



HAL
open science

Landmark-based Geopositioning with Imprecise Map

Noël Nadal, Jean-Marc Lasgouttes, Fawzi Nashashibi

► **To cite this version:**

Noël Nadal, Jean-Marc Lasgouttes, Fawzi Nashashibi. Landmark-based Geopositioning with Imprecise Map. VEHITS 2025 - International Conference on Vehicle Technology and Intelligent Transport Systems, Apr 2025, Porto, Portugal. hal-04911247

HAL Id: hal-04911247

<https://inria.hal.science/hal-04911247v1>

Submitted on 24 Jan 2025




HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Landmark-based Geopositioning with Imprecise Map

Noël Nadal¹^a, Jean-Marc Lasgouttes¹^b and Fawzi Nashashibi¹^c

¹*Inria, France*

{noel.nadal, jean-marc.lasgouttes, fawzi.nashashibi}@inria.fr

Keywords:

Autonomous Navigation, Autonomous Vehicles, Landmark-Based Localization, Fusion and Estimation

Abstract:

This paper introduces a novel approach for real-time global positioning of vehicles, leveraging coarse landmark maps with Gaussian position uncertainty. The proposed method addresses the challenge of precise positioning in complex urban environments, where global navigation satellite system (GNSS) signals alone do not provide sufficient accuracy. Our approach is to achieve a fusion of Gaussian estimates of the vehicle's current position and orientation, based on observations of the vehicle, and information from the landmark maps. It exploits the Gaussian nature of our data to achieve robust, reliable and efficient positioning, despite the fact that our knowledge of the landmarks may be imprecise and their distribution on the map uneven. It does not rely on any particular type of sensor or vehicle. We have evaluated our method through our custom simulator and verified its effectiveness in obtaining good real-time positional accuracy of the vehicle, even when the GNSS signal is completely lost, even on the scale of a large urban area.

1 INTRODUCTION


Global positioning or geopositioning (Kumar and Muhammad, 2023) is one of the fundamental tasks required for autonomous vehicle navigation. The need is to be able to obtain a decimeter or even a centimeter position accuracy, in terms of global accuracy. Satellite signal receivers (GNSS) such as Galileo, GPS, Glonass or Baidu are widely used for less precise geolocation. However, when it comes to navigation in urban environments, these solutions suffer from unavailability and conceptual failures (Bresson et al., 2017): multiple paths due to urban canyons, unfavorable satellite configurations, atmospheric conditions, etc. On the other hand, solutions based on GNSS RTK with centimeter accuracy are very expensive.


In order to remedy this, the positioning solutions envisaged are based on the use of on-board and remote digital maps (Chalvatzaras et al., 2022; Elghazaly et al., 2023). Such maps are assumed to provide the autonomous vehicle with information, some of which can be perceived by their sensors and represented by point clouds (Li et al., 2021), occupancy grids (Al-


sayed et al., 2015), or lane graphs (Zürn et al., 2021; Büchner et al., 2023).

Another possibility is to use maps holding geopositioned local information (or landmarks) (Qu et al., 2018). These landmarks can be modeled as two- or three-dimensional points (Stoven-Dubois, 2022; Zhang et al., 2023) or as more complex geometric shapes (Ruan et al., 2023), and should be natural features (e.g. trees) or artificial features (e.g. poles, road signs, traffic lights, building corners, road markings) that are easily recognizable by vehicles on the road. In order to estimate a geo-referenced position, a vehicle uses its on-board perception sensors to detect certain landmarks in the immediate environment, either on its own (Stoven-Dubois, 2022; Weishaupt et al., 2024) or collaboratively (Chen et al., 2020), and then proceeds with matching them with the ones stored in the map. Naturally, the more vehicles detect landmarks, the more there will be in the map, which requires performance issues to be tackled as well (Alsayed et al., 2015). Another challenge is then to keep the landmarks map robust and accurate (Dissanayake et al., 2002; Sun et al., 2022; He et al., 2022).

In this paper, we assume that we have at our disposal a map of the urban road environment and an associated landmark map with Gaussian uncertainty (the reason for this choice will be explained later).

^a  <https://orcid.org/0009-0005-0340-6836>

^b  <https://orcid.org/0000-0001-7507-0378>

^c  <https://orcid.org/0000-0002-4209-1233>

Each landmark comes with a Gaussian error and we propose a global positioning method using this imprecise information. Here, we are addressing the problem of geopositioning in a city. Landmarks should be visible and detected with sufficient reliability by the algorithms at our disposal, despite the specific constraints of an urban environment: for instance, the risk that landmarks can be partially or totally occluded by surrounding mobile or parked vehicles. The proposed solution must enable the vehicle to position itself precisely, and without the help of a GNSS system.

