coefficients of these functions are modeled assuming a normal distribution, and are estimated by the maximum likelihood method. However, their model is only for simulations, and is difficult to apply to the prediction in real traffic situations because the velocity model can be generated only from a fixed position (the entry position into the intersection). In the trajectory prediction, it is necessary to be able to generate and update the model at any position.

We propose a trajectory prediction method for turning vehicles at any intersection and any position. We construct a novel desired velocity model for trajectory prediction based on intersection geometry and observed velocities. Assuming that the velocity becomes minimum at around the crosswalk, we calculate the velocity model by fitting past sequential velocities and estimated minimum velocity to the cubic function. We define the position where the velocity becomes minimum on the map, and refer to this position as the "velocity control point". Furthermore, in order to predict the trajectory for many time-steps into the future, we integrate our desired velocity model with Extended Kalman Filter (EKF). Experiments are conducted in the real environment. Compared with conventional methods, the proposed prediction method can enhance the state-prediction performance.

II. RELATED WORKS

Recently, the problem of predicting the trajectory of traffic participants has been actively studied in the field of robotics [2]. In [3], [5], prediction methods based on vehicle motion model were proposed for a lane change situation. In [3], [4], the trajectory components are modeled as a quintic polynomial. Kalman filters have been widely used to predict trajectories while accounting for uncertainty of vehicle motion [5], [6]. Kim et. al. [5] defined the desired yaw rate which is required for lane changes and curves, and incorporated the desired yaw rate in EKF. Furthermore, probabilistic trajectory prediction methods using Gaussian Mixture Models [7] and Monte-Carlo Simulation [8] have also been proposed.

Machine learning techniques have also been used to predict trajectories. The model parameters of vehicle trajectories are learned through a Gaussian process [9]. More sophisticated models which consider vehicle interactions have been introduced, including the coupled hidden Markov model [11] and dynamic Bayesian network[10]. In recent years,

[1] Atsushi Kawasaki is with the Corporate R&D Center, Toshiba Corporation, Japan. atsushi1.kawasaki@toshiba.co.jp

[2] Tsuyoshi Tasaki was with the Corporate R&D Center, Toshiba Corporation, Japan. He is now with Meijo University, Japan. tasaki@meijo-u.ac.jp

978-1-5386-4451-5/18/$31.00 ©2018 IEEE

prediction methods using deep neural networks have been proposed. In [12], a dynamic occupancy grid map is used as input to a deep convolutional network. Among the several DNN architectures, the recurrent neural network is widely employed for learning sequential structures. Kim et. al. [13] fed a sequence of vehicle coordinates into long short-term memory and produced probabilistic information about the future location over an occupancy grid map. However, these machine learning methods depend on training data and suffer from the disadvantage that they can not predict what will happen in an unknown environment.

Prediction methods specialized for intersections have recently been developed. As mentioned in Section I, Wolfermann et. al. [1] modeled the behavior of turning vehicles at any intersection, but their model is for simulations and cannot be generated from any position. Tran et. al. [9] proposed a velocity model for turn maneuvers by using Gaussian process regression. However, their model is focused on one intersection situation and cannot be applied to any intersection. Liebner et. al. [14] also proposed a velocity model which takes into account the curvature and can be applied to any intersection. Inspired by the above previous works, we aim to predict the trajectory of turning vehicles at any intersection and any position in the real environment. Our method yields better performance by using more intersection information (such as intersecting angle, curvature radius, lateral distance from the sidewalk, and position of the crosswalk)

III. System Overview

Measurement data was acquired from the real traffic scenarios at urban intersections. We used an experimental vehicle which is equipped with a visual sensor and a 360° laser scanner, Velodyne HDL-32e. The raw data acquired by the laser scanner are in the form of a 3D point cloud. This paper concerns intersections which are four road junction and have crosswalks, and does not concern scramble crossings or roundabouts.

