Robust Map-Augmented Localisation Using Particle Filters

COMP2550 Project Report

Sam Toyer*
u5568237@anu.edu.au

June 15, 2015

Abstract

In this paper, we propose a particle filtering method which fuses GPS, odometry, and the information in a digital street map to localise a road vehicle with superior accuracy to methods based on GPS and odometry alone. We demonstrate that our algorithm is robust to noise and GPS dropouts, and give a variant of the algorithm which dispenses with the requirement for odometry.

1 Introduction

Localisation is a critical task for intelligent transportation systems, and features in a variety of applications ranging from personal navigation devices (White et al., 2000) to self-driving cars (Levinson et al., 2011). Accurate localisation data can also be used to augment vehicle-based systems not directly related to navigation, like visual road detection (Álvarez et al., 2014). Precise information about the location of the vehicle is clearly of the utmost importance in such systems.

This task is complicated by the noise present in sensor readings. Despite its usefulness, GPS is a major offender in this regard: GPS accuracy can range from around 2m in an open area to up to 15m in heavily built-up areas (Modsching et al., 2006). Dead reckoning also suffers from significant errors, although unlike GPS error these tend to gradually increase over time. For instance, Vlcek et al. (1993) report that low cost dead reckoning systems—typically based on odometry and gyroscope data—yield localisation errors of between 2% and 5% of the distance travelled by the vehicle.

Fortunately, road vehicles in motion are almost always situated on public roads, so it is possible to improve localisation accuracy by taking into account the information present in a street map. With the advent of the OpenStreetMap project[1]—which provides, free of charge, detailed road network maps covering most of the globe—use of map information in this way has become a cheap and convenient method of augmenting the accuracy of existing localisation systems.

The most common method of incorporating this information is through "map matching", where localisation is performed under the simplifying assumption that the true location of the vehicle will *always* coincide with a roadway, footpath or other mapped feature. In this paper, we will not make such a highly restrictive assumption. Instead, we will merely assume that the vehicle is *most likely* near a roadway, and attempt to use this assumption to improve localisation accuracy as greatly as we can without losing the ability to localise vehicles which stray from the road network.

2 Related work

The approaches which presently dominate the map assisted localisation literature are map matching methods which use sets of hand-crafted rules to determine which road segment in a street map a vehicle is most likely to be travelling on, then attempt to localise the vehicle within that road segment. Typically, these approaches work by first assigning a "score" to each road segment in a map based on the segment's distance from the vehicle's estimated position, the segment's heading, and so on. After this, the algorithm will take the highest scoring segment, and localise to the point on that segment which is nearest to the last GPS fix or Kalman filter state estimate.

These heuristic methods can achieve very high accuracy. For example, Quddus et al. (2006) reported that the road segment scoring portion of their algorithm was able to correctly identify which road segment their test vehicle was travelling on with 99.2% accuracy on a 4,605 segment map. However, such high accuracy is not representative of all heuristic map matching algorithms—Quddus et al. reported previous algorithms achieving 70%-98% segment identification accuracy on the same data set for which theirs achieved 99.2%—and usually requires large sets of hand-tuned rules. Moving between urban, suburban and rural environments can sometimes reduce the usefulness of these rules, necessitating the production of yet more rules and heuristics for different environments (Velaga et al., 2012). Furthermore, heuristic map-matching algorithms are highly sensitive to initial conditions, as producing an incorrect first match will likely cause subsequent matches to be incorrect. This is due to the fact that heuristic algorithms often assign low scores to road segments far away from previously matched segments (Syed and Cannon, 2004).

*Supervisor: Dr. José M. Álvarez

[1] http://www.openstreetmap.org/

Probabilistic approaches to map aided localisation manage to avoid some of these pitfalls. Rather than producing a single "best guess" of the vehicle's state at each time step, probabilistic methods internally maintain a probability distribution over all possible vehicle states. This distribution may be updated sequentially and used to produce the expectation of the vehicle's position at each time step. This approach makes it straightforward to incorporate almost any kind of sensor data: for instance, Brubaker et al. (2013) used a probabilistic approach based on Gaussian mixture models to localise a vehicle using only street map information and visual odometry—a feat which would be impossible with traditional heuristic map matching approaches, which rely on GPS. Probabilistic approaches also make it possible to relax the map matching assumption that the vehicle is always on a mapped feature, as it is straightforward to expand a distribution over possible vehicle states to cover both on-road and off-road positions.

Particle filtering—also known as Monte Carlo localisation—is a popular probabilistic localisation method, and has been applied to map aided localisation in a number of different ways. Chausse et al. (2005) applied a particle filter to fuse odometry, GPS, vision and the information in a street map, whilst Selloum et al. (2009) and Toledo-Moreo et al. (2009) used particle filtering with odometry, GPS and map information alone. In each instance, the authors reported excellent results, with all algorithms providing lane-level positioning accuracy most of the time.

Unfortunately, the aforementioned approaches to probabilistic map aided localisation require sensors and types of maps which can be difficult to acquire. All three mentioned algorithms required extremely precise maps offering a level of detail far beyond what could be expected of an ordinary street map like those provided by OpenStreetMap. Furthermore, Selloum et al. and Toledo-Moreo et al. both assumed the availability of a service like the European Geostationary Navigation Overlay Service (EGNOS) to correct GPS errors, whilst Chausse et al. required that the vehicle be fitted with a camera in order for their algorithm to achieve lane-level accuracy. These constraints are all difficult to satisfy in low-cost systems and systems which operate in remote regions. Thus, our goal in this paper will be to produce an algorithm which retains the advantages of probabilistic map matching whilst dispensing with the onerous requirements of previous approaches.

