Estimating the Probabilities of Surrounding Vehicles’ Intentions and Trajectories using a Behavior Planner

Hatem Darweesh 1) Eijiro Takeuchi 1) 2) Kazuya Takeda 1) 2)

1) Nagoya University, Graduate School of Information Science 2) TierIV Inc.
Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8603 Japan (E-mail: hatem.darweesh@g.sp.m.is.nagoya-u.ac.jp)

Received on May, 17, 2019

ABSTRACT: Estimating the intentions and trajectories of other vehicles is critical to achieving stable, long-term planning and decision making in autonomous driving systems. This paper introduces a novel technique for estimating the intention and trajectory probabilities of surrounding vehicles. The first step is to use a deterministic behavior planner to identify possible trajectories and behaviors. The behavior planner models an average driver who follows the driving rules according to information provided by a road network map, and also provides the control signal used in the following step. Next, a customized particle filter is integrated with the planner to model the uncertainty of various trajectories and behaviors, using multiple sensing cues such as pose, velocity, acceleration and turn signal use. The proposed method supports various sensor modalities, depending on the availability of additional sensing information. Finally, by including the sensor data the probabilistic process is able to estimate the probabilities of various trajectories and intentions. The proposed method is generic to any driving situation supported by the behavior planner. Intentions such as ‘go forward’, ‘turn right’, ‘turn left’, ‘yield’ and ‘stop’ are supported by the proposed method. Our proposed method is evaluated using multiple, complex, simulated driving situations, and then comparing the simulation’s ground truth to the estimated probabilities. Evaluation criteria are how early and accurately our system can estimate the driving trajectory and intention probabilities of the vehicle. Our results show that the method can successfully estimate driver intention and trajectory in multiple complex situations, such as left turns, right turns and during passing, as well as at four-way intersections and bus stops.

KEY WORDS: Safety, intention determination, driving act, driver behavior, autonomous driving, intention estimation, trajectory estimation, uncertainty modeling, behavior planner.

1. Introduction

Developing autonomous vehicles systems has become easier, with many organizations and start-ups now competing to achieve safer and more reliable autonomous driving systems. Until all vehicles on the road become autonomous, the problem of interaction between human-driven and autonomous vehicles will remain. This stems from the difficulty autonomous vehicles have understanding the intentions of humans, due to driver uncertainty, indecision and human error. One solution is defensive autonomous driving, with the expectation that other drivers will operate their vehicles responsibly. Another solution is to estimate the other drivers’ probable intentions and future trajectories and then plan suitable action and a trajectory based on the estimation results [1][2][3].

Trajectory and intention estimation such in [4][5][6][7] uses deterministic model of the other vehicles, problem with this approach is embedding the other vehicles intention into the estimated trajectory. This results in the lack of explaining the ego vehicle decisions. Another problem is deterministic approaches ignores the uncertainty factor and require very accurate physical model of the other vehicles. Machine learning approaches are utilized in [8][9][10] such as Deep Neural Networks (DNN) and Recurrent Neural Networks (RNN). The problem with these approaches is that the system learns from limited set of data to handle one situation such as highway or four way intersection which results in lack of generalization, especially to situations that are rarely occur. Other research uses state machine and search based algorithms for intention estimation such as [11][12] although this approach is practical but it doesn’t consider the sensing uncertainty. Probabilistic based approaches to handle uncertainty is introduced in [8][13][14][15][16]. They utilize Hidden Markov Models (HMM), Markov Decision Processes (MDP) and Partially Observable MDP (POMDP). Although they show promising results but they are computationally expensive. Researchers often use very simple model of the surrounding vehicles to achieve better performance.

The main contribution of this work is to estimate intention and trajectory of surrounding vehicles for most of complex urban driving situations in real time. We accomplish that by using behavior planner such in [12] as the complex motion model integrated with non parametric probabilistic filter (particle filter) [17][18] to handle uncertainty. Behavior planners model both trajectories and behaviors of the surrounding vehicle providing a complete model not only to the motion of the vehicle but also to traffic rules constraints.

Other contribution is achieved, improving the particle filter performance to operate in real time by reducing the state space dimension using multiple particle filters for each trajectory and intention. The third contribution is the flexibility of the proposed method, as it allows the use of different kinds of sensor information similar to [19]. Continuous states such as position, heading, velocity and acceleration are used. Additional contextual
discrete sensing data can also be included with minimal implementation changes, such as driver gestures or Vehicle to Vehicle (V2V) communication to provide more sensing inputs. Figure 1 shows a block diagram of the introduced algorithm.

