# Lane-level map-matching with integrity on high-definition maps

Franck Li, Philippe Bonnifait, Javier Ibañez-Guzmán, Clément Zinoune

# Lane-Level Map-Matching With Integrity on High-Definition Maps

Franck Li[1,2], Philippe Bonnifait[1], Javier Ibanez-Guzman[2] and Clément Zinoune[2]

***Abstract*— Navigation maps provide important information for Advanced Driving Assistance Systems (ADAS) and Autonomous Vehicles. This paper presents a method estimating a set of likely map-matched hypotheses containing the correct solution with a high probability. This addresses the problems encountered when using a high definition map when a large number of ambiguities arise. These occur for instance, when only inaccurate prior information on position is available at initialization. The method uses lane-level accurate maps with dedicated attributes, such as connectedness and adjacency, and an automotive Global Navigation Satellite System (GNSS) receiver assisted with dead-reckoning (DR) sensors. GNSS can be so inaccurate that map-matching relies mainly on DR estimates, the GNSS fixes being used as uncertain estimates with protection levels. This paper proposes a formalization of the map-matching integrity problem as well as a sequential method using a Particle Filter providing a reliable set of map-matched hypotheses. The performance is evaluated using data acquired in public roads.**

## I. INTRODUCTION

To navigate, an intelligent vehicle needs a digital representation of the world as perceived by its on-board sensors. To this end, a priori information on features of the road network is mandatory. These are stored in digital road maps where description of the road network provides geometric and contextual information such as lane markings, traffic signs, etc.

A strong effort is currently made by map suppliers to meet the requirements of intelligent vehicles, that require high level of accuracy for new advanced tasks.

To access the relevant information, vehicles must be localized with respect to these maps. For this purpose, vehicles rely on Global Navigation Satellite Systems (GNSS) receivers which provide an absolute position on Earth. The process of associating these position estimates to roads on a map is known as *Map-Matching*. Considering GNSS measurements generally bear errors that can reach several meters [1], matching the true position on a lane-level map remains a challenging issue.

This paper presents a method for a lane-level map-matching using a Particle Filter (PF), focusing on the integrity of the result. Section II introduces the concept of map-matching integrity and demonstrates how PFs can be used in this sense. Secondly, Section III presents an optimized use of the map in the context of map-matching: as PF is usually calculation-heavy, efficiently use the map is very important. The implementation of the algorithm is explained in Section IV before presenting experimental results of this algorithm using data acquired in public roads in Section V.

[1] Sorbonne universités, Université de Technologie de Compiègne, CNRS, Heudiasyc UMR 7253, CS 60319, 60203 Compiègne cedex, France. {franck.li,philippe.bonnifait}@hds.utc.fr

[2] Renault s.a.s., Guyancourt, France. {javier.ibanez-guzman, clement.c.zinoune}@renault.com

This research has been carried out in the scope of the SIVALab laboratory shared between Renault, UTC and CNRS.

## II. MAP-MATCHING INTEGRITY

### A. Multiple Hypotheses Policy

The goal of the method described here differs from the most common use of map-matching for automotive application used in navigation systems for which map-matching should give a *single* position estimate to the user as a result. This corresponds to the usual need of such a system: a single position must be used to calculate a route and, in most cases, if this position is erroneous, the user is able to see the error and wait for a correct matching by disregarding the given information.

On the opposite, map-matching aimed to autonomous navigation systems must not be over-confident about its results: the worst-case scenario would be to provide an erroneous single position as there may be no human to detect the error and to apply a correct action. If there is an ambiguity in the matching, then the algorithm should keep it in mind and not make a decision. This is the notion of *map-matching integrity* [2]: the desired result is to get a method providing in real-time a set of likely matched lanes in which the correct lane is highly likely to make part. The size of the set has to be kept as small as possible. In other words, that would be a single lane if the pose estimate uncertainty is small and if there is no ambiguity or multiple ones that must include the real matched solution. This topic begins to be very important as progresses in intelligent vehicles are made and starts being developed in the literature [3], [4].

