

1 ABSTRACT

2 Due to the availability of GPS sensors in a variety of everyday devices, GPS trace-data is becoming
3 increasingly abundant. One potential use of this wealth of data is to infer, and update, the geometry
4 and connectivity of road maps, using what are known as “map generation” or “map inference”
5 algorithms. These algorithms offer a tremendous advantage when no existing road map data is
6 present – rather than incur the expense of a complete road survey, GPS trace-data can be used to
7 generate entirely new sections of the road map at a fraction of the cost. Road map inference can
8 also be valuable in cases where a road map is already available. Here, they may not only help
9 to increase the accuracy of drawn road maps, but also help to detect new road construction and
10 dynamically adapt to road closures – useful features for digital road maps being used for in-car
11 navigation. Proposed algorithms have been accompanied by evaluations that are largely qualitative
12 in nature, and feature little or no comparison against prior work. To address this lack of quantitative
13 and comparative evaluation, this paper makes the following contributions: (a) a comprehensive
14 survey of the current literature on map generation. (b) the first method described for evaluating
15 automatically generated maps. (c) a qualitative, quantitative, and comparative evaluation of three
16 reference algorithms. (d) open-source implementations of the three algorithms, a 118 hour trace
17 data set and ground truth map for unrestricted use by the automatic map generation community.

1. INTRODUCTION AND MOTIVATION

Road map inference is the process of automatically producing a directed, annotated road map from GPS traces. GPS traces are typically collected opportunistically, from vehicles that are anyway driving the roads for some other purpose. Map inference can be used to rapidly map unknown or constantly changing territory, to update and improve upon existing road maps, or to quickly adapt to detours and new construction.

Common throughout the existing literature on automatic map generation is a focus on qualitative evaluation. Virtually every published paper on the topic relies on visual inspection of the results, manually comparing generated maps against existing maps or satellite imagery. Moreover, comparison against existing work is virtually absent: out of 11 papers surveyed, only one (1) provides any comparison against prior work. The focus on qualitative evaluation in the literature, and the lack of comparative evaluation, has made it difficult to understand the relative merits of the various proposed map inference methods.

We conjecture that the lack of quantitative, comparative evaluation of map inference algorithms is due to three missing ingredients: (a) a sufficiently expressive and robust method of quantitatively evaluating the accuracy of a generated map, (b) publicly available implementations of proposed map inference methods, and (c) common sets of publicly available GPS traces and ground truth maps for use in evaluations.

Our contributions: In this paper, we thoroughly address the three problems identified above. We: (i) describe the first quantitative evaluation method for map inference algorithms, (ii) provide open-source implementations of three reference algorithms, and (iii) make available a 118 hour trace dataset, plus ground truth maps for unrestricted use by the map inference community. The data and implementations are available on our website (2). To provide a better understanding of prior work on map inference, we also provide: (iv) a comprehensive survey of the current literature on automatic map generation, and (v) a qualitative, quantitative, and comparative evaluation of our three reference algorithm implementations.

The remainder of the paper is structured as follows: §2 surveys the literature on the topic of automatic map generation, briefly describing the various methods proposed and the evaluation performed in each paper. §3 describes the quantitative evaluation method we propose. §4 visually illustrates the performance of the algorithms and offers certain qualitative observations. §5 quantitatively measures the performance and briefly discusses our implementations of three reference algorithms (3, 4, 5). §6 concludes the paper.

2. BACKGROUND AND SURVEY OF EXISTING LITERATURE

The base-line requirement of a map inference algorithm is to automatically turn raw GPS traces into a directed and annotated graph representing the connectivity and geometry of the underlying road network. Beyond such basic map generation, various additional objectives have been proposed, such as extracting detailed intersection geometries (6), the number and centerlines of lanes (6), and speed limit and road type (10). In this paper, we focus primarily on basic map generation, and only briefly mention these other aspects.

Paper	Class	Data	Ground Truth	Evaluation Method	Features	§
Edelkamp & Schrödl '03 (3)	k-means	250 synthetically perturbed traces	gen. GPS traces	lane error vs. amount of data	lane finding	2.3.1
Schroedl et.al. '04 (6)	k-means	250 synthetically perturbed traces	gen. GPS traces	lane error vs. amount of data	intersection geometry	2.3.2
Davies et. al. '06 (4)	KDE	1M GPS points	UK ordnance survey	eyeball vs. ground truth		2.5.1
Worrall & Nebot '07 (7)	k-means	traces from mining vehicles	none	compact vs. raw	compact represent.	2.3.3
Guo, Iwamura & Koga '07 (8)	k-means	synthetic GPS traces	none	relative error vs. amount of data		2.3.3
Chen & Cheng '08 (9)	KDE	traces from automobiles	Google Earth	eyeball vs. ground truth		2.5.2
Niehofer et al. '09 (10)	trace merge	7 traces	Google Maps	eyeball vs. ground truth, relative error vs. amount of data	edge classification	2.4.2
Cao & Krumm '09 (5)	trace merge	20M GPS points from campus shuttles	Bing Maps	eyeball vs. ground truth, route query vs. Bing Maps	GPS trace clarification	2.4.1
Shi, Shen & Liu '09 (11)	KDE	“massive amounts” of GPS traces	Google Earth	eyeball vs. ground truth		2.5.2
Jang, Kim & Lee '10 (12)	k-means	GPS traces	Naver maps (13)	eyeball vs. ground truth		2.3.2
Agamennoni et al. '11 (1)	k-means	5 days/15 open mine vehicle GPS traces	none	eyeball vs. (4) and (6)	“principal road path”	2.3.4

TABLE 1 Map Inference Papers in Chronological Order. Out of 11 Papers, 4 Quantitatively Evaluate the Accuracy of Generated Maps, and None Quantitatively Compare Against Existing Work.

2.1. Operational Overview

The map generation process is typically preceded by a filtering step, where traces are checked for any irregularities with regard to expected distance between points, speed traveled, acceleration, and abrupt direction change. Any point along a GPS trace that fails to satisfy these criteria is removed, with an interpolated point being inserted in its place.

After filtering, the approaches described in the literature can be categorized by their algorithmic foundations: the k -means algorithm (14), trace merging, or kernel density estimation (KDE) (15). Members of the first category perform variations of a 3-dimensional (latitude, longitude, direction) k -means algorithm to reduce the set of GPS points to a smaller set of cluster centroids. The centroids are then linked together to form the road geometry. Trace merging methods merge edges directly, without first reducing the set of raw GPS data. Finally, KDE-based methods first produce a kernel density estimate of the raw GPS points or edges, then co-opt image processing techniques to extract the road geometry from this density estimate.

2.2. Survey of Existing Literature

Table 1 summarizes the literature on automatic map generation. Out of the 11 papers, 6 report k -means based methods, 2 report trace merging methods, and 3 report KDE methods. As noted in the “Evaluation Method” column, the literature thus far has almost exclusively relied on a qualitative evaluation method (“eyeballing”). Typically, a sample of the the generated road geometry is overlaid on a map or satellite image (ground truth), from which conclusions are drawn manually. Surprisingly, despite the widespread availability of ground truth maps, quantitative evaluation has been exceedingly rare in this line of work. In the cases where any type of numerical accuracy result is reported (3, 6, 8, 10), no comparison is made to related work. Similarly, in cases where a comparison is made against related work (1), only a qualitative comparison is offered, with no numbers reported.

