Accurate and Robust Model-Based Vehicle Tracking Method Using Rao-Blackwellized and Scaling Series Particle Filters *

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

***Abstract*— Accurate and robust tracking technique is important for autonomous vehicle to interact with surrounding participants on road. In this paper, we proposed a model-based vehicle tracking method using *Rao-Blackwellized particle filter* (RBPF) and *scaling series particle filter* (SSPF).**

In comparison with common particle-based tracking methods like [1], we introduced the latest sensor measurement into motion estimate to improve the efficiency of motion estimate for accurate and robust tracking. A *dynamic Bayesian network* (DBN) and its theory base are derived to support this improvement of the general tracking method. Meanwhile, in order to track high dimensional state of target vehicle includes its position, orientation, motion and geometry in this general tracking method, we used SSPF for motion estimate and geometry fitting, and RBPF for geometry estimate.

We conducted an evaluation experiment with two autonomous vehicles around Nagoya University. The total distance traveled in this experiment is about 10km and the experiment environment contains variant types of road, for example, flat road, narrow street and ramp with large slope. We used the ground-truth extracted from our autonomous vehicles and the performance comparison to RBPF based method modified from [1] to demonstrate the accuracy and robustness of our tracking method.

I. Introduction

Vehicle tracking is an important technique for autonomous vehicle to interact with surrounding participants on road. Accurate tracking of target vehicles will lead to reliable prediction for safe driving. Meanwhile, irregular driving behaviors, for example, sudden brake, fast lane changing and sharp cornering, are strongly related to safety issues, and it requires that the tracking method is robust to such infrequent but critical irregular motion of target vehicle.

Bayesian approach is a common and efficient way to solve object tracking problem and *particle filter* (PF) is a kind of non-parametric *Bayesian filter* [2]. PF has the advantage of being able to represent complex belief, especially which has multiple high probability regions (called multi-mode), in comparison with parametric *Bayesian filters* include *Kalman filter* (KF) and its extensions, *extended Kalman filter* (EKF) and *unscented Kalman filter* (UKF).

However, PF's computational cost is less efficient than parametric *Bayesian filters* and is normally exponential in the dimensionality of the state, which is often referred to the *curse of dimensionality* [3]. The dimensionality of tracked vehicle's state is variant for different applications, but the vehicle's position, orientation, motion and geometry are always concerned while LiDAR (Light Detection And Ranging) is used for vehicle tracking. Such high dimensional state will lead to high computational complexity with basic PF.

Rao-Blackwellized particle filter (RBPF) and *scaling series particle filter* (SSPF) are two techniques to cope with the high dimensional state in basic PF. RBPF utilizes *Rao-Blackwellisation* technique to marginalize out some of the parameters in the state and use Gaussian estimate to track them for instead [4], which can decrease the state's dimensionality in basic PF. SSPF is an anneal-based iterative PF to perform search using a series of successive refinements, gradually scaling the precision from low to high [5], which can focus limited computational resources on the more interesting high probability regions and can thus solve relatively high dimensional problem with fewer particles in comparison to basic PF (we strongly recommend readers to read [5] or [6] for the details of SSPF).

Additionally, unlike the localization of ego-vehicle, a key problem of tracking is that the control input of target vehicle is always uncertain with large variance, which requires a large number of particles for motion estimate in basic PF. But SSPF has the advantage to handle large uncertainty with small number of particles.

In this paper we present an improved vehicle tracking method combining the use of RBPF ans SSPF. The fist example combining RBPF and SSPF for vehicle tracking can be found in [1] and then a similar work is implemented in [7]. They both used RBPF for the estimate of geometry parameters. However, SSPF is only used for vehicle's geometry fitting at the first step of tracking, the initialization of target vehicle, and then basic PF is used for target vehicle's motion estimate and pose update through the rest of tracking process. In our research, we deployed SSPF for not only geometry fitting but also motion estimate and pose update through entire tracking process. The estimate of geometry parameters is still handled by RBPF, which is similar to [1].

There are three key improvements to use SSPF through entire tracking process: 1) Our tracking method takes the latest sensor measurement into account for motion estimate and pose update, thus as a fast global optimization method, SSPF is suitable to get accurate and robust motion estimate result with few particles. 2) Because SSPF can solve relatively high dimensional problem with few particles, it is efficient

*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`, `shinpei@is.nagoya-u.ac.jp`

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

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

Preprint submitted to 2016 IEEE International Conference on Robotics and Automation. Received September 15, 2015.

to use complex non-linear motion model in our tracking method for accurate motion estimate and pose update. 3) The geometry fitting conducted by SSPF is represented by a set of particles around solution regions, which can represent multi-mode belief. Therefore, it is easy to linearize measurement likelihood with respect to geometry parameters for purpose of maintaining the vehicle's geometry belief in Gaussian form for RBPF implementation.