### B. Particle Filtering

Most of existing map-matching algorithms do not tackle this integrity problem as they perform at the macro-scale road-level. Ambiguities can be present at this scale, especially in dense areas, but advanced techniques, such as fuzzy logic map-matching applied by Quddus et al. [5] have a great capability to resolve them. A single solution can be found most of the time once the matching has converged. When coming down on the lane level, a lot of ambiguities rise. For example, Quddus' fuzzy inference system uses the vehicle's heading as an input criterion, but on a 2-lane road, both lanes have very close headings: it is therefore no longer a strong discriminant information. The algorithm will most likely not be able to decide on which lane to match and therefore fail.

With this perspective, a Particle Filtering approach is chosen here, as it is able to manage multiple hypotheses. Another

solution is to assign a Kalman filter to each hypothesis in a multiple model framework with a Gaussian mixture posterior approximation [6] but it leads to a more complicated implementation. Some matching algorithms have already been implemented using PF [7], [8], [9], but did not emphasize on the integrity possibilities of such methods.

In our model, the state of each particle is hybrid (continuous and discrete, see Eq. 1). $X_p^i = (x^i, y^i, \psi^i)$ represents the 2D pose and $ml^i$, the matched link of the $i$th particle. Each particle has an associated weight $w^i$ representing its likelihood.

$$X^i = (X_p^i, ml^i) = (x^i, y^i, \psi^i, ml^i) \tag{1}$$

Please note that the particles are not constrained to the links (e.g. [10]) which can induce curvilinear distortion at nodes or junctions difficult to handle and need careful management of the abscissa like proposed in [11].

The algorithm described in this paper uses an automotive GNSS receiver, DR sensors and a centimeter-accuracy lane-level map (as described in more details in Section III).

*1) Evolution model:* The evolution model is a unicycle since we measure the speed and yaw rate of the car:

$$\begin{cases} x_t^i &= x_{t-1}^i + v_t \cdot \Delta t \cdot \cos \psi_{t-1}^i \\ y_t^i &= y_{t-1}^i + v_t \cdot \Delta t \cdot \sin \psi_{t-1}^i \\ \psi_t^i &= \psi_{t-1}^i + \omega_t \cdot \Delta t \end{cases} . \tag{2}$$

$U_t^i = [v_t^i, \omega_t^i]^T$ is the input vector of the $i$th particle, with $v_t^i \sim \mathcal{N}(v_{raw}, \sigma_v^2)$ and $\omega_t^i \sim \mathcal{N}(\omega_{raw}, \sigma_\omega^2)$, based on raw measurements of the vehicle's speed and yaw rate $(v_{raw}, \omega_{raw})$ with an added random noise. The noise allows the particles to spread during their evolution, being added independently for each particle.

For matching the link, two approaches are used: particles are map-matched once at the filter initialization (see Section IV-A) and then follow the links logically using attributes stored in the lane-level map describing connectedness and adjacency (as described in Section III). This choice is very efficient in terms of real-time computation.

*2) Map Likelihood Computation :* When a map is processed as a raster (see [7]), a likelihood field can be computed in advance to accelerate the processing. In our case with a vectorial map, the likelihood is modeled as a Gaussian to compute the probability of belonging to a given lane. The lateral (also called cross-track) distance $y_t$ to the closest matched point is used to compute it. Gaussian cuts being fitted on links, the likelihood is maximum if the distance is null (the particle is right on the centerline) and quickly decreases as the particle gets to a distance greater than the half width of a driving lane. This likelihood is then used to update the weight of each particle recursively using a bootstrap strategy:

$$w_t^i = w_{t-1}^i \cdot p(y_t | X_k^i) \tag{3}$$

Note that this calculation takes into account the complete pose: the likelihood depends on the cross-track distance but also on the heading difference between the hypothesis and its matched link. Indeed, a vehicle is more likely to follow a heading close to the direction of the road it is following.

