Precise and Efficient Model-Based Vehicle Tracking Method Using Rao-Blackwellized and Scaling Series Particle Filters *

Mengwen He[1,3], Eijiro Takeuchi[1,3], Yoshiki Ninomiya[2,3] and Shinpei Kato[3,4]

Abstract— **A precise and efficient tracking technique is essential for an intelligent vehicle to interact with the surrounding vehicles on the road. In this paper, we propose a model-based vehicle tracking method using the Rao-Blackwellized particle filter (RBPF) and scaling series particle filter (SSPF).**

To ensure that the accuracy and speed of the proposed method are better than those standard particle-based tracking methods, we introduce the latest sensor measurement into the motion estimate. A dynamic Bayesian network and its theoretical base are derived to support this improvement. To track the high-dimensional states of the target vehicle including its position, orientation, motion, and geometry in the proposed general tracking method, we use the SSPF for the motion estimate and geometry fitting and the RBPF for the geometry estimate.

To demonstrate the effectiveness of our tracking method, we conduct an evaluation experiment using two intelligent vehicles around Nagoya University, Japan. The total distance traveled by the vehicles in this experiment is approximately 10 km, and the experimental environment contains flat roads, narrow streets, and steep ramps. To demonstrate the accuracy and speed of our tracking method, we extract the ground-truth from our intelligent vehicles and compare its performance with that of state-of-art RBPF-based approach.

I. Introduction

Vehicle tracking is an important technique for a smart vehicle to interact with the surrounding vehicles. Precise tracking of a target vehicle can sensitively detect the driver's intension and will lead to reliable predictions for ensuring safe driving. For example: (1) in the parking-lot, when a parked vehicle starts moving out, its small position change should be tracked for collision avoidance; (2) on the road, when a running vehicle tends to change lane, its small orientation change should be tracked for safe overtaking; and (3) at the crossing without traffic light, when a speedy vehicle comes out, its speed change should be tracked for deciding whether to pass the crossing.

The Bayesian approach is a common and efficient method to solve object-tracking problems, and a particle filter (PF) is a type of non-parametric Bayesian filter [1]. A PF has the advantage of being able to represent complex beliefs, especially those that have multiple high-probability regions (called multi-mode beliefs), unlike parametric Bayesian filters include the Kalman filter and its extensions, extended Kalman filter, and unscented Kalman filter.

However, the computational cost of PFs is higher than that of parametric Bayesian filters and usually increases exponentially according to the dimensionality of the tracking state, which is referred to the curse of dimensionality [2]. The dimensionality of a tracked vehicle's state varies for different applications, but the vehicle's position, orientation, motion, and geometry are always considered when the light detection and ranging (LiDAR) sensor is used for vehicle tracking. Such a high-dimensional state will lead to high computational complexity in the basic PF.

The Rao-Blackwellized PF (RBPF) and scaling series PF (SSPF) are two techniques developed to cope with the high-dimensional state in the basic PF. The RBPF utilizes the Rao-Blackwellization technique to marginalize some of the parameters in the tracking state and then use the Gaussian estimate to track them [3], which can decrease the state's dimensionality in the basic PF. The SSPF is an annealing-based iterative PF that performs searches using a series of successive refinements to gradually scale the precision from low to high [4]. Therefore, the SSPF can focus limited computational resources on the regions with higher probability and can thus solve the relatively high-dimensional problem using fewer particles than the basic PF (more details on the SSPF can be found in [4] or [5]).

Additionally, unlike the localization of an ego-vehicle, a fundamental problem with tracking is that the control input of the target vehicle is always uncertain with a significant variance, which requires a high number of particles for the motion estimate in the basic PF. However, the SSPF has the advantage of handling significant uncertainty using a small number of particles.

In this paper, we present an improved vehicle tracking method by combining the RBPF and SSPF. The use of the RBPF-SSPF combination for vehicle tracking was first proposed by Petrovskaya and Thrun [6], and then, a similar work was implemented by Wojke and Haselich [7]. They both use the RBPF for geometry parameter estimation. However, the SSPF is used only to fit the vehicle's geometry in the first step, target vehicle initialization, and then, the basic PF is used for the motion estimate and pose update through the rest of the tracking process. In our research, we deploy the SSPF for both the geometry fitting and motion estimate through the entire tracking process. The estimation

*This research is supported by the Center of Innovation Program (Nagoya-COI: Mobility Society leading to an Active and Joyful Life for Elderly) from Japan Science and Technology Agency.

[1]Graduate School of information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan alexanderhmw@ertl.jp, e_takeuchi@is.nagoya-u.ac.jp

[2]Institute of Innovation for Future Society (MIRAI), Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan ninomiya@coi.nagoya-u.ac.jp

[3]JST/COI, Nagoya 464-8603, Japan

[4]Graduate School of Information Science and Engineering, The University of Tokyo, 7 Chome-3-1 Hongo, Bunkyo-ku, Tokyo-to 113-8654, Japan shinpei@pf.is.s.u-tokyo.ac.jp

of geometry parameters is still handled by the RBPF, which is similar to the method of Petrovskaya and Thrun [6].

We made three key improvements in using the SSPF through the entire tracking process: (1) Our tracking method takes the latest sensor measurement into account for the motion estimate and pose update; thus, as a fast global optimization method, the SSPF is suitable to obtain accurate motion estimate results using few particles. (2) Because the SSPF can solve a relatively high-dimensional problem using few particles, it is efficient to use a complex non-linear motion model in our tracking method to obtain an accurate motion estimate and a pose update. (3) The geometry fitting carried out by the SSPF is represented by a set of particles around solution regions, which can represent the multi-mode belief. Therefore, it is easy to linearize the measurement likelihood on geometry parameters for maintaining the vehicle's geometry belief in the Gaussian form for RBPF implementation.

