Extracting line string features from GPS logs

Jörg Roth

Georg-Simon-Ohm-Hochschule Nürnberg
90489 Nürnberg
Germany
Joerg.Roth@Ohm-hochschule.de

Abstract

Geo data is an important foundation for any type of location-based service, but geo data often is expensive or distribution is limited by certain license restrictions. As a solution, open projects collect geo data from individuals and publish these data under a public licence. As a major drawback, the correction and integration of collected data is difficult and cost-intensive.
This paper describes an approach to automatically derive linear map data (roads, paths, highways etc.) from GPS logs. People just have to passively carry inexpensive GPS loggers whenever they drive or walk outdoors. The raw GPS logs then are automatically merged to a map. The approach contains error correction and sensor data fusion mechanisms. Our algorithm takes into account the specific measurement characteristic of GPS and is based on a probabilistic model. It performs two steps: first, track parts that represent the same paths are identified. Second, identifiable parts are fused to a single path, considering the Gaussian distribution of the input measurements.
We verified the approach with approx. 200 000 measurements in the area of Nuremberg that represents approx. 6 000 km driving distance. We show that an automatic processing of GPS logs to produce a road map is effectively possible.

1. Introduction

Geo data form the natural resource for location-based services. Geo objects describe the world in terms of natural, artificial and virtual entities that cover the Earth's surface. Important functions such as displaying maps, maintaining points of interests (POIs) and navigation strongly rely on the quality of geo data.

Official geo data often is expensive or protected by certain license agreements. As a solution, open projects collect geo data from individuals and publish these data under a public licence. For this, people measure path information with the satellite navigation system GPS using so-called *GPS loggers.* They periodically measure the current position (e.g. every 5s) that is stored in the persistant device memory. In projects such as *Open Street Maps* [OSM08], people can derive road information from GPS loggers that is integrated into a large geo database. As a major drawback, the integration of log data to maps is a manual, time-consuming task.

This paper describes an approach to *automatically* derive linear map data (roads, paths, highways etc.) from GPS logs. People just have to passively carry GPS loggers whenever they drive or walk outdoors. The raw GPS logs then are merged to a map that includes all collected data. The benefits of such an approach are:

- The collection of map data is simple and inexpensive. People just have to passively carry the GPS logger devices; further manual processing is not required. Thus, it is easier to get a higher number of contributors and even unpopular paths such as rarely used trails through forests can be detected.
- If people move on the same path multiple times, precision is improved by many measurements.
- Whereas existing geo data sources often represent a path as a single, bidirectional line, we can identify multiple lanes per road, at least different lanes for the two driving or walking directions.
- Additional statistics can be derived, e.g., the average driving speed at a certain position or statistics about driving directions. These statistics can be used for further evaluations e.g. to estimate positions of traffic lights or street sections with frequent traffic jams.

The approach does not detect the names of geo objects. Only line strings are extracted from the log data, not point-like or area-like geo objects.

2. Related Work

Related work that processes GPS logs tries to achieve three goals. First, logs can be used to detect special places that are visited more often. Second, logs can be used to get additional information about paths inside an existing map and third, logs can be used to derive line string features and build a new map from scratch. Even though, this paper presents an approach of the third kind, we briefly discuss related work of the first two types.

The problem to find special places (called *meaningful* places or *points of interest*) can be considered as a lower-dimensional variation of the line string feature extraction. Meaningful places are usually not modelled with their polygonal border but as centre points with a certain circular extension. Such points can easily be derived using clustering algorithms as proposed in [NK06] or Bayesian networks as presented in [YS07]. [JLO07] extends the clustering idea to identify moving objects.

Further approaches extend existing maps with the help of GPS logs. [JPR04] present a general framework to collect and store GPS log data to enrich existing maps, e.g. to store statistic about a car or the average speed, but no new roads are derived from the logs. [CJP05] proposed an approach to efficiently track moving objects. They take a predefined map and store the position history of an object into the given map. As a side effect, the road geometries of original maps are corrected according to the position log.