![Block Diagram of Proposed Algorithm](image)

Fig. 1 Proposed system architecture, consisting of two main modules; a Passive Behavior Planner and Particle Filters. Environment representation is very important for state modeling. Some studies as in [10][19][15] have used very simple, custom built road networks which provides some driving rules in a simulation environment, but not using a standard mapping format makes it difficult to integrate with other autonomous driving systems. In this work we used standard road network map format that is used in full autonomous driving stack [20].

Sensor configuration for this work depends on the framework and the utilized behavior planner being used. In the Autoware framework [20] and OpenPlanner [12] the data for surrounding vehicles come from LIDAR based object detection which outputs position and size. A fusion between LIDAR, RADAR and Camera could be used to improve detection accuracy and reduce uncertainty.

‘Intention’ and ‘behavior’ are generally used to describe a driver’s action, such as doing nothing, to the complicated, three-step passing of a slower vehicle. In this paper we will use ‘behavior’ to refer to the ego vehicle’s actions or the estimated actions of other vehicles before filtering. When we use ‘intention’ we mean the estimated set of actions after filtering [21][22].

In Section 2 we provide a system overview and describe the architecture of the particle filter algorithm. In Section 3 we discuss uncertainty modeling using the particle filter algorithm and our reason for choosing this method. One of the main contributions of this paper is the novel way we use the behavior planner and particle filter, and this is explained in Section 4. Section 5 describes our evaluation environments, scenarios and results. Conclusion and plans for future work are discussed in Section 6.

2. Method Overview

One of the ideas proposed in this paper is the extraction of all the possible driving trajectories and their embedded traffic rules from a road network map. These serve as a starting point for both the particle filter and the behavior planner. The extracted trajectories are used to initialize the prior distributions for the various driving behaviors to be estimated. From the point of view of the behavior planner, these trajectories are the planning base.

The main modification we introduce to the basic particle filter algorithm is the use of a deterministic behavior planner to estimate the motion model used to generate the control signal. This control signal is used to re-sample the remaining particles, which are then weighted using multiple sensing cues such as position, heading, velocity, acceleration and turn signals. Finally, the posterior probability for each particle filter is calculated and the best intention and trajectory is selected.

Figure 1 shows that proposed algorithm consists of three main modules; a trajectory extractor, a passive behavior planner and the particle filters. Tracked objects are the output of the third-party object detection and tracking module. Finally, the algorithm outputs intention and trajectory probabilities.

2.1. Trajectory Extraction

The trajectory extraction module extracts all of the possible trajectories from the road network map, along with the embedded traffic rules, as shown in Figure 2. In addition, we add kinematically feasible branches to model exiting from and merging onto main roads.

2.2. Passive behavior planner

The behavior planner [12] utilized in this paper uses road network maps and a state machine representing the available driving behaviors to generate a driving decision. The behavior planner then generates an obstacle-free trajectory for the low-level controller to execute. In addition to the map, the planner uses the ego vehicle’s behavior. The state machine used for behavioral planning is shown in Figure 3.

![Behavior Planner State Machine](image)

Fig. 2 Extracting all possible vehicle trajectories from a road network map. After all of the existing paths are extracted, the additional branching paths (right) are added.

Fig. 3 Behavior planner state machine.
The planner is modified to work passively, thus, it acts as the driver, and seeks out what behavior the vehicle should execute next, based on its previous intention and the current pose. It represents an expert driver in the loop by generating the behaviors that should be executed next. It does not take sensing uncertainties into account, however.

2.3. Particle Filter

The main reason for using a particle filter is to address uncertainties such as observational uncertainty and motion uncertainty. Although there are different types of filters that could be used to model these uncertainties, particle filters are chosen for two main reasons, one related to implementation and the other related to the fact that the probability distribution of the posterior is unknown \[17\],[19].

Many trajectory and intention estimation algorithms either ignore the type of sensor measurements used or strictly specify them. In this work, the implementation of the particle filter is intended to allow the integration of multiple sensing cues such as vehicle pose, heading, velocity, acceleration and turn signal information. In the future, this flexible design will be able to utilize vehicle to vehicle communication (V2V) \[15\] and driver gesture recognition \[23\].