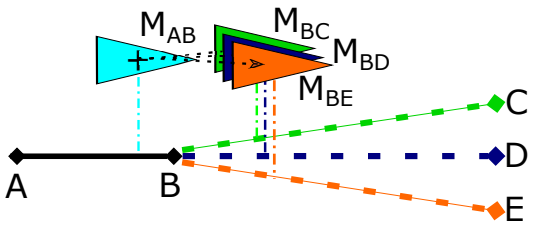


Fig. 1. Lane Forking where particle cloning occurs: the particle $M_{AB}$ is duplicated into $M_{BC}$, $M_{BE}$ and $M_{BD}$

*3) GNSS positions as points with Protection Levels:* In order to provide map-matching with high integrity, estimated positions computed by the GNSS receiver are only used to limit the spread of the particles and not to update the weights of the particles. We suppose that every position is associated with a Horizontal Protection Level (HPL) which is a bound of the estimation error (associated with an integrity risk) and used in aeronautics for monitoring the integrity of a GNSS position [12]. As positions can be greatly biased, especially in urban environments, the filter considers an area around every position computed by the GNSS receiver instead of a single point as an input. This area is the HPL, i.e. an area where the true position is highly likely to be located: all the particles inside this HPL are considered valid whereas the other are eliminated by the filter. This strategy can be seen as a gating process. The HPL depends greatly on the environment the vehicle evolves in: it can be conservative in poor satellites visibility conditions (e.g. tens of meters) or, inversely can be very small with high accuracy receivers and modern point computations like *Precise Point Positioning* [13].

*4) Resampling Strategy:* To avoid the degeneracy of the particles set, resampling is applied using Kitagawa's strategy [14], with a threshold of 66% effective particles. A low variance resampling [15] is performed to redraw the set of particles. Kitagawa's method is preferred to a systematic resampling at every step to favor particles spreading which allows a better exploration of the 2D-space.

*5) Particle Cloning:* In the same perspective of throughout exploration, a new strategy is adopted at lane forking: to be sure to explore all possible path, a particle arriving on a lane forking is cloned and each clone follows one of the connected path hypotheses (see Fig. 1). This creates a variable size of the set of particles (which is not a problem in practice if enough memory has been allocated for the filter). To avoid an exponential rise in complexity, a maximum number of particles is set (e.g. 150% of the original number of particles). Moreover, at each resampling, only the original number of particles is redrawn; the additional particles are of course taken into account during this step, but only the most likely will survive.

## III. SEMANTIC USE OF THE MAP

The map used in this study is considered faultless and highly accurate like in [11] and [16]. This allows to focus

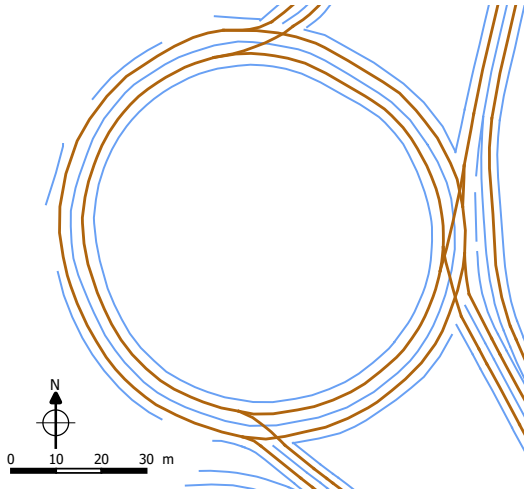


Fig. 2. Detail of the map of a roundabout in Compiègne, France. The centerline of the lanes are drawn in brown and the lane markings in blue. Complete connectedness is notably visible in the roundabout entrances and exits

on the map-matching algorithm considering no error from the map. This assumption is acceptable as map providers are currently working on high accuracy maps of public road with a large coverage. To comply with this assumption, a high accuracy lane-level map has been made by a mapmaker, covering 4 km of open roads in Compiègne (details on Fig. 2). An absolute accuracy of 2 cm is guaranteed. This prototype study map is oriented towards an intelligent automotive use: it includes additional details that are not present in most maps used in current navigation maps (e.g. lane markings information).

### A. Lane-Level Map Structure

This *mesoscale* map [17] is stored in a SQLite database. The *Spatialite* library is used, extending SQL functionalities with useful geographic utilities such as spatial requests and data structures (e.g. 2D/3D points, polylines). The main layers in this map are described as follows:

- *Links* that represent each driving lane's centerline. They are represented by polylines. A polyline is a sequence of *Shapepoints* (2D or 3D points, depending on the type of map) that follows the geometry of a drivable lane. This allows to not be limited in the representation of straight lanes but also curves. Subdivision of links (delimited by two shapepoints) will be referred to as *segments*.
- *Nodes* that binds consecutive links together; they denote most of the time intersections, but could also mark lane merging, splitting. Nodes represent the connectedness information in the link network. They enable fast link searching methods by storing parent and child links IDs.
- *Lane markings* are also stored in the database. In addition to the geometric description of the road marking, this layer contains attributes to identify the marking type (e.g. solid line) and associated link ID.