Below, we introduce the three categories of map inference algorithms in more detail and briefly discuss the variations on each offered in the literature.

2.3. Map Inference Based on the k -means Algorithm

The k -means based approach, originally described in (3) and illustrated in Figure 1, begins by distributing a series of “cluster seeds” at locations drawn from the set of trace data, with the constraint that every trace point must be within a fixed distance d and bearing difference δ of a cluster seed.

Using the cluster seeds as initial guesses, minor variations on the k -means algorithm are then used to find seed locations and headings that best describe the raw traces. Once the seed locations are identified, seeds are linked to form road segments, based on the pattern of raw traces passing through the seed locations. We provide qualitative and quantitative evaluations of the basic k -means algorithm in §4-5.

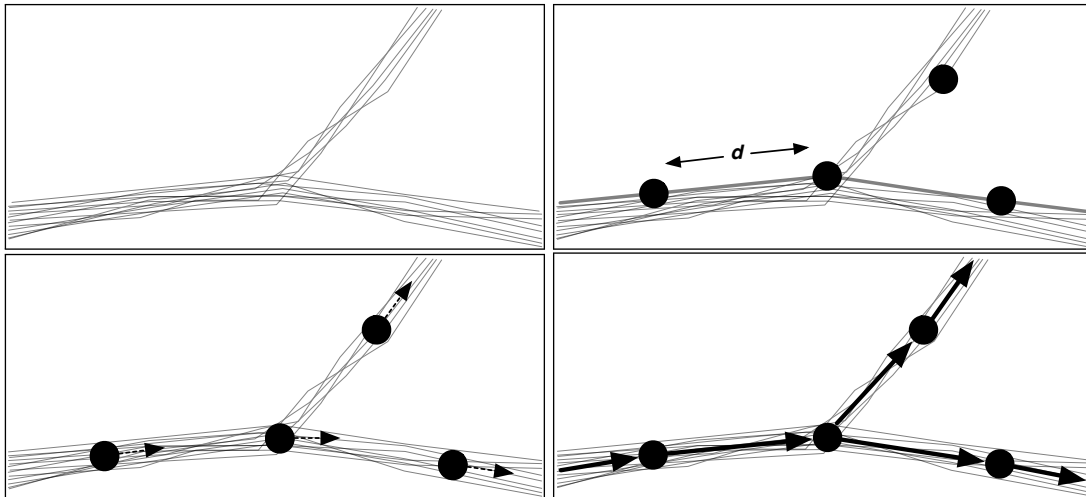


FIGURE 1 Map inference using the k -means algorithm. Raw traces (top left), initial seeds (top right), updated means with bearing (bottom left) and final graph after linking means (lower right).

1 2.3.1. *Edelkamp and Schrödl (2003) (3)*

2 In addition to being the original k -means based method, this paper further refines the road network
 3 model by fine-tuning the location of intersections, representing the road centerline by a fitted spline
 4 rather than a series of straight lines, and performing lane finding. Lane finding is done by clustering
 5 raw traces based on their respective distance-offset from the road centerline.

6 This work was evaluated by comparing the number of lanes detected, and the lane position
 7 error against a base map generated by the authors, using survey-grade differential GPS equipment.

8 2.3.2. *Schroedl et al. (2004) (6)*

9 Based on (3), this paper describes a process for additionally refining the intersection geometry,
 10 modeling individual lanes and the transitions and turn restrictions between them. This is accom-
 11 plished by first identifying intersections and their bounding boxes. Traces through an intersection
 12 are then grouped by entry and exit points, and a spline-fitting technique (16) is used for each group,
 13 to produce the final turn-lane geometry of the intersection.

14 This work was evaluated by visually comparing generated intersection geometries against
 15 NavTeq maps (17). A heuristic performance optimization for this method was described by Jang,
 16 Kim and Lee (2010) in (12), and evaluated by visually comparing the generated map against the
 17 map provided by Korean web portal Naver (13).

18 2.3.3. *Worrall and Nebot (2007) (7)*

19 Here, a method based on (3) is presented for inferring a road network represented as a compact set
 20 of lines and arcs. In order to create a compact representation of the generated road map, the set of
 21 clusters is first segmented into regions of “constant curvature”. The best line or arc is then fitted

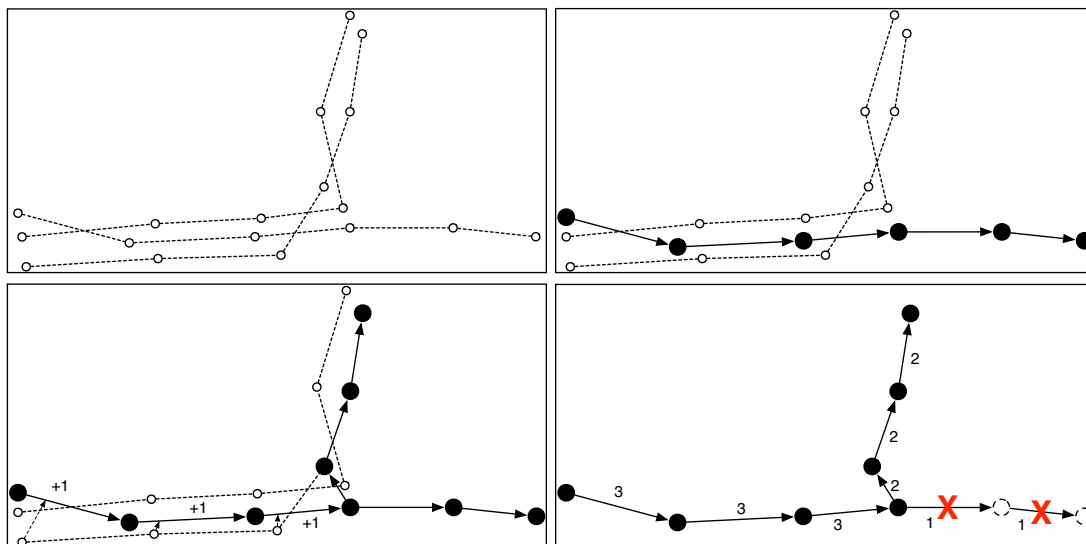


FIGURE 2 Trace merging approach. Raw GPS traces (upper left) are traversed one by one, adding new edges to the graph if no existing edge is sufficiently close in distance and bearing (upper right). Otherwise, the weights of matching edges are incremented (lower left). Edges with insufficient weight are pruned from the final graph (lower right).

1 to each segment. Standard regression analysis (18) is used for regions representing straight lines,
 2 whereas regions representing arcs are fitted using non-linear least-squares fitting (18).

3 This work was evaluated by measuring differences between the compressed map and the orig-
 4 inal generated map. A similar approach is presented in (8).

5 2.3.4. *Agamennoni, Neito and Nebot (2011) (1)*

6 The approach taken by Agamennoni, Neito and Nebot in (1) is similar to (6), but the focus is to
 7 extract “principal road paths”, which are curves as defined in (19).

8 In (20), a visual comparison is made against the original k -means approach (6), as well one
 9 based on kernel density estimation (4). A limited quantitative evaluation of this method is also
 10 presented in (20), using a GPS trace dataset collected by the authors.

11 2.4. Map Inference Based on Trace Merging

12 The trace merging methods proposed thus far use a greedy approach, illustrated in Figure 2. Iter-
 13 ating through each recorded GPS trace, edges from the raw trace are added to the map unless an
 14 edge sufficiently similar in location and bearing already exists. Should such an edge already exist,
 15 its “weight” is instead incremented. In post-processing, any edges with weight below a threshold
 16 are removed.