The rest of the paper is organized as follows. Section III presents a general overview of our tracking method, including the representation of the vehicle tracking problem and the fundamental mathematical derivation in theory. Based on the derived theory, a general framework for the model-based vehicle tracking method using the Rao-Blackwellized and scaling series PF (RBSSPF) is proposed. Section IV presents our implementation of the general tracking framework. Section V discusses the evaluation experiment results in terms of particle number, computation time, and accuracy. Section VI summarizes our work and discusses future work.

II. Related Works

The detection and tracking of moving objects (DATMO) is a core task in mobile robotics as well as in the field of intelligent vehicles. The most commonly used sensors in DATMO are cameras and LiDAR.

Vision-based vehicle detection and tracking have been topics of great interest to researchers over the past decade [8] and a variety of PF-based tracking methods have been developed [9], [10], [11], [12], [13], [14], [15]. In particular, Manz et al. [15] use a very accurate three-dimensional (3D) model of the target vehicle along with a non-linear Ackerman-steering model for the basic PF. Because they use the 3D model to obtain prior information, the motion parameters include only the steering angle and absolute longitudinal velocity; accordingly, 800 particles are used in their implementation for accurate tracking. However, the precise 3D model of the target vehicle is usually unknown. Therefore, the center of the rear axle should also be regarded as an additional parameter to be estimated, which leads to high computational complexity for the basic PF.

Some LiDAR-based DATMO approaches have been developed over the past decade [16], [17], [18], [19], [6], [7], [20]. One of these approaches—model-based DATMO [6], [7]—does not require separate data segmentation and association steps because the geometric object model helps to associate data points with targets directly. However, its main challenge is to ensure that the filter is sufficiently efficient to meet the high demands of an autonomous vehicle [1]. As stated in Section I, Rao-Blackwellization and the scaling series algorithm are two methods used to improve the inference efficiency of the basic PF, and the RBPF-SSPF combination was first introduced by Petrovskaya and Thrun [6]. In this study, model-based vehicle tracking is carried out in two-dimensions using a linear motion model, and the geometry of the vehicle is represented by a rectangle. Therefore, the state of the tracked vehicle contains eight parameters. By using the RBPF, The pose and velocity are estimated by the basic PF, and the geometry is approximated by the Gaussian distribution. Meanwhile, the SSPF is used to initialize geometry parameters involving geometry fitting under conditions of significant uncertainty. However, during the following tracking process, the basic PF is still employed for the motion estimate and a greedy local search method is used for the geometry fitting.

Our method enhances the method of Petrovskaya and Thrun [6] through efficiency improvement by fully exploiting the advantages of the SSPF. Therefore, we can use a high-dimensional non-linear motion model for the motion estimate, and from the particle set of the SSPF geometry fitting result, we can obtain the linearization of measurement likelihood in the RBPF for the geometry estimate. Another source of efficiency improvement is similar to the efficiency improvement achieved when switching from FastSLAM 1.0 to FastSLAM 2.0 [21]; in our tracking method, the motion estimate and pose update of the target vehicle take the latest sensor measurement into account. The SSPF itself is an optimization method, and thus, it can play the role of scan-matching in FastSLAM 2.0.

A rectangle or cube is a straightforward and common measurement model used for model-based vehicle tracking. In a previous study [22], sparse LiDAR data are combined with dense camera data to represent the vehicle model so that a more accurate measurement model is obtained, which can be easily incorporated into traditional Bayesian filtering techniques such as the Kalman filter or PF to obtain a more precise motion estimate.

III. Tracking Method Overview

In this section, we first discuss the different dynamic Bayesian network (DBN) model (compared to the model in [6]) used in our tracking method. The purpose of changing the DBN is that we need to include the latest measurement into the motion estimate, which is suitable for using SSPF because as a global optimizer, the SSPF can directly solve high-dimensional, non-linear, and multi-mode motion estimate. Then, based on the DBN, a mathematical derivation is carried out to obtain a new update equation for a Bayesian belief. Finally, based on the derived theory, a general framework for the model-based vehicle tracking method with the RBSSPF is presented.

A. Representation

For a target vehicle and a given sensor measurement Z_t at time t, we estimate the vehicle's pose X_t , motion V_t, and

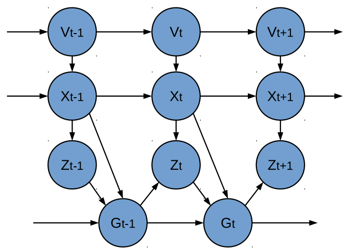

Fig. 1. Dynamic Bayesian network model of the target vehicle's pose X_t, motion V_t, geometry G_t, and LiDAR's measurements Z_t.

geometry G_t. We define the details of each variable used in our tracking method in Section IV. Here, we only present a general probabilistic model for the model-based vehicle tracking method, and Fig. 1 presents its corresponding DBN.