We developed and demonstrated our tracking method using our two autonomous vehicles as shown in Fig. 5. In the following, Section III presents a general overview of our tracking method, first including the representation of vehicle tracking problem and key mathematical derivation in theory. And then, based on the derived theory, a general method framework for model-based vehicle tracking using *Rao-Blackwellized and scaling series particle filter* (RBSSPF) is proposed. Section IV presents our implementation of the general tracking framework. Section V discusses the evaluation experiment results in aspect of accuracy, robustness and speed. Section VI summarizes our work and discusses future work.

II. Related Works

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 popular sensors used in DATMO are camera and LiDAR.

The vision-based vehicle detection and tracking has been a topic 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]. Especially for [15], they used a very accurate 3D model of target vehicle with a non-linear Ackerman-steering model for basic PF. As the 3D model is already given as priori information, their motion parameters to be estimated only include steering angle and absolute longitudinal velocity. Accordingly, 800 particles are used in their implementation for accurate tracking. However, accurate 3D model of target vehicle is normally unknown. Therefore, the center of rear axle should also be regarded as an additional parameter to be estimated, which leads to high computational complexity for basic PF.

Meanwhile, a number of 2D/3D LiDAR based DATMO approaches have been developed over the past decade [16], [17], [18], [19], [1], [7], [20]. Among these approaches, *model-based DATMO* [1], [7] does not require separate data segmentation and association steps, because the geometric object model helps associate data points to targets directly. However, its main challenge is to make the filter efficient enough to meet the high demands of autonomous vehicle [2]. As stated in Section I, *Rao-Blackwellisation* and *scaling series algorithm* are two ways to improve inference efficiency of basic PF and a combination of RBPF and SSPF is first introduced in [1].

In [1], the *model-based vehicle tracking* is conducted in 2D world with a *linear motion model* and the geometry of vehicle is represented by a rectangle. Therefore, the state of tracked vehicle contains 8 parameters: $X = (x, y, \theta), v, \Omega = (W, L, C_x, C_y)$, which would lead to high computational complexity with basic PF. By using RBPF, X and v are estimated by basic PF and Ω is estimated by Gaussian distribution. Meanwhile, in order to improve the performance of vehicle detection involves geometry fitting under conditions of large uncertainty, SSPF is used to initialize geometry parameters. However, during the following tracking process, the basic PF is still employed for motion estimate and a greedy local search method is used for geometry fitting.

A contribution of our method is the efficiency improvement of [1] by fully exploiting the advantages of SSPF. With the help of SSPF, high dimensional non-linear motion model can be used for motion estimate and the linearization of measurement likelihood with respect to geometry parameters is easier than that in [1] for geometry estimate with RBPF.

Another efficiency improvement is similar with that from *FastSLAM 1.0* to *FastSLAM 2.0* [21], the motion estimate and pose update of target vehicle in our tracking method takes the latest sensor measurement into account. SSPF itself is an optimization method, and thus it can play the role of *scan-matching* in *FastSLAM 2.0*.

Rectangle or cube is a simple and common model used in measurement model for model-based vehicle tracking. [22] combines sparse LiDAR data and dense camera data to represent vehicle model for more accurate measurement model, which could be easily incorporated into traditional Bayesian filtering techniques, such as Kalman or particle filters, for more accurate and robust motion estimate.

III. Tracking Method Overview

In this section, we will first discuss about a different *dynamic Bayesian network* (DBN) model (compared with [1]) used in our tracking method. Then based on this DBN, a mathematical derivation is conducted to get new update equation of *Bayesian belief*. Finally, based on the derived theory base, a general model-based vehicle tracking framework using RBSSPF is presented.

A. Representation

For each target vehicle at time t, given sensor measurement Z_t, we estimate its pose X_t , motion V_t and geometry G_t. The detail of each variables used in our tracking method is defined in Section IV. Here, we will present a general probabilistic model, whose DBN is shown as Fig. 1, for model-based vehicle tracking method.