The inputs of the prediction algorithm come in two main varieties: 1) other vehicle state, such as position, orientation, velocity, yaw rate, and acceleration; and 2) intersection geometry, such as a two-dimensional trajectory map, an intersection angle, a curvature radius, and a lateral distance from a sidewalk. The state is obtained from a vehicle detection and tracking system. The detection and tracking system is based on Particle filter whose input is 3D point cloud data. The likelihood of Particle filter is calculated by fitting a rectangle to the point cloud based on [15]. Fig. 1 (b) shows the result of the detection and tracking system. The two-dimensional trajectory map is prepared corresponding to each oncoming lane and maneuver, and consists of a point sequence at intervals of 1 m. We refer to the point in the point sequence as “waypoint”. The map is created by B-spline curves whose control points are selected from the ground truth positions of traffic participants. Black dots in Fig. 6 second column show the trajectory map. Additionally, because we focus on the trajectory prediction of turning vehicle, the maneuver of the traffic participant is assumed to be known.

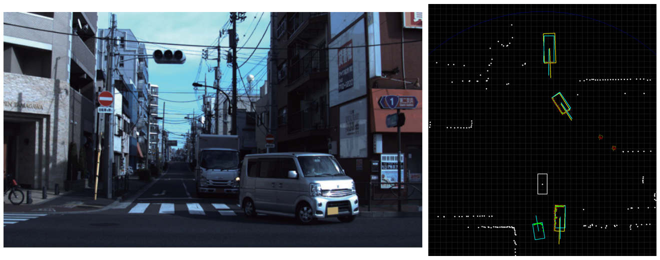

(a) (b)

Fig. 1. (a) Traffic scenario showing the vehicles which are the object of prediction. (b) Results of the vehicle detection and tracking system. The white rectangle shows the ego-vehicle, cyan rectangles show detected results by particle filter, and yellow rectangles show the ground truth.

The prediction algorithm generates the desired velocity for turning vehicles at intersections. Additionally, one generates the desired yaw rate based on [5]. Future position of other traffic participants is predicted by integrating these desired models and the current other vehicle state with EKF. The prediction state is computed every 0.1s up to a maximum prediction horizon of 4.0 s. The timestep of the observation is also 0.1s, and the prediction is conducted at each timestep.

IV. Trajectory Prediction of Turning Vehicle

A. Generating Desired Velocity Model

In order to predict the trajectory of a turning vehicle, we generate the future desired velocity. Our desired velocity model consists of two cubic functions and is generated by using intersection geometry. To solve the problem that the velocity model of [1] cannot be created from any position, we establish the velocity control point in the map. The velocity control point is defined as the position where the velocity becomes minimum. We assume that the velocity becomes minimum at around the crosswalk in consideration of the interaction between vehicles and pedestrians. In this paper, the velocity control point is equal to the waypoint which is the closest to the crosswalk. Fig. 2 shows the spatiotemporal relationship between the velocity control point and the desired velocity. The desired velocity model is divided into two parts, an inflow part $v_{in}(t)$ and an outflow part $v_{out}(t)$, and the boundary between them is defined as the moment the vehicle reaches the minimum velocity:

$$v_{in}(t) = c_{1,in}t^3 + c_{2,in}t^2 + c_{3,in}t + c_{4,in} \tag{1}$$

$$v_{out}(t) = c_{1,out}t^3 + c_{2,out}t^2 + c_{3,out}t + c_{4,out} \tag{2}$$

First, we focus on the calculation method of $v_{in}(t)$. We calculate $v_{in}(t)$ by fitting the observed past sequential velocities and estimated minimum velocity to the cubic function. Because the current velocity of the traffic participants which is observed by the tracking system is unstable, we utilize the sequential velocities over a period of 1.0 s. Additionally, the minimum velocity can be statistically calculated based on [1], but we define its velocity as the initial estimate $\overline{v}_{min}$

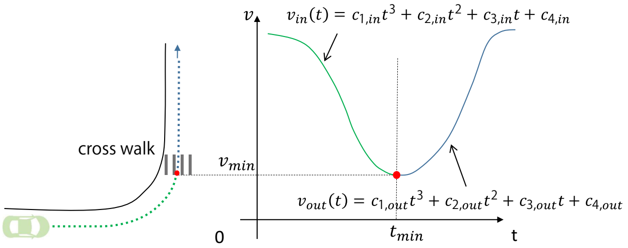

$v_{in}(t) = c_{1,in}t^3 + c_{2,in}t^2 + c_{3,in}t + c_{4,in}$
v
cross walk
v_{min}
$v_{out}(t) = c_{1,out}t^3 + c_{2,out}t^2 + c_{3,out}t + c_{4,out}$
0
t_{min}
t

Fig. 2. Spatiotemporal relationship between the velocity control point and the desired velocity.

