A Survey of Map Matching Algorithms

COMP2550 Literature Survey

Sam Toyer
u5568237@anu.edu.au

March 20, 2015

Abstract

Map matching is the process of taking a series of observations of a vehicle's environment—for instance, Global Positioning System (GPS) fixes, or the distance travelled by the vehicle's rear wheels in a certain time period—and attempting to fit the path described by those observations onto a digital map. In this survey, we explore several general approaches to this problem and compare the efficacy of current state-of-the-art algorithms for real-time map matching.

1 Introduction

Whilst satellite navigation systems are widely used for navigation, the positioning error inherent in these systems can be problematic in some applications, especially in built-up areas where buildings interfere with signals. For vehicles which are assumed to be on roadways, these errors can be reduced by restricting the possible positions of a vehicle to the roads listed on a digital map. The task of a map matching algorithm is to take noisy or biased observations from a localisation or odometry system and attempt to find the most accurate projection of these measurements onto a given digital map.

For the purposes of this survey, we will divide map matching algorithms into two categories. Firstly, we will consider heuristic algorithms which combine simple geometric and topological features of the road network in order to localise the vehicle. Secondly, we will consider probabilistic map matching algorithms, which model the vehicle's position as a probability distribution over all roads and localise the vehicle by finding the mode of this distribution.

2 Heuristic algorithms

Initial solutions to the map matching problem utilised only simple geometric heuristics, like the distance to each road segment, to localise vehicles. Later algorithms also took into account the topology of the road graph and the location history of the vehicle to try and match the "shape" of the vehicle's path to the map.

Whilst these early approaches offer middling performance on their own (Quddus et al., 2007), we nonetheless consider some of them in this section, as they form the basis for many of the more advanced algorithms which constitute the state-of-the-art today.

White et al. (2000) introduced the most influential of these algorithms, and we will briefly summarise two of them here.

The first algorithm which we will consider, *point-to-curve matching*, simply finds the nearest road segment in a digital map and calculates the orthogonal projection of the most recent GPS measurement onto that segment. As White et al. point out, this algorithm not only fails to make use of the vehicle's heading, velocity, past positions, and so on, but it also results in a highly unstable match at intersections, where the reported GPS reading may be approximately equidistant to two or more road segments.

The second algorithm which we will consider, *curve-to-curve matching*, attempts to incorporate both historical position data and map topology. It begins by drawing a piecewise linear curve through a fixed number of recent GPS fixes. For each road junction near the most recent of these fixes, it constructs additional piecewise linear curves of the same length as the curve produced from the vehicle's positioning history, each of which follow the road network outwards from the junction in question. Finally, it considers the area between each of the curves on the map and the curve representing the vehicle's path, and selects the route corresponding to the curve with the smallest area. The output of the algorithm is a projection of the most recent GPS

fix onto the best curve. This yields better performance than point-to-curve matching, as shown in Table 1, but at a greater computational cost.

Simple matching methods like point-to-curve and curve-to-curve can be combined using a number of schemes to produce improved map matching algorithms. These more advanced algorithms typically follow a three-step process:

1. Firstly, the algorithm estimates the vehicle's global position. This estimate can come directly from a GPS sensor, but is often paired with odometry data and passed through an Extended Kalman Filter (EKF) in order to smooth out noise. This step is independent of the choice of map matching algorithm, so we will treat it as a black box for the remainder of this survey.
2. Next, the algorithm attempts to find the specific road segment which the vehicle is travelling on.
3. Finally, the algorithm produces a best guess as to where the vehicle is within the selected road segment.

Step 2 is the most complex part of the matching process, as an incorrectly identified road segment will lead to a poor estimate of position. This step is often split into several different "modes", depending on the state of the vehicle. For instance, if it is not known which road segment the vehicle was travelling on during the previous time step, then the algorithm may take additional precautions—like waiting until reported GPS error is below a fixed threshold—in order to ensure that the first segment is matched correctly. If the first match is not correct, then it is likely that subsequently identified segments will not be correct either, as map matching algorithms often choose not to consider segments which are not neighbours of the previously matched segment. A similar issue occurs at junctions, where the vehicle can turn onto one of several different road segments, and the consequences for incorrectly inferring which segment the vehicle has branched onto are similar to those for incorrectly inferring the initial segment.