1) Motion Estimate: This DBN contains the motion model and measurement model as shown in (1). Obviously, our measurement model is a little different with common one. For example, in [1], their measurement model is $p(Z_t|X_t, G)$ and only one geometry node G exists in DBN. This difference is related to the introduction of latest measurement into motion estimate. In our method, the motion estimate (V_t) and pose update (X_t) of target vehicle at time t will consider the latest measurement Z_t, which is related to geometry G. However, the latest geometry estimate G_t is still unknown. Fortunately, the geometry G is constant and the geometry estimate represented in Gaussian form will

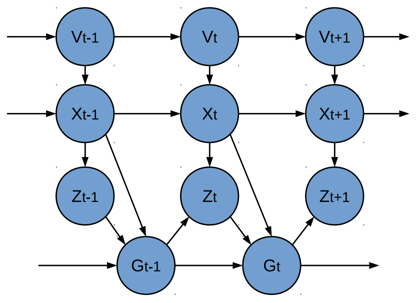

Fig. 1. Dynamic Bayesian network model of the target vehicle's pose X_t, motion V_t, geometry G_t, and LiDAR's measurements Z_t.

converge to it with some uncertainty. Therefore, we can use the last geometry estimate G_{t-1} for motion estimate. Note: the geometry parameter with high uncertainty will have low weight in motion estimate and vice versa.

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

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

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

B. Mathematical Derivation

For convenience, we will write state set $X^t = (X_1, X_2, ...X_t)$. Then, at time t, we produce an estimate of *Bayesian belief* about a history of estimate on pose, velocity and geometry of target vehicle based on a set of sensor measurements:

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

Similar to [1], we first define vehicle's motion posterior R_t and geometry posterior S_t conditioned on its motion:

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

Then we split up the belief (3) to get its update equation:

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

In our method, R_t is estimated by SSPF and approximated using a set of particles; S_t is estimated by RBPF and approximated using a Gaussian distribution. But it is possible to use other optimization methods other than SSPF for the estimate of R_t.

1) Motion Estimate: As we mentioned in last subsection, in addition to motion model $p(V_t|V_{t-1})$ and $p(X_t|X_{t-1}, V_t)$, a vehicle's motion estimate is also governed by 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:

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

Therefore, we can use SSPF to estimate vehicle's motion posterior R_t for belief update based on X_{t-1}, V_{t-1}, G_{t-1} and Z_t. And we will get vehicle's latest estimate on pose X_t and motion V_t, which is approximated by a set of particles.

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

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

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

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

We use a Gaussian approximation for the geometry posterior S_t. In order to maintain S_t in a Gaussian form, we need to linearize the measurement likelihood $p(Z_t|X_t, G)$ with respect to geometry G. In [1], they used the *Laplace approximation* 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 equals to conduct geometry fitting. However, in their implementation, they searched for the "maximum" via local optimization around last geometry estimate G_{t-1}. In our method, we directly use SSPF to search for the global maximum of the measurement likelihood $p(Z_t|X_t, G)$. After geometry fitting with SSPF, the optimized geometry G^* is represented by a set of particles, which could be directly used to approximate the Gaussian variance σ.

C. Method Framework

The theory base of our model-based vehicle tracking method and our implementation of RBSSPF in this method are present as above. A brief summary of tracking state update is listed as below and the corresponding general method framework is shown in Fig. 2.

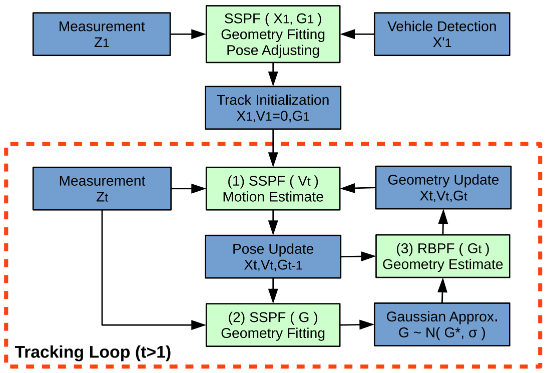
Measurement Z1
SSPF (X1, G1) Geometry Fitting Pose Adjusting
Vehicle Detection X'1
Track Initialization X1,V1=0,G1
Measurement Zt
(1) SSPF (Vt) Motion Estimate
Geometry Update Xt,Vt,Gt
Pose Update Xt,Vt,Gt-1
(3) RBPF (Gt) Geometry Estimate
(2) SSPF (G) Geometry Fitting
Gaussian Approx. G ~ N(G*, σ)
Tracking Loop (t>1)

Fig. 2. The framework of our tracking method

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 $S_t \propto N(G^*, \sigma) \cdot S_{t-1}$

IV. Tracking Method Implementation

In this section, our detail implementation of the general method framework is present, includes 1) the parameter definition of tracked vehicle's state (X, V, G), 2) the Ackerman-steering model, 3) the virtual scan as measurement (Z), 4) the corresponding measurement model. Readers can implement the general method framework with difference configurations for different applications.

A. Parameter Definition and Ackerman-steering Model

The parameter definition of tracked vehicle's state is presented in Tab. 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. Meanwhile, the basic rectangle is used as vehicle's geometry model. For the exist of anchor point, the rectangle is represented as distances to 4 edges (wl, wr, lf, lb) as Fig. 3 left.