We propose a vehicle geopositioning approach using landmarks whose precise positions are not necessarily known. This method is effective enough to be ran in real time, assuming that the vehicle's speed corresponds to what is generally observable in an urban environment. Based on the sensors' and the map's respective accuracy, we are able to determine both the most likely position of the vehicle and its Gaussian error. This is not a SLAM (Simultaneous Localization And Mapping) (Durrant-Whyte and Bailey, 2006) approach as the construction of the map (which is more sparse than in most SLAM settings) is handled separately, although it influenced the way we defined our map while working on this approach.

The paper is organized as follows: Section 2 gives further details regarding our methodology and the workflow of the method. The experiments performed on our simulator are then reported and analyzed in Section 3. Finally, Section 4 outlines future work.

2 METHODOLOGY

2.1 Data Association

We need to associate the perception-based observations and the landmarks that are already known and stored in the map. We take into account the covariance between the positions coordinates, in order to take advantage of the Gaussian model adopted. We have adopted a GCB (Geometric-based Compatibility Branch and Bound) (Neira et al., 2003; Breuel, 2003) algorithm, in order to carry out this type of data association.

This algorithm attempts to construct a set of pairs, each consisting of an observation and a landmark, that satisfy the given compatibility constraints. The search for the best-fitting such set is performed by incrementally constructing the tree of the solution space, which allows for efficient search space pruning. Starting from an empty set, the algorithm proceeds in a depth-first branch and bound manner. At the leaf level of the tree, the algorithm checks whether it has come up with

a hypothesis that has a better compatibility score than the current best one. For this purpose, we consider a unary constraint and a binary constraint.

For the unary constraint, we calculate a compatibility score between each observation and each landmark. This score depends on both the Euclidean distance between their positions, and on the covariance associated with these data. This score is then compared with a threshold, that is set at the beginning of the experiment, to determine whether or not the association is possible, which leads to the unary constraint. Formally, let us consider that an observation is given by a multivariate Gaussian distribution vector \mathbf{o} and a covariance matrix $\Sigma_{\mathbf{o}}$, and a landmark position by a multivariate Gaussian distribution vector \mathbf{m} and a covariance matrix $\Sigma_{\mathbf{m}}$. We define $\mathbf{d} = \mathbf{m} - \mathbf{o}$ and $\Sigma_{\mathbf{d}} = \Sigma_{\mathbf{o}} + \Sigma_{\mathbf{m}}$. The constraint between this observation and this landmark is then defined as:

$$\text{unary}(\mathbf{m}, \mathbf{o}) \stackrel{\text{def}}{=} \mathbf{d}^T \Sigma_{\mathbf{d}}^{-1} \mathbf{d} < \chi_{2, \alpha}^2 \quad (1)$$

where $\chi_{2, \alpha}^2$ is a χ^2 random variable with two degrees of freedom, and such that $\alpha = 0.95$.

While the unary constraint measures the possibility of a match, the binary constraint evaluates the compatibility between two matches. Given two landmarks i and j , we denote $\Sigma_{\mathbf{m}|ij}$ the covariance submatrix of $\Sigma_{\mathbf{m}}$ using only rows and columns that correspond to i and j . We then define the following.

$$\mathbf{b}_{ij}^{\mathbf{m}} \stackrel{\text{def}}{=} f_{ij}(\mathbf{m}_i, \mathbf{m}_j) = \begin{pmatrix} m_{jx} - m_{ix} \\ m_{jy} - m_{iy} \end{pmatrix}, \quad (2)$$

$$\mathbf{J}_i^{\mathbf{m}} \stackrel{\text{def}}{=} \left. \frac{\partial f_{ij}}{\partial \mathbf{m}_i} \right|_{(\mathbf{m}_i, \mathbf{m}_j)}, \quad (3)$$

$$\mathbf{J}_j^{\mathbf{m}} \stackrel{\text{def}}{=} \left. \frac{\partial f_{ij}}{\partial \mathbf{m}_j} \right|_{(\mathbf{m}_i, \mathbf{m}_j)}, \quad (4)$$

$$\mathbf{P}_{ij}^{\mathbf{m}} \stackrel{\text{def}}{=} \mathbf{J}_i^{\mathbf{m}} \Sigma_{\mathbf{m}|ij} \mathbf{J}_i^{\mathbf{m}T} + \mathbf{J}_j^{\mathbf{m}} \Sigma_{\mathbf{m}|ij} \mathbf{J}_j^{\mathbf{m}T} + \mathbf{J}_i^{\mathbf{m}} \Sigma_{\mathbf{m}|ij} \mathbf{J}_j^{\mathbf{m}T} + \mathbf{J}_j^{\mathbf{m}} \Sigma_{\mathbf{m}|ij} \mathbf{J}_i^{\mathbf{m}T}. \quad (5)$$

We define similar notations for any two observations \mathbf{o}_i and \mathbf{o}_l , with covariance matrix $\Sigma_{\mathbf{o}|kl}$ defined in the same fashion as above.