1 2.4.1. *Cao and Krumm (2009) (5)*

2 The method proposed by Cao and Krumm in (5) precedes the standard trace merging method with
3 a “clarification” step. The clarification step is a type of particle simulation (21), where a strong,
4 but short-range attractive force is applied to pull together nearby traces, and a weaker, but long-
5 range retractive force is used to keep traces from straying too far from their original location. This
6 reduces the effects of GPS noise by pulling together traces that originate from the same road,
7 forming tight bands along the road centerlines.

8 This work was evaluated by visually comparing the generated map against satellite imagery
9 from Bing Maps, and also by visually comparing shortest-path routes from the generated map
10 against those from Bing Maps.

11 We provide qualitative and quantitative evaluations of this trace merging algorithm in §4-5.

12 2.4.2. *Niehofer et al. (2009) (10)*

13 The method described by Niehofer, Burda, Wietfeld, Bauer, and Lueert in (10) modifies the stan-
14 dard merging approach by adjusting the position of existing road segments when merging a new
15 trace segment into the already-existing map. This technique allows the location of road segments
16 in the base map to be steadily refined as more traces are added, in a similar in spirit to the “clar-
17 ification” pre-processing step used in (5), except here it is used during the merge procedure. This
18 paper also describes means of automatically classifying road types, such as highways, streets or
19 walkways, as well as entry and exit ramps, bridges, and tunnels.

20 A qualitative evaluation of this work was conducted by visually comparing the generated map
21 against the same region depicted in Google Maps. A quantitative evaluation was also performed
22 against a “reference map” generated using all available traces, whereby the relative position error
23 of a road segment is shown to rapidly decrease with increasing amounts of data.

24 **2.5. Map Inference Based on Kernel Density Estimation (KDE)**

25 Map inference methods based on kernel density estimation (a) compute an approximate kernel
26 density estimate (15) of trace points or edges over the area of interest, (b) apply a threshold to
27 produce a binary image of the roads, and (c) apply a variety of methods to produce road centerlines
28 from this binary image, illustrated in Figure 3.

29 To produce the kernel density estimate, the geographical area of interest is divided into a 2-
30 dimensional grid of cells whose width is a fraction of the road width. Iterating over each of the
31 raw GPS points or edges, a 2-dimensional histogram is produced, containing the number of points
32 or edges that fall in each grid cell. The histogram is then convolved with a Gaussian smoothing
33 function (22), producing (an approximation of) the desired kernel density estimate. This is an
34 approximation due to the discretization created by the use of grid cells, where the accuracy of the
35 approximation is inversely proportional to the grid size.

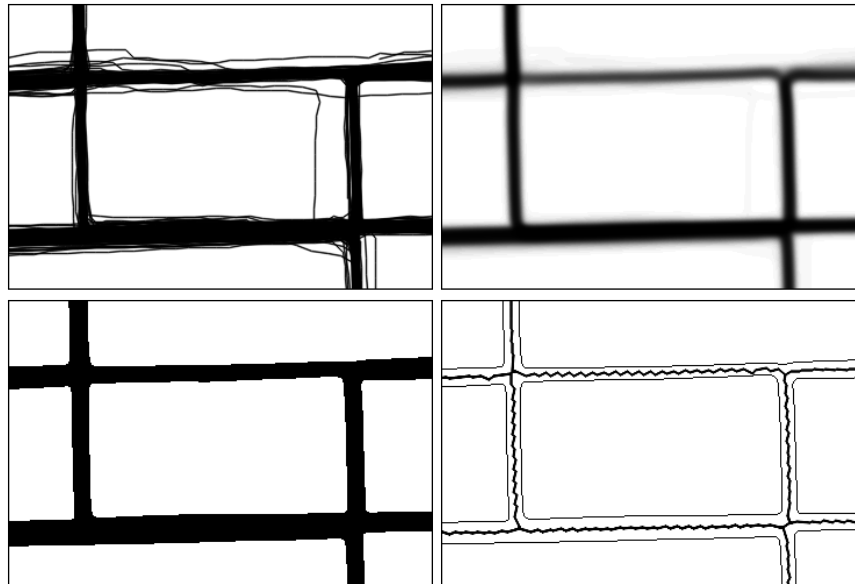


FIGURE 3 Map inference based on kernel density estimation (4). From the raw traces (top left), the KDE is computed (top right). A threshold is then applied to identify road/non-road cells (bottom left). Finally, road outlines and centerlines are computed (bottom right).

1 2.5.1. *Davies, Beresford and Hopper (2006) (4)*

2 In the scheme proposed by Davies, Beresford and Hopper (4), edges are accounted for in each grid
3 cell they pass through by an amount proportional to the length of line between the two points that
4 passes through the cell, in the anti-aliasing (23) fashion common in graphics.

5 After thresholding the density estimate, the outlines of the produced road “bitmap” are ex-
6 tracted using a contour follower (24). To find the centerlines of these outlines, which are likely
7 to coincide with the centerlines of the underlying roads, the Voronoi graph (25) of points evenly
8 spaced along the contours is produced, followed by the removal of edges that fall outside the con-
9 tour, or that are of insufficient length. Finally, separately produced KDEs of traces in each of the 8
10 cardinal and ordinal directions are used to annotate each road segment with its allowed directions
11 of travel.

12 This work was evaluated by visually comparing the generated map against that of a UK Ord-
13 nance Survey. We provide qualitative and quantitative evaluations of this KDE-based algorithm in
14 §4-5.

15 2.5.2. *Chen and Cheng (2008) (9)*

16 The KDE-based method described by Chen and Cheng in (9) produces a point-based density esti-
17 mate, and takes an image-processing approach to extract the road map from the bitmap image
18 produced after the thresholding step. Morphological “dilation” and “closing” operations are used
19 to produce a smooth and contiguous image of the road boundaries. Then, a “thinning” operation
20 deletes all pixels on the boundaries of a pattern until only a skeleton remains along the road cen-

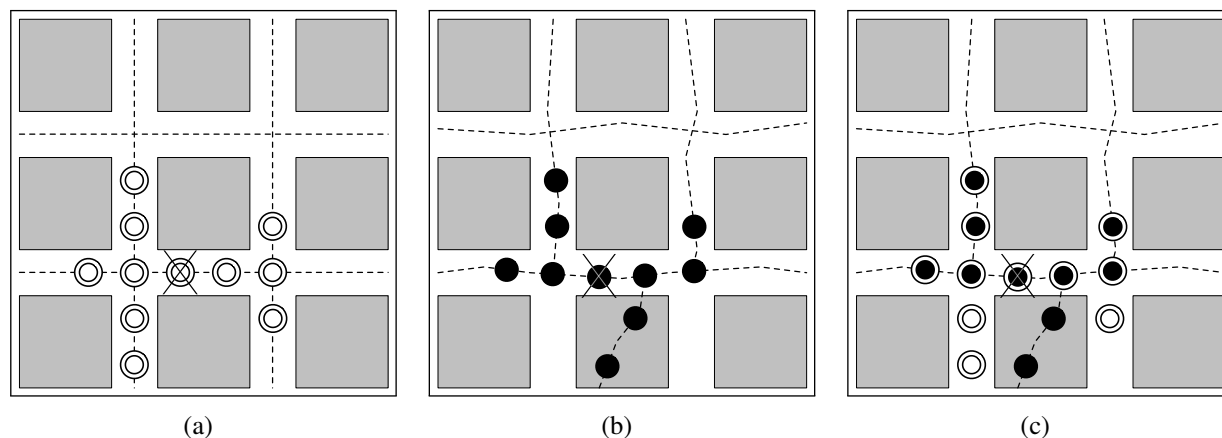


FIGURE 4 Overview of map comparison algorithm. (a) Holes are dropped at even intervals along edges of the ground truth map. (b) Marbles are dropped at even intervals along edges of the generated map. (c) Marbles from generated map fill holes where the maps overlap.

