Sensory Augmentation for Increased Awareness of Driving Environment

Chiyu Dong
John M. Dolan

Feb. 15, 2016

Technologies for Safe and Efficient Transportation (T-SET) UTC
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

DISCLAIMER

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Table of Contents

Problem

The goal of this project was to develop a a lateral localization framework for autonomous driving in urban areas. Vehicle location is significant information for the controller, planner and behaviors systems. Lateral location is extremely important for safe and reliable self-driving, due to dense traffic, small lane width and varying road geometry. Though RTK GPS has centimeter-level accuracy output in open areas, it can have half-meter lateral error in urban areas, which is extremely dangerous for urban driving. It is therefore desirable to precisely identify the lateral position by combining with other sensors.

Approach/Methodology

The proposed framework is based on a particle filter and flexibly extends to the use of any number of sensors. In the implementation, low-cost sensors such as wheel speed encoders, steering wheel angle sensor and camera are used for correction. A dead-reckoning system is also designed for situations in which lane markings are absent. By combining the particle filter framework with the dead-reckoning system, our approach can continuously provide reliable localization information. We focus on lateral localization in urban scenarios, including driving on normal roads and making turns in intersections. Since RTK GPS is not always reliable in urban areas, we also use a verification method to compare the result with LIDAR landmarks, serving as ground-truth.

The particle filter-based lateral localization framework requires a lane-level map (RNDF) in advance and takes two kinds of inputs: movement and observation. In our implementation, we take wheel speed and steering wheel angle as movement inputs to predict particles' locations;
and we take low-cost GPS and lane marking location as observation inputs to weight and re-sample particles. In particular, we first initialize hundreds of particles on the road (orthogonal to

the driving direction). Second, we treat vehicle lateral speed as a control parameter (movement model for the particle filter) to update the particle locations. Third, using GPS and lane marking information as observation, we generate the particle weights. Finally, we use the weights to resample the particles, and obtain the Maximum A Posteriori (MAP) location. We also establish criteria for the particles' re-initialization to avoid particle deprivation. The update rate of our algorithm is about 20 Hz, which is high enough for urban driving, because of low speed limits (usually 10 to 35 mph).

A. Map Model Generation. Before running the localization framework on vehicles, a lane-level map (RNDF) needs to be prepared in advance. The RNDF can be established by using high-resolution aerial images or mounting a RTK GPS on vehicle and driving around. Both ways involve map errors: the high-resolution aerial image has at most 30cm error, and RTK has 1-5cm if fixed. An accurate RNDF needs to take advantage of both aerial images and RTK, since sometimes RTK may not be fixed, and may have 30cm or greater error if in float or a more degraded mode. The map error can also be included in the framework.

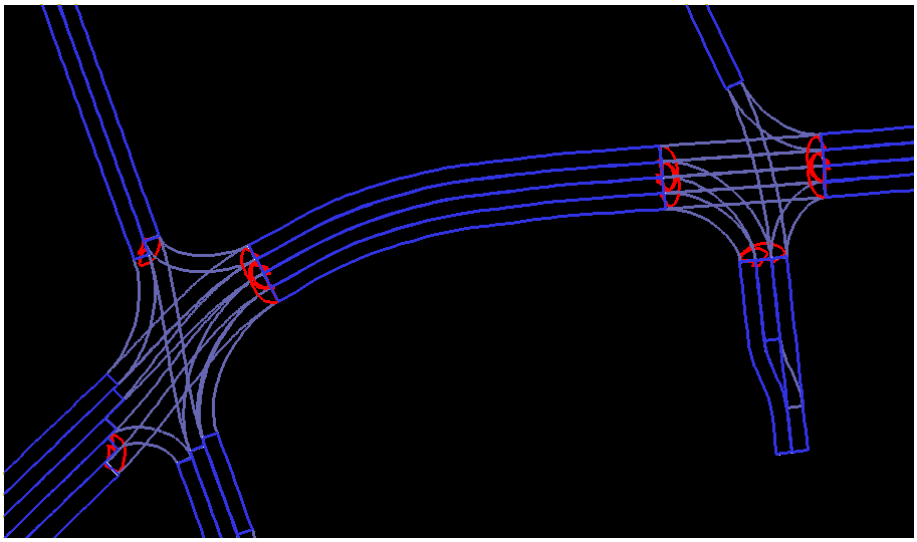


Figure 1. RNDF map model. This 2D map model accurately records number of lanes, lane length, lane width, curvature, etc.