The binary constraint between the matches $(\mathbf{o}_k, \mathbf{m}_i)$ and $(\mathbf{o}_l, \mathbf{m}_j)$ is then defined as

$$\text{binary}(\mathbf{o}_k, \mathbf{m}_i, \mathbf{o}_l, \mathbf{m}_j) \stackrel{\text{def}}{=} (\mathbf{b}_{kl}^{\mathbf{o}} - \mathbf{b}_{ij}^{\mathbf{m}})^T (\mathbf{P}_{kl}^{\mathbf{o}} + \mathbf{P}_{ij}^{\mathbf{m}})^{-1} (\mathbf{b}_{kl}^{\mathbf{o}} - \mathbf{b}_{ij}^{\mathbf{m}}) < \chi_{2, \alpha}^2 \quad (6)$$

where again $\alpha = 0.95$.

If more than one possible matches are found, then the matching score is defined as being the mean of $\text{unary}(\mathbf{m}, \mathbf{o})$ and $\text{binary}(\mathbf{o}_i, \mathbf{m}_j, \mathbf{o}_k, \mathbf{m}_l)$ values. The matching with the lowest score is chosen.

In order to optimize its execution time, the algorithm only takes into account landmarks which satisfy the following property: there is at least one observation such that the compatibility score between this observation and the landmarks is below the threshold set previously. Landmarks that are known to be at a large distance of the vehicle's estimated position are discarded beforehand, as they have negligible chance of being associated to an observation by the algorithm.

Once this data association is performed, $\ell \leq m$ observations have been matched to an existing landmark. For each such observation i , using the map's information regarding the corresponding landmark, we compute an estimation $(x_i^{\text{est}}, y_i^{\text{est}})$ of the vehicle's position. This leads to a 2ℓ -dimensional Gaussian distribution vector \mathbf{z} , composed of all x_i^{est} followed by all y_i^{est} .

2.2 Fusion of Gaussian Vectors

Once the data association has been carried out, each observation provides an estimate of the vehicle's current position, as well as the covariance between these estimates (which are map data). These estimates are then merged to determine the most likely vehicle position, in the form of a Gaussian variable.

In this part, we have 2ℓ measurements (ℓ for each coordinate), together with the corresponding covariance matrix Σ_ℓ , and $\mathbf{A}_\ell = \Sigma_\ell^{-1}$ the corresponding precision matrix (both being square matrices of size $2\ell \times 2\ell$). To perform the fusion, we maximize the probability density of this distribution, by differentiating the associated log-likelihood with respect to the unknown vehicle position \mathbf{v} . A unique maximum is then found.

Formally, we consider a vector \mathbf{z} of estimates of size 2ℓ , the first ℓ coordinates corresponding to estimates of the \mathbf{x} coordinate and the ℓ others to estimates of the \mathbf{y} coordinate. Let, $\mathbf{e} = (1, \dots, 1)^T$ a vector of size 2ℓ and \mathbf{E} of size $2 \times 4\ell$ defined as

$$\mathbf{E} \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{e} & \mathbf{0} \\ \mathbf{0} & \mathbf{e} \end{pmatrix}. \quad (7)$$

If the true value measured by the vehicle is \mathbf{v} , then the probability density of our measurements is now

$$p(\mathbf{z}|\mathbf{v}) = \frac{1}{Z} \exp\left(-\frac{1}{2} (\mathbf{E}\mathbf{v} - \mathbf{z})^T \mathbf{A}_\ell (\mathbf{E}\mathbf{v} - \mathbf{z})\right), \quad (8)$$

where Z is a constant.

We seek to maximize this value by differentiating the log-likelihood with respect to \mathbf{z} , which leads us to

$$\mathbf{E}^T \mathbf{A}_\ell \mathbf{E} \mathbf{v} - \mathbf{E}^T \mathbf{A}_\ell \mathbf{z} = 0, \quad (9)$$

which is easily solved to yield the following estimation for the position of the vehicle

$$\mathbf{v} = \mathbf{M}\mathbf{z}, \text{ with } \mathbf{M} = (\mathbf{E}^T \mathbf{A}_\ell \mathbf{E})^{-1} \mathbf{E}^T \mathbf{A}_\ell. \quad (10)$$

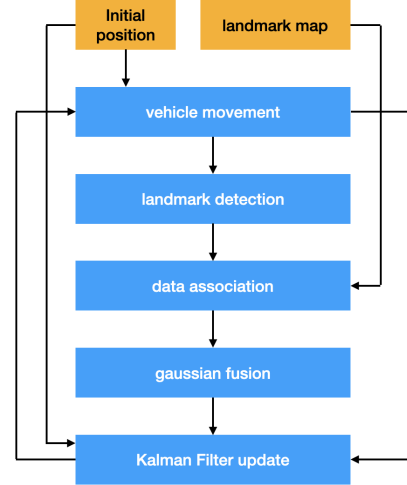


Figure 1: Summary of our approach. The initial position and the landmark maps are known to the vehicle at the beginning.