These three tables form the basis on which the database is built.

### B. Semantic Information

A digital road map can be viewed only as a Geographic Information Database, containing the coordinates of the different road structures. This is the natural approach when dealing with a digital map. But the most recent digital maps, such as the one used in this article, contain richer information that enables a more interesting processing. Once matched on the map, a hypothesis can heavily rely on the semantic information about the road network to evolve. For instance, every link accessible from a given matched position is easily accessible from the database, without the need of costly distance calculation. The map thus provides an evolution framework relatively independent to the 2D-plan geometry, as such a position can be projected on the map for a 1D evolution.

### C. Adjacent Links

Another important feature in our research map is the adjacency information: every link is aware of the links on its sides. This is mandatory to check for matching ambiguities, as adjacent links present the biggest challenge for map-matching. This information being available directly from the map, no costly calculation is neither needed and the exploration of hypotheses is greatly improved. This adds the second dimension of the map exploration, after the longitudinal one, provided by the link succession information. The matching algorithm thus have a complete framework to use efficiently the map, removing part of the heavy calculation and taking advantage of the efficient map design.

## IV. METHOD IMPLEMENTATION

In this section, we describe the main steps to implement efficiently the map matching method described in the previous sections and illustrated by Fig. 3. SQL based map format are efficient when using large map (i.e. large database size) due to the possibility to make spatial queries. Although the query is fast to execute, query formatting and returned data parsing cause significant CPU load. Its use is therefore limited to punctual queries using caching methods as implemented by Bonnifait et al. [18], where the map information was stored into memory as the car is moving from one position to another. The approach presented here loads the whole map information once when starting, as its size is relatively limited.

The initial filter sample set is populated and matched to the road network to finalize the initialization process. The filter then runs the real time execution loop. These two processes (see Fig. 3) are described in the following paragraphs.

### A. Initialization Step

After loading the SQL map into an internal data structure, the filter initializes on the first valid GNSS position received. Particles are then generated around this position, in a circular pattern to cover the full area corresponding to the associated HPL. Each particle is then matched to its corresponding link. This step follows a *point-to-curve* method which selects

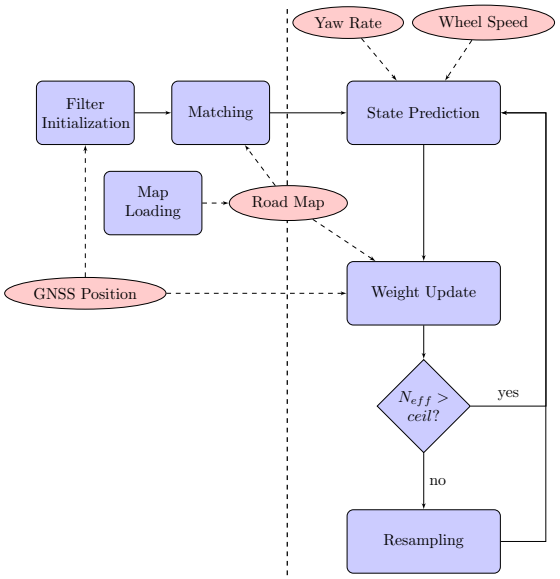


Fig. 3. Flowchart of the filter. Heavy calculation is kept out of the main processing loop