Fig. 1 is an RNDF example from one of our testing routes. In the RNDF, areas between intersections are road segments with multiple lanes which are well defined by waypoints on center lines; intersections are defined by specific zones; and entries and exits of lanes are linked by virtual lanes. From the RNDF, we know the number of lanes, lengths of lanes, lane direction and lane width at specific points. In Fig. 1, blue lines denote lane limits, brighter blue lines describe virtual lanes, and red lines show virtual lanes that have very high curvature.

B. Sensor Pre-processing. Some sensor data cannot be directly used by the particle filter framework, but require pre-processing. For example, the wheel speed and steering wheel angle cannot be directly used in the algorithm, though they can be used to roughly estimate heading information and lateral speed. [1] introduces a ground vehicle lateral motion model, and here we apply an approximation, which is shown in equations (1) and (2). Vehicle speed V can be easily generated from the wheel speed encoder. Heading angle α (yaw) and lateral speed *v* of the vehicle are strongly related to steering wheel angle and vehicle speed *V*. The steering wheel has angle φ, the steering ratio is γ, and the drifting ratio is ω_d. The drifting ratio reflects sideslip of the vehicle while running at high speed. For urban application, ω_d can be considered a constant, and experimental results show that $\omega_d = 1.1$.

$$\alpha = w_d \frac{\phi}{\gamma} \qquad (1)$$

$$v = V sin(\alpha) \qquad (2)$$

Though these estimates are not as accurate as those of high-performance IMU and the steering ratio may not be constant at high speed, experiments show they are accurate enough to serve as motion inputs for our algorithm in urban testing scenarios. Another indirectly used sensor output is lane marking detection. In this framework, only immediate left/right lane markings are

considered. Suppose lane markings on both sides are detected, and the distance from the vehicle is d_l (left) and d_r (right), respectively. The algorithm must obtain the bias from the lane center from the lane marking information, defining the lane center as zero position, where w denotes current lane width. If lane markings on both sides are visible, $d_r \geq 0$ and $d_l \geq 0$. If only one side is invisible, then the distance to this side is 0, and the distance to the other side is greater than 0, (e.g. if only the left side lane marking is visible, $d_l \geq 0$, $d_r = 0$). The bias from the lane center is:

$$b = \begin{cases} (d_l - d_r)/2 & (d_l > 0, d_r > 0) \\ d_l - w/2 & (d_l > 0, d_r = 0) \\ w/2 - d_r & (d_l = 0, d_r > 0) \end{cases} \quad (3)$$

C. General Particle Filter. Since both the map and GPS are inaccurate, other information is required to achieve accurate lateral localization. To combine data from several sensors, we use a particle filter. As mentioned, this requires a movement model, observation model and prior distribution to obtain the posterior distribution for parameters of interest. Moreover, in the particle filter implementation, prior and posterior distributions do not have to be in analytic form. The main benefit of using a particle filter is that there is no need to assume a prior distribution of lateral locations. Secondly, a small number of particles is enough to easily find the Maximum A Posteriori (MAP) location in real time. Thirdly, it can be extended by adding more sensors. Finally, particles can automatically take previous movement and location into account, since the particle distributions come from past iterations, and are affected by movement and observations in these iterations. This paragraph will describe the calculation of the movement model and observation model. Sampling is the most significant step in the particle filter algorithm, wherein all particles should be re-sampled according to their weights. The result from this step is a simulated posterior distribution. First, as shown in Fig. 2, we evenly spread N particles p^i across

each lane. Secondly, by using lateral speed, which is returned with its standard deviation by an onboard speed sensor, we update the particle locations. To simplify our implementation, we assume the lateral speed is a normally distributed random variable *V* whose mean and standard deviation are the speed sensor result. We define movement to the left as negative, and movement to the right as positive, assigning a speed *v* to each particle by randomly drawing a value from the normal distribution. The time ΔT between two instants is easily measured by the system. The translation of a single particle is therefore $v\Delta T$.