[RLW99] is one of the first approaches to derive line string features from GPS logs, but it requires an existing map that is extended. Note that the problem of modifying an existing map is much simpler because a general network of roads already exists. In [SWR+04] tracks are segmented with the help of points that are either derived from an existing map or computed using a clustering algorithm. This means, the problem to find line strings is reduced to the point clustering problem, but the line strings between the segment points are not adequately modelled. [MMB04] uses thresholds to identify corresponding linear path segments, but does not take into account the probabilistic distribution of measurements.

Discussion

Approaches that create line string maps have to face three problems as illustrated in fig. 1.

- *Line Clustering Problem*: Even though measurements may describe the same path, individual positions are not necessarily located close together. This is because the measurements are triggered using a fix clock (e.g. every 5s) or by movement (e.g. every 10m). Thus, a specific measurement randomly resides on the path line. A point clustering algorithm thus cannot be applied.
- *Segment Problem*: Users may drive on same roads, but whole tracks usually are not identical. The problem is to find out, which measurements of two tracks represent same paths and which not.

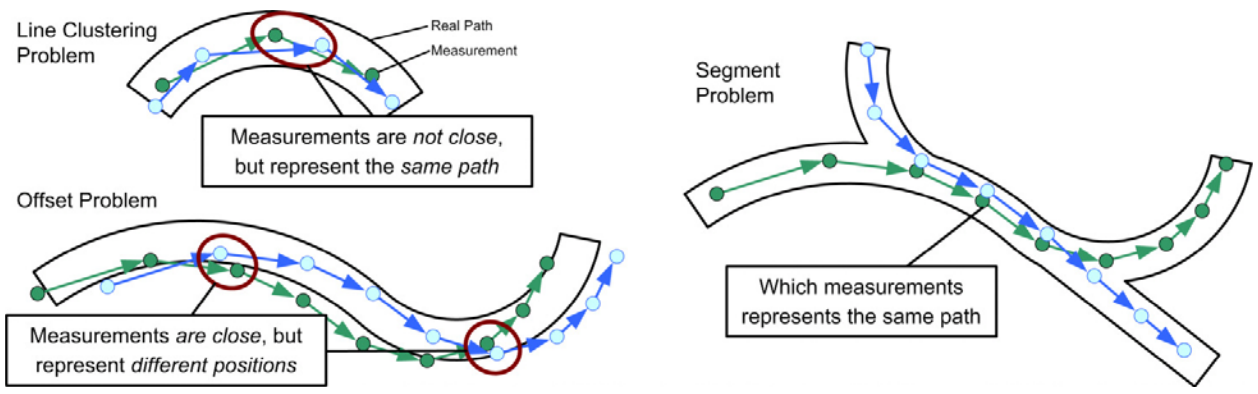
Line Clustering Problem
Real Path
Measurement
Measurements are *not close*, but represent the *same path*
Offset Problem
Measurements *are close*, but represent *different positions*
Segment Problem
Which measurements represents the same path

Fig. 1: Problems to fuse tracks

- *Offset Problem*: GPS logs have a high precision (i.e. subsequent measurements of the same position are close), but often have low accuracy (the difference of measured and real position often is high). If a GPS receiver receives a certain set of satellites, all measurements nearly have the same constant offset error and a track shape follows the actual road shape. Two corresponding tracks may contain similar positions that represent different path points. An approach thus has to explicitly assume a constant offset between multiple tracks and should not simply identify nearby measurements.

Many approaches consider the output of a GPS measurement as a true position without representing the error characteristic of a typical physical measurement. We strongly believe that a suitable approach has to model measurements as a probability distribution rather than a certain point in space.

3. Probabilistic Track Fusion (PTF)

We assume that GPS logs contain sequences of 2D positions and the respective time stamps. In addition, our approach has to know the variances in two dimensions for each measurement. They can be derived from information about the GPS measurement: the number of received satellites, whether *Differential GPS* was available and the so-called *Horizontal Dilution of Precision* that describes errors based the satellite constellation. If such values are not available, the standard variance of GPS is used, i.e., 78m^2 for 2dRMS(95%)=25m.
In a first step, the sequences of measurements are converted into a so-called *track model* that contains *coherent* sets of measurements:

- First, breaks are identified. Breaks are phases with zero speed for a certain time. The required time to define a break depends on the average speed nearby a break.
- Leading and trailing ends of a track (only some seconds) are removed as they often contain high measurement errors. This is because such measurement are often indoors and GPS has low accuracy before it completely fails or when it acquires a new set of satellites.
- The tracks are automatically classified according to their speed profile into classes such as pedestrian, bicycle, or car.