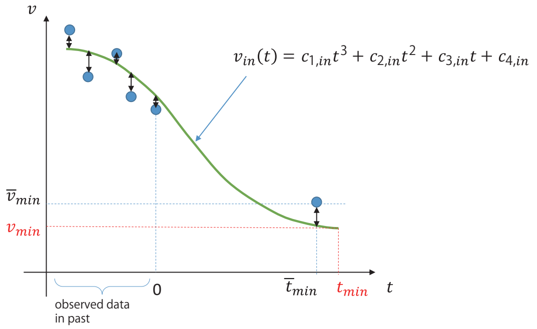


Fig. 3. The inflow part $v_{in}(t)$ of the desired velocity is calculated by minimizing the sum of the squared residuals which is the difference between observed past velocities and initial minimum velocity $\bar{v}_{min}$, and the fitted value provided by a model.

before the fitting. We fit the data to the cubic equation by minimizing the error between the cubic function and the velocities which contain the observed velocities and $\bar{v}_{min}$. Fig. 3 shows a schematic of the fitting.

The initial minimum velocity $\bar{v}_{min}$ incorporates the influences by the intersection geometry and driver characteristics, such as an approach speed v_{int}, an intersection angle θ_{int}, a curb radius r_{int} and a lateral distance of the vehicle in the exit from the sidewalk, l_{int}. The approach speed v_{int} is defined as the velocity at the stop line in [1]. In this paper, v_{int} is represented by calculating the velocity when arriving at the stop line assuming constant acceleration. Assuming that $\bar{v}_{min}$ follows a normal distribution, $\bar{v}_{min}$ is empirically estimated as a linear function of the influencing factors as follows:

$$\bar{v}_{min} = a_1 + a_2 v_{int} + a_3 r_{int} + a_4 \theta_{int} + a_5 l_{int} \tag{3}$$

where $a_1, \ldots, a_5$ are calculated by maximum likelihood estimation from the real traffic data. We use the parameters described in [1].

Additionally, the fitting requires the initial estimates of $v_{in}(t)$ and t_{min} at the time when the velocity becomes minimum. In order to calculate these values, the following equations are acquired from the constraint condition of the velocity control point:

$$0 = 3c_{1,in}t_{min}{}^2 + 2c_{2,in}t_{min} + c_{3,in} \tag{4}$$

$$L = \frac{1}{4}c_{1,in}t_{min}{}^4 + \frac{1}{3}c_{2,in}t_{min}{}^3 + \frac{1}{2}c_{3,in}t_{min}{}^2 + c_{4,in}t_{min} \tag{5}$$

where L is the path length from the current vehicle position to the velocity control point. We define the initial estimates of $v_{in}(t)$ and t_{min} as $\bar{v}_{in}(t)$ and $\bar{t}_{min}$(hereinafter, the over line shows the initial estimate). If the state of the traffic participant is accurately observed, $c_{4,in}$ and $c_{3,in}$ is equal to v_0 and a_0 which is the velocity and acceleration at the current time. However, in order to consider the observation error, in the estimation step $\bar{c}_{4,in}$ and $\bar{c}_{3,in}$ are defined as variables which change in step of 0.5 in the range of $[v_{min} - 3.0, v_{min} + 3.0]$ and $[-0.3G, 0]$, respectively. G denotes the gravitational acceleration. The variable range of $c_{4,in}$ is decided based on the standard deviation of observed velocity error, and the variable range of $c_{3,in}$ is based on the evidence that comfortable deceleration is generally less than $0.3G$. Equations (1), (4), and (5) are solved for $t_{min}, c_{1,in}$, and $c_{2,in}$, as follows:

$$\bar{t}_{min} = \frac{-3(\bar{c}_{4,in} + \bar{v}_{min}) + \sqrt{9(\bar{c}_{4,in} + \bar{v}_{min}) + 12L\bar{c}_{3,in}}}{\bar{c}_{3,in}} \tag{6}$$

$$\bar{c}_{1,in} = \frac{\bar{c}_{3,in}}{\bar{t}_{min}{}^2} + \frac{2(\bar{c}_{4,in} - \bar{v}_{min})}{\bar{t}_{min}{}^3} \tag{7}$$

$$\bar{c}_{2,in} = \frac{3\bar{c}_{1,in}\bar{t}_{min}{}^2 + \bar{c}_{3,in}}{2\bar{t}_{min}} \tag{8}$$