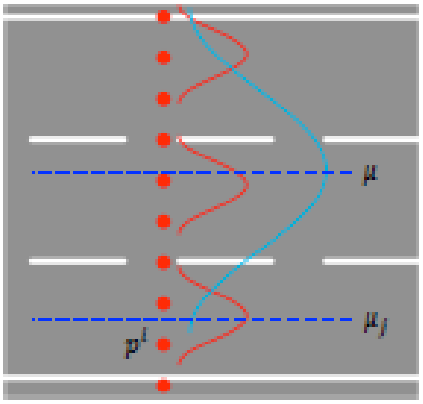


Fig. 2: Initialized particles pi (red points), GPS observation distribution (cyan blue) with mean value μ and lane marking location distribution on *j*th lane (red) with mean value μ_j. In this figure, the lane marking location distribution has been converted to the distribution of the vehicle position after detecting lane markings.

Equation 3 gives the particle update:

$$\hat{p}_t^i = v\Delta T + p_{t-1}^i \tag{4}$$

where $\hat{p}_t^i$ denotes the position of the *i*th particle at moment *t*, after applying the movement model. Third, each particle updates its weight from observation for resampling in the next step. Here we have two observations: one is the GPS, and the other is the lane marking detection. Both the GPS and the lane marking location are considered random variables, and we assume these random variables are statistically independent. The conditional joint probability density, which equals the

weight, is calculated as the product of two separate conditional probability densities. The first conditional probability is from the GPS and the second one is from lane marking detection:

$$W_t^i = P(og_t, ol_t|p_t^i) = P(og_t|p_t^i)P(ol_t|\hat{p}_t^i) \quad (5)$$

where W_t^i denotes the observation weight for particle i at moment t, og_t denotes the GPS observation at t and ol_t denotes the lane marking detection observation. It is known that the GPS signal has a normal distribution, and we use the mean value μ and standard deviation σ directly from the GPS module, and adjust these parameters online. The weight from the GPS can then be calculated by the following formula:

$$P(og_t|p_t^n) = w_{gps}^i = \mathcal{N}(p^i, \mu, \sigma) \quad (6)$$

where p^i is the location of ith particle. We only consider the left and right marking of our lane. If no lane marking was detected, set w_{lane}^i = 1; if at least one side lane marking is detected, (3) is used to calculate μ in (7). Furthermore, if one side lane marking is detected, we set σ = 0.1 and if both side lane markings are detected, we set σ = 0.05. If there are multiple lanes, this distribution is applied to all lanes on the road and the maximum density from these distributions is obtained, since lane marking detection has no information about which lane the vehicle is traveling in. The equation below shows how to calculate weights from the lane marking detection observations:

$$P(ol_t|p_t^n) = w_{lane}^i = \max(\mathcal{N}(p^i, \mu^j, \sigma^j)) \quad (7)$$

where μ^j, σ^j are the bias and its standard deviation from the lane marking in the jth lane, as Fig. 2 shows. For example, if the left lane marking is detected and is 1.2 m from the vehicle center, then the distribution of the vehicle position has a mean of 1.2 m from the left lane marking and a corresponding σ. Since lane marking detection does not report any information about the number

of lanes, we assume that the vehicle position will have this distribution in every lane on the road. In Fig. 3, three red curves denote weights on different particles from lane marking detection. In addition, if no lane marking is detected, $w_{lane}^{i} = 1$ and (5) yields:

$$W_t^i = P(og_t, ol_t|p_t^i) = P(og_t|p_t^i) \cdot 1 = w_{gps}^i \tag{8}$$

In that case, GPS will dominate the localization, and lateral localization relies on historical positions and GPS. The system can run in this dead reckoning mode if lane markings disappear, so our approach can still perform reliable lateral position estimation while changing lanes or making turns. We combine these two distributions (GPS and lane marking detection) for weight updating. Finally, we resample particles depending on their logarithmic weights. The MAP of the updated particles is the vehicle lateral location. In order to simplify and stabilize the result, we use a histogram to count the updated particles, then calculate the MAP (maximum a posteriori) location.