1 terlines, which is then converted into road segments. A very similar method was proposed by Shi,
 2 Shen and Liu in (11).

3 Both were evaluated by visually comparing the generated maps against images from Google
 4 Earth.

5 3. ROBUST QUANTITATIVE EVALUATION OF GENERATED MAPS

6 In this section, we describe a method of evaluating the accuracy of a map, with respect to a second
 7 “ground truth” map. This is a necessary requirement for a quantitative, comparative evaluation of
 8 competing map inference algorithms.

9 The accuracy of a generated map depends on two primary aspects of the map: geometry and
 10 topology. Here, the geometry of the map describes the geographical location of roads, while the
 11 topology describes the interconnections between roads.

12 A large body of work on “graph similarity” looks at the problem of comparing graphs (26).
 13 This includes exact methods testing for graph isomorphism (27), and inexact methods measuring
 14 graph edit distance (28). However, the problem of measuring graph similarity is fundamentally
 15 different from that of measuring map similarity: crucially, graphs and their nodes and edges lack
 16 any notion of geographical location. Thus, while graph similarity algorithms are able to measure
 17 the degree of topological similarity, their nodes and edges can be freely transposed in order to find
 18 the closest “match”. In a map, however, the geographical location of vertices and edges must be
 19 taken into account, in addition to their topological relationships.

20 To simultaneously measure the geometric and topological similarity of maps, we propose a
 21 method based on the following idea: starting from a random street location, explore each map
 22 outward within a maximum radius. This produces two sets of locations, which are essentially
 23 spatial samples of a local map neighborhood. By comparing the two sets of samples, and repeatedly
 24 sampling the maps in this fashion, we obtain a robust measure of the difference between the two

1 maps. If one of the maps is the “ground truth”, this difference represents the accuracy of the other.

2 The operation of our map comparison algorithm is depicted in Figure 4. First, we select the
 3 “street location” uniformly at random from the ground truth map. This point is marked with an “X”
 4 in Figure 4(a). From the starting point, we follow the road in question, dropping virtual “holes” at
 5 fixed intervals until a maximum radius r from the original location is reached. When an intersec-
 6 tion is encountered, we follow all connecting roads that lead away from the original location, as
 7 turn restrictions and one-way streets allow. Restricting the process to roads leading away from the
 8 origin elegantly de-emphasizes unlikely driving patterns such as u-turns, and improves the robust-
 9 ness of the map comparison operation. We then repeat this process starting from the closest point
 10 on the generated map, marked with an “X” in Figure 4(b). Following the same procedure, we drop
 11 virtual “marbles” at fixed intervals out to a radius r .

12 Intuitively, if a marble lands close to a hole, it falls in. This represents our matching process.
 13 As illustrated in Figure 4(c), marbles that are too far from a hole remain where they landed, and
 14 holes with no marbles nearby remain empty. In the figure, holes that are filled correspond to
 15 “matched locations”, where the geometry and topology of the two maps overlap. Unmatched
 16 marbles correspond to spurious parts of the generated map, and holes that remain empty correspond
 17 to parts of the ground truth map that are missing in the generated map. Counting the number of
 18 unmatched marbles and empty holes, we quantify the accuracy of the generated map with respect
 19 to the ground truth according to two metrics: the proportion of spurious marbles,

$$spurious = \frac{spurious_marbles}{spurious_marbles + matched_marbles},$$

20 and the proportion of missing locations (empty holes),

$$missing = \frac{empty_holes}{empty_holes + matched_holes}.$$

21 To produce a combined performance measure from these two measures, we use the well-known
 22 F-score (29), which is computed as follows:

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} = 2 \cdot \frac{(1 - spurious)(1 - missing)}{(1 - spurious) + (1 - missing)}.$$

23 The higher the F-score, the closer the match. Sampling the maps locally is an important aspect
 24 of our approach as it provides us the ability to capture the connectivity of the maps at a very
 25 fine-grained level (i.e., as they would be traveled), therefore allowing us to measure topological
 26 similarity. Conversely, one could imagine taking a global approach to this problem where we
 27 simply cover every edge in the ground truth map with holes, and every edge in the generated map
 28 with marbles, yielding a single pair of sets to match. While such a process would capture the
 29 geometric similarity between maps, it would fail to capture local topological similarity, a crucial
 30 aspect of overall map similarity. Repeated local sampling at randomly chosen locations yields an
 31 accurate view of local geometry and topology throughout the graph.

1 **3.1. Constructing the “Ground Truth”**

2 In order to measure the accuracy of the generated map, we need an accurate ground truth map for
3 comparison. We base our ground truth map on the OpenStreetMap (OSM) (30) database. However,
4 this map data contains many road segments that were never visited by the vehicles in our data set.
5 Comparing the generated maps against the full map would not be productive. Instead, we restrict
6 our ground-truth map to those street segments that were actually traversed by a vehicle in our data
7 set. This reduced ground-truth map reflects the most accurate road topology that can be inferred
8 by the available traces.

9 **4. QUALITATIVE EVALUATION**

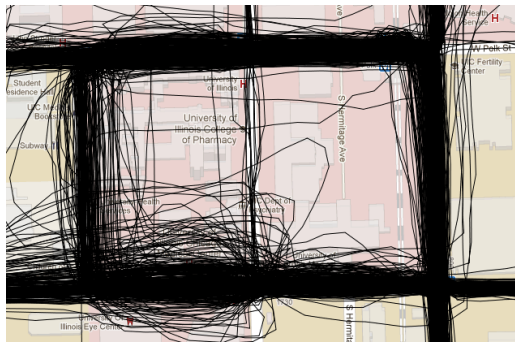
10 In this section, we qualitatively evaluate the road maps generated by three representative algo-
11 rithms, one from each of the three classes described in §2. To make this comparison, we first
12 re-implemented the algorithms as described in their respective papers. For k -means, we use the
13 algorithm by Edelkamp and Schrödl (*Edelkamp*) (3). For trace merging, we use the algorithm by
14 Cao and Krumm (*Cao*) (5), and for KDE-based map inference we use the algorithm by Davies,
15 Beresford and Hopper (*Davies*) (4).

16 **4.1. Evaluation Data**

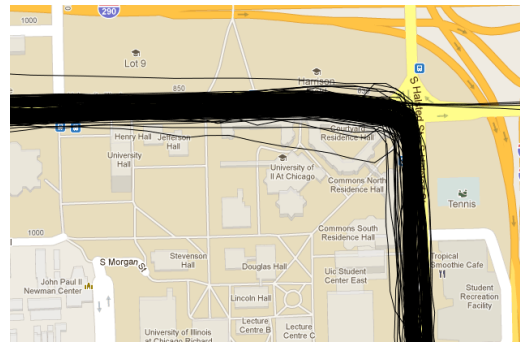
17 For trace data, we use 118 hours of GPS traces from the UIC campus shuttles. In addition to
18 traveling around campus, these pass through two different areas containing relatively tall buildings.
19 These areas contain significant GPS error. Figure 5(a) provides an example of the distribution of
20 traces and GPS error found in our dataset, in one of these high GPS error areas. In our evaluation,
21 we study both the entire dataset and a subset of the data drawn from an area of low-rise buildings
22 where there is very little GPS error (see a sample in Figure 5(b)). Because GPS error can be a
23 problem for map inference algorithms, partitioning our data this way allows us to test and compare
24 their performance on both a realistic, and a somewhat idealized dataset. The road maps generated
25 by each algorithm are depicted in Figures 5(c)-(h).