3. Uncertainty Modeling

Different driving behaviors have different observation profiles, based on their velocity, acceleration, orientation, turn signal and current road rule information. The more information that is available from the object detection and tracking step, or from other sensing modules, the more accurate the probability estimation. Clusters of samples (particles) for each behavior can be generated based on the assumption that the range of driving behaviors is limited to the ones matching those generated by expert drivers. The same behaviors are modeled by the behavior planner.

3.1. Reasons for using Particle Filters

Particle filters allow the use of several weak measurement cues to be accumulated into a strong estimator, and more sensing cues can be added later for better estimation. These cues could be situation or sensor reading dependent. In the case of sensor reading dependency, co-variance could be included in the particle weighting process. Another reason for choosing particle filtering is that the posterior probabilities could be at any distribution, rather than only a Gaussian one, because they are non-parametric Bayesian filters \[17\].

3.2. Algorithm Setup

The key role of the particle filter is to represent the posterior for calculating the belief state of \( x_{bel}(x) \) using a set of random state samples drawn from its distribution. The basic particle filter algorithm is shown in Algorithm 1. The default algorithm cannot be used due to missing control signal \( u \), so only the detected state for the surrounding vehicle is available. Control signal \( u \) is calculated from sensing signal \( z \) and road network map data using the passive behavior planner, as shown in Equation 1.

Particle filter set up consists of defining a state space, observations and particle representation. Table 1 describes the problem state space.

\[ u_t \approx \text{OpenPlanner} \left(Z_p, X_{t-1}, \text{map} \right) \] (1)

Observations could come from different modules, such as object detection and tracking, or turn signal detection. In this study we use modules inside the Autoware \[20\] open-source software package. Table 2 shows the utilized observations.

A separate particle filter for each trajectory and behavior is used to improve performance by reducing the dimensions of each particle filter. Each particle moves according to the behavior planner's estimated \( u \). Particle re-sampling uses a normal distribution as the prior distribution, as shown in Figure 4. When a new trajectory appears, a new set of particles are sampled. For the branching trajectories, the particles are re-sampled at a fixed time interval. The supported intention states are shown in Figure 5.

Table 1 State space elements.

| \( pose(x, y, \theta) \in \mathbb{R}^3 \) | \( v \in \mathbb{R} \) | \( a \in \mathbb{R} \) |
| \( T, S \in \{-1,0,1,2\} \equiv \{\text{None}, \text{L}, \text{Both}, \text{R}\} \) |
| \( Traj \in \text{waypoints} (p_0, \ldots, p_n); (x, y) \in \mathbb{R}^2 \) |
| \( Beh \in \{\text{Forward, Stop, Yield, Right, Left}\} \) |

Table 2 Observations.

| \( pose(x, y, \theta) \in \mathbb{R}^3 \) | \( v \in \mathbb{R} \) | \( a \in \mathbb{R} \) |
| \( T, S \in \{-1,0,1,2\} \equiv \{\text{None}, \text{L}, \text{Both}, \text{R}\} \) |

Fig. 4 Prior sampling distribution, manually designed with fixed parameters for the particle filter’s motion model initial condition.

Copyright © 2019 Society of Automotive Engineers of Japan, Inc. All rights reserved
The main modifications introduced to uncertainty representation due to the use of particle filters are, first, the use of a deterministic behavior planner as the motion model for particle filter, Equation 1. The behavior planner is assumed to function as an expert driver who will follow the traffic rules as indicated by the road map. This also enables us to use the simple motion model estimated by the behavior planner, thus replacing the behavior planner with a better one should result in a better model. Second, using particle filters increases flexibility, as it is easy to add more measurement cues and behaviors. Adding more sensing information such as turn signal and gesture detection is very straightforward, making it easy to integrate them with the current algorithm, improving estimation accuracy.

4. Steps of Estimation Algorithm

The proposed algorithm consists of four main steps; trajectory extraction, particle sampling, measurement update (weight calculation) and resampling.

4.1. Trajectory Extraction

We extract possible driving trajectories from the road network map and then add two additional trajectories to represent branching right and branching left. The left side of Figure 3 shows the center lines of the road network map, while the right side of the figure shows the four extracted trajectories; the two branching trajectories, one for going straight and another one for turning left. The long path represents the global plan for the simulated vehicle.