When Equation (9) becomes minimum changing $\bar{c}_{3,in}$ and $\bar{c}_{4,in}$, $\bar{v}_{in}(t)$ is set as the initial estimate of $v_{in}(t)$.

$$\bar{E} = \sum_{i=-9}^{0} (v_{obs}(i\Delta t) - \bar{v}_{in}(i\Delta t))^2 + (\bar{v}_{min} - v_{in}(\bar{t}_{min}))^2 \tag{9}$$

where v_{obs} is represented by the observed past velocity.

Next, the cubic function $v_{in}(t)$ is optimized using the initial value $\bar{v}_{in}(t)$. $c_{3,in}$ and $c_{4,in}$ are removed from $v_{in}(t)$ based on the constraint conditions of the velocity control point, equation (4) and (5), as follows:

$$\begin{aligned} v_{in}(t) = {} & c_{1,in}(t^3 - 3t_{min}{}^2 t + \frac{5}{4}t_{min}{}^3) \\ & + c_{2,in}(t^2 - 2t_{min}t + \frac{2}{3}t_{min}{}^2) + \frac{L}{t_{min}} \end{aligned} \tag{10}$$

where the objective parameter is defined as $\mathbf{C} = [\begin{array}{ccc} c_{1,in} & c_{2,ijn} & t_{min} \end{array}]^{\mathrm{T}}$. The sum of squared residuals is given by

$$E = \sum_{i=-9}^{0} w_1(v_{i\Delta t} - v_{in}(i\Delta t))^2 + w_2(\bar{v}_{min} - v_{in}(\bar{t}_{min}))^2 \tag{11}$$

where w_1 and w_2 are the weights. Since the number of observed velocities are larger, the weights are set as follows:$w_1 = 0.1, w_2 = 1.0$. Fitting the cubic function is conducted by using an iterative non-linear Gauss Newton-method:

$$\mathbf{C}^{(s+1)} = \mathbf{C}^{(s)} - (\mathbf{J}^{\mathrm{T}}\mathbf{J})^{-1}\mathbf{J}^{\mathrm{T}} e_t(\mathbf{C}^{(s)}) \tag{12}$$

where $e_t(\mathbf{C})$ is the residual error and $\mathbf{J}$ is the Jacobian matrix, $\mathbf{J} = \partial e_t(\mathbf{C})/\partial \mathbf{C}$. The superscript shows the iteration number of the optimization. In this paper, the maximum iteration is 10. This generates the inflow part of the desired velocity model.

Next, we explain the outflow part of the velocity model. We calculate $v_{out}(t)$ which passes through v_{min} and the legal speed v_{max} and which has a derivative of zero at v_{min} and

v_{max}. The unknown variables $c_{1,out}$ and $c_{2,out}$ are obtained by using [1]. These variables are assumed to follow a gamma distribution. The parameter of these distributions are also estimated as linear functions of the influencing factors, such as intersection geometry and legal speed limit.

B. Integrating Desired Velocity with EKF

We use EKF to predict the vehicle position for many time steps into the future. Prediction of the vehicle position needs not only desired velocity but also desired yaw rate. We implement the desired yaw rate model based on [5]. Their model outputs the yaw rate for smoothly passing following the trajectory map. We define the vehicle state as $\mathbf{x_p} = \begin{bmatrix} p_{x,p} & p_{y,p} & \theta_p & v_p & \gamma_p & a_p & \dot{\gamma}_p \end{bmatrix}^T$ where each state is a longitudinal position, lateral position, angle, velocity, yaw rate, acceleration, and yaw acceleration.