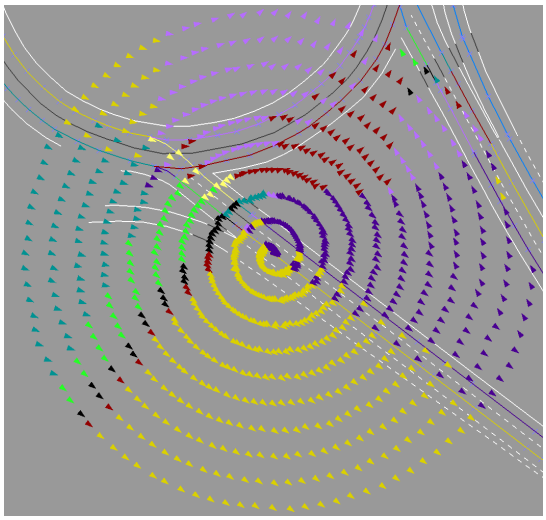

Fig. 4. Particle Initialization: colors denote the matched link. The initial heading corresponds to the matched link. Some particles are far from the links due to the 50 m HPL (high value chosen to be very conservative), but will be quickly eliminated during an resampling step.

the link candidate with the lowest Euclidean distance to the particle.

The algorithm makes the assumption that the vehicle is driving on-road, and that this road is in the map. Therefore, it is reasonable to initialize the particles' heading to be the link's (see Fig. 4). This initialization ensures considering all the links present on the map around this position. Some particles are notably created outside the drivable area represented in the map. It illustrates the fact the particles are not strongly constrained onto the map and can evolve in the 2-dimensional space and not only on the centerlines. This provides a spatial flexibility to the filter, only limited by the decreasing likelihood of particles that get too far away from a link. These particles will be quickly eliminated by the filter during the update step.

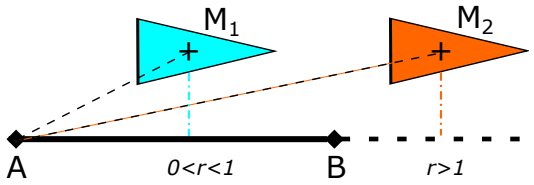


Fig. 5. Particle $M_1$ is still on the current segment (AB) while $M_2$ leaves it. This is determined by calculating the ratio described by Eq. 4

### B. Main Filtering Loop

Once associated to the road network, the free 2D spatial evolution of the particles becomes an evolution heavily tied to the map. The distance calculation is therefore simplified and the filter efficiency is improved. This process is synchronized with the proprioceptive information input rate (10 Hz).

The ID evolution of the particle is easily determined by simply calculating if its projection has left the currently matched segment. It is done simply by projecting the particle as shown on Fig. 5. The ratio

$$r = (\overrightarrow{AB} \cdot \overrightarrow{AM})/||\overrightarrow{AB}||^2 \tag{4}$$

is computed and if $r > 1$ (respectively $r < 0$), the particle has left the current segment and has to be associated with the next one (respectively the previous one) by simply using the connectedness information of the table of the map.

This is the only calculation needed to make the particles evolve on the map. Cloning and resampling are then applied as explained before. Each time a new GNSS position is available, the HPL gating is applied. The filter is designed to need as little calculation as possible.

### C. ID Estimates Of The Map-Matched Points

To estimate the matching hypotheses, the filter simply takes the weighted mean of the particles' pose. This calculation is done separately for each different link hypothesis:

$$X_{hyp_j} = \sum_i \overline{w_j^i} \cdot X_p^i. \tag{5}$$

$X_{hyp_j}$ is the $j$th matching hypothesis, $\overline{w_j^i}$ is the normalized weight of the $i$th particle of the $j$th hypothesis and $X_p^i$, its pose. This computes an estimate for each hypothesis with its associated weight. It is therefore possible to determine the most probable one.

## V. RESULTS

### A. Experimental Setup

A C++ implementation of the algorithm has been developed using the Pacpus framework[1], that provides easy integration in the laboratory's test vehicle and offline data replay. An experimental vehicle was used for real road data acquisition. The car was equipped with a Septentrio PolaRx4 GNSS receiver and DR information was accessible directly from the vehicle CAN bus.