1) Motion Estimate: The DBN contains the motion model and measurement model as shown in (1). Our measurement model is slightly different from commonly used models. For example, in [6], the measurement model is $p(Z_t|X_t, G)$ and only one geometry node G exists in the DBN. This difference is related to the introduction of the latest measurement into the motion estimate. In our method, the motion estimate (V_t) and pose update (X_t) of the target vehicle at time t consider the latest measurement Z_t, which is directly related to the true geometry G^{**}. However, both the true geometry G^{**} and the latest geometry estimate G_t are still unknown. Nevertheless, the true geometry G^{**} is constant, and the geometry estimate G_t represented in the Gaussian form will converge to the true geometry G^{**} with some uncertainty. Therefore, we can use the last geometry estimate G_{t-1} for the motion estimate[1].

$$\begin{array}{rcl} \text{Motion model} & : & \begin{cases} p(V_t|V_{t-1}) \\ p(X_t|X_{t-1}, V_t) \end{cases} \\ \text{Measurement model} & : & p(Z_t|X_t, G_{t-1}) \end{array} \tag{1}$$

2) Geometry Estimate: In [6], the geometry G is represented as a separate variable independent of time t. However, in our method, G_t is used to represent the geometry estimate at every time t and to explicitly illustrate its update equation (2) in the DBN. Therefore, we estimate the latest geometry G_t based on the latest pose estimate X_t, last geometry estimate G_{t-1}, and latest sensor measurement Z_t.

$$\text{Geometry update} \quad : \quad p(G_t|X_t, G_{t-1}, Z_t) \tag{2}$$

B. Mathematical Derivation

For convenience, set $A^t = (A_1, A_2, ...A_t)$ is defined as a time-series states of A. Then, at time t, we produce an estimate of the Bayesian belief according to the history of the estimate on the pose, velocity, and geometry of the target vehicle based on a set of sensor measurements:

$$Bel_t = p(X^t, V^t, G^t|Z^t) \tag{3}$$

[1]The geometry parameter with high uncertainty will have low weight in motion estimate and vice versa.

Similar to [6], we first define the vehicle's motion posterior R_t and the geometry posterior S_t conditioned on its motion:

$$\begin{array}{rcl} R_t & = & p(X_t, V_t|X^{t-1}, V^{t-1}, G^{t-1}, Z^t) \\ S_t & = & p(G_t|X^t, V^t, G^{t-1}, Z^t) \end{array} \tag{4}$$

Then we split belief (3) to obtain its update equation:

$$\begin{array}{rcl} Bel_t & = & p(G_t|X^t, V^t, G^{t-1}, Z^t) \\ & & \cdot p(X_t, V_t|X^{t-1}, V^{t-1}, G^{t-1}, Z^t) \\ & & \cdot p(X^{t-1}, V^{t-1}, G^{t-1}|Z^t) \\ & = & S_t \cdot R_t \cdot Bel_{t-1} \end{array} \tag{5}$$

In our implementation of this general tracking method, R_t is estimated by the SSPF and approximated using a set of particles, and S_t is estimated by the RBPF and approximated using a Gaussian distribution. However, it is possible to use other optimization methods other than the SSPF for estimating R_t.

1) Motion Estimate: As mentioned in the last subsection, in addition to the motion models $p(V_t|V_{t-1})$ and $p(X_t|X_{t-1}, V_t)$, a vehicle's motion estimate is also governed by the measurement model $p(Z_t|X_t, G_{t-1})$ in our method. These three probabilistic laws are related to the motion prediction distribution as follows:

$$\begin{array}{rcl} R_t & \propto & p(Z_t, X_t, V_t|X^{t-1}, V^{t-1}, G^{t-1}, Z^{t-1}) \\ & = & p(Z_t|X^t, V^t, G^{t-1}, Z^{t-1}) \\ & & \cdot p(X_t|X^{t-1}, V^t, G^{t-1}, Z^{t-1}) \\ & & \cdot p(V_t|X^{t-1}, V^{t-1}, G^{t-1}, Z^{t-1}) \\ & = & p(Z_t|X_t, G_{t-1}) \cdot p(X_t|X_{t-1}, V_t) \cdot p(V_t|V_{t-1}) \\ & \propto & p(X_t, V_t|X_{t-1}, V_{t-1}, G_{t-1}, Z_t) \end{array} \tag{6}$$

Therefore, we can use the SSPF to estimate the vehicle's motion posterior R_t for the belief update based on X_{t-1}, V_{t-1}, G_{t-1}, and the latest measurement Z_t. In addition, we obtain the vehicle's latest estimate on the pose X_t and motion V_t, which are approximated by a set of particles. Please refer to [4] or [5] for the details of how to implement the SSPF to estimate the motion posterior R_t.

2) Geometry Estimate: After obtaining the motion estimate, we obtain the latest X_t and V_t. Then, we have update equation for S_t:

$$\begin{array}{rcl} S_t & \propto & p(Z_t, G_{t-1}, X_t, G_t|X^{t-1}, V^t, G^{t-2}, Z^{t-1}) \\ & = & p(Z_t|X^t, V^t, G^t, Z^{t-1}) \\ & & \cdot p(G_{t-1}|X^t, V^t, G_t, G^{t-2}, Z^{t-1}) \\ & & \cdot p(X_t|X^{t-1}, V^t, G_t, G^{t-2}, Z^{t-1}) \\ & & \cdot p(G_t|X^{t-1}, V^t, G^{t-2}, Z^{t-1}) \\ & \propto & p(Z_t|X_t, G_{t-1}) \cdot S_{t-1} \end{array} \tag{7}$$

In this update equation, the first term $p(Z_t|X_t, G_{t-1})$ on the right side is the measurement model given in (1). However, after obtaining the motion estimate, G_{t-1} becomes out of date for the geometry estimate. Therefore, we replace $p(Z_t|X_t, G_{t-1})$ with the measurement likelihood $p(Z_t|X_t, G)$ with respect to the geometry variable G to obtain a more accurate estimate of G_t:

$$S_t \quad \propto \quad p(Z_t|X_t, G) \cdot S_{t-1} \tag{8}$$

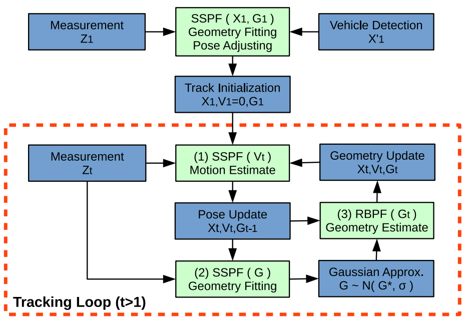

Fig. 2. The framework of our tracking method

We use a Gaussian approximation for the geometry posterior S_t. To maintain S_t in a Gaussian form, we need to linearize the measurement likelihood $p(Z_t|X_t, G)$ on geometry variable G. In [6], the Laplace approximation is used to fit a conditional Gaussian distribution $p(G|X_t, Z_t) \sim N(G^*, \sigma)$ at the global maximum of the measurement likelihood $p(Z_t|X_t, G)$, which is equal to carrying out geometry fitting. However, in their implementation [6], they searched for the "maximum" via local optimization around the last geometry estimate G_{t-1}. In our method, we directly use the SSPF to find the global maximum of the measurement likelihood $p(Z_t|X_t, G)$. After the geometry fitting with the SSPF, the optimized geometry G^* is represented by a set of particles, which can be directly used to approximate the Gaussian variance σ.

C. Method Framework

Summary of Tracking Method

- Bayesian belief of tracking and update equation
  - $Bel_t = p(X^t, V^t, G^t|Z^t)$
  - $Bel_t = S_t \cdot R_t \cdot Bel_{t-1}$
- Motion estimate and pose update (X_t and V_t)
  - SSPF on $R_t \propto p(X_t, V_t|X_{t-1}, V_{t-1}, G_{t-1}, Z_t)$
- Geometry fitting ($p(G|X_t, Z_t) \sim N(G^*, \sigma)$)
  - SSPF on $p(Z_t|X_t, G)$ with respect to geometry G
- Geometry estimate (G_t)
  - RBPF on $S_t \propto N(G^*, \sigma) \cdot S_{t-1}$

We present the theory of our model-based vehicle tracking method and our implementation of the RBSSPF in this method. A brief summary of tracking state update is listed in the box above, and the corresponding general method framework is shown in Fig. 2.

IV. Tracking Method Implementation

To implement the general tracking framework discussed above, we first need to define the details of the tracked vehicle's state and the sensor measurement, and then construct the motion model and the measurement model for the SSPF in the motion estimate and geometry fitting.

This section presents the detailed implementation of the general method framework, including (1) the parameter definition of the tracked vehicle's state (X, V, G), (2) the Ackerman-steering motion model, (3) the measurement (Z) in the form of the virtual scan, and (4) the corresponding measurement model. The general method framework can be implemented with different configurations for different applications.

TABLE I

Notation of Parameters in Our Implementation

	Parameters	Note
Pose (X)	x, y	position of anchor point
	θ	orientation of vehicle
Motion (V)	α	orientation offset of velocity
	v	velocity
	κ	curvature
Geometry (G)	wl, wr	distance to left/right edge
	lf, lb	distance to front/back edge

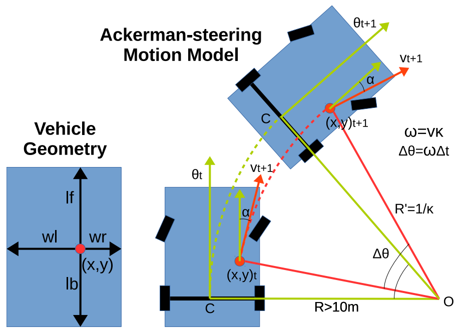

Fig. 3. The illustration of vehicle's geometry and Ackerman-steering motion model.

A. Parameter Definition and Ackerman-steering Motion Model

The parameter definition of the tracked vehicle's state is presented in Table I, and the corresponding illustration is shown in Fig. 3. In our implementation, an anchor point (x, y) on a target vehicle and the vehicle's orientation (θ) are tracked. A basic rectangle is used as the vehicle's geometry model. For the anchor point, the rectangle is represented in terms of the distances from the anchor point to the four edges (wl, wr, lf, lb), as shown in Fig. 3 (left).

The Ackerman-steering motion model is used in our motion estimate. The motion parameters (α, v, κ) are defined in Table I and illustrated in Fig. 3 (right). In this model, the motion of the anchor point is estimated as a circular trajectory (red dotted line and radius). Because, the anchor point (x, y) is randomly selected from the vehicle detection result (Fig. 2 upper blocks) and the geometry is temporarily unknown, an orientation offset (α) is present, which adjusts the velocity to be tangential to the circular trajectory. In practice, the orientation offset (α) is related to not only the steering angle of the front wheels but also the possible slip motion and four-wheel steering model (back wheels may also be slightly steered).

B. Virtual Scan and Measurement Model

Similar to [6] and [7], we also compress a 3D point cloud as a 2D virtual scan (an array of range values with different

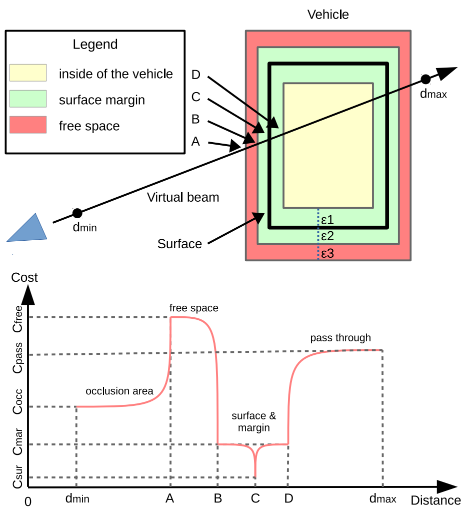

Fig. 4. The illustration of vehicle's measurement model and cost function. d_{min}–A corresponds to the occlusion area, A–B corresponds to the free space, B–D corresponds to the vehicle's surface, and D–d_{max} corresponds to the passing through the vehicle.