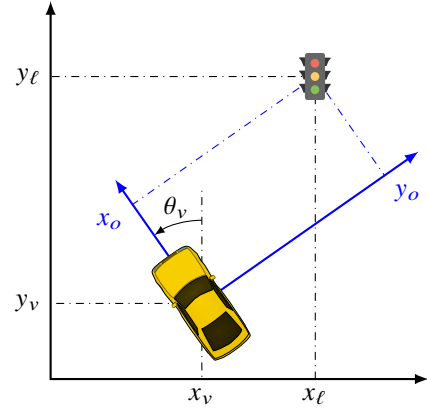


Figure 2: Coordinates of a landmark in global and local frame of reference.

The covariance matrix of \mathbf{v} is given by:

$$\Sigma_{\mathbf{v}} = \mathbf{M}^T \Sigma_\ell \mathbf{M} \quad (11)$$

2.3 Workflow

We assume that the vehicle has local access to a map with n landmarks, whose positions are known up to a given accuracy. Assuming that the two-dimensional points describing landmarks follow a multivariate Gaussian distribution, the map thus represents them as a $2n$ -dimensional Gaussian vector, the n first coordinates being the x -coordinates of the landmarks, and the n last coordinates being their y -coordinates. Let $\boldsymbol{\mu}$ of dimension $2n$ be its location, $\boldsymbol{\Sigma}$ its $2n \times 2n$ covariance matrix, and $\mathbf{A} = \boldsymbol{\Sigma}^{-1}$ its precision matrix.

Our approach is summarized in Figure 1. Initially, we assume that our knowledge of the vehicle's coordinates are accurate up to a centered Gaussian error ϵ_i . The algorithm is run once after each time interval Δt .

At each iteration, the vehicle is assumed to move on the map on a straight line, with a speed v and an angle θ . We apply centered Gaussian errors with standard deviations ϵ_v and ϵ_θ on these two values, and this leads to a new estimation of the vehicle’s position.

Then, the vehicle detects $m \leq n$ landmarks, initially not associated to any of the map. We assume though that the map contains all the landmarks that could possibly be detected, meaning that any observation necessarily corresponds to a known landmark. Each observation corresponds to the position of one of these landmarks in the vehicle’s frame of reference, as shown in Figure 2. We take into account the imperfection of the sensors used, assuming that they are correctly calibrated to avoid bias, so that the coordinates of each landmark l at time t are determined up to an error that is a centered Gaussian with standard deviation $\epsilon_{\text{lmk},l,t}$. These coordinates can then be converted into absolute coordinates, which can then be compared with the coordinates of the map landmarks.

We then perform data association between the map landmarks and the detected landmarks using the Geometric Compatibility Branch and Bound (GCBB) algorithm described in Section 2.1. Each matched observation leads to an estimation of the vehicle’s current position, under the form of a Gaussian vector, which give us $\ell \leq m$ such vectors. Right after this data association we adjust the vehicle’s estimated angle if this leads to a better score for this specific matching: we check values above the base estimate with a step of $\Delta\theta$ radians, until the compatibility score no longer decreases, then do the same for lower values.

These $\ell \leq m$ vectors are then used to perform a Gaussian fusion (Section 2.2), that leads to a single estimation for the vehicle’s position, with the corresponding covariance matrix. Finally, the new vehicle position estimation is obtained through a linear Kalman Filter (Kalman, 1960), which state space only contains the vehicle’s coordinates \mathbf{v} . The Gaussian fusion estimation, along with the previously estimated position after the vehicle moved, are used as inputs to predict and update the vehicle’s coordinates using results from previous iterations. Then the algorithm loops with the corrected vehicle position.

3 EXPERIMENTATION

3.1 Simulator

This model was tested using a simulator (see Figure 3), implemented in C++ using the RTMaps software (Nashashibi, 2000). The idea is to have a vehicle move on a local map with a constant speed. This

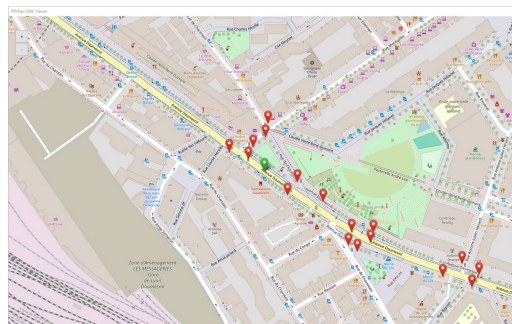


Figure 3: Illustration of our simulator. The vehicle (green marker) geopositions itself thanks to the landmarks (red markers).

Table 1: Numerical values used for the simulation.