On the other hand, Step 3 is straightforward, as it usually consists of a simple projection of the last GPS fix onto the matched road segment. As such, we will not consider it for the remainder of this section so that we can focus on the more complex link selection process.

Ochieng et al. (2003) propose an algorithm which is representative of these three-step map matching procedures. Upon initialisation, it selects an initial link based primarily on the link's distance from the vehicle's estimated global position. It then enters into a subsequent matching phase, during which it projects GPS coordinates onto a previously selected link. If the vehicle begins to turn or nears a junction, the algorithm re-enters its initial link selection phase to detect which road segment the vehicle has entered.

One of the drawbacks to this approach is that it involves a lot of "magic numbers"; for instance, heading difference thresholds are required to determine when the vehicle is turning, and distance thresholds are required to determine when the vehicle is near a junction. Ochieng et al. tweak these constants manually, but this method is labour intensive, and may need to be repeated for different environments, like rural or suburban environments, in which different thresholds may be advantageous.

Velaga et al. (2009) present a similar algorithm, albeit one which selects links a using weighted sum of features related to vehicle heading, link heading, distance along a link and link connectivity. As a result of their use of weighting for link features, their algorithm is even more dependent on tunable constants than that of Ochieng et al., and Velaga et al. address this issue by proposing an accompanying optimisation procedure for finding weights.

Optimising weights in a map matching algorithm is challenging, as the most useful error function for a given set of weights—the proportion of mismatched roads—can generally only be computed by performing a complete run of the map-matching algorithm over the available data. Velaga et al. handle this by assuming that map matching error can be approximated by a simple function of the weights used by their algorithm. Once this approximation to the error function is found using an optimisation algorithm, Velaga et al. choose the weights used by their map matching algorithm to be the set of weights which minimise this learnt approximation.

In a later paper, Velaga et al. attempt to tweak this algorithm slightly in order to improve its map matching accuracy, and in the process illustrate an important consideration in map matching algorithm design: by deriving three different sets of weights—one for rural areas, one for suburban areas, and one for urban areas—and choosing the appropriate set of weights for the environment in which the vehicle is operating, they produce an mismatch rate decrease of around two percentage points, as shown in Table 1 (Velaga et al., 2012). This demonstrates the sensitivity of weights used in heuristic algorithms to the environment in which they are used, and should be considered when evaluating other weighted map matching algorithms.

Fuzzy logic is another commonly used method of incorporating multiple heuristics for map matching. Fuzzy logic systems consider a variety of numerical inputs—for instance, distance to a road segment d_i or speed of a vehicle s—and "fuzzify" these according to "membership functions", which map a numerical input like velocity to a "degree of applicability" for some label, like "fast" or "slow". The fuzzy inference system can then apply a number of rules like those listed below, which attempt to determine how likely it is that a

vehicle is on a given road segment r_i:

If d_i is short and s is fast then r_i is unlikely
If d_i is long and s is medium then r_i is likely

Each of the above rules will use some predefined method to find the degree to which d is "short" or "long" and the degree to which s is "medium" or "fast", then combine the weights for these labels together and use them to find the degree to which r_i is "likely" or "unlikely". These degree to which each label is applicable is then combined back into a real number using one of several methods, and this number is taken to represent how likely it is that the vehicle is on segment r_i.

The precise way in which input variables are mapped onto labels like "fast" and "slow", as well as the way in which these labels are combined to produce a numerical result, varies from application to application, and a number of concrete examples are given by Zadeh (1988). However, the common quality of fuzzy logic systems is that they allow the designer to incorporate their own domain knowledge into the system in a natural way using inference rules like those given above.

Quddus et al. (2006) propose a particularly effective fuzzy-logic based method which merges odometry, heading information, road segment connectivity, and GPS fixes. Their road segment selection mechanism operates in one of three modes: one for determining the initial link when no map-matched positions are available, one for determining subsequent links when travelling between junctions, and one for determining subsequent links at junctions. Each of these modes uses a separate set of fuzzy inference rules, which take into account GPS precision as reported by the receiver, the vehicle's speed, heading and position relative to the link, distance and direction to the nearest junction, and the vehicle's change in heading over time. The results produced by this algorithm are amongst the best reported in recent years, as detailed in Table 1.