3 Our approach

3.1 Particle filtering

As mentioned previously, particle filtering is a probabilistic localisation method that is well suited to map aided localisation. Internally, a particle filter maintains a fixed-size set of "particles", each of which represents a single sample from the following distribution (Fox et al., 1999):

$$p(s_t \mid o_t, a_{t-1}, o_{t-1}, a_{t-2}, \ldots, o_0)$$

Where:

s_t is the vehicle state at time t. This generally consists of a latitude, longitude and heading, but may contain more information in some cases.

o_t represents the set of observations made by the vehicle's sensors at time t.

a_t is the action taken by the vehicle at time t. Whilst these actions *can* correspond to throttle or steering wheel positions, it is also possible to define actions to be sensor measurements—for instance, the distance travelled by the wheels of a vehicle in a single time step. Although these sensory observations are not "actions" in the strictest sense of the word, we will soon see that it is convenient to treat them separately to the other observations represented by o_t.

Over time, a particle filter updates its internal particle set using incoming observations and actions so that the particles remain distributed according to $p(s_t \mid o_t, a_{t-1}, \ldots, o_0)$. This process begins at time $t = 0$ by distributing the particles $\left\{s_0^{(n)}\right\}_{n=1}^{N}$ according to some chosen prior distribution over vehicle states. In our case, the positions of the initial particle set are samples from a Gaussian centered on the first available GPS fix, with particle yaws distributed uniformly.

At each subsequent time step t, the vehicle takes an action a_t to yield a new state s_{t+1}. After this, we may receive a subsequent sensory observation o_{t+1}, which in our case will be a GPS fix. Using this new information, we may produce a new particle set $\left\{s_{t+1}^{(n)}\right\}_{n=1}^{N}$ distributed according to $p(s_{t+1} \mid o_{t+1}, a_t, \ldots, o_0)$ from our previous particle set using the following process:

1. **Predict:** A new set of particles $\left\{s_{t+1}^{(n)\prime}\right\}_{n=1}^{N}$ is sampled from the transition distribution, $p(s_{t+1} \mid s_t, a_t)$, using the last observed action a_t and the set of particles from the previous time step:
$$s_{t+1}^{(n)\prime} \sim p(s_{t+1} \mid s_t^{(n)}, a_t)$$

2. **Update:** Each particle $s_{t+1}^{(n)\prime}$ is then assigned a weight $w_{t+1}^{(n)}$ reflecting the likelihood of the most recent observation o_{t+1} in state $s_{t+1}^{(n)\prime}$:
$$w_{t+1}^{(n)} = \tilde{p}(o_{t+1} \mid s_{t+1}^{(n)\prime}) \tag{1}$$
Where $\tilde{p}(o_{t+1} \mid s_{t+1}^{(n)\prime})$ is the (unnormalised) likelihood of the observation o_{t+1} being made in state $s_{t+1}^{(n)\prime}$.

3. **Resample:** Finally, a new set of particles $\left\{s_{t+1}^{(n)}\right\}_{n=1}^{N}$ is produced by drawing N particles, with replacement, from $\left\{s_{t+1}^{(n)\prime}\right\}_{n=1}^{N}$ using the weights $\left\{w_{t+1}^{(n)}\right\}_{n=1}^{N}$. In some cases, this resampling step is omitted, and the weight set is instead carried over to the next iteration, in which case the weights for the next set of particles will be calculated using the following equation rather than Equation 1:

$$w_{t+1}^{(n)} = w_t^{(n)} \tilde{p}(o_{t+1} \mid s_{t+1}^{(n)\prime})$$

Some authors choose to perform the resampling step only once the effective number of particles, N_{eff}, falls below some threshold (Gustafsson et al., 2002), where N_{eff} is defined as:

$$N_{\text{eff}} = \left(\sum_{n=1}^{N} {w_t^{(n)}}^2\right)^{-1}$$

We have opted to set the threshold for N_{eff} at two thirds of the size of the particle set.

Finally, the expected vehicle position at time $t+1$ can be expressed as:

$$\mathbb{E}[s_{t+1}] \approx \begin{cases} \frac{1}{N}\sum_{n=1}^{N} s_{t+1}^{(n)} & \text{if resampled} \\ \sum_{n=1}^{N} w_{t+1}^{(n)} s_{t+1}^{(n)} & \text{otherwise} \end{cases}$$

In statistics, this process is known as Sequential Importance Resampling (SIR), and is explained at length by Smith and Gelfand (1992). One important requirement of SIR is that states obey the Markov property; that is, the state s_{t+1} is dependent only on the state s_t, the action a_t and the observation o_{t+1} (Dellaert et al., 1999). In Section 3.4, we will investigate a case in which this requirement is violated.

3.2 Basic particle filter configuration

In our implementation, we have chosen the state representation at time t, denoted $s_t = (x_t, y_t, \theta_t)^T$, to contain an easting x_t, a northing y_t, and a yaw θ_t. The observations $o_t = (f_{xt}, f_{yt})^T$ consist of an easting f_{xt} and northing f_{yt} for the most recent GPS fix, whilst the actions $a_t = (v_t, \omega_t)^T$ contain a forward speed v_t and a change in heading ω_t, as might be reported by a speedometer and gyroscope, respectively.