Δ_t	40 ms	iteration duration
$\Delta\theta$	0.005 rad	angle adjustment step
ϵ_i	0.1 m	initial position error
ϵ_m	0.1 m	landmark position error (map)
ϵ_v	0.056 m/s	vehicle speed error
ϵ_θ	0.0044 rad	vehicle angle error

local map consists in a simplified version of an OpenStreetMap export of the 12th arrondissement of Paris. We retrieve the information regarding the streets and their nodes to create an oriented graph on which the vehicle may move. In addition, we provide a set of landmarks, which we assume detectable by our simulated vehicle. Such landmarks would correspond in reality to those detectable by vehicles with current technology, some of which were listed in Section 1. Given that these landmarks’ coordinates are *a priori* accurate, and in order to simulate the fact that the landmarks of the actual map will not be accurate, a centered Gaussian error with a standard deviation of ϵ_m is assigned to each coordinate. In this way, the vehicle will detect landmarks at positions that are close yet different from the ones that are stored in the map that the vehicle uses to position itself. Even though our method relies on the GCBB algorithm, the fact that the landmarks in the map are not correlated implies that the binary constraint of this algorithm is useless in this particular simulation setup. In Section 4 we will go back to this point in order to explain how we intend to build a map with covariance.

The simulator simultaneously determines two positions: the vehicle’s actual position (where it really is), and the vehicle’s perceived position (where it thinks it is). Each time the vehicle moves, the actual movement and the landmark detection are returned with additional error, using the numerical values in Table 1. For the sake of the experimentation, the vehicle moves at a constant speed of 30 km/h.

Table 2: Cumulative distribution of the geopositioning errors depending on the landmark density. For example, with an average of 1 landmark per 10.5m, about 81.2% of the position estimates have an error of less than 0.1m.

Error (m)	1 p. 10.5m	1 p. 14m	1 p. 21m
< .05	35.2%	35.5%	30.8%
< .1	81.2%	80.8%	75.4%
< .15	96.8%	96.6%	94.2%
< .2	99.6%	99.5%	98.6%
< .4	100%	100%	100%

In these experiments we also tried to simulate the fact that, in real life, certain landmarks are likely to be hidden (for example, by another vehicle, construction work, buildings. . .). To this end, at each iteration every landmark that is currently visible has a 0.001 probability to become hidden for a given number of iterations ranging between 1 (40 ms) and 1000 (40 s), following a uniform law.

For simplicity, we assume that only landmarks that are less than 50 m away from the vehicle are possibly visible, regardless of the location and shape of nearby buildings. Finally, for performance purposes, we limit the number of landmarks that can be detected at a given moment to 5: if there are more than 5 visible landmarks at some point then the 5 landmarks with the largest estimated distance to the vehicle are selected. Indeed, in practice, more distant landmarks yield a better vehicle angle estimation.

The hardware used in practice inevitably has an impact on the performance of this approach. For instance, the measured errors will necessarily depend on the type of sensor used to detect the landmarks, such as cameras, radars, or LiDARs. However, most of these technical details are modeled in this paper by the correlation matrix, in Gaussian form. For example, irrespective of the sensors used, it is assumed that the detection of a landmark by a vehicle is represented by the estimated coordinates of the landmark together with the corresponding error. Similarly, the issues surrounding vehicle-server communications are not studied. Hence, our method does not rely on a specific type of sensor and may be used without the need of taking specific technical constraints into account. As one of our goals is to provide an effective geopositioning without the use of expensive sensors, we have deliberately used mediocre values for the mean error estimations to make sure that the results that we obtain are still acceptable despite this additional constraint and degraded operational conditions.

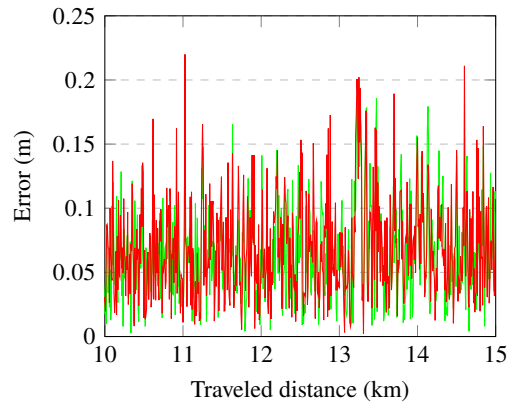


Figure 4: Evolution of the positioning error for the fusion estimation (green) and on the Kalman filter estimation (red) between kilometer 10 and kilometer 15 (restricted for clarity). Average landmark density is one every 21 m.