4.2. Particle Sampling

We sample new particles for new behaviors and trajectories, thus when a new trajectory appears in a scene, new particles need to be generated. Figure 5 shows the state network representing the intentions of other vehicles and the transitions between them. Extracting the trajectory from the previous step helps us to represent the initial probability distribution needed to achieve particle sampling. Figure 4 shows the distributions that we draw for each particle filter.

4.3. Weight Calculation

In this step the sampled particles are filtered by calculating how far they are from the original distribution and sensing cues. The weight for each particle is calculated against the sensing data, such as position, direction, velocity, acceleration and turn signal information. It is important to take sensing noise into consideration. We can use the same sensing noise value for each measurement, but that will not be accurate because the sensors vary in their degree of precision.

The following set of equations shows our method of weight calculation for a single particle. In Equation 2, the total weight \( W_{all} \) is calculated by aggregating the measurement cues weights. \( P \) is the importance factor for each measurement cue such that \( P_{pose} + P_{dir} + P_{vel} + P_{acc} + P_{sig} = 1 \). This enable choose which sensing cue should contribute to the posterior distribution the most. In Equations 3 and 4, \( W_{pose} \) is the distance deference between observed position \((x, y)\) and particle position \((\hat{x}, \hat{y})\). In Equation 5, direction weight \( W_{dir} \) is calculated using the difference between both observed \( \theta \) and sampled \( \hat{\theta} \). Discontinuity in angle differences could be problematic, so a special function is used in Equations 5 and 9 which contains information about vehicle heading semantics. In Equation 6, velocity weight \( W_{vel} \) is the difference between velocity as measured by the object tracker, \( v \), and the expected velocity calculated by the passive behavior planner, \( \nu \). In Equation 7, acceleration weight \( W_{acc} \) is the difference between acceleration measured by tracking, \( a \), and the expected acceleration from the passive planner, \( \nu \). In Equations 8 and 10, turn signal weight is the similarity between what is observed by the vision module, \( S \), and the expected signal direction as estimated by the passive planner, \( \nu \). Each measurement cue weight is multiplied by an importance factor \( P \), as shown in Equation 2.

\[
W_{all} = W_{pose} \cdot P_{pose} + W_{dir} \cdot P_{dir} + W_{vel} \cdot P_{vel} + W_{acc} \cdot P_{acc} + W_{sig} \cdot P_{sig}
\]

\[
d = \sqrt{(x_{z} - x)^2 + (y_{z} - y)^2}
\]

\[
W_{pose} = \left\{ \begin{array}{ll} \frac{1}{d}, & \text{if } d > pose\_err \\ \frac{1}{pose\_err}, & \text{otherwise} \end{array} \right.
\]

\[
W_{dir} = \left\{ \begin{array}{ll} \frac{1}{dir\_err}, & \text{if } |\theta_{diff}| > dir\_err \\ \frac{1}{\theta_{diff}}, & \text{otherwise} \end{array} \right.
\]

\[
W_{vel} = \left\{ \begin{array}{ll} \frac{1}{vel\_err}, & \text{if } |v_{z} - v| > vel\_err \\ \frac{1}{v_{z} - v}, & \text{otherwise} \end{array} \right.
\]

\[
W_{acc} = \left\{ \begin{array}{ll} \frac{1}{acc\_err}, & \text{if } |a_{z} - a| > acc\_err \\ \frac{1}{a_{z} - a}, & \text{otherwise} \end{array} \right.
\]

\[
W_{sig} = S_{diff}(S_{z}, S)
\]

\[
\theta_{diff}(\theta_{1}, \theta_{2}) = \left\{ \begin{array}{ll} 2\pi - (\theta_{1} - \theta_{2}), & \text{if } \theta_{1} > \theta_{2} \\ 2\pi - (\theta_{2} - \theta_{1}), & \text{if } \theta_{2} > \theta_{1} \end{array} \right.
\]
There are multiple approaches for weight calculation; in our experiments we tried several methods. Finally we selected equation 2-10 as they are the most computationally efficient. Theoretically equation 2 is supposed to calculate the joint probability of all sensing information \[^{[17]}\]. One problem with using joint probability and multiply all probabilities is lots of particle doesn’t lie in the vicinity of the correct state. That leads to particle deprivation which requires the use of huge number of particles. Another problem is when one sensing cue importance factor becomes very low other sensing weights contributions decrease too. To solve both problems in equation 2 we used summation of the weights instead of the joint probability with the assumption that all sensing cues are independent.