The transition distribution $p(s_{t+1} \mid s_t, a_t)$ chosen for our application updates particles utilising the vehicle's forward speed v_t and angular velocity ω_t at time t. Concretely, each particle particle $s_t^{(n)} = (x_t^{(n)}, y_t^{(n)}, \theta_t^{(n)})^T$ is updated using the following Euler integration:

$$\begin{aligned} \theta_{t+1} &= \theta_t + \omega_t' \Delta t \\ x_{t+1} &= x_t + v_t' \cos(\theta_t) \Delta t \\ y_{t+1} &= y_t + v_t' \sin(\theta_t) \Delta t \end{aligned}$$

Where $\omega_t' \sim \mathcal{N}(\omega_t, \sigma_\omega^2)$ and $v_t' \sim \mathcal{N}(v_t, \beta v_t^2)$ are random variables drawn from distributions representing our uncertainty about the true speed and angular velocity of the vehicle (Thrun et al., 2001). We found that the parameters $\sigma_\omega^2 = 0.15$ and $\beta = 0.36$ worked well for this purpose.

When odometry is unavailable, it is no longer possible to directly use the forward speed v_t and angular velocity ω_t of the vehicle to update particles using the aforementioned transition distribution. In Section 3.4, we will discuss the changes which must be made to the state representation and transition distribution in order to support localisation in this more challenging case.

Having defined a transition model and representations for states, actions and observations, all that remains is to define an observation likelihood model for use in the particle filter's update step. If a GPS fix at easting f_{xt} and northing f_{yt} was observed between time steps t and $t+1$, we set the weight $w_{t+1}^{(n)}$ associated with the particle $s_{t+1}^{(n)} = (x_{t+1}, y_{t+1}, \theta_{t+1})^T$ to:

$$w_{t+1}^{(n)} = \exp\left(-\frac{1}{2}\left(\begin{bmatrix} x_{t+1} \\ y_{t+1} \end{bmatrix} - \begin{bmatrix} f_{xt} \\ f_{yt} \end{bmatrix}\right)^T \Sigma^{-1} \left(\begin{bmatrix} x_{t+1} \\ y_{t+1} \end{bmatrix} - \begin{bmatrix} f_{xt} \\ f_{yt} \end{bmatrix}\right)\right) \tag{2}$$

This corresponds to an unnormalised Gaussian distribution with mean $(f_{xt}, f_{yt})^T$ and precision Σ^{-1}. We have found that a precision of the form $\sigma^{-2} I$, where σ^{-2} is derived empirically from GPS sensor data, yields good results.

In reality, GPS likelihood is not uniformly distributed, as GPS errors instead follow a complicated distribution governed by atmospheric interference, multipath fading, and other factors (Bajaj et al., 2002). However, other authors have found similar Gaussian-based modelling strategies to be effective (El Najjar and Bonnifait, 2005; Selloum et al., 2009; Toledo-Moreo et al., 2009), so we have adopted one here.

If GPS data is temporarily unavailable, we simply initialise weights uniformly, rather than applying Equation 2. Intuitively, this will cause the transition distribution to spread particles outwards until GPS fixes become available again to decrease the weight of particles far away from the vehicle's true position. This behaviour is examined as part of the qualitative comparison performed in Section 4.

Every time a GPS fix becomes available, we also choose to scatter a small proportion (approximately 1%) of particles in a normal distribution about the GPS fix. This ensures that the vehicle is able to re-localise itself correctly in the event that the particle filter's internal state estimate diverges too far from the vehicle's true position. Although there is no direct analogue to this strategy in the traditional SIR framework, Fox et al. (1999) argue that

it is theoretically sound in cases where the scattered particles are subsequently weighted correctly according to $p(o_t \mid s_t)$ and are sampled from a distribution which is nonzero over all locations where $p(s_t \mid o_t, a_{t-1}, \ldots, o_0)$ is nonzero.

3.3 Incorporating map data

Whilst other authors have attempted to incorporate street map data into the particle filtering process by extending each particle's state to include an associated road segment (Selloum et al., 2009; Toledo-Moreo et al., 2009), we have found that the fastest way of incorporating map data is by way of a pseudo-likelihood re-weighting during the particle filter's update step. Concretely, for each particle $s_t^{(n)}$, we calculate the distance $d_t^{(n)}$ to the nearest road segment in the stored street map, then update the weight $w_t^{(n)}$ of the stored particle using:

$$w_t^{(n)} \leftarrow \frac{w_t^{(n)}}{\left(1 + {d_t^{(n)}}^2\right)^{1.1}} \tag{3}$$

Whilst this pseudo-likelihood is not reflective of the true likelihood $p(d_t^{(n)} \mid s_t^{(n)})$, we have nonetheless found that it does a reasonable job of forcing particles to remain near the road without preventing the particles from "drifting" away from the map when the vehicle's true position is far from a mapped feature. This ensures that the vehicle is still able to localise itself when it is travelling on unmapped roads.

To ensure that the algorithm can still localise a vehicle in the absence of map information, our implementation does not perform map weight updates whenever more than 95% of the particles are at least 15m from the nearest roadway. This also allows the algorithm to perform localisation when the vehicle briefly travels over an unmapped feature, as demonstrated in Section 4.2.

Despite being an efficient option amongst different methods of incorporating map data, computing the distance between each particle and its nearest road segment is still by far the most expensive step in the particle filtering process, so it is imperative that this computation be fast if real-time performance is desired. We have found that the k-d tree implementation provided by Alliez et al. (2015) yields good performance for this task, taking around 50ms to perform the calculations for 2000 particles on a map of 3220 segments using a 2.5GHz Intel Core i5 processor.