26 **4.2. Davies, Beresford and Hopper (4)**

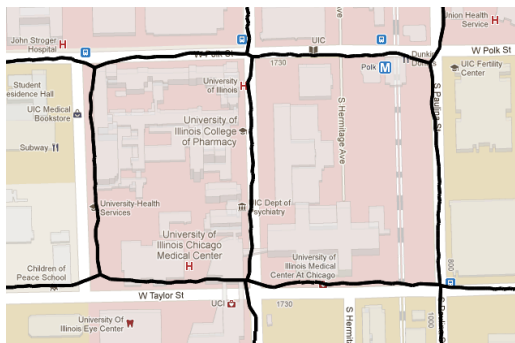
27 Visually, this algorithm produces superior maps in areas with large GPS error. Because this method
28 avoids treating GPS traces individually, and instead uses them in aggregate to find the road bound-
29 aries, it is able to create one road from a large collection of relatively diverse traces. We can see
30 this aspect of the algorithm illustrated in the difficult high-error case in Figure 5(c), where it accu-
31 rately extracted the road topology without adding any extraneous edges. In the low-error case in
32 Figure 5(d), the result is largely the same. However, note the absence of the less-traveled road seg-
33 ment on the right. This illustrates the fundamental trade-off of this algorithm: any given threshold
34 value must compromise between introducing noise in high-density areas, and losing infrequently
35 traveled edges in low-density areas.



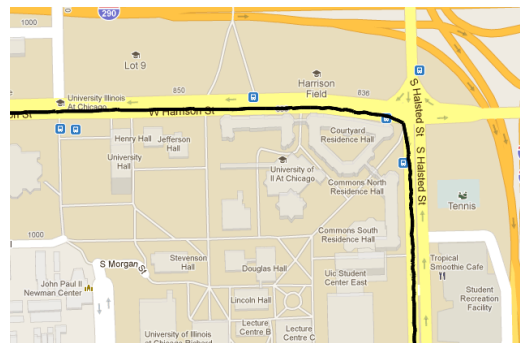
(a) Raw Data, high-error sample



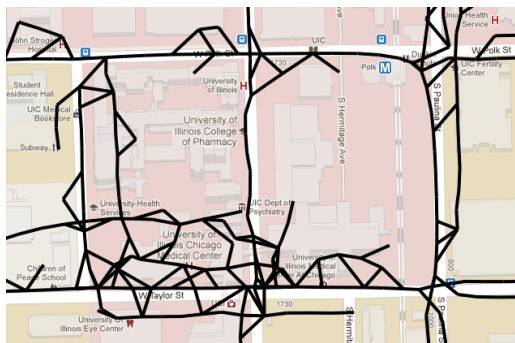
(b) Raw Data, low-error sample



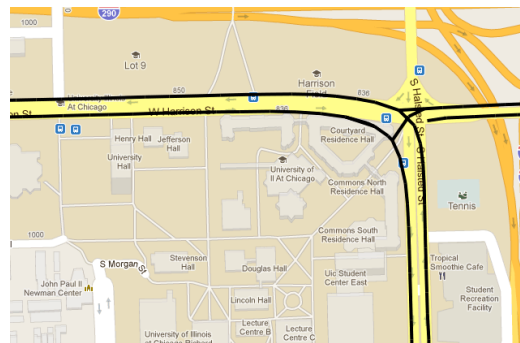
(c) Davies, high-error sample



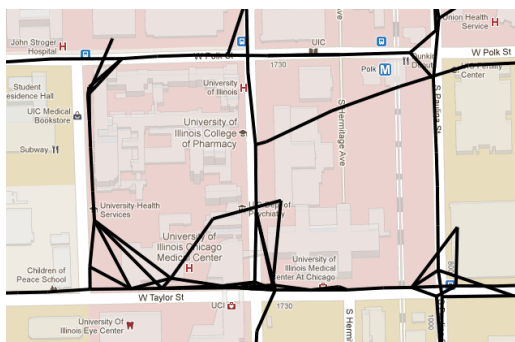
(d) Davies, low-error sample



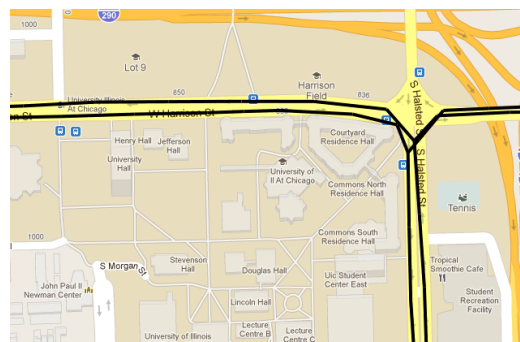
(e) Cao, high-error sample



(f) Cao, low-error sample



(g) Edelkamp, high-error sample



(h) Edelkamp, low-error sample

FIGURE 5 Raw data and results from the three implemented algorithms in areas with high and low GPS error.

4.3. Cao and Krumm (5)

The clarification pre-processing step performed in this algorithm helps reduce GPS noise prior to map generation, and in noise-free areas results in cleanly generated maps as demonstrated in Figure 5(f). However, clarification has its limitations in areas with high GPS error, where spatially dispersed traces are unable to become tightly banded. As a result, when the trace merging method is applied to the clarified data, residual noise results in the spurious roads seen in Figure 5(e). Although this algorithm attempts to prune spurious roads after map generation, its efforts are largely futile in areas of high GPS noise where edge volume is widely distributed.

4.4. Edelkamp and Schrödl (3)

This algorithm creates road segments by joining clusters based on the underlying trace data, and works well in areas with low GPS-noise, as we can see in Figure 5(h). However, this cluster-joining method is easily led astray by GPS noise and results in spurious roads being produced, illustrated in Figure 5(g). Because this algorithm does not attempt to prune spurious roads after generation, all of these roads remain in the final road map. Worth noting here is that our implementation does not include intersection refinement and lane-finding. This would likely have improved its results on the low-error data set, but would have little or no effect on the high-error dataset.

5. QUANTITATIVE EVALUATION

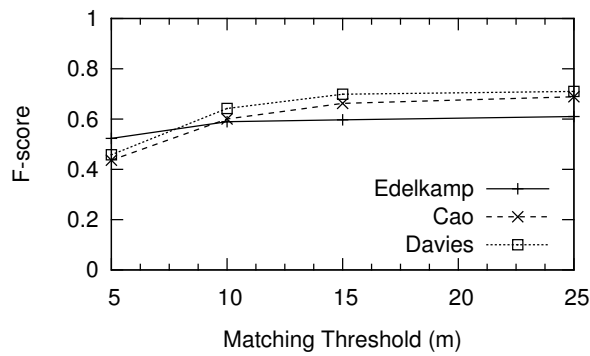
In this section, we use the map comparison method described in §3 to quantitatively evaluate the three representative algorithms described in §4. We also discuss parameter selection and implementation details below.

