Probabilistic Modeling of Traffic Lanes from GPS Traces

Yihua Chen
Department of Electrical Engineering
University of Washington
Seattle, WA 98195, USA
yhchen@u.washington.edu

John Krumm
Microsoft Research
Redmond, WA 98052, USA
jckrumm@microsoft.com

ABSTRACT

Instead of traditional ways of creating road maps, an attractive alternative is to create a map based on GPS traces of regular drivers. One important aspect of this approach is to automatically compute the number and locations of driving lanes on a road. We introduce the idea of using a Gaussian mixture model (GMM) to model the distribution of GPS traces across multiple traffic lanes. The GMM naturally accounts for the inherent spread in GPS data. We present a new variation of the GMM that enforces constant lane width and GPS variance in each lane. For fitting the GMM, we also introduce a new regularizer that is sensitive to the overall spread of the GPS data across the road. Our experiments on real GPS data show that our new GMM is better at counting lanes than a more traditional GMM, and it gives more consistent results across our data set.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models—*statistical*

General Terms

Algorithms

Keywords

GPS, road map, Gaussian mixture model

1. INTRODUCTION

Digital road maps are growing in importance with the increasing popularity of Web-based maps and in-vehicle navigation systems. However, creating and maintaining these maps is expensive—companies rely on specially equipped vehicles with trained drivers traveling the roads to record street details. In addition to the expense, it is difficult for a method like this to keep up with changes in the roads due to construction or disasters.

An alternative to driving the roads is to use aerial imagery, represented by the early work of Tavakoli and Rosenfeld [15]. This approach is limited by the expense of taking enough images to keep up with road changes and the fact that still images are inadequate for determining dynamic aspects of the road network like driving direction, traffic controls, and turn restrictions.

Another way to infer the road network is to use GPS data recorded from regular vehicles as they drive normally. This is close to the approach taken by OpenStreetMap [11], which uses user-contributed GPS traces, aerial imagery, and other freely available data to create free digital maps that are open for editing from registered users. Likewise, WikiMapia [4], Google Maps [1], and TomTom's Map Share^{TM} [3] let users update maps. All of these efforts, including OpenStreetMap, require users to make their updates manually.

There has also been work on completely automated methods aimed at building maps from GPS traces. This technique appears to have started with the work by Rogers et al. in 1999 [13]. They used differential GPS (DGPS) traces to find the roads' centerline and then clustered the traces to find traffic lanes. In 2003, Edelkamp and Schrödl presented extensive work on creating and updating road maps with DGPS traces using a clustering technique to find the roads and lanes within the roads [9]. Schroedl et al. built on this work with constrained clustering that explolits domain knowledge [14]. Brüntrup et al. [6], Worrall & Nebot [17], and Cao & Krumm [7] also presented algorithms for clustering GPS traces into road representations.

One of the most important parts of turning GPS traces into a road map is to infer the roads' detailed lane structure. This is a prerequisite for instructing a driver which lane to use in preparation for a turn and for creating an accurate rendering of the road network. Updated knowledge of the lane structure is especially useful on roads undergoing construction, because construction often leads to frequent, temporary, and sometimes confusing changes in lanes. The lane structure, especially the number of lanes, can also be important for inferring the type of road (for example, arterial vs. neighborhood road) and for estimating traffic flow capacity.

This paper presents a new method for inferring the lane structure of roads from GPS traces. Specifically, we fit a Gaussian mixture model (GMM) to perpendicular cross sections of the traces across the road, based on the assumption that GPS traces will tend to cluster near the center of each

Figure 1: We tested our lane-finding algorithm on the GPS data from the three intersections shown above.

lane with some spread due to GPS noise and other vagaries. Using a GMM lets us invoke the considerable prior work on GMM fitting along with an exploration of GMM extensions customized to our particular problem. In addition, the GMM output gives a probability distribution of lane traffic, preserving and representing the inherent uncertainty in GPS traces that can be propagated to higher-level inference modules.

We tested our basic technique and its variations on GPS data collected from 55 shuttle vehicles driving around our institution. We describe this data in Section 2, followed by a detailed explanation of our approach in Section 3. The experimental results are shown in Section 4, and we conclude in Section 5 with a discussion of future extensions of this work.