3.4 Unavailability of odometry

If odometry is unavailable, the transition model proposed previously will cease to work, so we must produce a new transition model and associated state representation which does not depend on odometry.

One alternative model would be to adopt a stationary transition distribution for particle positions, like a Gaussian centered on each particle. Such a transition model would be efficient and simple, but would grossly violate the assumption that vehicle states form a Markov chain, since clearly a vehicle which moves by a certain amount and in a certain direction at one time step is likely to move by a similar amount and in a similar direction at the next. Concretely, violating this assumption means that particles either move too little or too much at each time step, depending on the vehicle's velocity.

In theory, the Markov state transition assumption can only be satisfied without odometry by expanding the internal state representation to contain all relevant physical properties of the vehicle, including position, velocity, acceleration, heading, angular velocity, angular acceleration, and so on. As a compromise, we have chosen to expand the state representation to contain only velocity and position. This does not truly satisfy the Markov state transition assumption, since the changes in a vehicle's velocity are heavily correlated over time, but we have nonetheless found that it produces acceptable results.

Operating without odometry requires that we replace our previous state vector $s = (x, y, \theta)^T$ with a new state vector $s = (x, y, v_x, v_y)^T$, where v_x and v_y are easting and northing velocities, respectively. Given a state s_t, our transition model becomes:

$$\begin{aligned} x_{t+1} &= x_t + v_{xt}\Delta t \\ y_{t+1} &= y_t + v_{yt}\Delta t \\ v_{xt+1} &= v_{xt} + \epsilon_x \\ v_{yt+1} &= v_{yt} + \epsilon_y \end{aligned}$$

Where $(\epsilon_x, \epsilon_y) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ and Δt is the time elapsed between time steps t and $t+1$. We have found that $\sigma^2 = 0.01$ works well, although in general it should be possible to fit an appropriate Gaussian from recorded data.

4 Results

In order to evaluate the performance of the map-based localisation algorithm proposed in Section 3, we utilised both quantitative and qualitative assessment. In the quantitative tests, we attempted to analyse the performance of the algorithm on partially artificial data for which it was straightforward to measure divergence from the ground truth. On the other hand, the qualitative tests attempted to examine algorithm performance using more realistic consumer-grade localisation data, which unfortunately did not include a ground truth for use in numerical comparisons.

The code and data used to produce all figures and statistics in this paper is available online.[2]

[2] https://github.com/qxcv/comp2550/tree/master/project

4.1 Quantitative tests

The quantitative tests utilised the data set presented by Brubaker et al. (2013), which includes a number of GPS traces along with OpenStreetMap data for the regions immediately surrounding the traces. The supplied dataset included exceptionally accurate RTK-GPS positioning data, but no samples from a consumer-grade GPS receiver. As a result, we had to use a synthetic noise model to benchmark the algorithm. The noise model included the following "roughening" steps:

1. Downsampling of GPS-like positioning information from 10Hz to 1Hz.
2. Addition of Gaussian-distributed white noise with zero mean and covariance $8^2\mathbf{I}$ to the positioning information.
3. Rescaling of reported speed by a small but random factor of at most 1% in order to emulate a subtly miscalibrated odometer.
4. Addition of Gaussian noise with zero mean and standard deviation 4×10^{-4} deg/s to the angular velocity data.

These parameters do not closely reflect the characteristics of real sensor noise. In particular, as mentioned previously, real GPS error tends to be highly time-correlated, but with an unpredictable distribution. Nonetheless, this "roughening" process does an excellent job of illustrating the difference in performance between plain particle filtering and map aided particle filtering in the presence of non-ideal sensory input.

The results of this comparison are shown in Figure 1. Clearly, map aided particle filtering offers a significant advantage over plain particle filtering, with the horizontal positioning error of the map aided method falling to as low as half the horizontal positioning error of the non-map-aided method.

The primary effect of the use of map information is to constrain the movement of particles in a direction perpendicular to the nearest road. Figure 2 illustrates this phenomenon. Whilst the map prior does nothing to constrain the spread of the particles in the direction of the road, it does constrain the spread of the particles perpendicular to it. It is anticipated that this property could be especially useful in urban canyon environments, where GPS satellites on either side of a vehicle are typically blocked out by buildings. This has the effect of increasing positioning error perpendicular to the road without affecting positioning error in the direction of the road, so a weighting step which constrains particles to positions near the road should yield a significant increase in particle filter performance (Modsching et al., 2006).

As Table 4.1 attests, our implementation is able to run comfortably in real time on a desktop processor. Whilst use of map information incurs a significant performance penalty, it is still possible to produce predictions using thousands of particles at 10Hz. The current implementation, written in Python, is a proof-of-concept with few optimisations, so it is anticipated that even greater performance could be achieved by moving to a lower level language. Many of the sub-tasks performed by the algorithm—updating particle weights, predicting particle positions, etc.—are trivial to parallelise, so moving to a multi-threaded architecture could also boost performance.

4.2 Qualitative tests

The qualitative assessment used a GPS trace provided by J. Álvarez (personal communication, May 12, 2015), which was produced from several kilometers of urban driving, including a brief GPS outage as the vehicle travelled through a tunnel. The results of this test are presented in Figure 4. After the final GPS fix, the map aided particle filter ensures that particles remain distributed along the lane on which the vehicle was travelling. Even after merging with a second lane, the particles have continued to travel forwards along the correct road, as expected. The non-map-aided filter, on the other hand, continues to provide a slightly less accurate state estimate, but it can be observed that the variance of its particle set is growing significantly over time. Without a GPS weight for resampling, the spread of these particles will not be constrained, so the particle filter will rapidly lose accuracy until the vehicle regains GPS reception.