4.4. Re-Sampling

In this step, particles with small weights are removed and are replaced with new samples drawn from particles with the highest confidence level. This sampling step relies on the motion model that estimates the path the observed vehicle is expected to follow, which in this study is the behavior planner. It estimates the motion and intentions of other vehicles according to the observed circumstances. Equations 11 to 17 define the re-sampling step and our motion model.

\[
\begin{align*}
    (v_p, \omega_p, s_p, beh_p) &= \text{OpenPlanner}(x_{t-1}, beh_{t-1}, traj_{t-1}) \\
    x_t &= \text{Move}(x_{t-1}, v_p, \omega_p, beh_p)
\end{align*}
\]

In this experiment, Figure 6, there are two possible trajectories the simulated vehicle can follow. Forward trajectory until the intersection and then continuing forward towards Goal \(F\), or a forward trajectory until the intersection and then turning left towards Goal \(L\). Assumptions for this experiments are that the traffic light is green and that turn signal sensing is not available.

5. Evaluation Results

In this section we present a detailed analysis of the results of our estimation process. We evaluated our proposed intention and trajectory estimation system using multiple simulated driving situations. The objective of these evaluations is to demonstrate that the proposed method can accurately estimates trajectories and intention probabilities at various driving scenarios, such as three-way intersection, four-way intersection, intersection with and without stop signs, and bus stops. These driving scenarios are depicted from the National Highway Traffic Safety Administration (NHTSA) report \[^{[24]}\].

The simulated vehicles operate using an active version of the behavior planner. State and generated behaviors for the simulated vehicles are considered as the ground truth. There is no information sharing between the planner for the simulated vehicles and the ego vehicle’s planner. An additional perception simulation module received the ground truth for vehicle pose and size, which was then published in a similar message format as the object detection algorithm. The perception simulator added random noise to the position and point cloud data before publishing. Starting conditions are fixed for each experimental trial in order to facilitate a comparison between the results. Finally, the ground truth from the simulated vehicle and the estimation results from our proposed method are compared to measure performance. All simulations used actual intersections from the road network map of the area around Nagoya University.

Two values are used to discriminate intentions and trajectories. Average weight, which is the average of all of the particles’ normalized weights, represents signal behavior at each time step. The average is calculated after the re-sampling step. The second value is probability, which is the ratio between the remaining particles (after elimination of the weak ones) to the maximum number particles. This probability is calculated before the re-sampling step. Both measurements can be used in different ways, but usually average weight changes more smoothly over time.

In the particle filter re-sampling step, weak particles are replaced with new particles. These new particles are randomly drawn from the normal distribution around the previous best state. Sometimes when new particles are generated, the weight calculation spikes. After few iterations it converges to the general trend. Increasing the number of particles minimize these spikes but hurts the performance. In the experiments we conducted these changes (spikes) never big enough to affect the estimation process.

Intentions is abbreviated in the graphs by the first character of each intention, Moving Forward (\(F\)), Stopping (\(S\)), Branching Left (\(L\)), Branching Right (\(R\)), Yielding (\(Y\)), Accelerating (\(A\)), and Parking (\(P\)).

5.1. Three-way Intersection

In this experiment, \(F\) and \(L\) offer forward trajectory until the intersection and then continuing forward towards Goal \(F\) or a forward trajectory until the intersection and then turning left towards Goal \(L\). Assumptions for this experiments are that the traffic light is green and that turn signal sensing is not available.
Figure 7 shows the speed profile and the state transition for the planner generated behaviors. The target intentions are mapped to the velocity profile graph. Figure 8 shows the average weights of the particles associated with each intention (Forward, Stop and Yield). The most probable intention to be selected by the planner at each time step for Trajectory L is shown in Figure 9. For Trajectory F, Intention weights associated with it is shown in Figure 10.

Trajectory probability estimation can be achieved either by averaging the weights of all of the particles associated with the trajectory, or by only using the particles associated with the Forward intention. Intuitively, Forward intention weights should be sufficient to discriminate between trajectories, and the data from our experiments support that intuition.

5.2. Four-way Intersection

In this experiment, Figure 11, the simulated vehicle starts from the common starting position and moves towards one of three goals; Forward (F), Left (L) or Right (R). The experiment repeats from starting position to each goal, and during each trial we enable or disable the ability to use turn signals as a sensing cue. In addition, we enable or disable stopping at the stop sign. The analysis of our results shows the effect of adding more sensing information on the accuracy of the resulting probabilities.