Ackerman-steering model is used in our motion estimation. Motion parameters (α, v, κ) are defined in Tab. I and illustrated in Fig. 3 right. In this model, the motion of anchor point is estimated as a circular trajectory (red dot line and radius). Because, the anchor point (x, y) is random selected from vehicle detection result (Fig. 2 upper part) and the geometry is temporarily unknown, there will be an orientation offset (α) , which adjusts the velocity to be tangent to the circular trajectory. In practice, the orientation offset (α) is not only related with the steering angle of front wheels, but also related with the possible slip motion and

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

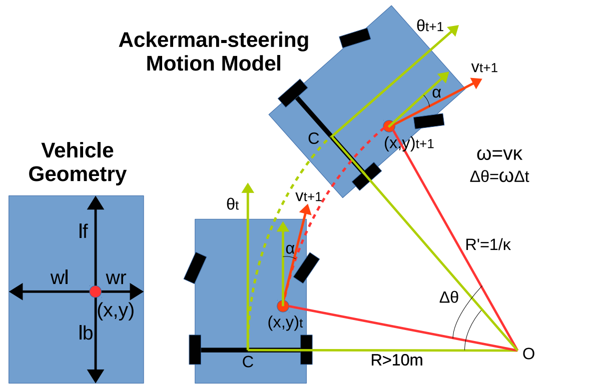

Fig. 3. The illustration of vehicle's geometry and Ackerman-steering model.

four-wheel steering model (back wheels may also be slightly steered).

B. Virtual Scan and Measurement Model

Similar to [1], [7], we also compress 3D point cloud as 2D virtual scan (a set of beams with bearing and range values), and then perform 2D vehicle tracking. However, in our research, we developed a fast and robust virtual scan generation method, which can generate 2000 beams within 15ms and handle the road with large slope, to guarantee the stability of measurement. Because this paper focuses on vehicle tracking, we will present the new virtual scan generation method in another paper.

The measurement model is shown in Fig. 4 and is also similar to [1]. However, for the measurement model will always be used in SSPF (motion estimate and geometry fitting) to derive particles' weight and SSPF works best for beliefs that are continuous and differentiable [6]. The discontinuous cost functions in [1] are approximated by exponential functions (red curved lines) in our implementation.

In [20], they used the same cost function with [1] (constant cost around surface). In order to get accurate geometry fitting result, they further used a least squares method for optimization. However, in our cost function, there is a peak at vehicle's surface, which directly enables SSPF to find the optimized results of geometry fitting as well as motion estimate.

V. Experiment

A. Settings and Overview

This paper mainly focuses on the efficiency improvement of motion estimate by introducing latest measurement and

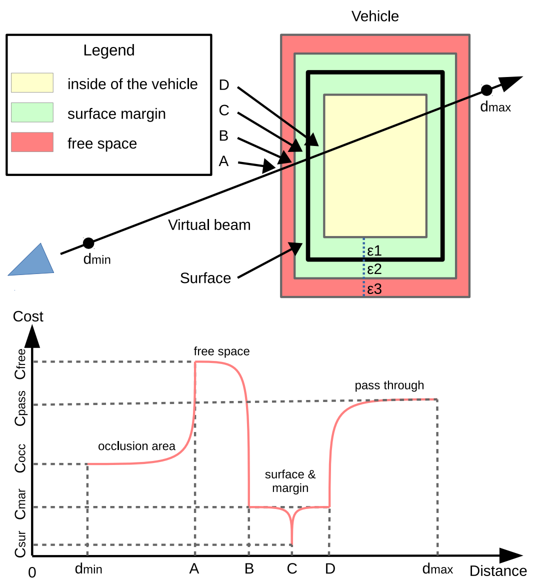

Fig. 4. The illustration of vehicle's measurement model and cost function.

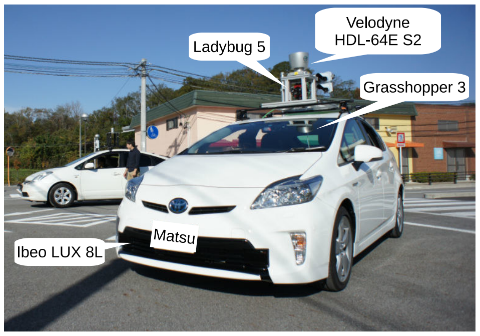

Fig. 5. Autonomous Vehicle "Matsu" equipped with Velodyne 64-S2 and Grasshopper 3 HD Camera.