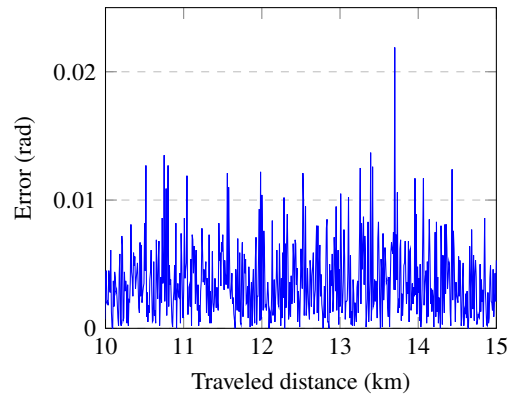


Figure 5: Evolution of the error on vehicle's angle estimation between kilometer 10 and kilometer 15 (restricted for clarity). Average landmark density is one every 21 m.

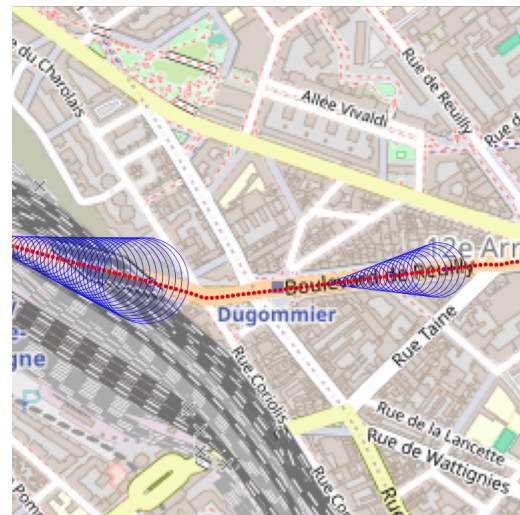


Figure 6: Evolution of the position estimation standard deviation (in blue) on a short portion of the simulation (not to scale, maximum standard deviation is slightly less than 15 cm in practice). The vehicle's trajectory is in red.

Table 3: Cumulative distribution of the angle errors depending on the landmark density.

Error (rad)	1 p. 10.5m	1 p. 14m	1 p. 21m
< .005	71.7%	71.9%	70.8%
< .01	96.9%	96.8%	96.1%
< .015	99.9%	99.9%	99.7%
< .05	100%	100%	100%

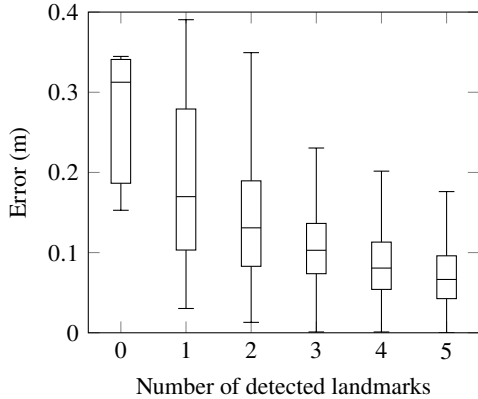


Figure 7: Boxplots (median/quartiles/1.5*IQR) of the geopositioning error of the Kalman filter’s estimation, conditionally on the number of detected landmarks. Average landmark density is one every 21 m.

3.2 Results

The numerical results presented here correspond to a 1 h simulated ride, using a map which contains landmarks that were generated for the sake of the experiment. In order to explore the impact of varying landmark densities in the urban environment, each of these experiments features a different number of known landmarks on the map. These landmarks are generated randomly, with an algorithm that ensures that they are sufficiently close to the roads on which the vehicle is likely to travel. The maps that have been produced for this experiment contain respectively 2000, 3000 and 4000 landmarks. If we assume that all these landmarks are supposed to be on the side of a road, we can translate this into the average number of meters of road per landmark: respectively one landmark per 10.5 m, 14 m and 21 m. Another way to describe it is that landmark densities are respectively 2185, 1639 and 1092 per square kilometer, based on the area exported from OpenStreetMap.

Table 2 and Table 3 respectively show the distribution of the vehicle’s position and angle errors values. Except for these two tables, the results reported in this section correspond to the less dense landmark map, with 2000 items.

Figure 4 and Figure 5 respectively show the evolution of the distance between the vehicle’s estimated

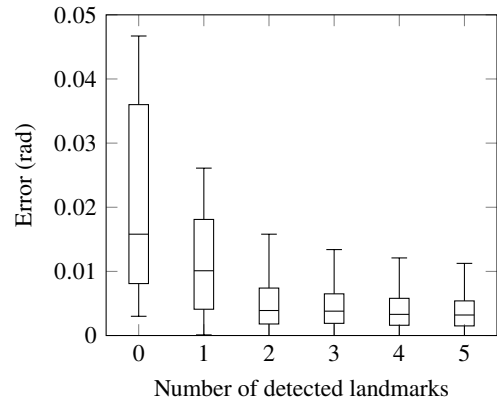


Figure 8: Boxplots (median/quartiles/1.5*IQR) of the angle error of the Kalman filter’s estimation, conditionally on the number of detected landmarks. Average landmark density is one every 21 m.

coordinates and the real coordinates and the absolute difference between the vehicle’s estimated headway and the real angle during the 1 h long runs.