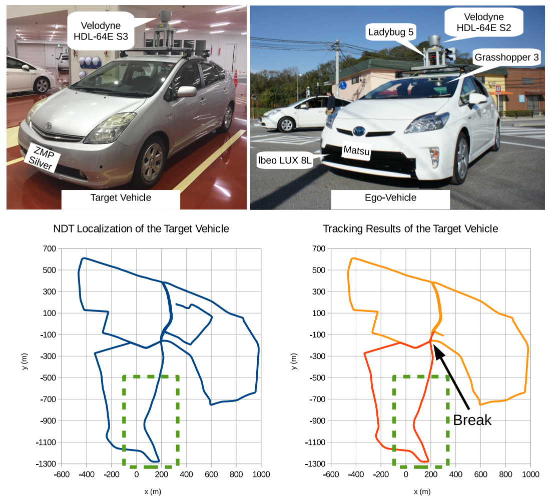

Fig. 5. Top-Left: the target vehicle "ZMP Silver" equipped with the Velodyne 64-S3. Top-Right the ego-vehicle "Matsu" equipped with the Velodyne 64-S2 and Grasshopper-3 HD Camera. Bottom-Left: the localization result of the target vehicle using NDT scan-matching method. Bottom-Right: the tracking result of the target vehicle using the RBSSPF. A break caused by the traffic light divides the tracking result into two parts, the red path and the orange path. The green rectangle is the "challenge" area for evaluation.

bearings on a 2D horizontal plane) and then perform 2D vehicle tracking. However, in our research, we develop a fast and robust virtual scan generation method, which can generate 2000 beams within 15ms and handle roads with large slopes, to guarantee measurement stability. Because this paper focuses on vehicle tracking, the detail of the new virtual scan generation method is described in [23].

Our measurement model is derived from that in [6], and [6]'s measurement model is rooted from the independent beam model [24], [25], [26] and likelihood field model [27]. [6] models each beam's likelihood as a zero-mean Gaussian of variance computed with respect to a constant cost selected based on the relationship between the beam and the vehicle like the occlusion area, free space around the vehicle, surface of the vehicle, and passing through the vehicle.

However, our measurement model should always be used with the SSPF (motion estimate and geometry fitting) to derive particles' weight, and the SSPF gives the best results for continuous and differentiable beliefs with varying gradients [5]; therefore, as shown in Fig. 4, the discontinuous constant cost functions in [6] are approximated by exponential functions (red curved lines) in our implementation.

Additionally, in [20], the same cost function as that in [6] (constant cost around a surface) is used. To obtain an accurate geometry fitting result, the least squares method was further used in [20] for optimization. However, in our cost function, a peak (C) exists at the vehicle's surface, which directly enables the SSPF to find the optimized results of the geometry fitting as well as motion estimate.

V. Experiment

A. Overview

1) Reference Tracking Method: This paper focuses on the efficiency improvement for obtaining the motion estimate by introducing the latest measurement and implementing the SSPF; therefore, for comparison, we implemented another model-based vehicle tracking method using the basic PF for the motion estimate (the RBPF still handles the geometry estimate). This basic PF implementation is based on [6] but with four main modifications: (1) the Ackerman-steering model is used instead of the point model, (2) the geometry configuration (wl, wr, lf, lb) listed in Table I is used instead of (W, L, C_x, C_y), (3) the measurement model as shown in Fig. 4 is used instead of the constant cost functions, and (4) the SSPF is used for the geometry fitting instead of greedy local search. Thus, we can equally compare their performance in obtaining the motion estimate with these modifications. In the following discussion, the basic PF tracking implementation is simply referred to as the RBPF and our tracking implementation as the RBSSPF.

2) Ground-Truth: To obtain the ground-truth of the tracked vehicle and hence demonstrate the precision of our tracking method, we perform a tracking evaluation experiment using two similar intelligent vehicles around Nagoya University in Japan, as shown in Fig. 5 (top). Both the vehicles are equipped with Velodyne HDL-64E sensors for accurate localization [28], and we can determine the true velocity from their CAN data.

The total distance traveled by the vehicle in this experiment is approximately 10 km as shown in Fig. 5 (bottom-left), and the driving environment around Nagoya University contains various types of roads, for example, flat roads, narrow streets and steep ramps. The maximum distance between the two intelligent vehicles is approximately 60 m, whereas the valid maximum range of the virtual scan on a flat road is 80 m. The maximum speed of the target intelligent vehicle is approximately $60km/h$, and its minimum speed is $0km/s$. The dataset is published on the Autoware as a rosbag file[2].

As shown in Fig. 5 (bottom-right), our tracking method can continuously track the target intelligent vehicle for approximately 30 min with only one accidental break caused by a traffic light (only the target vehicle crosses the stop line before the red light). Sometimes, the target vehicle is shortly occluded (less than 2 s) by a road corner or ramp peak, which can be recovered by the particles. The entire experiment record and the visualized tracking result can be viewed on the website[3].

After observing the entire tracking result, we found an area, the green rectangle in Fig. 5 (bottom), to be relatively "challenging" for tracking. First, it has a very sharp corner at the bottom of the selected area. Then, after the sharp turning, the road is broad, curved, and sloped. On this path, the target vehicle can drive at 50 km/h with several instances of lane changing. Therefore, in this area, the tracking result is unstable when a small number of particles are used, and our efficiency evaluation is performed in this local area.

3) SSPF's Particle Number and Computation Time: We need to find out the number of particles used in the RBPF and the RBSSPF to compare their efficiency, because generally speaking, the computation time and precision of the PF-based application is related to the particle number. Therefore, we defined two terms: (1) the required particle number (R-PNUM) is the maximum re-sampled particle number and is used to control the PF (in the following evaluation, we set up nine groups of experiments according to the R-PNUM ranging from $2^5(32)$ to $2^{13}(8192)$); and (2) the total particle number (T-PNUM) counts all the particles used in the motion estimate and determines the accuracy and computation time. Fig. 6 is used to explain the particle number and computation time in the RBSSPF.