The results of a second qualitative comparison—this time using "roughened" synthetic data from Brubaker et al.—are presented in Figure 5. The aim of this comparison was to demonstrate the behaviour of the filter when the vehicle is not on the road network. Immediately after leaving the road network ($t = 7.5$), we see that the accuracy of the state estimate drops significantly, as the map weight decimates particles which are far from the last road on which the vehicle was travelling. During the next several seconds, a number of GPS fixes are received and some particles are scattered around each fix. By $t = 12.5$, more than 95% of the particles are at least 15m from the nearest road, so map weights are no longer incorporated in the particle filtering process. This increases accuracy until $t = 17.5$, at which point map weights are incorporated again and the filter begins to re-localise the vehicle as it had done before the vehicle veered off the previous road.

This experiment demonstrates that a map aided particle filter is a poor choice of localisation algorithm for applications in which the vehicle is likely leave the mapped road network frequently.

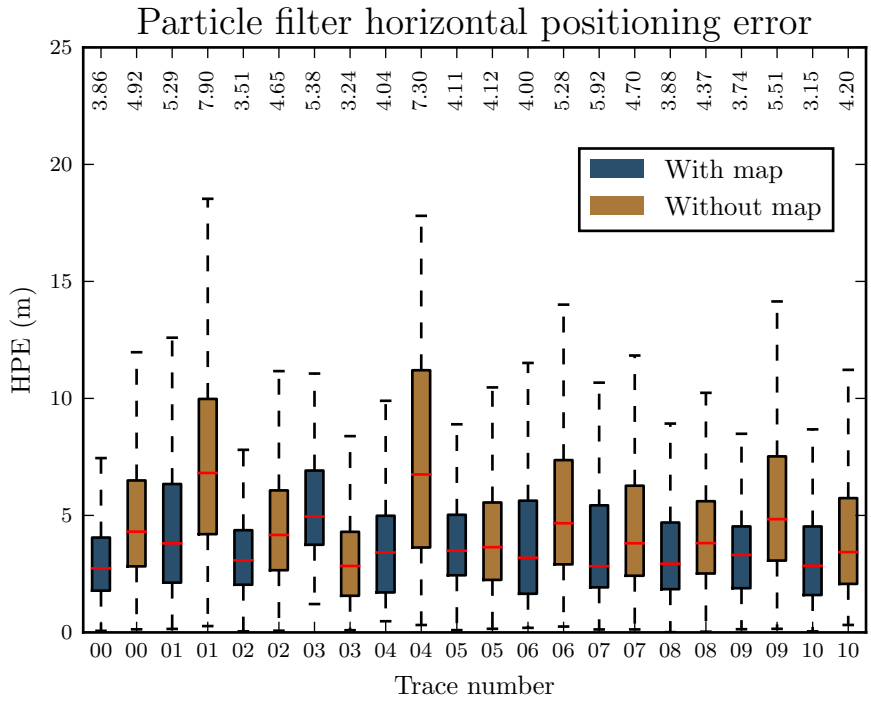

Figure 1: Distribution of Horizontal Positioning Error (HPE) when using particle filtering with and without map information. Numbers at the top of each column indicate mean HPE. HPE was measured at 10Hz on each of the 11 vehicle localisation traces provided by Brubaker et al. (2013). 2000 particles were used for each filter.

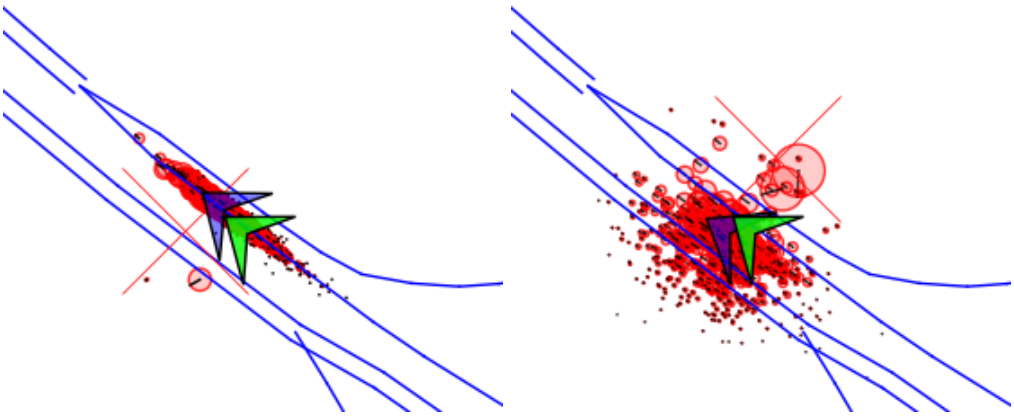

Figure 2: Images showing the particle spread induced by a map aided particle filter (left) versus that induced by a non-map aided particle filter (right). In each diagram, the road network is represented by the blue lines, the particle filter estimate is represented by a blue quadrilateral, the ground truth is represented by a green quadrilateral, and the particles associated with the particle filter are drawn as red circles, with area directly proportional to weight. The red cross indicates the location of the most recent pseudo-GPS measurement. For scale, both the quadrilateral representing the ground truth and the quadrilateral representing the state estimate are 15m wide.