Figure 6 shows the evolution of the standard deviation of the position estimation on a short portion of the simulation. We can see that the standard deviation increases gradually until a landmark is found, then stays very low until no landmark is found anymore.

Figure 7 shows the evolution of accuracy as a function of the number of landmarks the vehicle detects and matches to the map landmarks. Figure 8 describes the angle estimation error, again depending on the number of landmarks the vehicle detects and matches to the map landmarks.

3.3 Analysis

In terms of performance, each iteration of the algorithm runs in less than 40 ms, which means that our method can be used in real time. As expected, the data association part of the algorithm is the most CPU-consuming part, and taking into account more than 5 landmarks at once could cause the running time to grow drastically. The other CPU-intensive step in the algorithm is the initialization of the data association, which takes place in linear time in the number of landmarks in the map, which can increase significantly as we scale up the map size. Although this is not a primary focus of this paper, robust solutions exist to handle large urban environments, such as in (Alsayed et al., 2015), and can be easily integrated into our current work. Gaussian estimation fusion performs a number of matrix inversions and multiplications bounded by a constant, and the size of the matrices is the number of estimates resulting from the previous data association (which is less than 5), so this will

never be a performance issue.

Regarding the vehicle's position and angle estimation error: while the results using 3000 (1 p. 14 m) and 4000 (1 p. 10.5 m) landmarks were very similar, it seems that switching from 2000 (1 p. 21 m) to 3000 improves the results a little bit. Despite these differences, we can see that our goal of keeping the distance error below 10 cm is achieved at least 96% of the time.

While this error stays below 0.15 m most of the time, we notice that it occasionally spikes. A deeper analysis of the data reveals that this happens in a few specific areas of the map the vehicle drives in.

As expected, errors are likely to increase in areas where the number of detected landmarks is low. For example, when the vehicle detects only one landmark, the error on the vehicle's position is highly dependent on the error on the single detected landmark. This is likely to occur when the concentration of landmarks is too low: in practice, for example, traffic lights tend to be grouped around major junctions, while bus stops are more isolated from each other. Even when the vehicle is travelling through an area with a satisfactory number of landmarks, it may happen that some or all of these landmarks are temporarily hidden, so that position accuracy may deteriorate for a short time in such an area. In these experiments, the vehicle detected 3 or more landmarks at least 98% of the time.

In our experimental setting, decimeter accuracy is unlikely to be achieved consistently if there are not always enough landmarks visible at the same time. The question then arises as to how to make the most of the urban environment to keep the number of landmarks detectable by a moving vehicle to a maximum.

Regarding the impact of the number of landmarks observed at once by the vehicle, although the results for 0 landmark are not necessarily very representative, given the rarity of such a situation, we can nevertheless see that, as expected, the results improve with the number of matches with known landmarks from the map. In the specific case of the vehicle's angle's estimation, two landmarks seem to be already enough to significantly improve the estimation's accuracy.

Although this is rare, we have observed that the error can occasionally be significant even with a large number of landmarks observed and associated with the map. First of all, it should be noted that, as the number of detected landmarks increases, it generally takes a few iterations for the error to converge to a more satisfactory value, which explains why there can still be significant errors for 1 or more landmarks detected (this also explains why there can be very small errors for 0 or 1 landmarks detected, which is not a major issue since we are interested in the maximum error). Another explanation is that when the number of

landmarks observed is large, the density of landmarks in the area through which the vehicle is passing may be higher than average. This implies that some landmarks are very close to each other, leading to more frequent association errors than usual, and therefore to more position estimation errors.

We also investigated the question of whether the data association errors may have a significant impact on our method's accuracy. Generally speaking, matching errors involved landmarks that were generated very close to each other. Therefore, such wrong matches have a very minor impact on the vehicle's position estimation and alone do not compromise the goal of decimetric precision. Furthermore, some common practical methods to improve the data association, such as defining the landmarks' orientation, type (e.g. traffic sign vs. traffic light) or height have not been used in this simulator, which leave a lot of room for improvement in the event the accuracy of this data association algorithm becomes a critical issue.

Regarding the fact that we only consider at most 5 landmarks at once: in practice, it does not seem like increasing this threshold significantly improves the results when using the aforementioned numerical values. However, since the target is generally to have a decimeter or better precision, it may be necessary to increase that threshold when the map is too inaccurate. Future work could include the study of some of the hyperparameters of this simulation in order to understand more precisely how, for example, depending on the accuracy of the map, to find a compromise between the performance of the algorithm and the number of observations used to estimate the vehicle's position.

4 CONCLUSION

This paper shows how an imprecise landmark map can contribute in geopositioning an autonomous vehicle. The results obtained using our simulator show that, depending on the accuracy of the landmark map and the landmark density, the error in the vehicle's coordinates is small enough to replace a GNSS service in urban scenarios where such data is unreliable or unavailable.