The *Probabilistic Track Fusion* (*PTF*) converts the track model into a *path model*. In contrast to the track model, the path model represents each real path only once. The algorithm runs in two steps:
Step 1: Identify those parts of different tracks that represent the same path.
Step 2: Fuse corresponding track parts to a single path using a probabilistic model.

The algorithm solves the problems described in fig. 1, especially the offset problem. It is an *online algorithm*, i.e. tracks are consecutively integrated into a final path model. The track ordering has no influence on the final result. Note that step 1 is based on a heuristic, whereas step 2 uses closed mathematically founded formulas.

Step 1: Identification of corresponding track parts
This step is based on histograms. Consider two tracks T_1, T_2 with measured positions P_{1i}, P_{2k}. Let $L_i(\alpha)$ denote the straight line through P_{1i} with angle α and $S_i(\alpha)$ the set of intersections of $L_i(\alpha)$ with the segments (P_{2k}, P_{2k+1}). Let further denote

$$d_i(\alpha) = \begin{cases} \min\left\{ \left|(P_{1i}, s)\right| \; \middle| \; s \in S_i(\alpha) \right\} & \text{if } S_i(\alpha) \neq \varnothing \\ \text{undef} \;\; \text{if } S_i(\alpha) = \varnothing \end{cases} \tag{1}$$

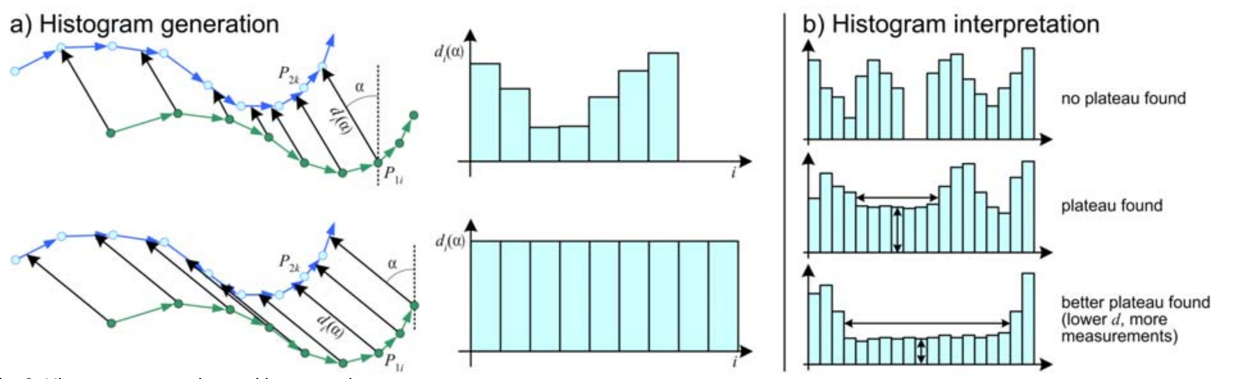

Fig. 2: Histogram generation and interpretation

A histogram for a certain angle α represents values of $d_i(\alpha)$ for all P_{1i} of T_1. Fig. 2a illustrates the histogram generation. Note that intersections in $S_i(\alpha)$ usually are not measured points P_{2k} of T_2 but are usually inside the segments (P_{2k}, P_{2k+1}). We thus build a *bidirectional* histogram that also measures the offset between points P_{2k} to segments (P_{1i}, P_{1i+1}) of T_1. For two tracks and for a set of values for α (e.g. every 10 degrees) the corresponding histogram undergoes an interpretation (fig. 2b). The goal is to find plateaus in the histogram, as they indicate a nearly constant distance between the two tracks under a certain angle α. This is typical for an offset error produced by GPS. Such plateaus can easily be detected – corresponding measurements both have a low average value d_{avg} and a low standard deviation d_{dev}. Note that according to the segment problem (fig. 1), a plateau only covers a subset $\{i_0,\dots\ i_1\}$ of all histogram entries. To identify the "best" plateau, we use the formula

$$v = \frac{i_1 - i_0}{d_{avg} + d_{dev}} \tag{2}$$

where higher values represent better plateaus. After the plateau with the highest v is found, the remaining track parts again undergo a plateau search, until no further plateaus are detected. After this we know for each pair of tracks:

- whether parts of the tracks represent the same path,
- the measurements of each track that can be identified,
- the offset vector between the corresponding parts of the tracks.