5.1. Main Quantitative Results

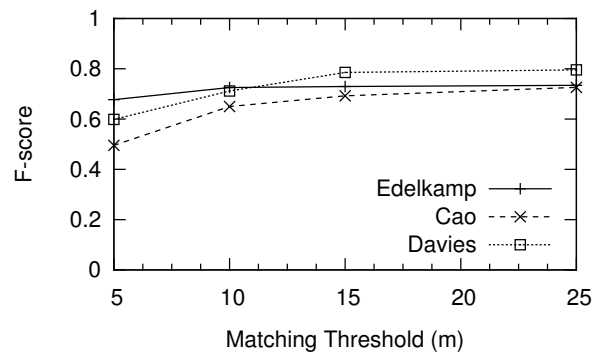
Figure 6 illustrates our main results, showing the performance of the three algorithms for a varying “matching threshold”. The matching threshold is the allowable distance between a “hole” and a “marble”. Detailed discussion of these results follows.

In Figure 6(a) we can see the performance of the three algorithms as tested on the full data set. For small matching thresholds, the fine-grained spatial accuracy of the algorithm is significant. Both Davies and Cao underperform Edelkamp for a 5-meter matching threshold. For Davies, this is explained by the fact that it generates a single, bi-directional centerline, whereas the others produce one edge in each direction. On a wide road, the distance between the centerline and the center of the road in one direction may well exceed 5 meters. A similar problem is exhibited by Cao, where lanes in opposite directions are artificially spread apart to improve legibility. This introduces a slight error, which shows up as poor matching performance for a 5-meter threshold.

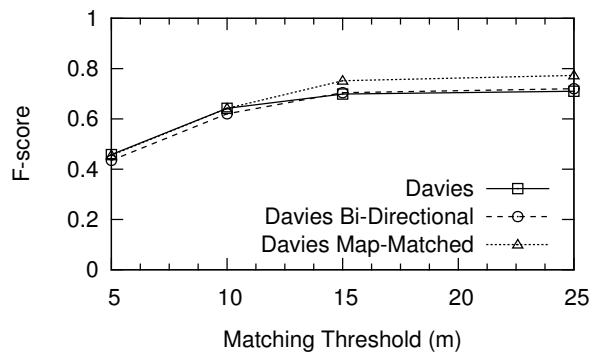
In the low-error data set, all of the algorithms unsurprisingly achieve a higher F-score. However, Edelkamp and Davies achieve a larger increase in performance compared to Cao, reversing the performance ordering of Edelkamp and Cao for this low-error data set.



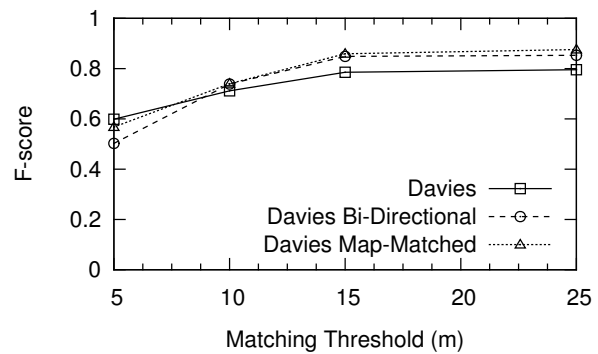
(a) All algorithms, full data set



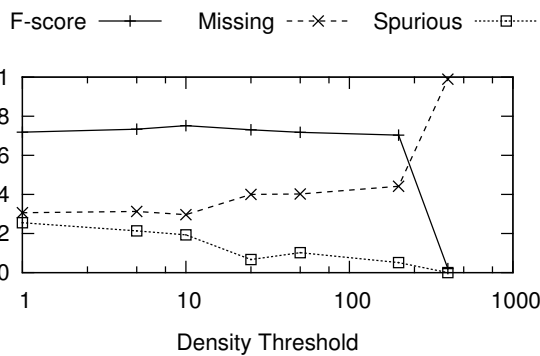
(b) All algorithms, low-error data set



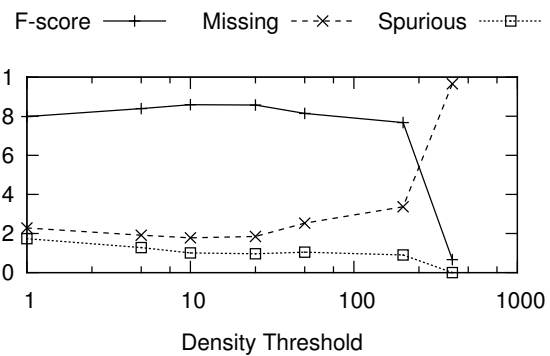
(c) Davies algorithms, full data set



(d) Davies algorithms, low-error data set



(e) Davies map-matched algorithm, full data set



(f) Davies map-matched algorithm, low-error data set

FIGURE 6 F-score results of the three algorithms for several matching distance thresholds, on the full (a) and low-error (b) data set. F-score results for the three Davies algorithms on the same full (c) and low-error (d) data sets. F-score, missing and spurious results for varying density threshold with the Davies map-matched algorithm, on the full (e) and low-error (f) data sets.

1 5.1.1. Addressing Directionality Finding in (4)

2 In contrast with the other two approaches, Davies generates a single centerline for each street, with
3 a directionality annotation. The authors in (4) use a technique that measures the direction of travel
4 through each grid cell (as mentioned in §2.5.1) to extract the allowed directions of travel. However,
5 in our testing this method did not fare well: many streets that are bi-directional in the ground-truth,
6 were not represented as such in the inferred map. To correct for this problem we experimented
7 with two different modifications to the Davies algorithm.

8 First, we discarded the direction-finding component of the Davies algorithm, making every
9 road bi-directional. While this now allowed for bi-directional roads whose directionality was in-
10 correctly inferred to be correctly represented, it also introduced the problem of incorrectly repre-
11 senting one-way streets as bi-directional. On the full dataset, which consists of a large number of
12 one-way and bi-directional streets, this tradeoff resulted in no net performance gain, as can be seen
13 in Figure 6(c). In our low-error data set however, which consists predominantly of bi-directional
14 roads, we can see that the bi-directional Davies implementation achieved a notable performance
15 gain (see Figure 6(d)).

16 As an alternative to the directionality-finding technique in (4), we map matched our GPS traces
17 onto the inferred bi-directional map above, using Viterbi map-matching (31). We then used the
18 resulting sequences of road segments to infer road directionality. The relative performance of the
19 three techniques is shown in Figures 6(c) and 6(d). Here, the line marked “Davies” is identical to
20 the one in Figures 6(a) and 6(b). Building on an already strong performance, the map-matching
21 technique gives the KDE-based algorithm by Davies a significant lead over the alternative ap-
22 proaches.

23 5.1.2. Remaining Challenges

24 Some challenges remain, even with our map-matching improvements to the Davies algorithm.
25 Primarily, the choice of a density threshold for Davies is made globally across the entire map.
26 If the chosen threshold value is too low, excess noise is added to the map in the form of spurious
27 roads. If the value is too high, portions of the map with relatively low density are treated as spurious
28 and removed, resulting in missing roads. This behavior is illustrated in Figures 6(e) and 6(f). Here,
29 the F-score is largely constant over a wide range of density threshold values. However, by studying
30 the components that make up the F-score, missing and spurious, the trade-off is clearly seen. As
31 the density threshold increases, the proportion of spurious edges decreases, while the proportion
32 of missing edges increases. We believe this insight is the key to further improvements to the KDE-
33 based method—only marginal improvements will be made as long as the constant threshold used
34 in the current algorithm remains.