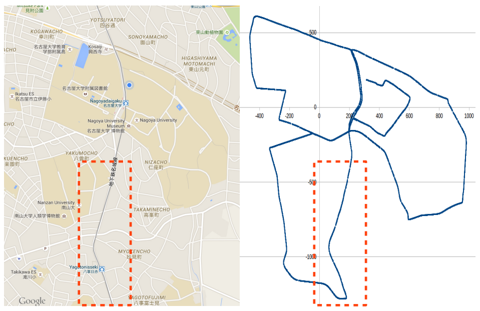

Fig. 6. The path of our experiment around Nagoya University (image is from Google Map). The localization is derived by NDT scan-matching algorithm [23]. The red rectangles indicate the evaluation area.

the implementation of SSPF. Therefore, for comparison, we implemented another model-based vehicle tracking method using basic PF for motion estimate (geometry estimate is still handled by RBPF). This basic PF implementation is based on [1] and there are 3 main modifications: 1) use Ackerman-steering model instead of point model, 2) use geometry configuration (wl, wr, lf, lb) in Tab. I instead of (W, L, C_x, C_y), 3) use SSPF for geometry fitting instead of greedy local search. Thus we can equally compare their performance on motion estimate with these modifications. Below we will simply note the basic PF tracking implementation as RBPF and our tracking implementation as RBSSPF.

In order to use large number of particles for real-time evaluation, we implemented both tracking methods on GPU for acceleration. Therefore, in our evaluation, the number of particles for motion estimate ranges from $2^5(32)$ to $2^{13}(8192)$ (9 groups of evaluation in total) and the corresponding time cost for tracking ranges from $10ms$ to $50ms$.

In order to get ground-truth of tracked vehicle to demonstrate the accuracy and robustness of our tracking method, we conducted an tracking evaluation experiment using two similar autonomous vehicles (Fig. 5) around Nagoya University (Fig. 6). Both vehicles are equipped with Velodyne HDL-64E for accurate localization [23], and we can also get true velocity from their CAN data. Therefore, the ground-truth we have includes target vehicle's (Velodyne) position (x, y), orientation (θ), velocity (v) and geometry $(W = wl + wr, L = lf + lb)$.

The total distance traveled in this experiment is about $10km$ and the environment around Nagoya University contains various types of road, for example, flat road, narrow street and ramp with large slope. Our tracking method can continuously track target autonomous vehicle for about 30 minutes without lost, except a break by traffic light (only target vehicle passed the stop line). The max distance between these two autonomous vehicles is about 60m, while the max valid range of virtual scan on flat road is 80m. Sometimes the target vehicle is shortly occluded (less than 2 seconds) by road corner or ramp peak, which can be recovered by particles. However, in consideration of safety on public road, irregular motion tracking is not conducted in this experiment.

After observing entire tracking result, we found a area is relatively "challenging" for tracking (red rectangle in Fig. 6). First, it has a very sharp corner (Fig. 6 bottom). Then, after the sharp corner, the road is broad, curved and sloped. On this road, the target vehicle can drive at $50km/h$ with several times of lane changing. Therefore, in this area, only RBPF tracking implementation is unstable with small number of particles. Our evaluation is conducted in this area and more details could be found in this paper's video.

B. Evaluation and Discussion

Fig. 7 shows the ground-truth of target vehicle includes position (x, y), orientation (θ) and velocity (v). The position and orientation are derived from 3D localization using scan-matching between Velodyne and 3D point cloud map [23]. The velocity is extracted from CAN data and is related to the vehicle's four-wheel's speed (provided by vehicle manufacturer). Additionally, from the official website of Toyota Prius, we get target vehicle's width $(W = wl+wr)$ equals to 1.48m. With the ground-truth, we can use RMSE (Root Mean Squared Error) to separately evaluate the accuracy of tracking results on position, orientation, velocity and geometry.

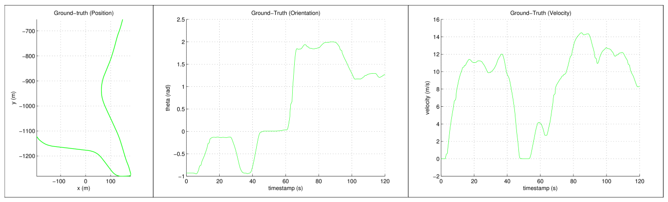

Fig. 7. Ground-truth of target vehicle's position (x, y), orientation (θ) and velocity (v).

For RBPF and RBSSPF tracking implementations, we have conducted 9 groups of evaluation experiments with different particle numbers [1] ranges from $2^5(32)$ to $2^{13}(8192)$. Then we compared the estimate on position, orientation, velocity and geometry among ground-truth, RBPF and RB-SSPF with different particle numbers like Fig. 8. Finally, the corresponding RMSE statistical results are listed in Tab. II and also illustrated in Fig. 9.