Once we know the corresponding parts, we now have to fuse these parts to a single path representation.

Step 2: Fusion of corresponding track parts

Given two track segments that represent the same path, what is a *single* sequence of positions that represents both track inputs? We model measurements as a triple (x, y, var), where x, y is the measured position and var the variance of the corresponding Gaussian distribution related to the measurement. Note, even though we actually deal with two-dimensional variances, the following formulas only contain one variance value for both axes for better readability.

Our approach converts two sequences of (x, y, var) from the two input tracks again to a sequence of (x, y, var) – the output path. This is a great benefit, as the result again can be used as an input track for further fusions without to switch to another representation. From the histogram approach above we combine corresponding parts of two tracks T_1, T_2 with measured positions P_{1i}, P_{2k} as illustrated in fig. 3.

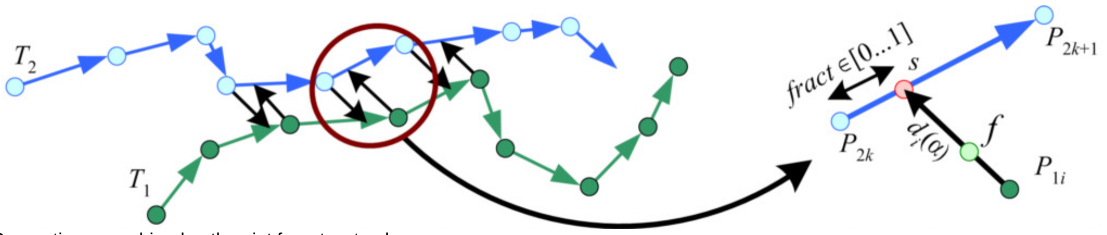

Fig. 3: Generating a combined path point from two tracks

Our model contains

- the intersection $s = (x_s, y_s, \mathrm{var}_s)$ of (P_{2k}, P_{2k+1}) and $L_i(\alpha)$ with distance $d_i(\alpha)$ to P_{1i}
- the fraction $fract = |(P_{2k}, s)| \,/\, |(P_{2k}, P_{2k+1})|$
- the position $f = (x_f, y_f, \mathrm{var}_f)$ that represents the combined path point after fusing the tracks T_1, T_2.

As $L_i(\alpha)$ is given by the histogram, the position (x_s, y_s) of s is known, but we still have to compute var_s. We use the formula

$$\mathrm{var}_S = \mathrm{var}_{P_{2k}} \cdot (1 - \mathit{fract}) + \mathrm{var}_{P_{2k+1}} \cdot \mathit{fract} \tag{3}$$

that represents the linear interpolation between the two measurement variances of P_{2k}, P_{2k+1}. We also could take into account that between two measurements we do not have definite knowledge about the positions, especially near the centre. We could reflect the missing knowledge by a higher variance, with e.g., the formula

$$\mathrm{var}_S = \left(\mathrm{var}_{P_{2k}} \cdot (1 - \mathit{fract}) + \mathrm{var}_{P_{2k+1}} \cdot \mathit{fract}\right) \cdot (2 - (2\,\mathit{fract} - 1)^2) \tag{4}$$

that adds the variances of P_{2k} and P_{2k+1} for fract = 0.5. Note that other estimations with similar characteristics are conceivable. Experiments show that formula (3) produces reasonable result, but an optimal formula is based on a motion model (see future work).
From $(x_s, y_s, \mathrm{var}_s)$ and $(x_{P1i}, y_{P1i}, \mathrm{var}_{P1i})$ we now can compute the combined position f using Bayes conditional probabilities applied to Gaussian distributions of GPS measurements. We get

$$f = \begin{pmatrix} x_f \\ y_f \\ \mathrm{var}_f \end{pmatrix} = \begin{pmatrix} \left(x_S\, \mathrm{var}_{P_{1i}} + x_{P_{1i}}\, \mathrm{var}_S\right) / \left(\mathrm{var}_{P_{1i}} + \mathrm{var}_S\right) \\ \left(y_S\, \mathrm{var}_{P_{1i}} + y_{P_{1i}}\, \mathrm{var}_S\right) / \left(\mathrm{var}_{P_{1i}} + \mathrm{var}_S\right) \\ \left(\mathrm{var}_{P_{1i}} \cdot \mathrm{var}_S\right) / \left(\mathrm{var}_{P_{1i}} + \mathrm{var}_S\right) \end{pmatrix} \tag{5}$$