EKF is an extension of Kalman Filter, which estimates the states of a dynamical system:

$$\mathbf{x_p}[t] = \mathbf{f_p}(\mathbf{x_p}[t-1]) + \mathbf{w_p}[t-1] \tag{13}$$

$$\begin{aligned} \mathbf{z_k}[t] &= \mathbf{H_p} \cdot \mathbf{x_p}[t] + \mathbf{v_p}[t] \\ &= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \cdot \mathbf{x_p}[t] + \mathbf{v_p}[t] \\ &= \begin{bmatrix} v_{des} & \gamma_{des} \end{bmatrix}^T \end{aligned} \tag{14}$$

with

$$\mathbf{w_p}[t] \sim N(0, \mathbf{W_p}[t]) \tag{15}$$

$$\mathbf{v_p}[t] \sim N(0, \begin{bmatrix} V_v & 0 \\ 0 & V_\gamma \end{bmatrix}) \tag{16}$$

where $\mathbf{f_p}$ shows uniformly accelerated motion model. The desired velocity v_{des} and the desired yaw rate γ_{des} are used as the virtual measurement of EKF. The desired velocity v_{des} consists of equation (1) and (2). If $t < t_{min}$, v_{des} is equal to $v_{in}(t)$, else to $v_{out}(t)$. Suppose that the process noise $\mathbf{w_p}$ and the measurement noise $\mathbf{v_p}$ have a normal distribution. For the noises V_γ and $\mathbf{W_p}$, refer to [5]. The noise in the desired velocity V_v is the variance between the observed present velocity and the predicted past velocity corresponding to the present time. We define V_v as follows:

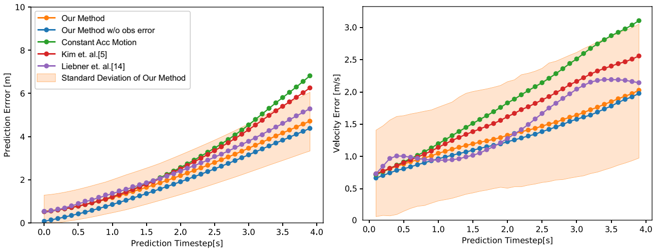
$$V_v = \begin{cases} V_{v,min} & (V_{v,\sigma} \leqq V_{v,min}) \\ V_{v,\sigma} & (V_{min} < V_{v,\sigma} < V_{v,max}) \\ V_{v,max} & (V_{v,max} \leqq V_{v,\sigma}) \end{cases} \tag{17}$$

$$V_{v,\sigma} = \frac{1}{N} \sum_{i=1}^{N} ((v_0 - v_i[i]) \times \frac{10}{i})^2 \tag{18}$$

where v_i is the predicted velocity before time-step i. In our implementation, the number of v_i, N, is set to 10. A difference in predicted time steps is normalized by multiplying by $10/i$. $V_{v,min}$ and $V_{v,max}$ are constant values which are experimentally acquired, and set to 1.0 and 15.0, respectively.

V. EXPERIMENT

This chapter presents some results of the prediction algorithm evaluating real data from intersections at Kanagawa, Japan. The dataset was collected from 10 different intersections and consist of 65 left-turn vehicles. The ground truth positions of traffic participants are manually taught from the point cloud data. Some of the observed traffic situations contain pedestrians who cross at a crosswalk. The pedestrian positions are not used in our algorithm, but are taught from the point cloud data.



(a) Prediction error (b) Velocity error

Fig. 4. Comparison of prediction error and velocity error of our method with those of other methods. Orange and blue show our method with observation error and without observation error, respectively.