For the RBPF's basic forward sampling method, its R-PNUM is equal to its T-PNUM. However, for the SSPF's annealing-based iteration, we can control its R-PNUM, but its T-PNUM is dynamic because as shown in Fig. 6 (top), the re-sampled particle number in each iteration is dynamic. The relationship between the R-PNUM and the average T-PNUM is shown in Fig. 6 (middle). Specifically, although the T-PNUM increases along with the increase of the R-PNUM, the PNUM ratio defined in Fig. 6 (middle) indicates that the increase rate of the T-PNUM is slower than the R-PNUM, which is helpful to the RBSSPF with large R-PNUM.

Fig. 6 bottom presents the computation time of the CPU (Xeon E5-1660 v2, 3.70GHz*12) series implementation of the RBSSPF (CPU-RBSSPF) and the GPU (GeForce GTX TITAN) parallel implementation of the RBSSPF (GPU-RBSSPF). The computation time increases along with the increase of the R-PNUM as well as the T-PNUM. Although the CPU-RBSSPF with one thread is time-consuming, but the GPU acceleration enables the RBSSPF tracking method to work in real-time (only costs 42ms for the R-PNUM=8192).

[2]https://github.com/CPFL/Autoware/

[3]https://drive.google.com/file/d/0B2ipFDl4dLZHSWlaZEFFQUtTaW8/view?usp=sharing

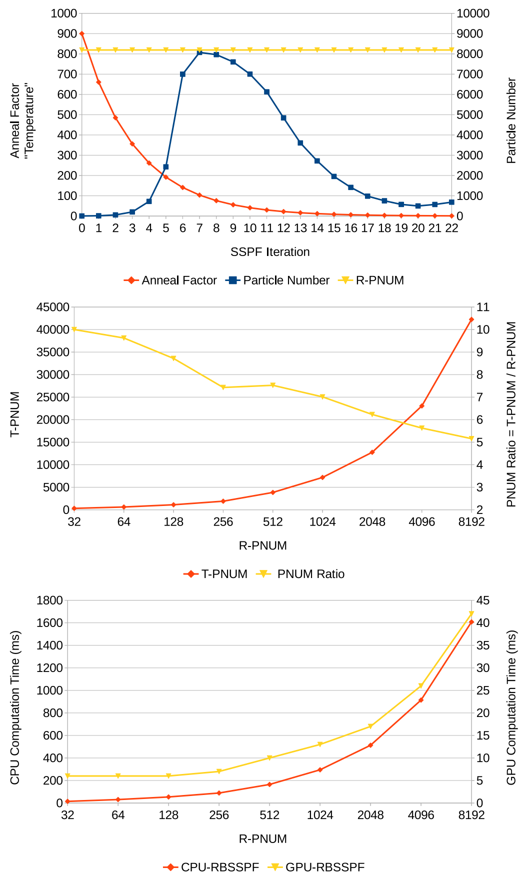

Fig. 6. Top: in this SSPF example, the R-PNUM is set to 8192, and the re-sampled particle number during each SSPF's iteration is dynamic. Middle: The relationship between the R-PNUM and the T-PNUM of the RBSSPF. Bottom: The computation time of the CPU and GPU implementation of the RBSSPF. We can find that as the R-PNUM increases, the computation time of both implementations increases, but the GPU implementation only costs about 40ms with the R-PNUM=8192, which is sufficient for both accurate and real-time applications.

B. Efficiency Evaluation

The following evaluation is carried out from the "challenge" area in Fig. 5.

1) Ground-truth and Tracking Error: Fig. 7 shows the ground-truth (orange line) of the target vehicle including the orientation (θ), velocity (v) and width ($W = wl + wr$). The position and orientation are derived from 3D localization using scan-matching between Velodyne data and the 3D point cloud map [28] (its standard variance ranges from 3cm to 10cm). The velocity is extracted from CAN data and is related to the speeds of the vehicle's four wheels (provided by the vehicle manufacturer). Additionally, from the official website of Toyota Prius, we obtain the target vehicle's width $W = 1.48\ m$.

In Fig. 7, we select and present a pair of tracking results from the RBPF (blue line with the R-PNUM=2048) and the RBSSPF (red line with the R-PNUM=64), because from the visual checking, the tracking result of either the RBPF with the R-PNUM=2048 or the RBSSPF with the R-PNUM=64 is significantly more stable than that with smaller R-PNUM.

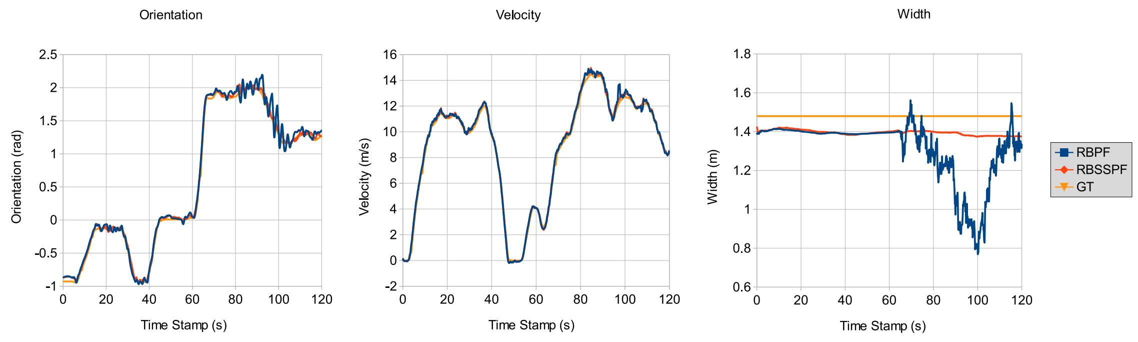

Fig. 7. Orange line: Ground-truth of the target vehicle's orientation (θ), velocity (v), and width ($W = wl + wr$). Blue (RBPF, R-PNUM=2048) and red (RBSSPF, R-PNUM=64) lines: one example of the tracking results and their difference to the ground-truth. (Compared with the position scale, the error of position is too small to illustrate)