2. GPS TRACES

Our GPS data came from a fleet of 55 shuttle vehicles that drive between the Microsoft corporate buildings in the Seattle area. We equipped each shuttle with a standard GPS logger: a RoyalTek RBT-2300 with a SiRF Star III GPS chipset and WAAS (Wide Area Augmentation System) enabled. Based on our previous work with these loggers, we estimate the standard deviation of the latitude/longitude measurements to be about 4 meters [12]. We configured the loggers to record a time stamped latitude/longitude measurement every second. Each vehicle was recorded for an average of 12.7 days, and in total we collected about 20 million GPS points.

For testing our lane-finding algorithm, we concentrated on roads around three intersections, shown in Fig. 1. We chose intersections for two reasons. First, the lane structure at intersections is often challenging with turn lanes appearing on the approach road and disappearing on the other side of the intersection. Second, for lane-level navigation, to correctly identify the lane structure is especially important around intersections since we need to infer the turn rules of each lane.

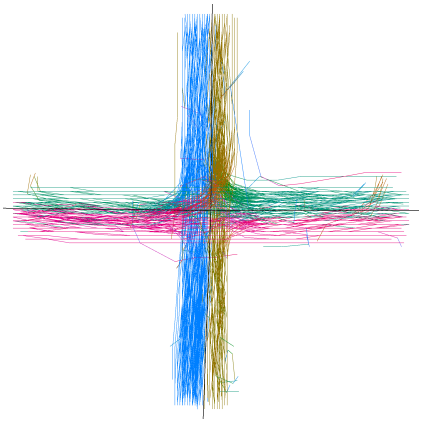
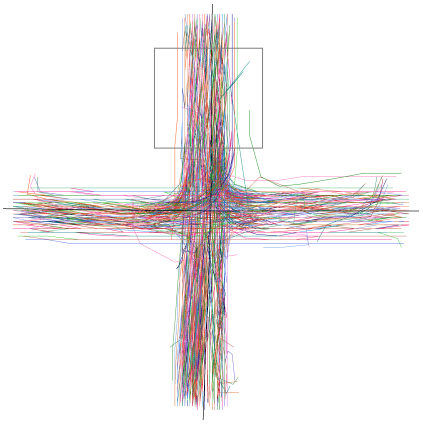


Figure 2: The GPS traces around Intersection 1 (doubly-circled in Fig. 1) are shown above; each line segment between two temporally adjacent GPS points is viewed as a vector and colored according to its direction.



(a)

(b)

Figure 3: The GPS traces around Intersection 1 (doubly-circled in Fig. 1) are shown in (a), where each separate trip is mapped to a single color, and (b) zooms in on the area in the gray square, which has 4 lanes approaching the intersection and 2 lanes departing from the intersection.

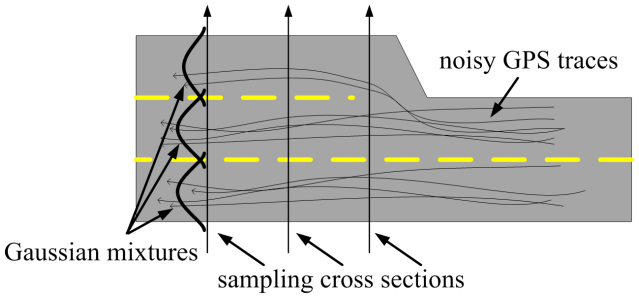


Figure 4: In order to find the lane structure, we fit a Gaussian mixture model (GMM) to the intersections between the GPS traces and a sampling line perpendicular to the road's centerline.

Due to the inherent inaccuracy of GPS, it is not easy to identify the lane structure from aggregated GPS traces. This can be seen from Fig. 2 and Fig. 3, which show the GPS traces around Intersection 1. Satellite images show that the area in the gray square has 6 lanes in total; however, as shown in Fig. 3(b), the traces from these 6 lanes are certainly not separated, and thus it is extremely difficult, if not impossible, to identify the lane structure by eye inspection. An agglomerative clustering method was used in [13] and [9] for finding lanes, while [14] added a constrained k-means clustering method [16]. Both methods are based on the assumption that GPS traces from different lanes are well separated. In our case, this assumption is seriously violated, and therefore we are motivated to use a probabilistic method to extract the lane structure from a mass of GPS traces such as this. In particular, we chose to model the GPS traces in each lane as a Gaussian distribution. The resulting mixture of Gaussians implicitly allows for poor separation of traces in different lanes, which is what we observed. Our algorithm automatically finds the best number of Gaussians and their means, which correspond to the number of lanes and the lane centers, respectively. We believe this is the first time that a GMM has been applied to the problem of finding lanes from GPS traces. The next section describes how we used the GMM, including a novel GMM formulation specifically derived for our problem.