5.2.1. Goal F: No Stopping, turn signal sensing on

From the perspective of a human driver, it is highly probable that the vehicle will continue to move forward without stopping or slowing down if no right or left turn signal is observed and the vehicle is moving faster than turning speed. Thus, the estimated intention should be Forward and the estimated trajectory should be the ‘go forward’ trajectory, Trajectory F. Figure 12 shows the resulting average weights used to estimate the intention of the observed vehicle in Figure 13. At times, the weights are similar, so confidence when selecting an intention is low, but at other times it is clear that there is only one likely intention, although there is still a small possibility of other intentions. The critical intersection zone is indicated in the graph, approximately between the 150th and 400th time steps. In Figure 12 it is obvious that the predicted intentions match the ground truth behaviors with high confidence most of the time, especially in the intersection zone. In this experiment, Trajectory F was predicted with high probability, as shown in Figure 14, where the probability of Trajectory F is about 0.8.
5.2.2. Goal F: No stopping, turn signal sensing off

Results in Figures 15 and 16 are the same as in the previous experiment, indicating that in this scenario turn signal information does not have much effect on the final estimation results. However, when we compare Figures 12 and 15, we see that the intention weights are slightly higher when turn signal information is included. The same relationship could be observed for trajectory estimation when comparing Figure 14 (with turn signal information), and Figure 17 (without turn signal information).

5.2.3. Goal F: Stopping, turn signal sensing on/off

Figure 18 shows the weights and selected intentions when turn signal information is not included. Figure 19 shows the weights and selected intentions when turn signal sensing is included. We can see from the results how using the turn signal data affects the results.

When the vehicle stops at the intersection, we lose forward velocity as a clue for helping choose the most likely trajectory, so there is not enough evidence for the particle filter to converge towards one of the three trajectories, especially when turn signal information is not taken into consideration. In Figure 20, we can see that the algorithm was confused and couldn’t estimate the correct trajectory when the vehicle stopped for the stop sign, but that it estimated correctly once the vehicle started to move again. This problem is resolved by using the turn signal data, as shown in Figure 21, the prediction algorithm gives a higher confidence to the correct trajectory even though the vehicle is stopped. Although the algorithm selected the Forward trajectory as best choice, the Left and Right trajectories still have probabilities as high as the intended one. This could happen when the driver...
forgets to use the turn signal or changes his mind at the last moment.

5.2.4. Goal L

This experiment is similar to section 5.1 experiment. Figure 22 compares the effect of using turn signal data on how early we can estimate the vehicle’s trajectory. In the worst-case scenario (no turn signal data) the algorithm can still estimate the vehicle’s trajectory accurately once the vehicle starts turning at the intersection (within 1 to 2 seconds).

5.2.5. Goal R

When conducting the Goal R experiment, we obtained similar result as in our Goal L experiment. We are able to obtain consistent estimation results, but we still could not predict the driver’s intention when the vehicle stopped at the intersection without turn signal information. In fact, even human drivers cannot correctly predict which direction a vehicle will turn from a stop without turn signal information, until the moment that the vehicle starts to turn.

5.3. Parked vehicles and bus stop scenarios

Other frequent driving scenarios include attempting to avoid parked vehicles and buses waiting at bus stops. A simple rule that humans use is to observe and understand a driver’s hazard lights vs. turn signals, i.e., whether the lights on both sides of the vehicle are flashing, or only on the right or left side. Another strategy is to determine if the vehicle is stopped by the side of the road without any stop line or traffic light nearby.

5.3.1. Bus stop scenario

Fig. 23 A bus is stopping at a bus stop. The ego vehicle is traveling in the same lane as the bus, while another vehicle is traveling in the adjacent lane.
In Figure 23 the ego vehicle is coming from behind in the same lane as the bus. Another vehicle is moving forward in the adjacent lane, blocking the ego vehicle from passing the bus. The algorithm needs to estimate with high confidence that this bus is stopping and that the stop is not caused by other traffic circumstances, such as a traffic light, stop sign or normal traffic jam. To help handle such a scenario, we added a new intention (Parking). Figure 24 shows the estimated intentions of the bus against its tracked velocity profile. In this experiment hazard lights are used to differentiate general stopping from waiting by the roadside.

Fig. 24 Data associated with a bus stopping at a bus stop. Top: weights associated with each intention. Bottom: most probable intention at each time step.