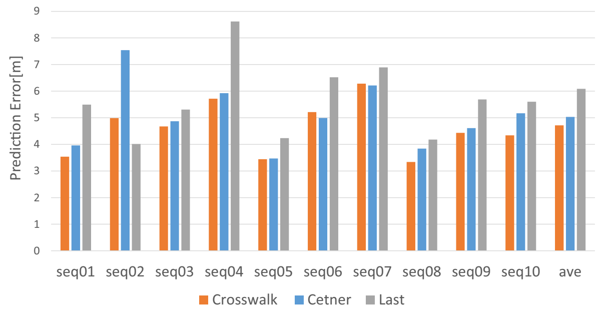
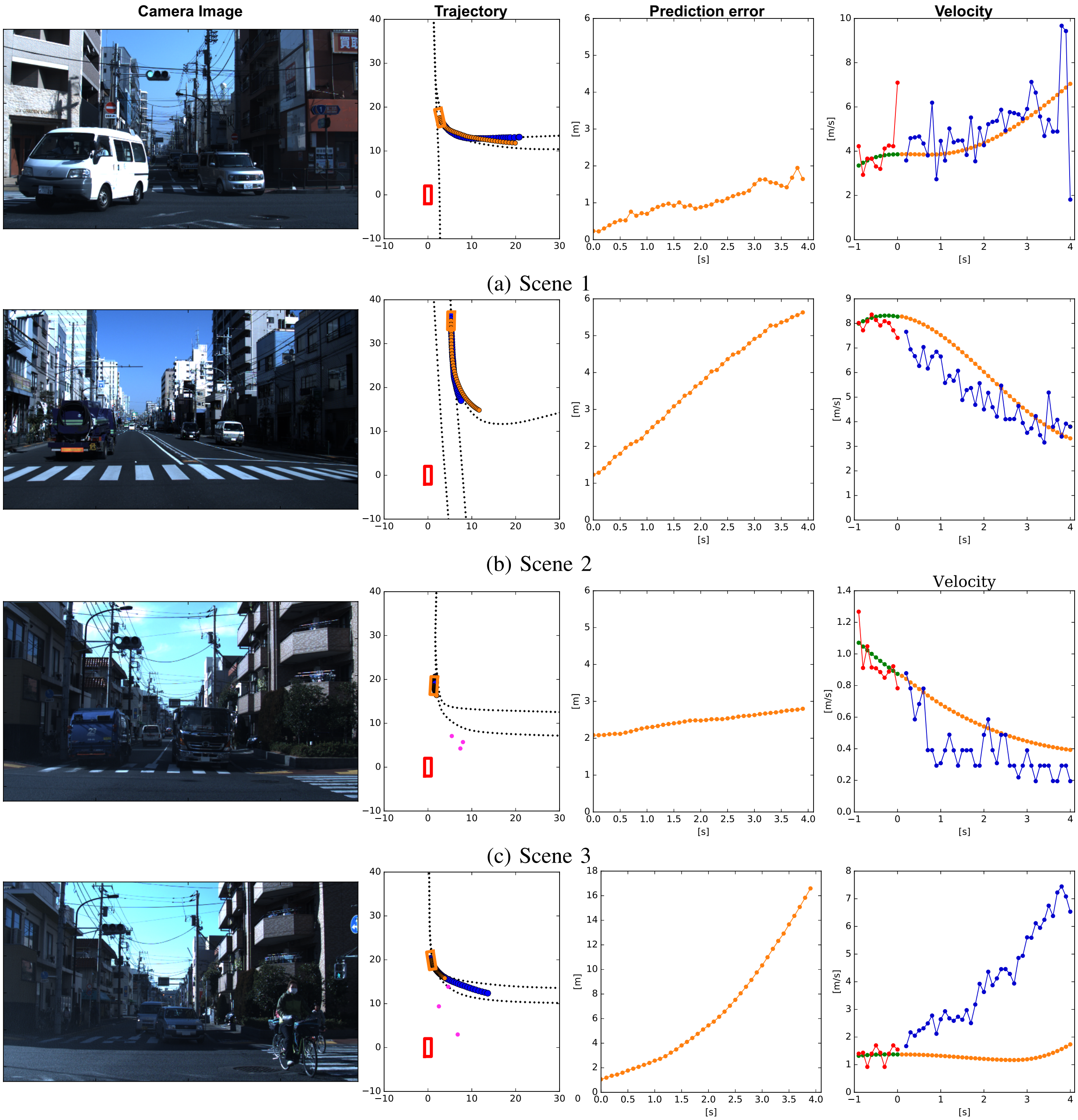


Fig. 5. Comparison of the prediction error in the position of the velocity control point.