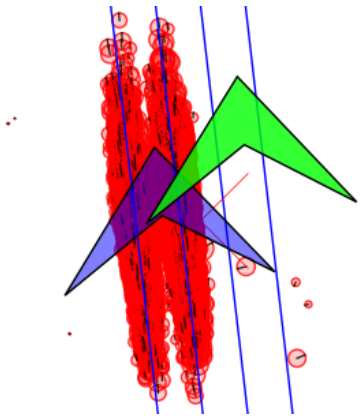

Figure 3: In this scenario, the map weighting has made particles concentrate on two lanes in which the vehicle is not travelling. In particular, the particle filter has produced a state estimate coinciding with lanes going in the opposite direction to the vehicle's true direction of travel.

Particles	500	1000	1500	2000
Time w/ map (s)	380.79	736.66	1117.00	1470.08
Time w/o map (s)	15.47	19.00	23.16	26.32
HPE w/ map ($\mu \pm \sigma$m)	4.86 ± 5.08	4.56 ± 4.58	4.06 ± 4.12	3.93 ± 3.68
HPE w/o map ($\mu \pm \sigma$m)	4.72 ± 3.19	4.65 ± 3.01	5.09 ± 3.18	4.72 ± 3.17

Table 1: Sum of processing times and averaged HPE for the filter over all traces in the Brubaker et al. data set using different numbers of particles. In each case, the filter produced position estimates at 10Hz for a total of 2320s (39 minutes) of traces. Statistics produced using a single core of a 2.8GHz AMD Phenom II processor.

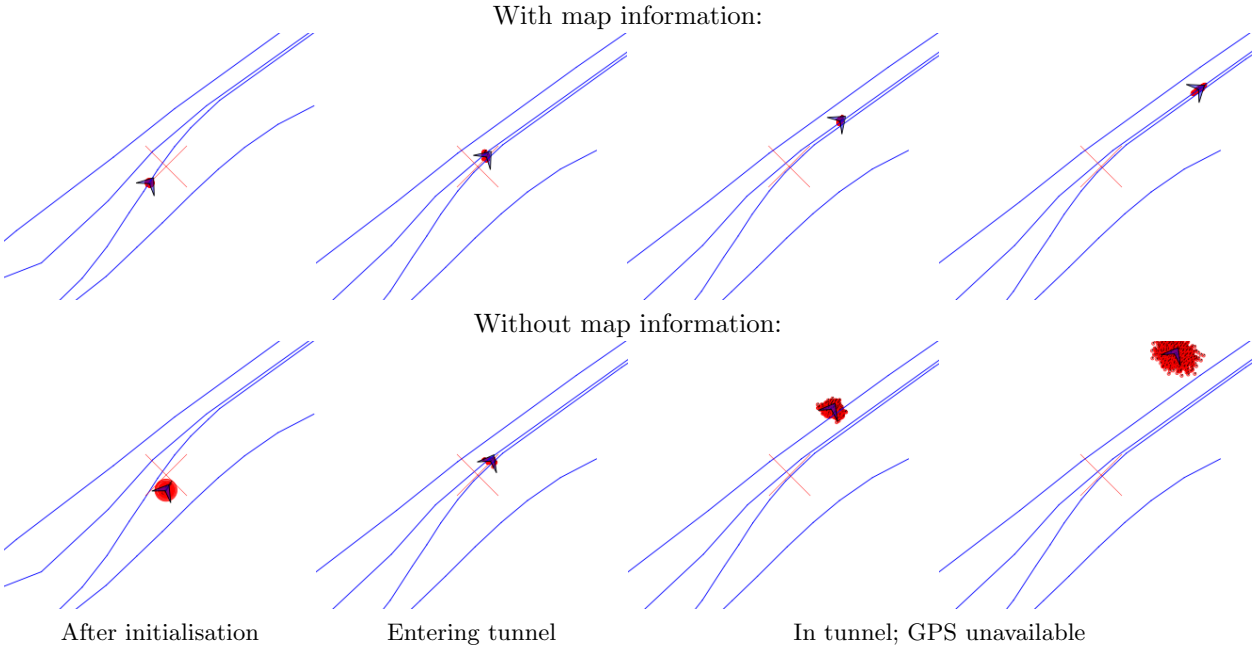

Figure 4: Qualitative comparison of the map aided particle filter (top) with an equivalent particle filter not making use of map information (bottom). Time flows from left to right. Graphical conventions are the same as those documented in Figure 2.