5.4. Passing scenario

In case of passing the parked bus, ego vehicle’s planner needs to identify if the vehicle in the adjacent lane is yielding or not. If a yielding intention is estimated with enough confidence, the ego vehicle can safely change lanes. Figure 25 shows the estimated intentions for the vehicle driving alongside the ego vehicle. The results indicate that the vehicle has yielded to let the ego vehicle pass the bus. In this case, the ego vehicle planner will decide a safe lane change trajectory.

Fig. 25 Data associated with the vehicle in the adjacent lane. Top: weights associated with each intention. Bottom: most probable intention at each time step.

6. Conclusion

In this study we proposed an intention and trajectory estimation method for predicting the behavior of surrounding vehicles. The final goal of the proposed method is to associate a probability with each intention and trajectory. This is a very important step before decision making in an autonomous driving system’s planning process.

Our method works by first associating a particle filter with each intention and trajectory. Then, a deterministic behavior planner is used to provide the expected motion model for each particle. After that, the particle weights are calculated by finding the difference between the expected state and the actual sensing information, using multiple sensing inputs such as position, velocity, acceleration and turn signal status. Our results show that our proposed method has the ability to accurately discriminate between various possible intentions and trajectories in a variety of complex driving situations. Contributions of this work are as follows:

- By using a behavior planner, we were able to model the intentions and trajectories of surrounding vehicles. Our results show that any new behavior the planner supports, including ‘socially aware’, ‘complex’ or ‘rare’, can be easily integrated into the estimator, and that our model works reliably prior to the filtering step. Explained in Section 5.
- By using a particle filter, we were able to address the problem of sensing uncertainty. Explained in Section 3.
- Particle filter performance was improved by reducing particle state dimensionality. This was achieved by assigning a separate filter to each intention or trajectory. Explained in Section 4.
- By using multiple sensing cues, our method allows earlier estimation and increases estimation confidence. The proposed method also has the ability to handle sensing problems such as variations in sensor accuracy, adding a new sensor or losing an existing sensor. Explained in Section 5.

The next step is to integrate the estimated probabilities with the ego vehicle’s behavior planner, in order to plan smoother velocity profiles and generate safer decisions.

From the target application point of view, this work is targeting level 4 and level 5 Autonomy. More computational platform based optimization is needed to support level 3 active safety application. The algorithm design is flexible enough to improve the performance using multi processor or GPU based in vehicle computational platforms.

ACKNOWLEDGMENTS

We would like to acknowledge that this work was supported by the Japan Science and Technology Agency (JST) project on Open Innovation Platform with Enterprises, Research Institute and Academia (OPERA). Also we would like to thank everyone who helped us with advice, or who took time to review this study.

REFERENCES

[1] C. Katrakazas, M. Quddus, W.H. Chen, and L. Deka, “Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions,” In
Transportation Research Part C: Emerging Technologies, 60: pp.416-442, 2015.

[2] S. Ammoun and F. Nashashibi, “Real time trajectory prediction for collision risk estimation between vehicles.” In Intelligent Computer Communication and Processing, IEEE 5th International Conference, pp. 417-422, 2009.

[3] A. Polychronopoulos, M. Tsogas, A. J. Amditis and L. Andreone, “Sensor fusion for predicting vehicles’ path for collision avoidance systems,” IEEE Transactions on Intelligent Transportation Systems 8, no. 3 (2007): 549-562.

[4] G. Xie, H. Gao, L. Qian, B. Huang, K. Li and J. Wang, “Vehicle Trajectory Prediction by Integrating Physics- and Maneuver-Based Approaches Using Interactive Multiple Models,” IEEE Transactions on Industrial Electronics 65, no. 7: 5999-6008, 2018.

[5] A. Houenou, P. Bonmifiat, V. Cherfaoui and W. Yao, “Vehicle trajectory prediction based on motion model and maneuver recognition,” Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., pp. 4363-4369, Nov. 2013.

[6] P. Liu, A. Kurt and U. Ozguner, “Trajectory prediction of a lane changing vehicle based on driver behavior estimation and classification,” Proc. 17th Int. IEEE Conf. Intell. Transp. Syst., pp. 942-947, 2014.

[7] R. Pepy, A. Lambert and H. Mounier, “Reducing navigation errors by planning with realistic vehicle model,” In Intelligent Vehicles Symposium, pp. 300-307, IEEE, 2006.