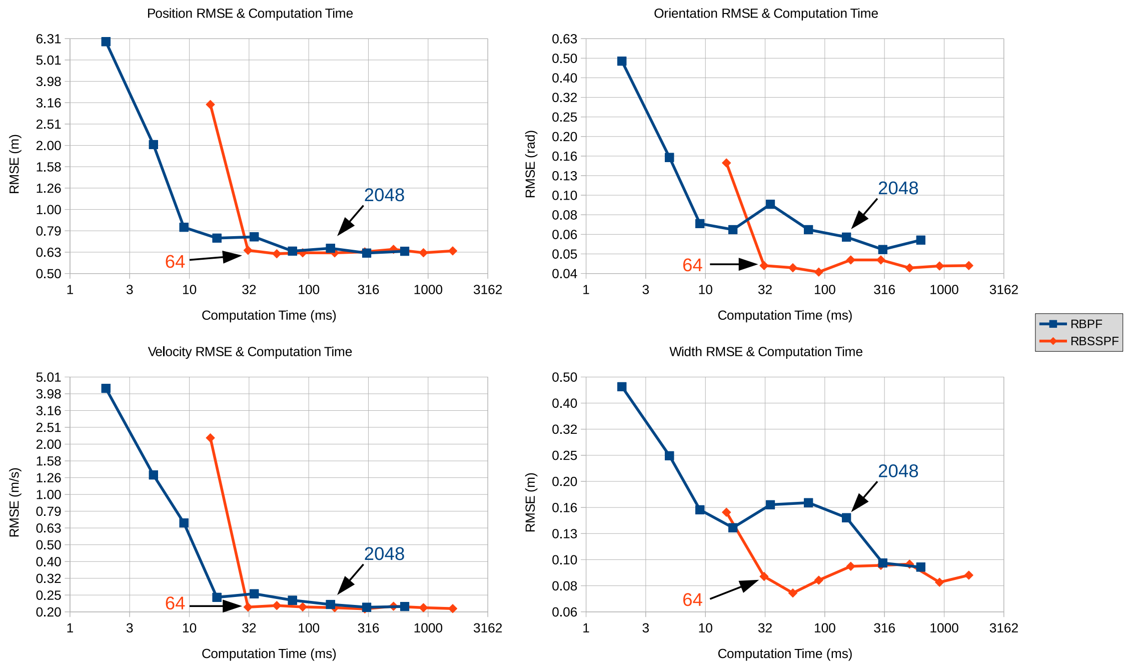

Fig. 8. The relationship between the computation time and the RMSE of position, orientation, velocity and width tracking results. The RMSE results of either the RBPF (blue line) or the RBSSPF (red line) contain nine groups of experiments with different R-PNUM ranging from $2^5(32)$ (most left) to $2^{13}(8192)$ (most right).

We can find that after 60 ms, where the target vehicle takes a sharp turn and then accelerates, the RBPF's orientation tracking result contains large turbulence, and its geometry tracking result moves away from the ground-truth[4]; whereas, the RBSSPF's tracking results are more stable. Therefore, we could measure the difference between the tracking results and the ground-truth, and thus, we can use the root mean squared error (RMSE) to measure the error of the tracking results for precision evaluation.

Combining the computation time of the CPU series implementations and the RMSE of the tracking results, we obtain the relationship between the computation time and the RMSE as shown in Fig. 8. We find that: (1) the minimum RMSE of the RBSSPF is not greater than that of the RBPF, and specifically, the RBSSPF is more precise in orientation and width estimation than the RBPF, (2) the RBSSPF spends less time (R-PNUM=64) to achieve the level of minimum RMSE than the RBPF (R-PNUM$\geq$1024), and (3) the RBSSPF with R-PNUM=32 performs worse in both precision and speed than the RBPF with 64$\leq$R-PNUM$\leq$256.[5]

From this experiment result, we demonstrate that if we constrain the computation time of tracking, the RBSSPF can obtain more precise results than RBPF, and if we require to reach a certain level of accuracy, the RBSSPF spends less

[4]The geometry estimate result is affected by the motion estimate result, because the geometry fitting needs to compensate the error of motion estimate and thereby causing unstable geometry estimate.

[5]From Fig.6, we know that if R-PNUM=32, the SSPF will reach the maximum re-sample particle number with very high anneal factor, which leads to unreliable estimate result.

time than the RBPF. Therefore, the RBSSPF is an efficient tracking method.

VI. Conclusion

In this paper, we present a precise and efficient tracking method by introducing the latest sensor measurement for the motion estimate; we also present the theoretical base of the method. To track the high-dimensional state of the target vehicle including its position, orientation, motion, and geometry, we used the SSPF for the motion estimate and geometry fitting and the RBPF for the geometry estimate. Based on this procedure, we proposed a general framework for our tracking method. In our implementation, we defined a ten-dimensional state for the target vehicle. Then, we used the 3D Ackerman-steering model as the motion model, a 2D virtual scan from Velodyne data as the sensor measurement, and a modified measurement model from a previous study [6]. To demonstrate the precision and efficiency of our tracking method, we performed an evaluation experiment using two intelligent vehicles and compared our method with the basic PF-based tracking method modified from a previous study [6]. Based on the experiment results, we demonstrated the efficiency improvement achieved by our tracking method in terms of accuracy and speed.