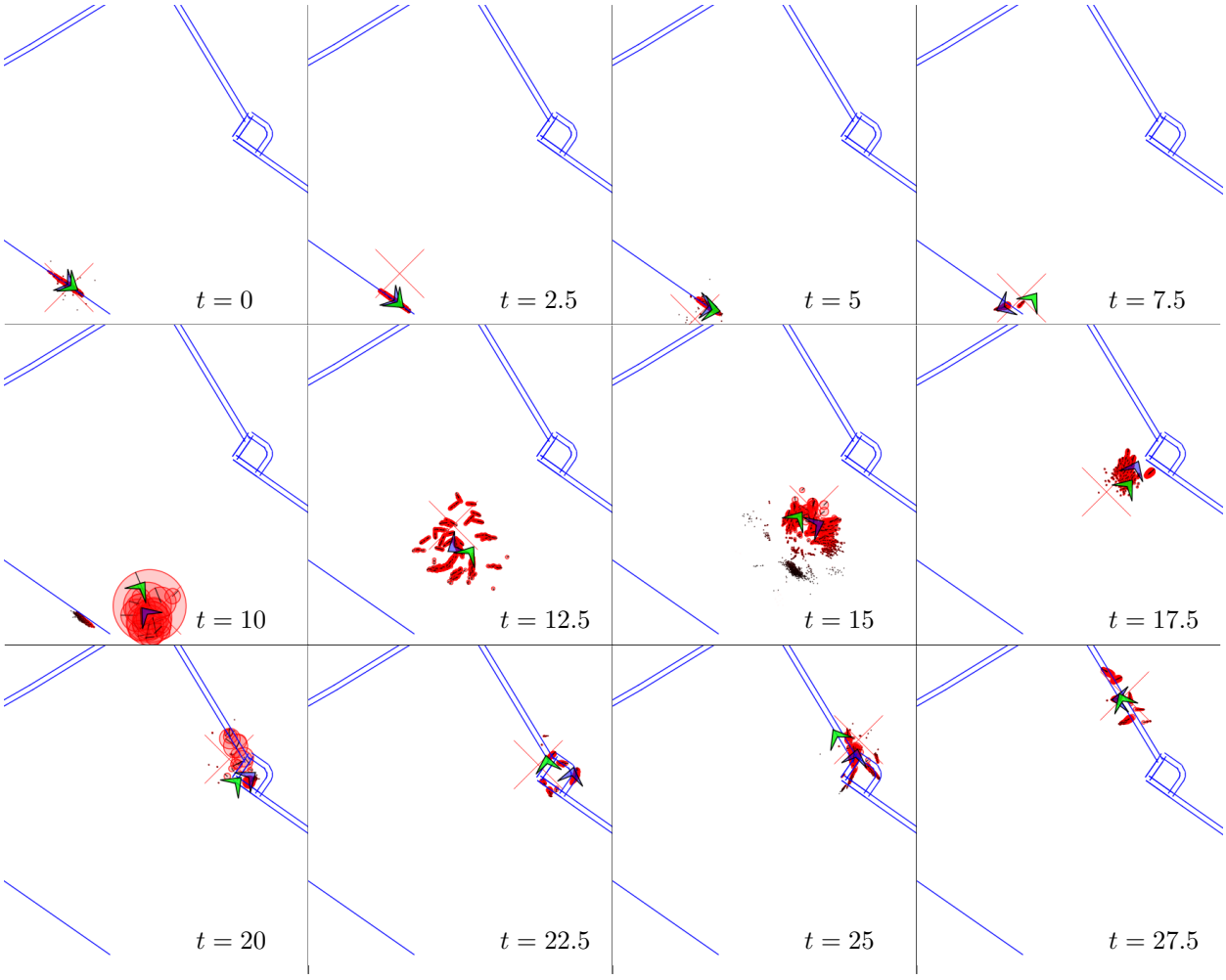

Figure 5: Evolution of the map aided particle filter state estimate (blue) and particle set (red) as the ground truth vehicle position (green) traverses a section of the map with a missing road. Times given in seconds since first frame. As before, graphical conventions are identical to those of Figure 2.

4.3 Shortcomings and possible improvements

Unfortunately, the algorithm's tendency to draw particles in towards the road can, in some cases, decrease localisation performance. An example of this is given in Figure 3: the particle filter has estimated that the vehicle is in one of the two left-hand lanes of the four lane road, but the vehicle is actually travelling in one of the right-hand lanes. Fixing this defect is challenging, since GPS fixes and motion information alone do little to reliably disambiguate between individual lanes, especially when the mapped lane locations may be inaccurate—as is almost certainly the case with some OpenStreetMap data. Other authors have avoided this problem by using more accurate maps and augmented GPS (Selloum et al., 2009; Toledo-Moreo et al., 2009). Another common approach is to first use a coarse localisation method to determine which road the vehicle is on, then employ computer vision algorithms to detect lane markings and/or curb locations in order to localise the vehicle *within* the road (Chausse et al., 2005; Montemerlo et al., 2008). If sub-lane accuracy were required, then the current algorithm would have to be extended using one of these approaches.

Figure 3 also points to another issue with the algorithm: bimodal particle distributions. Particle filtering typically attempts to find the expectation of the posterior distribution over vehicle states, but this information is not useful in cases where the distribution is multimodal, like when the distribution is split across two lanes of a road. In this case, *any* single state estimate will be misleading (Dellaert et al., 1999). However, if a single state estimate is always needed, then it may help to apply clustering to the particles, then to take the expectation of the highest weighted cluster.

As mentioned in Section 4.2, the algorithm's accuracy drops significantly when the vehicle is no longer travelling on a mapped road. Simultaneous Localisation and Mapping (SLAM)—where the internal map representation is updated to correspond to the observed road location—is one potential fix for this problem. However, this particular application is unusual in that the mapped features are not directly observed in the same way that, for instance, a wall might be observed by a laser range finder, which complicates the use of SLAM. As a result, we have not incorporated any form of SLAM into the algorithm, although it cannot be ruled out that some form of SLAM may improve localisation accuracy.

At present, heading estimates are also difficult for the algorithm. Each particle includes a heading estimate for use in the prediction step of the particle filter, but no external heading observation is used to prevent the drift of these heading estimates. At high speeds, incorporating the heading estimated by the GPS unit into the particle filter update step may help, although this estimate does become highly volatile at low speeds (Ochieng et al., 2003). Alternatively, weighting particles by the mismatch between their heading and that of the direction of traffic flow on the nearest road segments might prevent scenarios like that depicted in Figure 3, in which the majority of particles are travelling north, but are localised on lanes on which traffic can only go south! Unfortunately, incorporating this information would require additional manipulation of the street map, which could prove too expensive for real-time applications.