However, fuzzy logic algorithms like that of Quddus et al. have many of the same flaws as the simpler heuristic ones mentioned at the beginning of this section; like other heuristic algorithms, fuzzy logic algorithms have a large number of parameters and design decisions which can be difficult to optimise, and which are often determined through trial and error. Fuzzy logic algorithms also require a high degree of domain knowledge and experimentation to write inference rules, which are not required in simple weighted models.

3 Probabilistic algorithms

Probabilistic algorithms approach map matching as a problem of trying to find a probability distribution over all possible positions of a vehicle on the road network, given the available sensor data. This distribution can be approximated using a number of techniques, and the resultant approximation can then be used to provide a reasonable estimate of the vehicle's location on the map, as well as a measure of certainty for this estimate.

The most common probabilistic technique for map matching is particle filtering. A particle filter approximates a vehicle's position on the road using a fixed number of "particles" at different locations, each representing a different possible location for the vehicle. Each particle has an associated weight, which indicates how well the position of that particle agrees with the of sensory inputs to the system. Initially, the particles are scattered uniformly throughout some region, and all weights are equal. At each time step, the particle filter does the following, as described at length by Gustafsson et al. (2002):

1. Each particle's weight is updated using the likelihood of the most recently acquired sensor data, given the assumption that the true location of the vehicle is that of the given particle.
2. A new set of N particles is then sampled, with replacement, from the old set of N particles according to the particles' weights. The weights of the particles are then made equal.
3. Finally, the state of each particle is advanced according to a model of the vehicle's motion. Some noise may be added during this process to ensure that subsequent re-sampling steps do not eliminate all but the most probable particle state.

Both Toledo-Moreo et al. (2009) and Selloum et al. (2009) employ particle filters in a similar manner. The algorithms presented in each of these papers incorporate map data by forcing a particle's weight to zero whenever its distance to the nearest road centerline exceeds a half a standard lane width, ensuring that they will be eliminated at the next resampling step. The algorithms also incorporate data from onboard gyroscopes and odometers in order to advance the state of each particle, and can optionally use GPS data when it is available, which is a boon in environments where GPS signals are patchy.

One common disadvantage to these approaches is that they rely on the availability of "enhanced maps" (or "Emaps"), which use clothoids (also known as Euler or Cornu spirals) to represent road segments, and are thus able to more accurately express the path of curved roads. The algorithms also rely on information about lane widths, although this can be estimated from national road standards if necessary. Another drawback is that particle filters can cease to function properly when a large number of particles have their weights clamped to zero during an update step, as happens when a par-

ticle leaves the road, and neither paper addresses this weakness in detail.

Another more recent approach to probabilistic map matching, which still requires knowledge of lane location and direction, but dispenses with the requirement for lane width, is given by Brubaker et al. (2013). Rather than approximating the vehicle's location distribution using particles, Brubaker et al. use a weighted mixture of Gaussian distributions assigned to each road segment. At each time step, these Gaussians are updated using both simulated vehicle dynamics and odometry. The vehicle's location can then be inferred from the mode of the position distribution.

Despite only requiring odometry data, which can be derived either using computer vision methods or from GPS fixes, this approach was reported to be able to go from a uniform distribution over 2000km of roads to a positioning estimate within 3m of ground truth after only 20s of observations. The chief weakness of this approach is that it sometimes fails to localise the vehicle when there exist multiple possible routes on a map with approximately the same shape, as such routes are not distinguishable using odometry alone.

4 Existing results and future research directions

At present, the only commonly available metric for evaluating map matching algorithms is the proportion of road segments which are correctly identified. Other metrics like along-track error and cross-track error are seldom available, and even when they are, the data and code used to produce them is not readily available, making it difficult to compare algorithms presented in different papers. Furthermore, some benchmarks are performed on extremely limited datasets, which limits the conclusions which can be drawn about the algorithm. In spite of these caveats, the available performance data for the algorithms mentioned in this survey has been summarised in Table 1 of the appendix.

At present, the fuzzy logic algorithm presented in Quddus et al. (2006) has achieved the best self-reported accuracy of any algorithm tested on a large data set. However, other approaches utilising fuzzy logic have not been able to approach this level of accuracy (Syed and Cannon, 2004; Ren and Karimi, 2012). It may be worthwhile to investigate precisely which factors contribute to the effectiveness of fuzzy map matching algorithms in order to explain this discrepancy.