[8] D. Lenz, F. Diehl, M. Truong Le and A. Knoll, “Deep Neural Networks for Markovian Interactive Scene Prediction in Highway Scenarios,” In IEEE Intelligent Vehicles Symposium (IV) June 1114, 2017, Redondo Beach, CA, USA.

[9] J. Morton, T. A. Wheeler and M. J. Kochenderfer, “Analysis of Recurrent Neural Networks for Probabilistic Modeling of Driver Behavior,” IEEE Transactions on Intelligent Transportation Systems, pp. 110, 2016.

[10] D. J. Phillips, T. A. Wheeler and M. J. Kochenderfer, “Generalizable intention prediction of human drivers at intersections,” In Intelligent Vehicles Symposium (IV), pp. 1665-1670, IEEE, 2017.

[11] C. Urmon, J. Anhalt, H. Bae, J. D. Bagnell, C. Baker, R. E. Bittner, T. Brown, M. N. Clark, M. Darms, D. Demitrish, J. Dolan, D. Duggins, D. Ferguson, T. Galatali, C. M. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, M. Likhachev, B. Litkouhi, A. Kelly, M. McNaughton, N. Miller, J. Nickolaou, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, V. Sadekar, B. Salesky, Y. W. See, S. Singh, J. M. Snider, J. C. Struble, A. T. Stentz, M. Taylor, W. R. L. Whitaker, Z. Wolkowicki, W. Zhang and J. Ziglar, “Autonomous driving in urban environments: Boss and the urban challenge,” Journal of Field Robotics, Special Issue on the 2007 DARPA Urban Challenge, Part I, vol. 25, no. 1, pp. 425-466, June 2008.

[12] H. Darweesh, E. Takeuchi, K. Takeda, Y. Ninomiya, A. Sujjito, L. Morales, N. Akai, T. Tomizawa and S. Kato, “Open Source Integrated Planner for Autonomous Navigation in Highly Dynamic Environments,” J. Robot. Mechatron., Vol.29, No.4, pp. 668-684, 2017.

[13] S. Brechtel, T. Gindele and R. Dillmann, “Probabilistic Decision-Making under Uncertainty for Autonomous Driving using Continuous POMDPs,” IEEE 17th International Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, vol 17, 2014.

[14] M. Bouton, A. Cosgun and M. J. Kochenderfer, “State Planning for Autonomously Navigating Urban Intersections,” IEEE Intelligent Vehicles Symposium (IV) June 11-14, 2017, Redondo Beach, CA, USA.

[15] C. Hubmann, M. Becker, D. Althoff, D. Lenz and C. Stiller, “Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles,” In Intelligent Vehicles Symposium (IV), pp. 1671-1678. IEEE, 2017.

[16] H. Bai, S. Cai, N. Ye, D. Hsu and W. S. Lee, “Intention-aware online POMDP planning for autonomous driving in a crowd,” 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, pp. 454-460, 2015.

[17] S. Thrun, W. Burgard and D. Fox, “Probabilistic Robotics (Intelligent Robotics and Autonomous Agents),” The MIT Press, 2005.

[18] N. de Freitas, “A tutorial and exercises on particle filters for finite mixture models with unknown number of components,” 2001.

[19] S. Vacek, S. Bergmann, U. Mohr and R. Dillmann, “Rule-based tracking of multiple lanes using particle filters,” IEEE International Conference on Multi-sensor Fusion and Integration for Intelligent Systems, September 3-6, 2006, Heidelberg, Germany.

[20] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda and T. Hamada, “An open approach to autonomous vehicles,” IEEE Micro, vol. 35, no. 6, pp. 60-68, Nov 2015.

[21] K. Lidström and T. Larsson, “Model-based estimation of driver intentions using particle filtering,” 11th International IEEE Conference on Intelligent Transportation Systems, Beijing, China, October 12-15, pp. 1177-1182, 2008.

[22] A. Koenig, T. Rehder and S. Hohmann, “Exact inference and learning in hybrid Bayesian Networks for lane change intention classification,” In Intelligent Vehicles Symposium (IV), pp. 1535-1540. IEEE, 2017.

[23] Liu, Hongyi and Lihui Wang. “Gesture recognition for human-robot collaboration: A review,” International Journal of Industrial Ergonomics 68: 355-367, 2018.

[24] NHTSA, “Pre-Crash Scenario Typology for Crash Avoidance Research”, April 2007.