We plan to test our method on data collected from a real vehicle in the near future: this will be facilitated by the use of the RTMaps software, which allows real data to be easily collected. One of the challenges will be to identify families of landmarks that are convenient to detect in an urban setting and dense enough for our needs.

For the purposes of this work, we have assumed

that we are in possession of a map based on real landmarks that can be detected in practice using existing technology, but the imprecision of the position of these landmarks has been artificially defined. In the longer term, our goal is to use the measurements made by individual vehicles as crowd-sourced data for improving the map. In such a system, the vehicles send all the data that they obtained during a day to a central server that will consolidate and process them in order to produce a new landmarks map that will be sent back to all vehicles. The goal is to improve on the work of (Stoven-Dubois, 2022) to handle city-scale maps. Eventually, this will lead to a methodology for the creation of precise landmark maps from coarse ones without resorting to expensive solutions like GNSS RTK.

REFERENCES

- Alsayed, Z., Bresson, G., Nashashibi, F., and Verroust-Blondet, A. (2015). PML-SLAM: a solution for localization in large-scale urban environments. In *PPNIV-IROS 2015*.
- Bresson, G., Alsayed, Z., Yu, L., and Glaser, S. (2017). Simultaneous localization and mapping: A survey of current trends in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 2(3):194–220.
- Breuel, T. M. (2003). Implementation techniques for geometric branch-and-bound matching methods. *Computer Vision and Image Understanding*, 90(3):258–294.
- Büchner, M., Zürn, J., Todoran, I.-G., Valada, A., and Burgard, W. (2023). Learning and aggregating lane graphs for urban automated driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13415–13424.
- Chalvatzaras, A., Pratikakis, I., and Amanatiadis, A. A. (2022). A survey on map-based localization techniques for autonomous vehicles. *IEEE Transactions on intelligent vehicles*, 8(2):1574–1596.
- Chen, S., Zhang, N., and Sun, H. (2020). Collaborative localization based on traffic landmarks for autonomous driving. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.
- Dissanayake, G., Williams, S. B., Durrant-Whyte, H., and Bailey, T. (2002). Map management for efficient simultaneous localization and mapping (SLAM). *Autonomous Robots*, 12:267–286.
- Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: part I. *IEEE robotics & automation magazine*, 13(2):99–110.
- Elghazaly, G., Frank, R., Harvey, S., and Safko, S. (2023). High-definition maps: Comprehensive survey, challenges and future perspectives. *IEEE Open Journal of Intelligent Transportation Systems*.
- He, L., Jiang, S., Liang, X., Wang, N., and Song, S. (2022). Diff-net: Image feature difference based high-definition map change detection for autonomous driving. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2635–2641. IEEE.
- Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45.
- Kumar, D. and Muhammad, N. (2023). A survey on localization for autonomous vehicles. *IEEE Access*.
- Li, L., Wang, R., and Zhang, X. (2021). A tutorial review on point cloud registrations: principle, classification, comparison, and technology challenges. *Mathematical Problems in Engineering*, 2021(1):9953910.
- Nashashibi, F. (2000). *RT maps*: a framework for prototyping automotive multi-sensor applications. In *IEEE Proceedings of the IEEE Intelligent Vehicles Symposium 2000*, pages 99–103. IEEE.
- Neira, J., Tardós, J. D., and Castellanos, J. A. (2003). Linear time vehicle relocation in SLAM. In *ICRA*, pages 427–433.
- Qu, X., Soheilian, B., and Paparoditis, N. (2018). Landmark based localization in urban environment. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:90–103.
- Ruan, J., Li, B., Wang, Y., and Sun, Y. (2023). Slamesh: Real-time lidar simultaneous localization and meshing. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3546–3552. IEEE.
- Stoven-Dubois, A. (2022). *Robust Crowdsourced Mapping for Landmarks-based Vehicle Localization*. PhD thesis, Université Clermont Auvergne.
- Sun, S., Jelfs, B., Ghorbani, K., Matthews, G., and Gilliam, C. (2022). Landmark management in the application of radar SLAM. In *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 903–910. IEEE.
- Weishaupt, F., Tilly, J. F., Appenrodt, N., Fischer, P., Dickmann, J., and Heberling, D. (2024). Landmark-based vehicle self-localization using automotive polarimetric radars. *IEEE Transactions on Intelligent Transportation Systems*.
- Zhang, Y., Severinsen, O. A., Leonard, J. J., Carlone, L., and Khosoussi, K. (2023). Data-association-free landmark-based SLAM. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8349–8355. IEEE.
- Zürn, J., Vertens, J., and Burgard, W. (2021). Lane graph estimation for scene understanding in urban driving. *IEEE Robotics and Automation Letters*, 6(4):8615–8622.