The work of Velaga et al. (2012) in improving the performance of simple heuristic algorithms through automated optimisation of weights is another interesting area of research. Extending this solution to also learn thresholds for turn and junction detection, rather than just the weights used to score road segments, would likely lead to further improvements. Applying standard machine learning techniques, as opposed to the specialised methods proposed by Velaga et al., may also improve accuracy, although this is more challenging as it would necessitate redesigning the algorithm to make the objective function used in optimisation—which was given as the number of mismatched links over a complete run of the algorithm—more efficient to compute.

One other promising avenue of research is the use of maps from the OpenStreetMap[1]project, which offers free and extensive coverage of the road network, but at a resolution below that used in some past map matching papers. In particular, adapting the particle filtering approach presented in Toledo-Moreo et al. (2009) and Selloum et al. (2009) to function with these readily available maps, rather than with complex "Emaps", could yield an algorithm with low positional error and good tolerance of mismatches which would be appropriate for low-cost navigation systems.

Finally, future research in map matching could benefit greatly from standardised map matching benchmarking methods. At the very least, reporting link matching accuracy and horizontal positioning error would assist other researchers in making meaningful comparisons between algorithms, as would the use of publicly available data sets like that produced by Newson and Krumm (2009).[2]

5 Conclusion

In this survey, we explored two broad categories of map matching algorithms: heuristic algorithms, which use combinations of simple features like a vehicle's distance from a road link or its heading relative to a link in order to localise the vehicle, and probabilistic algorithms, which try to model a probability distribution of possible vehicle positions. The first category included techniques ranging from point-to-curve and curve-to-curve matching to more advanced weighted and fuzzy logic based algorithms, whereas the second category included algorithms based on particle filtering and Gaussian mixture models. We then concluded with a brief summary of reported performance data for state-of-the-art algorithms, and suggested some promising directions for future research, including further investigation of fuzzy logic methods, research into automated optimisation of weights used in map matching algorithms, and the use of more readily available map data.

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

[2] http://research.microsoft.com/en-us/um/people/jckrumm/MapMatchingData/data.htm

References

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. 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.

P. Newson and J. Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 336–343. ACM, 2009.

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.

M. Ren and H. A. Karimi. A fuzzy logic map matching for wheelchair navigation. *GPS solutions*, 16(3): 273–282, 2012.

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.

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.

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. Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems. *Transportation Research Part C: Emerging Technologies*, 17(6):672–683, 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. 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.

L. A. Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

Appendix: Algorithm performance data

Paper	Extra information	Algorithm type	Identification %	ATE	XTE	HE
White et al. (2000)	GPS	Point-to-curve	53.4–67.7% 76.8%†	29.5m (95%)†	10.1m (95%)†	32.0m (95%)†
		Curve-to-curve	61.7–72.6%	-	-	-
Gustafsson et al. (2002)	Wheel speeds, lane data	Particle filter	-	-	-	-
Ochieng et al. (2003)	GPS, gyroscope, odometer	Heuristic	100%	-	-	-
Quddus et al. (2006)	GPS, gyroscope, odometer	Fuzzy logic	99.2%	4.2m (95%)	3.2m (95%)	5.5m (95%)
Selloum et al. (2009)	GPS, lane data, gyroscope, odometer	Particle filter	73%	-	-	-
Toledo-Moreo et al. (2009)	GPS, lane data, gyroscope, odometer	Particle filter	-	-	-	1.9m (95%)
Velaga et al. (2009)	GPS, gyroscope	Heuristic	95.9–96.7%	7.36m (95%)	9m (95%)	9.8m (95%)
Velaga et al. (2012)	GPS, gyroscope	Heuristic	97.8%	2.11m±1.52m	3.19m±2.59m	4.19m±2.47m
Brubaker et al. (2013)	GPS (or visual) odometry, lane data	Gaussian mixture model	-	-	-	2.4m (mean)

Table 1: Performance data for map matching algorithms mentioned in this survey, as reported by their respective authors except where indicated otherwise. Figures marked with a superscript dagger† are taken from Quddus et al. (2006). Column contents are, from left to right: paper in which the algorithm appeared, sensors and additional map data required by the algorithm, algorithm family, correct segment identification percentage, along-track error (ATE), cross-track error (XTE) and horizontal error relative to ground truth from an augmented GPS system (HE). Positioning errors are given either as percentiles, means with standard deviations (mean±std. dev.), or just means, as indicated.