3. LANE MODELING USING GMMS

3.1 General Approach

We make an independent estimate of the road's lane structure at given intervals along the road. We assume that we have an approximate centerline of the road, perhaps derived from the GPS data itself, as in [6], [7], [9], [13], [14], or [17]. At points with fixed intervals along the road's centerline, we construct sampling lines perpendicular to the centerline, as shown in Fig. 4. For each sampling line, we find the intersection points between the sampling line and the GPS traces, yielding a set of one-dimensional points along the sampling line. We expect these points to cluster near the centers of the lanes. However, these points will inevitably be spread, due to the GPS noise, placement of the GPS receiver in the vehicle, placement of the vehicle in the lane, and occasional lane changes. We assume that the spread within a lane can be modeled as a Gaussian distribution, and thus with each lane contributing one Gaussian, we can model the distribution of the intersection points as a weighted sum of Gaussian distributions, which is the familiar Gaussian mixture model (GMM). For one-dimensional data $x \in \mathbb{R}$, a GMM with k components has the following density:

$$p(x) = \sum_{j=1}^{k} w_j \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right). \quad (1)$$

The parameters in (1) are:

- k: the number of Gaussian components, one for each lane, giving an automatic lane count.
- $w_1, \ldots, w_k$: the weight of each component, corresponding to the relative traffic volume in each lane. The weights have to be positive and normalized, that is, $w_j > 0$, $j = 1, \ldots, k$, and $\sum_{j=1}^{k} w_j = 1$.
- $\mu_1, \ldots, \mu_k$: the mean of the traces for each component, giving the centerline of each lane. The μ_j's should be distinct so that each one corresponds to a different lane.
- $\sigma_1^2, \ldots, \sigma_k^2$: the variance of the traces for each component, giving the spread of the traces.

For each sampling line, we fit two GMMs, one for traffic in each direction. The following subsections describe various approaches to learn the GMM parameters from the data, some taking advantage of inherent constraints like constant lane widths and equivalent variances in each lane.

3.2 Learning the GMM Parameters without Constraints

The expectation-maximization (EM) algorithm [8] is often used to learn the parameters of a GMM with a fixed k. We will defer how to choose k, the number of lanes, to Section 3.4. Given a sample of n points $x_1, \ldots, x_n$, the EM algorithm seeks a maximum likelihood solution in an iterative fashion. Specifically, after choosing the initial estimates $w_j^{(0)}$, $\mu_j^{(0)}$, $\sigma_j^{(0)}$, $j = 1, \ldots, k$, the EM algorithm iterates between the following two steps until it converges:

1. **E-step:** Compute

$$\gamma_{ij}^{(m)} = \frac{w_j^{(m)} \phi(x_i \,|\, \mu_j^{(m)}, \sigma_j^{(m)})}{\sum_{l=1}^{k} w_l^{(m)} \phi(x_i \,|\, \mu_l^{(m)}, \sigma_l^{(m)})},$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, k$, where

$$\phi(x \,|\, \mu, \sigma) \triangleq \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

and compute

$$n_j^{(m)} = \sum_{i=1}^{n} \gamma_{ij}^{(m)},$$

for $j = 1, \ldots, k$.

2. **M-step:** Compute the new estimates

$$w_j^{(m+1)} = \frac{n_j^{(m)}}{n},$$

$$\mu_j^{(m+1)} = \frac{1}{n_j^{(m)}} \sum_{i=1}^{n} \gamma_{ij}^{(m)} x_i, \tag{2}$$

$$\sigma_j^{(m+1)} = \sqrt{\frac{1}{n_j^{(m)}} \sum_{i=1}^{n} \gamma_{ij}^{(m)} \left(x_i - \mu_j^{(m+1)}\right)^2}, \tag{3}$$

for $j = 1, \ldots, k$.