A. Evaluating Prediction Performance

We evaluate the prediction accuracy qualitatively and quantitatively. Representative scenarios of the prediction results are shown in Fig.6. The first column shows a picture taken from the camera mounted on the experimental vehicle. The second column shows the predicted trajectories. Orange and blue indicate the predicted results and true trajectories, respectively. The black point sequences represent the trajectory maps. The third column shows the prediction error, which is defined as the Euclidean distance between the true position and predicted position at each prediction timestep. The last column represents the velocity graphs which consist of desired velocity (orange), the observed past velocity (red), past velocity after fitting (green), and the true velocity (blue). The true velocity is calculated from the difference between the true position at the current time and the true positions one and two timesteps earlier. From trajectory results and velocity graphs, we found that predicted trajectory is close to the true trajectory and the desired velocity forms a smooth curve in consideration of observed past velocities. The predicted trajectories bend along the trajectory map by the desired yaw rate. From Fig.6(a) and (b) we found that our method can predict the trajectory from any position. Fig.6(c) is an interaction scenario involving a vehicle and pedestrians. Because the desired velocity



(a) Scene 1

(b) Scene 2

(c) Scene 3

(d) Scene 4

Fig. 6. Each row shows the result of representative scenario. The first column shows camera images and the second column shows trajectory results. Red and orange rectangles indicate the ego-vehicle and other vehicles, respectively. Orange and blue trajectories show predicted and true trajectories, respectively. Purple circles show pedestrians. The third column shows prediction errors with observation error, and the fourth column shows the velocity graphs which consist of desired velocity (orange), observed past velocity (red), past velocity after fitting (green) and true velocity (blue).

becomes minimum at the crosswalk and the observed past velocities are low when there are pedestrians, the predicted minimum velocity is close to zero. On the other hand, Fig. 6(d) shows the scenario where the worst error occurs. The reason for the large error is interaction between the vehicle and pedestrians. As mentioned above, our method can reduce the minimum velocity close to zero when pedestrians cross the crosswalk. However, the intention of another driver to start their vehicle is unknowable because our method does not employ the pedestrian information. We consider that the more surrounding information, such as the position of pedestrians, are needed for the prediction of the timing of starting.

Next, we quantitatively compare our method with other velocity models. Four methods were used for comparison: our method, constant accelerated motion model, and two previous works [5], [14]. The method of [5] is based on the assumption that longitudinal acceleration decreases to zero with the corresponding time constant. The method of [14] generates the desired velocity model based on the curvature of the trajectory map. In order to make a fair comparison between the methods, the implementation other than the velocity model is same. Fig. 4 shows the average of the prediction error and the velocity error up to 4.0s. We found that the prediction error and velocity error of our method are the smallest. The constant accelerated motion model and the method proposed by [5] cannot reproduce the velocity changes such as slowing down and speeding up. The method proposed by [14] can reproduce the velocity change, but this method takes only the curvature into consideration and is not expressive enough.

Additionally, we evaluate the accuracy of only the prediction part. The prediction error above includes the observation error which occurs due to the vehicle detection and tracking. Prediction error and velocity error are calculated again by using the true position at the current time as input to the prediction algorithm. Each error is plotted in Fig. 4 (blue line). The average prediction error with and without observation error at the end of the prediction horizon of 4.0 s is 4.7 m and 4.3 m, respectively.

B. Evaluating the Position of Velocity Control Point

In our method, the velocity control point is set to the waypoint which is the closest to the crosswalk because it is thought that the velocity becomes minimum before the vehicle crosses the crosswalk. In this section, we quantitatively describe where the velocity control point is the best position. Three positions of the velocity control point were used for comparison. One is the waypoint which is the nearest to the crosswalk. Another is the center of the range including the waypoints whose curvatures are equal to or more than the threshold. The third is the last point of the above range. Fig.5 shows the average of the prediction error for each position of the velocity control point and for each sequence of the dataset. The rightmost graph represents the average of all sequences and shows that the best position of the velocity control point is the waypoint nearest to the crosswalk. When pedestrians do not exist (seq 6, 7), the accuracy in the case that the velocity control point sets to the center is slightly higher. On the other hand, when pedestrians exist, the accuracy in the case that the velocity control point is closest to the crosswalk is higher. It is possible to change the position of velocity control point depending on whether pedestrians exist or not. However, the safety of the ego-vehicle is higher in the case that velocity control point sets to the crosswalk because sometimes pedestrians cannot be detected due to obstacles such as other vehicles.