Note that the variance of f is always lower than the lowest variance of the two input positions (regardless of their distance), thus the precision always increases for each step.
Until now we only fuse two *different* tracks, but also a *single* track may represent same paths multiple times. If, e.g. a driver lost the way, he may drive on the same road twice. Such a track has to be cut into two independent parts in order to use to approach above. For this, prior to checking two tracks, each track undergoes a self-overlap test, which also uses the histogram approach above: a histogram is generated that uses the same track for both inputs. To avoid the meaningless zero-plateau with all $d_i(\alpha)$=0, the two values of i_0 have to keep a certain distance, thus only real self-overlaps are detected.
Whenever two track segments are fused, the number of resulting positions is approximately the sum of measurements of both input segments. The number permanently would increase, if no additional mechanism was applied. Thus, after a fusion step, the result path undergoes a resampling step. It removes the second position of three consecutive positions that nearly reside on a straight line. As a result, the number of positions remains nearly constant.

4. Sample Execution and Performance Discussion

We verified our approach with the help of two sample data sets mainly collected in the area on Nuremberg. Table 1 presents the statistics.

Table 1: Sample Execution Statistics

	Urban Example		Regional Example	
	Input	**PTF Output**	**Input**	**PTF Output**
Number of Tracks	518	649	1 252	1 378
Number of Measurements	120 501	48 747	221 700	99 302
Measured Distance	2 146 km	251 km	5 899 km	1 435 km
Processing Time (@3.4 GHz)	9.5 hours		11.0 hours	

Fig. 4 illustrates the results of the urban example. Note that even though the number of measurements (and thus the measured distance) dramatically is reduced by the PTF fusion process, the number of tracks slightly increased. This is because long tracks that cover multiple roads are split up into short paths.
This sample execution shows that the PTF approach is able to automatically produce the required output. The path model now can easily be used for further processing, e.g. to create a topology map or to generate statistics about the respective roads.
Some remarks about the complexity: Processing time mainly depends on the number of tracks t and the average number of measurements per track m. If we consider the number of checked angles (and thus the number of histograms) as constant, we can generate histograms for pairs of tracks in $O(m^2)$ steps. This means $O(tm^2)$ steps are required for the self-overlap test and $O(t^2m^2)$ to fuse all tracks. If $n = tm$ denotes the total number of measurements, PTF requires $O(n^2)$ steps.