35 5.2. Parameter Sensitivity and Implementation Details

36 Each of the algorithms described above includes several tuning parameters. We conducted a sensi-
37 tivity analysis in order to determine which of those most significantly impact performance. Some
38 of the results of this analysis are discussed below.

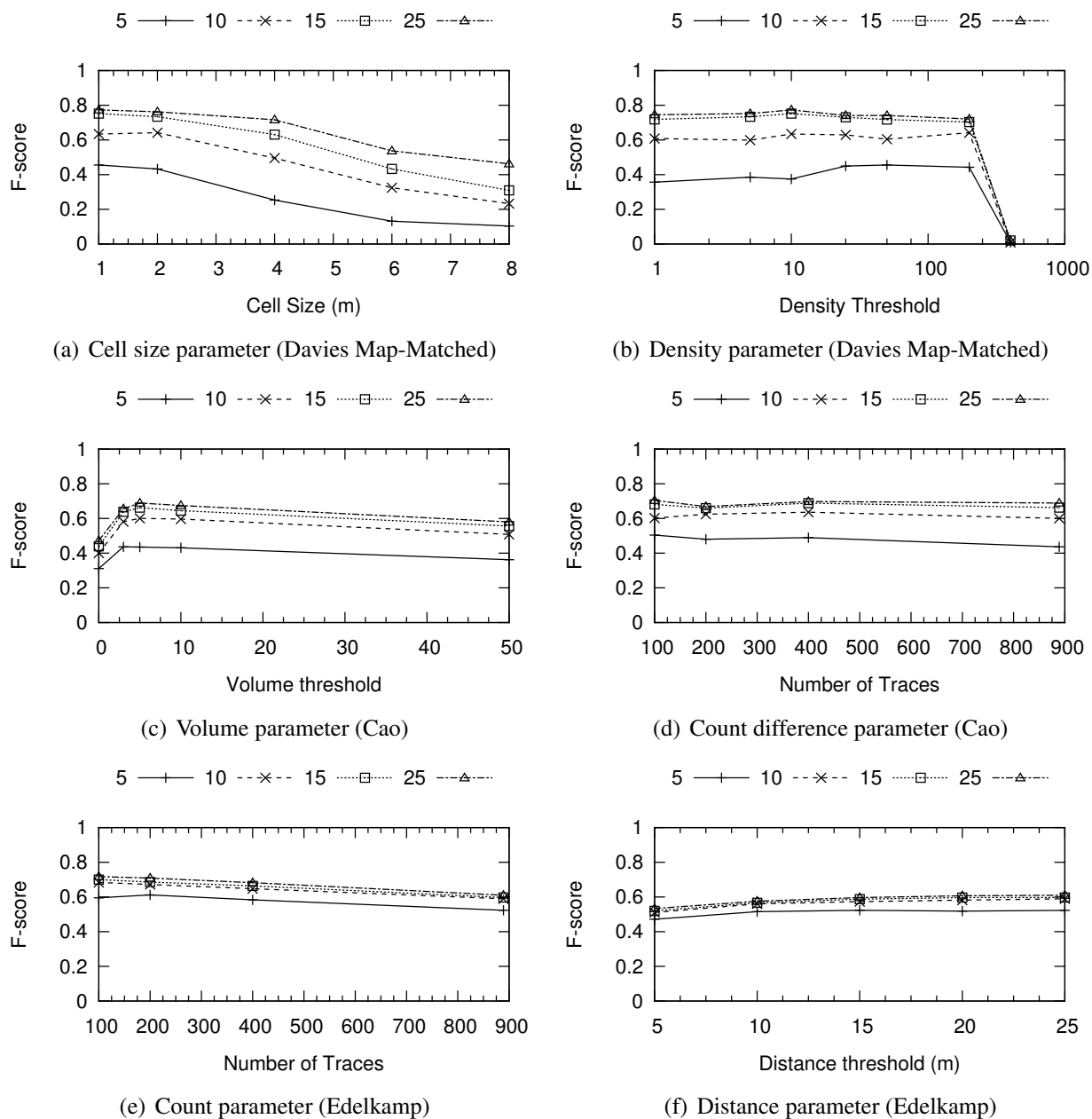


FIGURE 7 Parameter sensitivity testing over several matching distance thresholds for the three algorithms.

1 5.2.1. *Davies, Beresford and Hopper (4) Implementation*

2 Our implementation of this algorithm closely follows the description in (4), with the additional
3 modifications discussed in §5.1.1. The parameters for this algorithm are: cell size, density thresh-
4 old, kernel bandwidth, and number of traces used to generate the map. Figures 7(a) and 7(b)
5 illustrate this algorithm’s sensitivity to cell size and density threshold. In 7(a) we see that a smaller
6 cell size will always produce a better result with a fixed bandwidth kernel, as this simply improves
7 granularity. Curiously, the density threshold (see Figure 7(b)) is not a particularly sensitive param-
8 eter. This is due to the trade-off between missing and spurious roads, as discussed in §5.1.2.

9 5.2.2. *Cao and Krumm (5) Implementation*

10 Our implementation of this algorithm follows the description in (5) closely, with map generation
11 being preceded by clarification. The parameters for this algorithm are: edge volume threshold,
12 location distance limit, location bearing difference threshold, and number of traces used to generate
13 the map. Figures 7(c) and 7(d) illustrate this algorithm’s sensitivity to edge volume threshold and
14 number of traces used to generate the map. In 7(c) we see that increasing the edge volume threshold
15 – used for pruning spurious edges from the generated map – increases map accuracy, as spurious
16 edges are removed. However, we also see that if we prune too aggressively legitimate roads may
17 end up being incorrectly removed, decreasing accuracy. We can see in 7(d) that performance
18 decreases with the number of traces used to generate the map, a result of the increased noise that
19 comes with a larger dataset, and this algorithm’s inability to overcome that error.

20 5.2.3. *Edelkamp and Schrödl (3) Implementation*

21 Our implementation of this algorithm follows the description in (3), with the exception that no
22 intersection refinement is done. The parameters for this algorithm are: cluster seed interval, intra-
23 cluster bearing difference threshold, intra-cluster distance threshold, and number of traces used to
24 generate the map. Figures 7(e) and 7(f) illustrate this algorithm’s sensitivity to the number of traces
25 and intra-cluster distance threshold parameters. We can see in 7(e) that performance decreases with
26 the number of traces used to generate the map, as similarly to Cao, the larger dataset includes more
27 noise which this algorithm is unable to overcome. Also, we see in 7(f) that performance improves
28 with increasing intra-cluster distance, as this increases the clusters’ resistance to noise in the GPS
29 traces.

30 5.3. **Algorithm Runtime**

31 The runtime of these algorithms vary dramatically, due to differences in algorithmic complexity.
32 Particularly, Cao suffers in dense neighborhoods, where it exhibits quadratic complexity. On a
33 subset of 100 traces, Cao finished in 2.5 hours, Edelkamp in 73 seconds, and Davies in 8 seconds.
34 Our map-matched version of Davies finished in 106 seconds. On the full set of 899 traces, Cao
35 required 2.5 days, Edelkamp 15 minutes, and Davies 25 seconds. The map-matched version of
36 Davies finished in 14 minutes.