VI. CONCLUSIONS

We proposed a method of trajectory prediction of turning vehicles at urban intersections. Our method generated a novel desired velocity model for the trajectory prediction that takes into account observed velocities and intersection geometry, such as, intersection angle, corner radius, lateral distance from the sidewalk, and position of the crosswalk. Our method has the advantage of being able to predict the trajectory at any intersection and from any position. Our method yields better prediction performance than other prediction methods.

In future work, we need to solve the problem that the prediction error become large when interactions between vehicles and pedestrians occur. We will improve the method of generating the desired velocity in consideration of the multiple state of vehicles and pedestrians.

REFERENCES

[1] A. Wolfermann, W. K. Alhajyaseen, and H. Nakamura, Modeling speed profiles of turning vehicles at signalized intersections. International Conference on Road Safety and Simulation (RSS2011), 17 pp, 2011.

[2] S. Lefevre, D. Vasquez, and C. Laugier, A survey on motion prediction and risk assessment for intellgient vehicles, ROBOMECH Journal, vol. 1, no. 1, pp. 1-14, 2014.

[3] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, Vehicle trajectory prediction based on motion model and maneuver recognition, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2013), pp. 4363-4369, 2013.

[4] J. H. Kim and D. S. Kum, Threat prediction algorithm based on local path candidates and surrounding vehicle trajectory predictions for automated driving vehicles, IEEE International Conference on Intelligent Vehicles Symposium (IV2015), pp. 1220-1225, 2015.

[5] B. Kim, and K. Yi, Probabilistic and holistic prediction of vehicle states using sensor fusion for application to integrated vehicle safety systems, IEEE Transactions on Intelligent Transportation Systems, vol. 15, pp. 2178-2190, 2014.

[6] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, Map-based long term motion prediction for vehicles in traffic environments, IEEE International Conference on Intelligent Transportation Systems (ITSC2013), pp. 2166-2172, 2013.

[7] J. Wiest, M. Hoffken, U. Krebel, and K. Dietmayer, Probabilistic trajectory prediction with gaussian mixture models, IEEE International Conference on Intelligent Vehicles Symposium (IV2012), pp. 141-146, 2012.

[8] M. Schreier, V. Willert, and J. Adamy, An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments, IEEE Transactions on Intelligent Transportation Systems, vol. 17, pp. 2751-2766, 2016.

[9] Q. Tran and J. Firl, Modelling of traffic situations at urban intersections with probabilistic non-parametric regression, IEEE International Conference on Intelligent Vehicles Symposium (IV2013), pp. 334-339, 2013.

[10] T. Gindele, S. Brechtel, and R. Dillmann, R, Learning driver behavior models from traffic observations for decision making and planning, IEEE Intelligent Transportation Systems Magazine, vol, 7, pp. 69-79, 2015.

[11] M. Brand, N. Oliver, and A. Pentland, Coupled hidden Markov models for complex action recognition, IEEE International Conference on Computer Vision and Pattern Recognition(CVPR1997), pp. 994-999, 1997.

[12] S. Hoermann, M. Bach, and Klaus Dietmayer, Dynamic Occupancy Grid Prediction for Urban Autonomous Driving: A Deep Learning Approach with Fully Automatic Labeling, arXiv preprint arXiv:1705.08781, 2017.

[13] B. Kim, C. M. Kang, S. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi, Probabilistic Vehicle Trajectory Prediction over Occupancy Grid Map via Recurrent Neural Network, arXiv preprint arXiv:1704.07049, 2017.

[14] M. Liebner, F. Klanner, M. Baumann, C. Ruhhammer, and C. Stiller, Velocity-based driver intent inference at urban intersections in the presence of preceding vehicles, IEEE Intelligent Transportation Systems Magazine, vol. 5, pp. 10-21, 2013.

[15] T. D. Vu and O. Aycard, Laser-based detection and tracking moving objects using data-driven markov chain monte carlo, IEEE International Conference on Robotics and Automation(ICRA2009), pp. 3800-3806, 2009