Tab. II shows not only the RMSE result of all evaluation experiment results (col. 4-11), but also the evaluated particles in motion estimate (col. 2-3). As we have mentioned above, the particle number (col. 1) presents how many particles to represent motion estimate result. However, RBSSPF is an iterative and adaptive PF, therefore, the total number of particles evaluated for final motion estimate is different with its particle number. While, for RBPF, the number of evaluated particles equals to its particle number. We think it is better to take evaluated particles into consideration for more fair comparison.

Additionally, for the motion estimate evaluation, besides the position, orientation and velocity, we also include the geometry parameter width, which is seemingly not directly related with motion and should converge to real value in tracking process. Because in practice, bad motion estimate will generate unstable geometry estimate. The model-based vehicle tracking methods first sample the motion space (RBPF uses basic forward sampling and RBSSPF uses SSPF to generate good motion samples), then use geometry fitting result and previous geometry estimate to calculate weights for resampling. If the motion sample space cannot well cover real motion change (near or out of boundary of motion sample space), the motion estimate will have some error. Thus, the geometry fitting result will be apart from previous geometry estimate to compensate motion estimate error, which leads to unstable geometry estimation similar to Fig. 8 right part. Therefore, we also regard geometry width as an important evaluation indicator.

1) Accuracy Evaluation: We evaluate our tracking method's accuracy on motion estimate from Tab. II and Fig. 9. RBPF with 32/64 particles failed several times during tracking and this leads to large RMSE result. Meanwhile, RBSSPF with 32 particles also failed several times. Except for the failure cases, in general, RBPF's error decreases with the increase of particle number, which satisfies the property of PF algorithm. However, RBSSPF's error keeps in similar level. Compare the error between RBPF and RBSSPF, we find that RBPF's error is always larger than RBSSPF's error not only with same particle number but also with similar number of evaluated particles (e.g. Fig. 8). Meanwhile, the error level of RBPF with large particle number, which sometimes could be regarded as "ground-truth", finally converges to that of RBSSPF. But in some aspects, e.g. orientation and width, even the error of RBPF with 8192 particles is still larger than that of RBSSPF with only 64 particles.

[1]RBPF: particle number used in forward motion sampling; RBSSPF (iterative and adaptive PF): max particle number of final motion estimate.

2) Speed Evaluation: According to [6], due to SSPF's divide-and-conquer nature, it can run in log time (in the best case) compared to naive approaches, such as importance sampling or uniform grid. Therefore, RBSSPF is faster than RBPF tracking in theory. However, in our naive GPU implementation, the iterative SSPF requires a "for loop" on CPU (15 times of iteration in our implementation), which causes the loss of performance. Therefore, in our implementation, RBSSPF is slower than RBPF tracking. In order to evaluate the true computation load, we used the number of particles evaluated instead of pure time cost to represent the speed.

Comparing the number of evaluated particles in RBPF and RBSSPF, which are listed in Tab. II, RBSSPF totally evaluates 630 (particle number=64) particles to reach the error level achieved by using RBPF with 8192 particles. However, RBSSPF fails after evaluating 326 in total and RBPF works with 256 particles. There are two reasons: 1) 32 particles are not able to well represent final motion estimate of 3 dimensional Ackerman-steering model, 2) SSPF is an iterative anneal based algorithm and the number of particles evaluated in each iteration grows from 1 to max particle number with the decrease of "temperature". For 32 particle number limitation, the "temperature" is still very high when the algorithm reaches to the max particle number, which causes unstable motion estimate. But overall, RBSSPF requires less number of particles evaluated and this demonstrates that the RBSSPF is faster than RBPF tracking method.

3) Robustness Discussion: In Section I, we mentioned that the tracking method should be robust to irregular motion. However, in consideration of safety on public road, we did not conduct irregular motion in our evaluation experiment. Therefore, we cannot directly demonstrate the robustness of our tracking method. However, we used an indirect evidence to demonstrate its robustness.

At the back of irregular motion tracking failure is the mismatching between motion sample space and real motion change. Irregular motion generates large motion change, which requires large motion sample space as well as large number of particles for basic PF. Moreover, if the motion model is high dimensional, the required particle number is exponential to the motion change with large base. Therefore, compared with basic PF method (RBPF), if we can use same regular motion experiment data to demonstrate that our tracking method (RBSSPF) can achieve similar accuracy level with fewer particles and its error is less sensitive to

Preprint submitted to 2016 IEEE International Conference on Robotics and Automation. Received September 15, 2015.