E. Dead reckoning. Since the GPS signal is not stable in urban areas, especially when driving through intersections, the GPS localization can oscillate dramatically. It is crucial to prevent this pose jump to obtain accurate pose estimation when driving through intersections, which involves a turn where no lane marking is observable. Here we apply the bicycle model to analyze the motion in intersections. If the current heading angle is θ, current location is X, longitudinal translation is dx, and lateral translation is dy, then the dead-reckoning result in vehicle coordinates should be $X_r = (dx, dy)^T$, and the result $\hat{X}$ in global coordinates (North/East coordinates) should be:

$$\hat{X} = X + \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot X_r$$

Let w denote the wheelbase, φ the steering wheel angle, R the turn radius, v the vehicle's speed from ABS, dt the time interval, and Φ the turning angle. Actually, the rotation matrix is the transformation from the body (local) coordinates to the global coordinates [2]. $X_r = (dx, dy)^T$ can then be calculated by Equation (9):

$$\begin{pmatrix} R \\ \Phi \\ X_r \end{pmatrix} = \begin{pmatrix} R \\ \Phi \\ dx \\ dy \end{pmatrix} = \begin{pmatrix} w/\tan\phi \\ v \cdot dt/R \\ R\sin\Phi \\ R(1-\cos\Phi) \end{pmatrix} \tag{9}$$

Findings

To verify the performance, we conducted field tests on the CMU-SRX autonomous driving vehicle platform [3]. The vehicle has a Mobileye system to detect lane markings, RTK GPS to collect ground-truth poses, and six 4-layer LIDAR sensors which provide 360-degree coverage to detect and register landmarks. The testing loops are surrounded by buildings or hills. RTK GPS normally has 0.4m to 0.6m lateral error on average on our test segments (according to messages from RTK GPS) in our urban testing area and the pose from RTK GPS jumps occasionally. The vehicle cannot autonomously run in this area relying solely on RTK GPS. For the initial test, we just use RTK GPS and try to combine other low-cost sensors' outputs (steering wheel angle, wheel speed and lane locations) to correct the received RTK GPS result and achieve a smaller lateral error. To compare the result of our approach with ground truth, we combine LIDAR and RTK GPS to generate a landmark map. The results are organized into two categories: 1) Normal segments where a lane marking is visible on at least one side; 2) Intersections where the vehicle makes turns and no lane marking is visible.

A. Generating Ground Truth. Since RTK GPS is not always reliable in urban areas, it cannot be used as ground truth. We found that even though RTK GPS is not always reliable, it can have high performance (about 0.05m pose error) on some segments of the test route. Every time we test, these high-performance segments are different. Therefore, by executing a route 3 to 5 times, we can collect high-accuracy pose data on all segments of the test route. However, these offline data cannot be directly used as ground truth, since it is impossible to drive the exact the same trajectories in our tests that we used to generate ground truth. To utilize ground-truth poses, we therefore rely instead on the LIDAR system and landmarks. We pick and measure positions of permanent landmarks using the LIDAR system when RTK GPS is in high-accuracy mode (FIXED RTK), and save locations of these landmarks as a map. Examples of potential landmarks are light poles and fire hydrants. Small intersecting surfaces can limit the position error caused by the shape of landmarks.