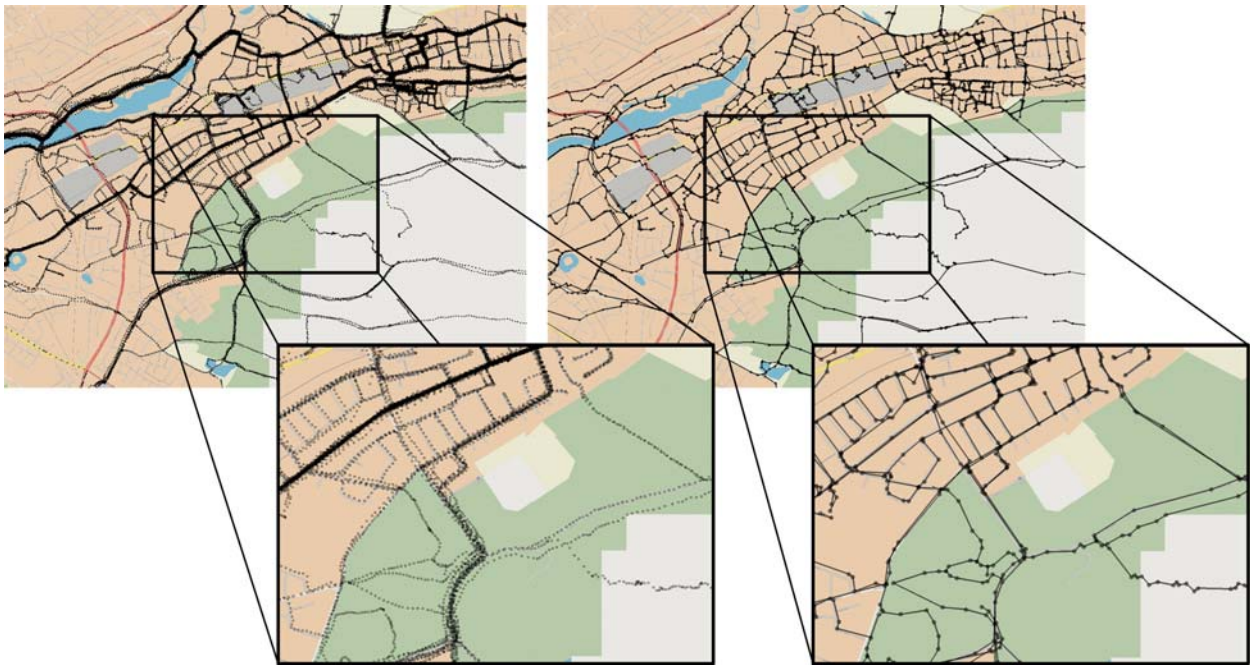
Fig. 4: Track Input (left) and generated Path Model (right)

As PTF does have to fulfil real time constraints, this complexity is acceptable. In addition, our implementation uses a spatial index to identify pairs of distant paths where the histogram generation is futile. This is why the regional example with widely spread paths does not require significant more time than the urban example.

5. Conclusion and Future Work

Our approach significantly simplifies the production of maps from GPS logs. It includes two steps: first, it creates and interprets histograms to identify of corresponding track parts and second, it combines paths with a probabilistic model that considers the error distribution of the GPS measurements. The approach is verified with the help of large sets of test data.

In the future we want to improve some parts of the approach. Especially the position and variance of the intersection point s is worth to be discussed. In reality, the trajectory between two measurements is not a straight line, but depends on velocity and acceleration of the moving object. We thus could model s much better and get better result for f, if we knew certain motion parameters such as the maximum acceleration. Thus, the next step is to derive a motion model from the given tracks that is integrated into PTF.

6. References

[CJP05] Civilis, A.; Jensen, C.; Pakalnis, S., Techniques for Efficient Road-Network-Based Tracking of Moving Objects, IEEE Transaction on Knowledge and Data Engineering, Vol. 17, No. 5, May 2005, 698-712

[JLO07] Jensen, C.; Lin, D.; Ooi, B., Continuous Clustering of Moving Objects, IEEE Transaction on Knowledge and Data Engineering, Vol. 19, No. 9, Sept 2007, 1161-1174

[JPR04] Jensen, C.; Lahrmann, H.; Pakalnis, S.; Runge, The infati data, J., TR-79, July 28, 2004, TIMECENTER Technical Report

[MMB04] Morris, S.; Morris; A.; Barnard, K.: Digital Trail Libraries, Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries, 2004, 63-71

[NK06] Nurmi, P.; Koolwaaij, J., Identifying meaningful locations, 3rd Annual International Conference on Mobile and Ubiquitous Systems - Workshops, 2006, 1-8

[OSM08] OpenStreetMap, http://www.openstreetmap.de/

[RLW99] Rogers, S.; Langley, P.; Wilson, C., Mining GPS Data to Augment Road Models, Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, 1999, 104-113

[SWR+04] Schroedl, S.; Wagstaff, K.; Rogers, S.; Langley, P.; Wilson, C., Mining GPS Traces for Map Refinement, Data Mining and Knowledge Discovery, Vol. 9, No. 1, July 2004, Springer-Verlag, 59-87

[YS07] Extracting Meaningful Contexts from Mobile Life Log, Youngseal, L., Sung-Bae C., Proc. of Intelligent Data Engineering and Automated Learning - IDEAL 2007, Springer LNCS 4881, 2007, 750-759