The algorithm has been tested using Pacpus data replay capability running in real time. The test trajectory (see

[1]developed at Heudiasyc. More info at `pacpus.hds.utc.fr`

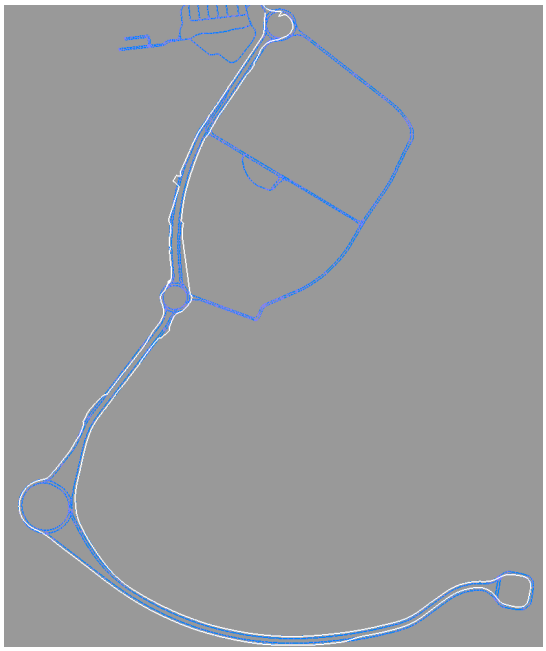

Fig. 6. Test GNSS trace (in white) over the lane map (in blue). Large GNSS errors are clearly visible.

Fig. 6) is representative of a peri-urban trip (2x2 lanes roads, including roundabouts). Some high-rise building are present, as well as open-sky conditions at the bottom of the figure. The algorithm runs in real-time conditions with 1000 particles on an AMD A8-4500M CPU@1.90 GHz with 8 Go RAM.

Fig. 7 shows the GNSS positioning error during the test. The error is relatively contained with a mean error of 0.5 m and a standard deviation of 0.82 m but can still interfere with a lane-level map-matching. Lots of spikes (up to 8 m) are notably present which can be highly problematic for the matching stability. The filter performs well even with these errors (note that losses of GNSS signal occur in the dataset, but are not reflected in Fig. 7 as no error can be computed). The HPL is set to 50 meters, largely exceeding the GNSS error. This high value reflects a low confidence on the GNSS position, the algorithm being very conservative. It could be tuned to reflect the situation more closely and put higher confidence to this input.

As the test trajectory is quite long (about 4 km in a 10 minute-long acquisition), the following graphs split it to focus on specific points. They represent the variation of the different hypotheses weights over time.

Fig. 8 shows the situation right after initialization, 4 hypotheses are found (the initialization is done next to a roundabout, a difficult situation). 2 of them are eliminated as the vehicle leaves the roundabout, leaving the 2 driving lanes as the only surviving hypotheses. Ground truth is the green line, fluctuation is visible between the two adjacent lanes, as ambiguities cannot be resolved by the filter at this time instant.

Inside roundabouts (Fig. 9), a high fluctuation is observable: this is explained by the high variation of yaw rate in these locations. It is a difficult scenario, but the algorithm still manage to determine a correct set of hypotheses.

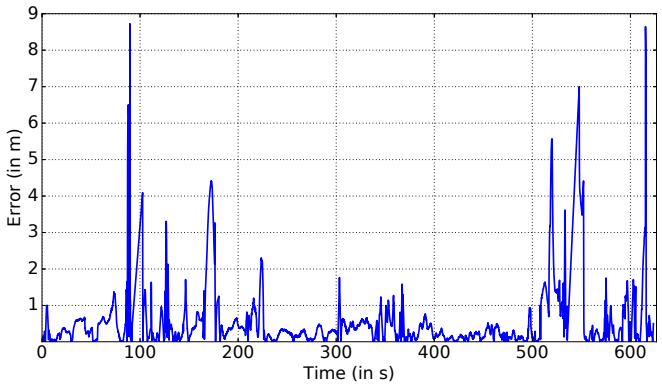


Fig. 7. Errors of GNSS fixes compared to real positions. Error spikes are clearly visible.