B. Landmark Registration. GPS is used to initialize the system and find the k closest observable landmarks. Even though absolute locations of found landmarks are inaccurate due to the urban canyon effect, the relative position between the vehicle and landmarks can be measured by the LIDAR system accurately. As shown in Fig. 3, *S* is part of our LIDAR coverage. We search in the neighbor area of the vehicle to find a nearest landmark *P* (red point) which is saved in our database, and then locate kNN (the k nearest-neighbor) LIDAR points (green points) around the found landmark within a given radius *r*. We use LIDAR to directly obtain relative pose **d** = **O'P'** from the vehicle *O*' to the average position of the selected points, and we assume that *P'* and *P* are equal in the global (North/East) coordinate system. Since the saved landmarks are collected when GPS is accurate enough (approximately 0:05m error from the real position), their locations are also reliable. Therefore the found landmark *P* in the

database can be used to calculate the vehicle ground-truth location by combining its current measured position (relative to the vehicle) d, i.e., distance and heading to the vehicle. If $P = (X, Y)^T$; $d = (x', y')^T$, the heading angle between local y-axis and global N-axis is θ, then the corrected location of the vehicle (the red car) is $O = (O_x; O_y)^T$:

$$O = P - \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \cdot d$$

We run a verification test, using FIXED RTK GPS to compare with our LIDAR landmark localization result. There is $0.06m$ error on average, which means that the LIDAR landmark localization method is accurate enough to serve as ground truth for comparing with the result of our approach.

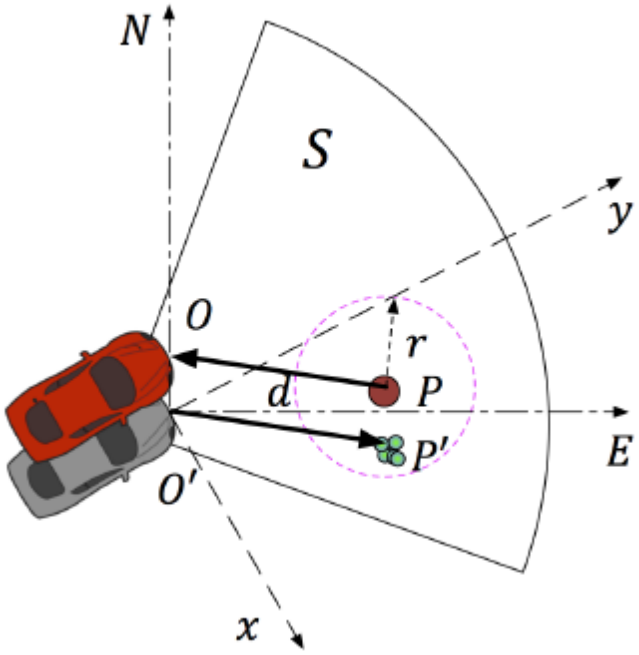
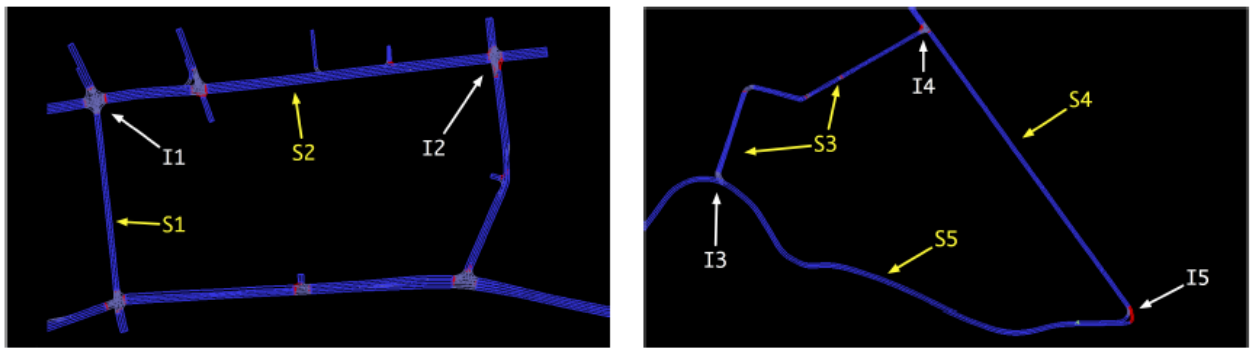


Fig. 3: Ground-truth landmark registration with LIDAR points and localization.