Our tracking method takes the latest sensor measurement into consideration for obtaining the motion estimate. Therefore, the measurement model is critical. However, in the present study, we used a basic rectangle as the vehicle model, which may produce inaccurate measurement results. In the future, we will use more accurate models for tracking, for example, [22]. Other improvements that can be implemented in the future are the extension of the general framework of the tracking method to 3D tracking and the inclusion of camera-based measurements.

References

[1] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, C. Laugier, U. Nunes, and P. Bessiere, "Awareness of road scene participants for autonomous driving," in *Handbook of Intelligent Vehicles*. Springer, 2012, pp. 1383–1432.

[2] D. J. MacKay, "Introduction to Monte Carlo methods," in *Learning in Graphical Models*. Springer, 1998, pp. 175–204.

[3] A. Doucet, N. De Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2000, pp. 176–183.

[4] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Bayesian estimation for autonomous object manipulation based on tactile sensors," in *IEEE International Conference on Robotics and Automation (ICRA), 2006*. IEEE, 2006, pp. 707–714.

[5] A. V. Petrovskaya, *Towards dependable robotic perception*. Stanford University, 2011.

[6] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots*, vol. 26, no. 2-3, pp. 123–139, 2009.

[7] N. Wojke and M. Haselich, "Moving vehicle detection and tracking in unstructured environments," in *IEEE International Conference on Robotics and Automation (ICRA), 2012*. IEEE, 2012, pp. 3082–3087.

[8] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 14, no. 4, pp. 1773–1795, 2013.

[9] ——, "A general active-learning framework for on-road vehicle recognition and tracking," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 11, no. 2, pp. 267–276, 2010.

[10] Q. Yuan, A. Thangali, V. Ablavsky, and S. Sclaroff, "Learning a family of detectors via multiplicative kernels," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 3, pp. 514–530, 2011.

[11] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 11, pp. 2259–2272, 2011.

[12] H. T. Niknejad, A. Takeuchi, S. Mita, and D. McAllester, "On-road multivehicle tracking using deformable object model and particle filter with improved likelihood estimation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 2, pp. 748–758, 2012.

[13] R. Danescu, F. Oniga, and S. Nedevschi, "Modeling and tracking the driving environment with a particle-based occupancy grid," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 4, pp. 1331–1342, 2011.

[14] D. Klein, D. Schulz, S. Frintrop, A. B. Cremers, *et al.*, "Adaptive real-time video-tracking for arbitrary objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010*. IEEE, 2010, pp. 772–777.

[15] M. Manz, T. Luettel, F. Von Hundelshausen, and H.-J. Wuensche, "Monocular model-based 3D vehicle tracking for autonomous vehicles in unstructured environment," in *IEEE International Conference on Robotics and Automation (ICRA), 2011*. IEEE, 2011, pp. 2465–2471.

[16] C. Mertz, L. E. Navarro-Serment, R. MacLachlan, P. Rybski, A. Steinfeld, A. Suppe, C. Urmson, N. Vandapel, M. Hebert, C. Thorpe, *et al.*, "Moving object detection with laser scanners," *Journal of Field Robotics*, vol. 30, no. 1, pp. 17–43, 2013.

[17] H. Cho, Y.-W. Seo, B. Vijaya Kumar, and R. R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," in *IEEE International Conference on Robotics and Automation (ICRA), 2014*. IEEE, 2014, pp. 1836–1843.

[18] H. Zhao, Q. Zhang, M. Chiba, R. Shibasaki, J. Cui, and H. Zha, "Moving object classification using horizontal laser scan data," in *IEEE International Conference on Robotics and Automation (ICRA), 2009*. IEEE, 2009, pp. 2424–2430.

[19] H. Zhao, C. Wang, W. Yao, F. Davoine, J. Cui, and H. Zha, "Omni-directional detection and tracking of on-road vehicles using multiple horizontal laser scanners," in *Intelligent Vehicles Symposium (IV), 2012*. IEEE, 2012, pp. 57–62.

[20] Z. Liu, D. Liu, and T. Chen, "Vehicle detection and tracking with 2D laser range finders," in *6th International Congress on Image and Signal Processing (CISP), 2013*, vol. 2. IEEE, 2013, pp. 1006–1013.

[21] D. Roller, M. Montemerlo, S. Thrun, and B. Wegbreit, "Fastslam 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2003.

[22] D. Held, J. Levinson, and S. Thrun, "Precision tracking with sparse 3D and dense color 2D data," in *IEEE International Conference on Robotics and Automation (ICRA), 2013*. IEEE, 2013, pp. 1138–1145.

[23] H. Mengwen, E. Takeuchi, Y. Ninomiya, and K. Shinpei, "Robust Virtual Scan for Obstacle Detection in Urban Environments," in *Intelligent Vehicles Symposium, 2016*. IEEE, 2016.

[24] H. P. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI magazine*, vol. 9, no. 2, p. 61, 1988.

[25] W. Burgard, D. Fox, D. Hennig, and T. Schmidt, "Estimating the absolute position of a mobile robot using position probability grids," in *Proceedings of the national conference on artificial intelligence*. Citeseer, 1996, pp. 896–901.

[26] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research*, pp. 391–427, 1999.

[27] S. Thrun, "A probabilistic on-line mapping algorithm for teams of mobile robots," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.

[28] E. Takeuchi and T. Tsubouchi, "A 3-D scan matching using improved 3-D normal distributions transform for mobile robotic mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006*. IEEE, 2006, pp. 3068–3073.