6. CONCLUSION

Robust quantitative evaluation methods, and rigorous comparison against prior work are important tools for furthering any field of scientific inquiry. Using the new tools presented here, we compared three algorithms from the literature. Overall, the algorithm by Davies et. al (4) was found to significantly outperform the others under a variety of conditions. Using our quantitative evaluation method, we were able to identify opportunities for further improvement to this algorithm. Some of these were implemented and evaluated here yielding a further, significant performance improvement. It is our hope and expectation that the new tools offered here will help bring significant advances to the study of map inference from GPS traces.

7. ADDENDUM

Implementations of the three reference algorithms (3, 4, 5), the 118 hour trace dataset and ground truth map used in this paper are available on our website (2), for unrestricted use by the map inference community.

REFERENCES

- [1] Agamennoni, G., J. Nieto, and E. Nebot, Robust Inference of Principal Road Paths for Intelligent Transportation Systems. *Intelligent Transportation Systems, IEEE Transactions on*, Vol. 12, No. 1, 2011, pp. 298–308.
- [2] *BITS Networked Systems Laboratory*, <http://bits.cs.uic.edu/>. Accessed July 22, 2011.
- [3] Edelkamp, S. and S. Schrödl, Route Planning and Map Inference with Global Positioning Traces. In *Computer Science in Perspective* (R. Klein, H.-W. Six, and L. Wegner, eds.), Springer Berlin / Heidelberg, Vol. 2598 of *Lecture Notes in Computer Science*, 2003, pp. 128–151.
- [4] Davies, J. J., A. R. Beresford, and A. Hopper, Scalable, Distributed, Real-Time Map Generation. *IEEE Pervasive Computing*, Vol. 5, 2006, pp. 47–54.
- [5] Cao, L. and J. Krumm, From GPS traces to a routable road map. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, New York, NY, USA, 2009, GIS '09, pp. 3–12.
- [6] Schroedl, S., K. Wagstaff, S. Rogers, P. Langley, and C. Wilson, Mining GPS Traces for Map Refinement. *Data Mining and Knowledge Discovery*, Vol. 9, 2004, pp. 59–87, 10.1023/B:DAMI.0000026904.74892.89.
- [7] Worrall, S. and E. Nebot, Automated Process for Generating Digitised Maps through GPS Data Compression. In *Australasian Conference on Robotics and Automation*, 2007.

- 1 [8] Guo, T., K. Iwamura, and M. Koga, Towards high accuracy road maps generation from mas-
2 sive GPS Traces data. In *Geoscience and Remote Sensing Symposium, 2007. IGARSS 2007.*
3 *IEEE International*, 2007, pp. 667–670.
- 4 [9] Chen, C. and Y. Cheng, Roads Digital Map Generation with Multi-track GPS Data. In *Edu-*
5 *cation Technology and Training, 2008. and 2008 International Workshop on Geoscience and*
6 *Remote Sensing. ETT and GRS 2008. International Workshop on*, 2008, Vol. 1, pp. 508–511.
- 7 [10] Niehoefer, B., R. Burda, C. Wietfeld, F. Bauer, and O. Lueert, GPS Community Map Gener-
8 ation for Enhanced Routing Methods Based on Trace-Collection by Mobile Phones. In *Ad-*
9 *vances in Satellite and Space Communications, 2009. SPACOMM 2009. First International*
10 *Conference on*, 2009, pp. 156–161.
- 11 [11] Shi, W., S. Shen, and Y. Liu, Automatic generation of road network map from massive GPS,
12 vehicle trajectories. In *Intelligent Transportation Systems, 2009. ITSC '09. 12th International*
13 *IEEE Conference on*, 2009, pp. 1–6.
- 14 [12] Jang, S., T. Kim, and E. Lee, Map generation system with lightweight GPS trace data. In
15 *Advanced Communication Technology (ICACT), 2010 The 12th International Conference on*,
16 2010, Vol. 2, pp. 1489–1493.
- 17 [13] Naver, <http://dev.naver.com/openapi/apis/map/>. Accessed November 14, 2011.
- 18 [14] Mitchell, T., *Machine Learning (Mcgraw-Hill International Edit)*. McGraw-Hill Education
19 (ISE Editions), 1st ed., 1997.
- 20 [15] Scott, D. W., *Kernel Density Estimators*, John Wiley and Sons, Inc., pp. 125–193, 2008.
- 21 [16] Piegl, L. and W. Tiller, *The NURBS book (2nd ed.)*. Springer-Verlag New York, Inc., New
22 York, NY, USA, 1997.
- 23 [17] NAVTEQ Maps and Traffic, <http://www.navteq.com/>. Accessed July 22, 2011.
- 24 [18] Wong, S. S. M., *Computational methods in physics and engineering*. World Scientific, 1997.
- 25 [19] Hastie, T. and W. Stuetzle, *Principal Curves*. University of Washington, 1988.
- 26 [20] Agamennoni, G., J. Nieto, and E. Nebot, *Technical Report: Inference of Principal Road*
27 *Paths Using GPS Data*. The University of Sydney, Australian Centre For Field Robotics,
28 <http://www-personal.acfr.usyd.edu.au/jnieto/Research2010>.
- 29 [21] Khanna, M., *Introduction to Particle Physics*. Prentice-Hall of India, 2004.
- 30 [22] Hastie, T., R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining,*
31 *Inference, and Prediction*. Springer, 2009.

- 1 [23] Leler, W. J., Human vision, anti-aliasing, and the cheap 4000 line display. In *Proceedings*
2 *of the 7th annual conference on Computer graphics and interactive techniques*, ACM, New
3 York, NY, USA, 1980, SIGGRAPH '80, pp. 308–313.
- 4 [24] Yokoi, S., J. ichiro Toriwaki, and T. Fukumura, An Analysis of Topological Properties of
5 Digitized Binary Pictures Using Local Features. *Computer Graphics and Image Processing*,
6 Vol. 4, No. 1, 1975, pp. 63 – 73.
- 7 [25] Aurenhammer, F., Voronoi diagrams—a survey of a fundamental geometric data structure.
8 *ACM Comput. Surv.*, Vol. 23, 1991, pp. 345–405.
- 9 [26] Conte, D., P. Foggia, C. Sansone, and M. Vento, Thirty years of graph matching in pattern
10 recognition. *International Journal of Pattern Recognition & Artificial Intelligence*, Vol. 18,
11 No. 3, 2004, pp. 265–298.
- 12 [27] Cormen, T., *Introduction to algorithms*. MIT electrical engineering and computer science
13 series, MIT Press, 2001.
- 14 [28] Bunke, H., On a relation between graph edit distance and maximum common subgraph. *Pat-*
15 *tern Recognition Letters*, Vol. 18, No. 8, 1997, pp. 689–694.
- 16 [29] Liu, B., *Web data mining: exploring hyperlinks, contents, and usage data*. Data-centric sys-
17 tems and applications, Springer, 2007.
- 18 [30] *OpenStreetMap*, <http://www.openstreetmap.org/>. Accessed July 22, 2011.
- 19 [31] Thiagarajan, A., L. Sivalingam, K. LaCurts, S. Toledo, J. Eriksson, S. Madden, and H. Bal-
20 akrishnan, VTrack: Accurate, Energy-Aware Road Traffic Delay Estimation Using Mobile
21 Phones. In *SenSys*, 2009.