C. Test setup and results. For initialization, we evenly distribute particles across lanes. To achieve stable localization accuracy for different lane widths, we set constant distance between adjacent particles, instead of initializing a constant number of particles in an arbitrary lane. In our experiment, we set the distance to 5 cm. As suggested in [4], lane width in local areas varies from 2.7 to 3.6 meters. Therefore, there are 54 to 52 particles per lane. The algorithm updates at a rate of 40 Hz, on an Intel i3-level processor. Considering the typical speed limit in urban areas ($30mph = 13m/s$), the longitudinal update interval is only 32 cm. Fig. 4 shows our two test loops: the Oakland loop (Fig. 4a) and campus loop (Fig. 4b). Both of them are 2:1km in length, and we traverse them in a clockwise direction. These loops are in an urban area in Pittsburgh and surrounded by high buildings or hills, where GPS is not always accurate. Test loops include two different scenarios: normal segments and intersections. In normal segments, lane markings on at least one side can be reliably identified. In intersections, no lane marking is available in intersections, so dead-reckoning is activated.



(a) Oakland Loop (b) Campus Loop

Fig. 4: Test loops in urban area near Carnegie Mellon University. S_i are normal segments, I_i are intersections.

In order to illustrate the performance of different modules in our system (particle filter localization and dead-reckoning), we organize the result by these two scenarios. Table I and Table II show test results of normal segments S_i and intersections I_i, respectively. We only show results on the S_1, S_2 segments and I_1, I_2 intersections for the Oakland loop because landmarks are hard to recognize in other segments, so there is no validation data for those segments. e_{gps}, σ_{gps}, and $\max_{gps}$ are respectively the mean error, standard deviation and max error with respect to ground truth of the RTK GPS alone. e_{pf}, σ_{pf} and $\max_{pf}$ are the same quantities for the particle filter.

TABLE I: Lateral errors from RTK GPS and our approach in normal segments (m)

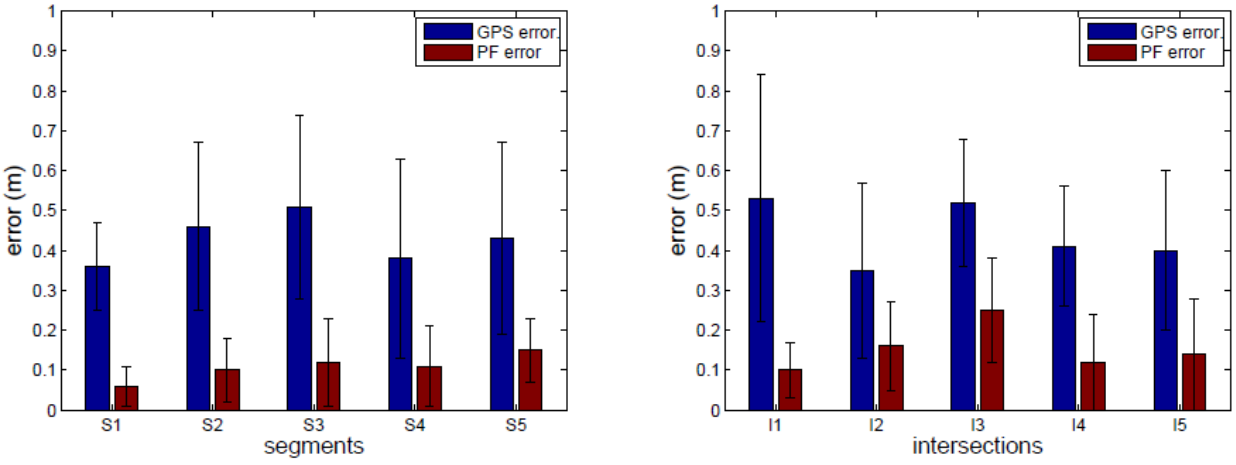
No.	e_{gps}	σ_{gps}	$\max_{gps}$	e_{pf}	σ_{pf}	$\max_{pf}$
S_1	0.36	0.11	0.66	0.06	0.05	0.27
S_2	0.46	0.21	1.03	0.10	0.08	0.41
S_3	0.51	0.23	1.17	0.12	0.11	0.73
S_4	0.38	0.25	1.19	0.11	0.10	0.44
S_5	0.43	0.24	1.12	0.15	0.08	0.50