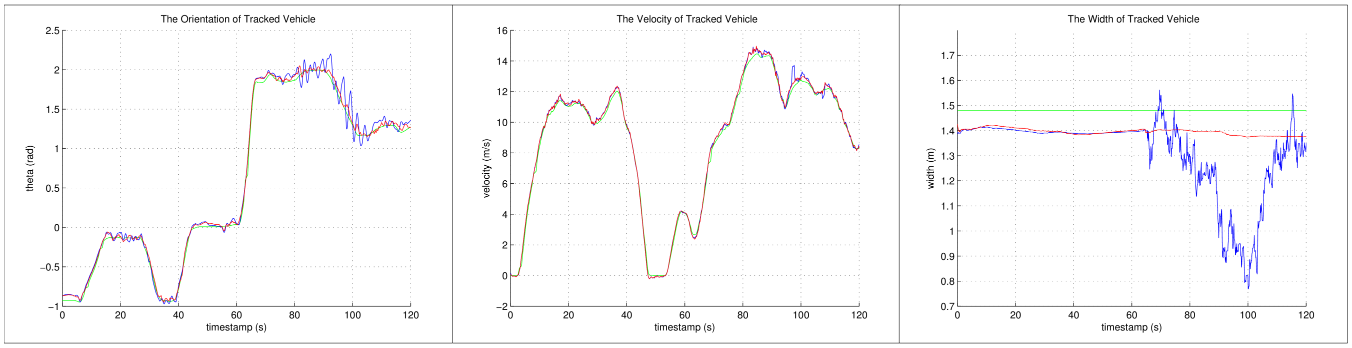

Fig. 8. One example (the RMSE is highlighted in Tab. II) of comparing the estimate on orientation (θ), velocity (v) and width ($W = wl + wr$) among ground-truth (green), RBPF (blue, 2048 particles) and RBSSPF (red,64 particles) with different particle numbers. (Compared with position scale, the error of position is too small to illustrate)

TABLE II

THE EVALUATED PARTICLES AND RMSE OF EVALUATION EXPERIMENT RESULTS

Particle Number[a]	Evaluated Particles		Position (m)		Orientation (rad)		Velocity (m/s)		Width (m)	
	RBPF	RBSSPF[b]	RBPF	RBSSPF	RBPF	RBSSPF	RBPF	RBSSPF	RBPF	RBSSPF
32 (2^5)	32	326	6.1039	3.1018	0.4841	0.1463	4.2869	2.1738	0.4600	0.1519
64 (2^6)	64	630	2.0132	0.6437	0.1561	0.0437	1.3062	0.2129	0.2502	0.0862
128 (2^7)	128	1611	0.8250	0.6200	0.0716	0.0426	0.6762	0.2175	0.1553	0.0745
256 (2^8)	256	2096	0.7342	0.6267	0.0667	0.0405	0.2435	0.2135	0.1326	0.0834
512 (2^9)	512	4082	0.7438	0.6262	0.0899	0.0467	0.2558	0.2111	0.1624	0.0943
1024 (2^{10})	1024	7779	0.6382	0.6328	0.0667	0.0467	0.2340	0.2079	0.1654	0.0951
2048 (2^{11})	2048	14270	0.6579	0.6504	0.0611	0.0425	0.2208	0.2154	0.1448	0.0961
4096 (2^{12})	4096	25929	0.6251	0.6268	0.0528	0.0435	0.2125	0.2112	0.0971	0.0819
8192 (2^{13})	8192	44167	0.6370	0.6396	0.0590	0.0437	0.2147	0.2087	0.0936	0.0872

[a] The particle number of forward motion sampling (RBPF) or final motion estimate (RBSSPF, also max particle number).
[b] The average of sum of particles used in each iteration.