Finally, the algorithm presented in this paper does not account for vehicle pitch, roll or altitude, nor for the elevation of the underlying road network. This information is seldom used in map matching algorithms, although there is at least one instance of vehicle elevation being used to improve GPS coverage by decreasing the number of satellites required for a GPS fix (Quddus et al., 2007). The algorithm presented in this paper could be extended to account for pitch, roll and altitude by modifying the state representation to include these quantities, as well as making use of attitude and altitude sensors during the particle filter update step. This could improve localisation in mountainous regions, where the distance travelled by the vehicle's wheels does not correspond well to the distance travelled over the map.

5 Conclusion

In this paper, we presented a particle filter-based algorithm for localising road vehicles using map data, GPS fixes and (optionally) odometry. In order to incorporate map data, we introduced a particle weighting factor dependent only on a given particle's distance to the nearest road in the mapped road network. By comparing the performance of particle filtering with and without this map update, we demonstrated that the use of map information can significantly improve localisation accuracy in scenarios where the vehicle remains on the road network. In contrast to earlier work, we showed that our algorithm functions even with low-quality crowd-sourced maps and highly noisy positioning sensors. Finally, we discussed and proposed solutions to some shortcomings of the presented algorithm.

References

P. Alliez, S. Tayeb, and C. Wormser. 3D fast intersection and distance computation. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.6 edition, 2015. URL http://doc.cgal.org/4.6/Manual/packages.html#PkgAABB_treeSummary.

J. M. Álvarez, A. M. López, T. Gevers, and F. Lumbreras. Combining priors, appearance, and context for road detection. *IEEE Transactions on Intelligent Transportation Systems*, 15(3):1168–1178, 2014.

R. Bajaj, S. L. Ranaweera, and D. P. Agrawal. GPS: location-tracking technology. *Computer*, 35(4):92–94, 2002.

M. A. Brubaker, A. Geiger, and R. Urtasun. Lost! leveraging the crowd for probabilistic visual self-localization. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3057–3064. IEEE, 2013.

F. Chausse, J. Laneurit, and R. Chapuis. Vehicle localization on a digital map using particles filtering. In *Intelligent Vehicles Symposium, 2005 IEEE*, pages 243–248. IEEE, 2005.

F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo localization for mobile robots. In *Robotics and Automation, 1999 IEEE International Conference on*, volume 2, pages 1322–1328. IEEE, 1999.

M. E. El Najjar and P. Bonnifait. A road-matching method for precise vehicle localization using belief theory and Kalman filtering. *Autonomous Robots*, 19(2):173–191, 2005.

D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. *AAAI/IAAI*, 1999:343–349, 1999.

F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *Signal Processing, IEEE Transactions on*, 50(2):425–437, 2002.

J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011.

M. Modsching, R. Kramer, and K. ten Hagen. Field trial on GPS accuracy in a medium size city: The influence of built-up. In *3rd Workshop on Positioning, Navigation and Communication*, pages 209–218, 2006.

M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, et al. Junior: The Stanford entry in the Urban Challenge. *Journal of Field Robotics*, 25(9):569–597, 2008.

W. Y. Ochieng, M. Quddus, and R. B. Noland. Map-matching in complex urban road networks. *Revista Brasileira de Cartografia*, 2(55), 2003.

M. A. Quddus, R. B. Noland, and W. Y. Ochieng. A high accuracy fuzzy logic based map matching algorithm for road transport. *Journal of Intelligent Transportation Systems*, 10(3):103–115, 2006.

M. A. Quddus, W. Y. Ochieng, and R. B. Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation Research Part C: Emerging Technologies*, 15(5):312–328, 2007.

A. Selloum, D. Betaille, E. Le Carpentier, and F. Peyret. Lane level positioning using particle filtering. In *Intelligent Transportation Systems, 12th International IEEE Conference on*, pages 1–6. IEEE, 2009.

A. F. Smith and A. E. Gelfand. Bayesian statistics without tears: a sampling–resampling perspective. *The American Statistician*, 46(2):84–88, 1992.

S. Syed and M. Cannon. Fuzzy logic-based map matching algorithm for vehicle navigation system in urban canyons. In *ION National Technical Meeting, San Diego, CA*, volume 1, pages 26–28, 2004.

S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial intelligence*, 128(1):99–141, 2001.

R. Toledo-Moreo, D. Bétaille, F. Peyret, and J. Laneurit. Fusing GNSS, dead-reckoning, and enhanced maps for road vehicle lane-level navigation. *Selected Topics in Signal Processing, IEEE Journal of*, 3(5):798–809, 2009.

N. R. Velaga, M. A. Quddus, and A. L. Bristow. Improving the performance of a topological map-matching algorithm through error detection and correction. *Journal of Intelligent Transportation Systems*, 16(3):147–158, 2012.

C. Vlcek, P. McLain, and M. Murphy. GPS/dead reckoning for vehicle tracking in the "urban canyon" environment. In *IEEE Vehicle Navigation & Information Systems Conference. Ottawa, Canada: IEEE*, 1993.

C. E. White, D. Bernstein, and A. L. Kornhauser. Some map matching algorithms for personal navigation assistants. *Transportation Research Part C: Emerging Technologies*, 8(1):91–108, 2000.