TABLE II: Lateral errors from RTK GPS and our approach in intersections (m)

No.	e_{gps}	σ_{gps}	$\max_{gps}$	e_{pf}	σ_{pf}	$\max_{pf}$
I_1	0.53	0.31	0.95	0.10	0.07	0.35
I_2	0.35	0.22	0.96	0.16	0.11	0.45
I_3	0.52	0.16	0.70	0.25	0.13	0.47
I_4	0.41	0.15	0.65	0.12	0.13	0.50
I_5	0.40	0.20	0.68	0.14	0.14	0.49

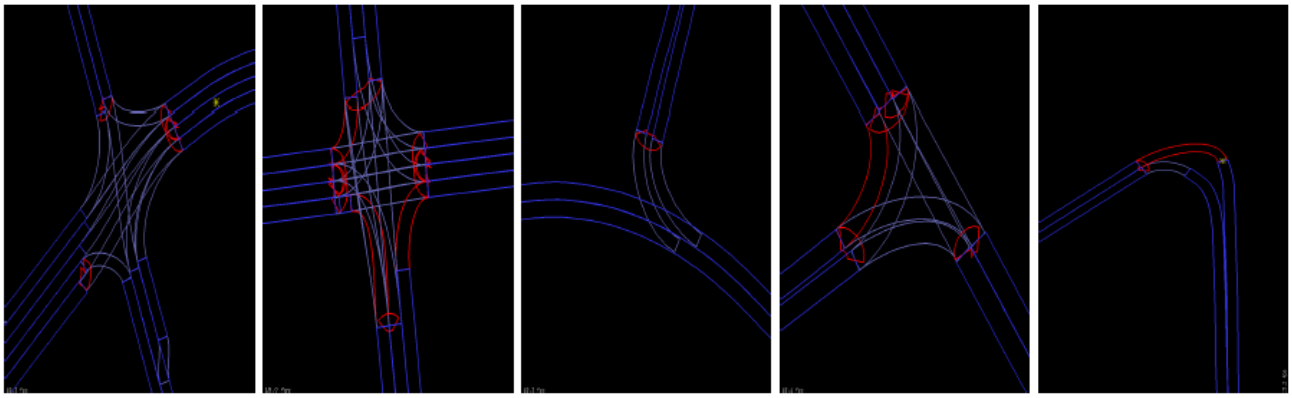
1) Normal segments: In normal segments, lane markings are available on at least one side. These segments include straight and curved roads. These segments are used to test the performance of our lateral localization system when lane markings are available and GPS is less

accurate. Even RTK GPS is not able to obtain accurate pose information, and it usually was 35cm to 51cm lateral error, as shown in the e_{gps} column of Table I. Normally, lane width is about 2.8m in urban areas, and vehicle width is about 2.0m, so even if the vehicle can perfectly follow the lane center, there is only 0.4m from each side of vehicle to the lane markings. Since the GPS can have 0.3m to 0.5m error, there is a high chance of crossing the lane marking and entering another lane. Moreover, even though it is legal to drive the car very close to a lane marking when remaining in one lane, other drivers cannot correctly estimate intention of the vehicle. Table I and Fig. 5a show the performance of the particle filter localization part of our approach. The particle filter solution provides more accurate and stable pose estimation than RTK GPS. Mean error of the new approach (column 5) is 29-36cm less than that of GPS (column 2). The standard deviation (column 6) of the new approach is also less than RTK GPS's (column 3), and the maximum error of the new approach (column 7) in each segment is less than that of RTK GPS (column 4), which means that the particle filter method is more stable and smooth.