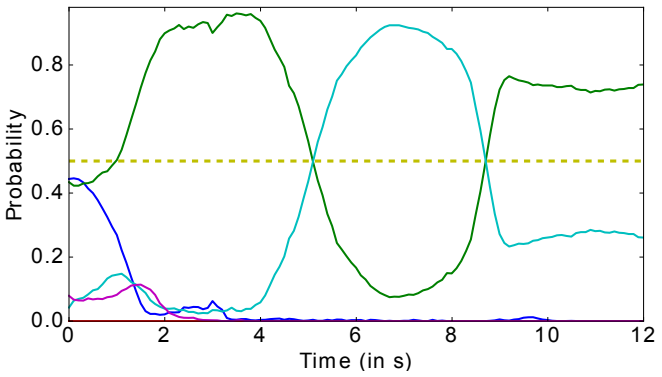


Fig. 8. Probabilities of the map-matched hypotheses with respect to samples. At initialization, 4 hypotheses are found but soon collapse into 2, corresponding to the 2-lane road (ground truth in green).

Fig. 10 shows a cloning event happening at the location shown on Fig. 11: the rightmost lane forks and forms a new lane (a deviation) which the test vehicle is taking. In the first half of Fig. 10, the ground truth is represented by the green line. Then the purple hypothesis emerges after the forking and becomes the new ground truth. The two old hypotheses lose weight while the new one gets more and more important, until becoming the dominant hypothesis.

Repeatability has been studied by executing the filter on the same input data 15 times. Ground truth has been determined by manually labeling the dataset with the correct

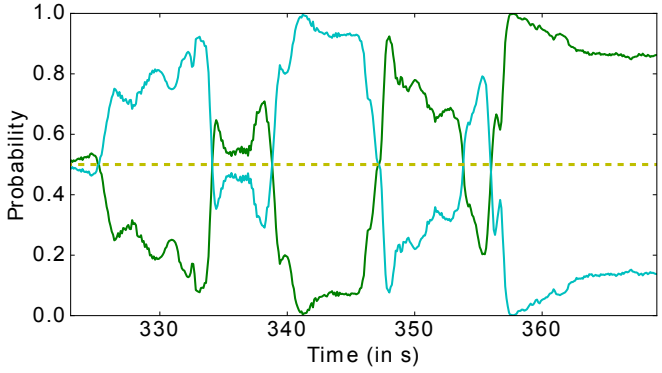


Fig. 9. Fluctuations inside roundabouts (high yaw rate variation, ground truth in green).

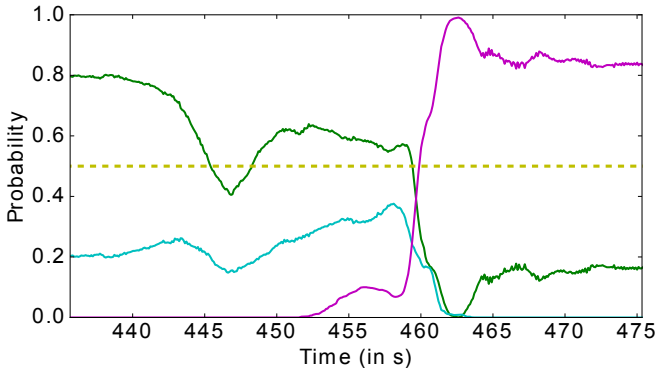


Fig. 10. Filter behavior at a lane forking: the hypothesis corresponding to the new lane (actual path taken) quickly gains importance, while the other two drop.

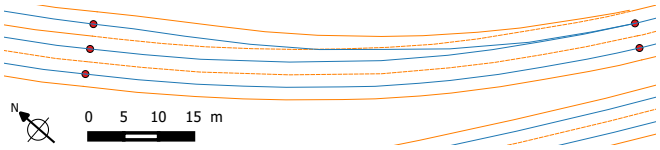


Fig. 11. Details of the location described in Fig. 10. A new lane is created a the rightmost (top of the figure): cloning happens.

lane. Results are shown in Table I: the filter does not keep the correct matching hypothesis only 2.4% of the time. Otherwise, the set of hypotheses always includes the ground truth. But the ground truth is only identified as the most probable hypothesis 51.3% of the time. This illustrates that the filter cannot discern two hypotheses on which ambiguity is still present. These two figures are in consistent with the idea of a map-matching integrity (keeping the correct matching but also the other likely candidates). Moreover, the availability of the algorithm is correct as it provides 2 or less hypotheses 76% of time, and 3 or less 94% of the time.