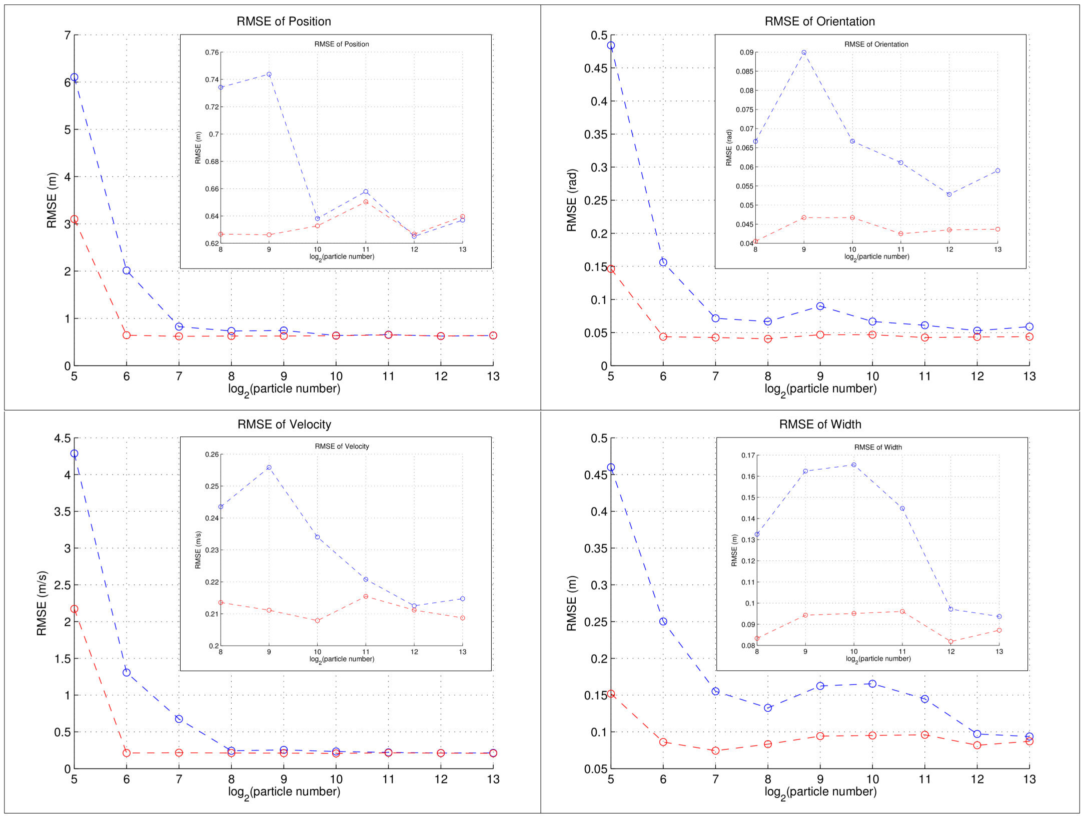

Fig. 9. The comparison of RMSE of position, orientation, velocity and width between RBPF (blue) and RBSSPF (red). Small overlapped graph shows the detail RMSE ranges from $2^8(256)$ to $2^{13}(8192)$.

Preprint submitted to 2016 IEEE International Conference on Robotics and Automation. Received September 15, 2015.

particle number, we can indirectly demonstrate RBSSPF is more robust than RBPF to irregular motion.

Fortunately, from the statement of accuracy evaluation, we indirectly demonstrate the robustness of RBSSPF with the RMSE in Tab. II and Fig. 9. Additionally, Fig. 8 presents another practical evidence, which is a comparison of tracking results between RBPF with 2048 particles and RBSSPF with 64 particles (630 evaluated particles). During the period from 60s to 120s, the target vehicle first takes a very sharp turn (orientation: 0rad$\rightarrow$2rad) and then drives at high speed (velocity: 10m/s-15m/s) on a curved and sloped road with several times of lane changing. For RBPF tracking result, the orientation estimate vibrates around ground-truth with large offset, the velocity estimate has some large error and geometry estimate is very unstable. While, the tracking result of RBSSPF is more accurate and stable.

VI. Conclusion

In this paper, we present an accurate and robust tracking method by introducing latest sensor measurement for motion estimate as well as its theory base. In order to track high dimensional state of target vehicle includes its position, orientation, motion and geometry, we used SSPF for motion estimate and geometry fitting, and RBPF for geometry estimate. Based on this, we proposed a general framework of our tracking method. In our implementation, we defined a 10 dimensional state for target vehicle. Then we used 3 dimensional Ackerman-steering model as motion model, 2D virtual scan from Velodyne data as sensor measurement and modified measurement model from [1]. In order to demonstrate the accuracy and robustness of our tracking method, we conducted an evaluation experiment using two autonomous vehicles and compared our method with the basic PF based tracking method modified from [1]. Based on the experiment results, we demonstrated the efficiency improvement on accuracy and robustness of our tracking method.

Our tracking method takes the latest sensor measurement into consideration for motion estimate, therefore, measurement model is very important. However, currently, we used basic rectangle as vehicle model, which may cause inaccurate measurement result. In the future, we will use more accurate model for tracking, for example [22]. Another improvement in implementation in the future is to extend the general framework of tracking method to 3D tracking and include camera measurement.

References

[1] 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.

[2] 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.

[3] D. J. MacKay, "Introduction to monte carlo methods," in *Learning in graphical models*. Springer, 1998, pp. 175–204.

[4] 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.

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

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

[7] N. Wojke and M. Haselich, "Moving vehicle detection and tracking in unstructured environments," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. 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 *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. 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 *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. 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 *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. 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 *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. 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*. IEEE, 2012, pp. 57–62.

[20] Z. Liu, D. Liu, and T. Chen, "Vehicle detection and tracking with 2D laser range finders," in *Image and Signal Processing (CISP), 2013 6th International Congress on*, 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 *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1138–1145.

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

Preprint submitted to 2016 IEEE International Conference on Robotics and Automation. Received September 15, 2015.