(a) Normal segments errors. (b) Intersections errors.

Fig. 5: Error results in normal segments and intersections for using RTK GPS and our method.



(a) I_1 (b) I_2 (c) I_3 (d) I_4 (e) I_5

Fig. 6: Test intersections in urban area.

2) Intersections: Fig. 6 shows the geometry of our test intersections in detail. I_1 and I_2 are normal four way intersections, and I_3, I_4 and I_5 are T-intersections (One arm of I_5 is not drawn on our map, but it does not affect the result). The intersection scenarios test dead-reckoning performance. In normal segments, it is easy to observe lane markings and then calculate accurate lateral position by applying our particle filter approach. The system should also be able to report accurate pose when there is no lane marking available, and also when the car makes turns in intersections. If the system can achieve accurate localization in both normal segments and intersections, it can be applied to urban autonomous driving. Table II and Fig. 5b show the performance of the dead-reckoning part of our approach. It returns more accurate and stable results than the RTK GPS. The mean error of our method is less than half of RTK GPS's error. The standard deviation of our approach is also smaller and the maximum error is about 15- 20cm less than that of GPS, which means that pose from the new approach is more stable than RTK GPS. In each row of the tables, standard deviations of our approach (σ_{pf}) are less than that of GPS (σ_{gps}). Since the LIDAR landmark localization system naively uses LIDAR points to register known landmarks and LIDAR sensors' outputs are not stable, registered landmarks can keep jumping around their desired position. Also, a landmark object is not a point without area;

for example, cross-sections of poles are circles with radius of 5-10cm, and it is hard to determine the center of the landmark. Such errors result in unstable measurement. Therefore the standard deviation of our approach is close to the mean error. The maximum errors occur when there is incorrect landmark registration. For example, a walking pedestrian is detected by LIDAR and the points from the pedestrian may be accidentally registered to a nearby landmark. In this case, a wrong measurement is used to calculate the vehicle's pose, resulting in large error. Since such a case rarely occurs, the mean error and standard deviation are much smaller than those maxima.

Conclusions/Recommendations

We have created a lateral localization system that consistently gives accurate and stable estimates of lateral location in a typical urban environment. This approach is robust and accurate not only for straight or single lanes, but also when lane markings are not visible and making turns in intersections by applying dead-reckoning system. Even when the vehicle only receives a poor GPS signal with large drifting and jumping, more accurate lateral location can be achieved by combining other sensors' outputs, such as wheel speed, steering wheel angles and lane marking locations. In addition, we make specific adjustments to the general particle filter. The updating and weighting method make the particle filter work for this special application, and the re-initialization criteria and an anti-deprivation method make the algorithm more effective and reliable.

Future work should include:

- Developing an automatic landmark identification system which can generate the landmark map autonomously
- Developing a more sophisticated method for registration between LIDAR points and saved landmarks

- Move from a high-accuracy, high-cost GPS (which still has dropouts in urban canyons) to a low-cost GPS more relevant for production vehicles

Works Cited

[1] Popp, Karl, and Schiehlen, Werner. Ground Vehicle Dynamics. Springer Verlag, Berlin-Heidelberg, 2010.

[2] Jazar, Reza N., Vehicle Dynamics: Theory and Application. Springer Science & Business Media, 2013.

[3] Wei, Junqing, Snider, Jarrod M., Kim, Junsung, Dolan, John M., Rajkumar, Raj, and Litkouhi, Bakhtiar. "Towards a viable autonomous driving research platform." In Intelligent Vehicles Symposium (IV), 2013 IEEE, pp. 763-770. IEEE, 2013.

[4] "A policy on geometry design of highways and streets," AASHTO, 5th Edition, ISBN 1-56051 (2004): 263-266.