## VI. CONCLUSION

The lane-level map-matching algorithm presented in this paper developed a new approach for solving the problem. It aims at providing results with high integrity by estimating more than an unique solution that could be erroneous. The map-matching output can therefore be a set of multiple lane hypotheses in ambiguous situations or when pose uncertainty is high. This result is beneficial for all the systems that need to get confidence indicators associated with the map-matching procedure. The proposed method has been designed to exploit as much as possible the semantic information stored in the high definition map. As shown by the results, its behavior is reliable, the convergence is quite fast (in the order of seconds after a first GNSS fix, with a moving vehicle and dead-reckoning information). This quick convergence is crucial when guiding an autonomous vehicle, in particular if the approach resides only on relative localization

To be used more effectively such a technique has to be further improved. The current algorithm uses only a GNSS receiver in a very careful way thanks to HPL indicators. To improve the matching, a next step is to add data from a perception system and fuse them with the algorithm output. This data fusion is necessary to remove more ambiguity on the lane-level matching. The filter could also be implemented using a graphics processor (GPGPU) to associate the parallel paradigm of these architectures with the highly parallel nature of Particle Filtering.

TABLE I

MATCHING CORRECTNESS IN PERCENTAGE FOR 15 REPETITIONS

| Metrics | % |
|---|---|
| Set including Correct Matching | 97.6 |
| Set of 3 or less hypotheses | 94.1 |
| Correct Best hypothesis | 51.3 |

## REFERENCES


[1] E. D. Kaplan and C. J. Hegarty, *Understanding GPS*, 2006.

[2] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Integrity of map-matching algorithms," *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 4, pp. 283–302, 2006.

[3] M. Jabbour, P. Bonnifait, and V. Cherfaoui, "Map-Matching Integrity Using Multihypothesis Road-Tracking," *Journal of Intelligent Transportation Systems*, vol. 12, no. 4, pp. 189–201, oct 2008.

[4] C. Fouque and P. Bonnifait, "Matching raw gps measurements on a navigable map without computing a global position," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 887–898, June 2012.

[5] M. Quddus, R. Noland, and W. Ochieng, "A High Accuracy Fuzzy Logic Based Map Matching Algorithm for Road Transport," *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, vol. 10, no. 3, pp. 103–115, sep 2006.

[6] D. Alspach and H. Sorenson, "Nonlinear bayesian estimation using gaussian sum approximations," *IEEE transactions on automatic control*, vol. 17, no. 4, pp. 439–448, 1972.

[7] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[8] J. Rabe, M. Necker, and C. Stiller, "Ego-lane estimation for lane-level navigation in urban scenarios," in *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, jun 2016, pp. 896–901.

[9] I. Szottka, "Particle filtering for lane-level map-matching at road bifurcations," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE, oct 2013, pp. 154–159.

[10] P. Merriaux, Y. Dupuis, P. Vasseur, and X. Savatier, "Wheel odometry-based car localization and tracking on vectorial map," in *2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 1890–1891.

[11] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 420–425.

[12] S. Feng, W. Y. Ochieng, D. Walsh, and R. Ioannides, "A measurement domain receiver autonomous integrity monitoring algorithm," *GPS Solutions*, vol. 10, no. 2, pp. 85–96, 2006.

[13] M. D. L. Samper, M. M. R. Merino, G. T. Gonzáles, and D. B. Martí, "PPP for advanced precise positioning applications, including reliability bound," in *27th International Technical Meeting of The Satellite Division of the Institute of Navigation*, 2014, pp. 2478–2493.

[14] G. Kitagawa, "Monte carlo filter and smoother for non-gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[15] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 1999–2000, mar 2002.

[16] R. Toledo-Moreo, D. Betaille, and F. Peyret, "Lane-Level Integrity Provision for Navigation and Map Matching With GNSS, Dead Reckoning, and Enhanced Maps," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 100–112, mar 2010.

[17] Jie Du and M. Barth, "Next-Generation Automated Vehicle Location Systems: Positioning at the Lane Level," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 48–57, mar 2008.

[18] P. Bonnifait, J. Laneurit, C. Fouque, and G. Dherbomez, "Multi-Hypothesis Map-Matching Using Particle Filtering," in *16th World Congress for ITS Systems and Services*, 2009, pp. 1–8.