However, maximum likelihood estimation (MLE) in general is an ill-posed problem for GMM due to the fact that its likelihood function is not bounded above [5].[1] To avoid possible singularities or degeneracies, Fraley and Raftery proposed to replace MLE with maximum *a posteriori* (MAP) estimation [10]. Specifically, they assumed a uniform prior on the weights $w_1, \ldots, w_k$, and used an inverse gamma prior on the variances:

$$\sigma_j^2 \sim \text{Inv-Gamma}\left(\frac{\nu}{2}, \frac{\varsigma^2}{2}\right), \quad j = 1, \ldots, k,$$

that is, each σ_j^2 has density

$$p(\sigma_j^2 \,|\, \nu, \varsigma^2) = \frac{\left(\frac{\varsigma^2}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)} \left(\sigma_j^2\right)^{-\frac{\nu+2}{2}} \exp\left(-\frac{\varsigma^2}{2\sigma_j^2}\right),$$

where $\Gamma(\cdot)$ is the Gamma function, and ν and ς^2 are hyperparameters. The mean of this inverse gamma distribution is $\frac{\varsigma^2}{\nu-2}$. For the means, they used a normal prior conditioned on the variances:

$$\mu_j \,|\, \sigma_j^2 \sim \mathcal{N}\left(\eta, \frac{\sigma_j^2}{\kappa}\right), \quad j = 1, \ldots, k,$$

where η and κ are hyperparameters. The EM algorithm can be easily extended to MAP estimation: one only needs to change (2) and (3) to

$$\mu_j^{(m+1)} = \frac{\kappa\eta + \sum_{i=1}^{n} \gamma_{ij}^{(m)} x_i}{\kappa + n_j^{(m)}},$$

and

$$\sigma_j^{(m+1)} = \sqrt{\frac{\varsigma^2 + \kappa\left(\mu_j^{(m+1)} - \eta\right)^2 + \sum_{i=1}^{n} \gamma_{ij}^{(m)} \left(x_i - \mu_j^{(m+1)}\right)^2}{n_j^{(m)} + \nu + 3}},$$

respectively. The results in [10] show that with proper choice of the hyperparameters, the MAP estimation, which can be viewed as Bayesian regularization, is more robust than MLE.

3.3 Learning the GMM Parameters with Constraints

The methods discussed in Section 3.2 estimate the GMM parameters without any hard constraint; however, the underlying structure of our data implies certain restrictions on the model. First, we expect the widths of the lanes on the same sampling line to be approximately the same. This observation can be translated into the constraint that μ_j's are equally spaced, that is,

$$\mu_j = \mu + (j-1)\Delta\mu, \quad j = 1, \ldots, k, \tag{4}$$

where $\Delta\mu$ is the change between two adjacent μ_j's, and μ is the mean of either the leftmost or rightmost component along the sampling line, depending on the sign of $\Delta\mu$. Second, we expect the causes of the trace spread remain approximately the same among the lanes on the same sampling line, and therefore, we let all the Gaussian components share the same variance, that is,

$$\sigma_j^2 = \sigma^2, \quad j = 1, \ldots, k. \tag{5}$$

Substitute (4) and (5) into (1), and then for the restricted model, we obtain the following density:

$$p(x) = \sum_{j=1}^{k} w_j \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu - (j-1)\Delta\mu)^2}{2\sigma^2}\right). \tag{6}$$

Intuitively, (6) is a more reasonable description of our data, and for a fixed k, we reduce the model parameters from $\left\{w_i, \mu_i, \sigma_i^2\right\}_{i=1}^{k}$ to $w_1, \ldots, w_k$, μ, $\Delta\mu$, and σ^2, which also reduces the training time.

For robust estimation and for being able to incorporate prior knowledge, we consider MAP estimation of the parameters in (6), similar to what we presented for MAP estimation in Section 3.2. Specifically, we assume uniform priors on the weights $w_1, \ldots, w_k$ and μ,[2] respectively. For the shared variance σ^2, as in [10], we use an inverse gamma prior:

$$\sigma^2 \sim \text{Inv-Gamma}\left(\frac{\nu}{2}, \frac{\varsigma^2}{2}\right).$$

For $\Delta\mu$, we use a normal prior conditioned on σ^2:

$$\Delta\mu \,|\, \sigma^2 \sim \mathcal{N}\left(\eta, \frac{\sigma^2}{\kappa}\right).$$

The derivation of the EM algorithm for learning these parameters can be found in Appendix A.

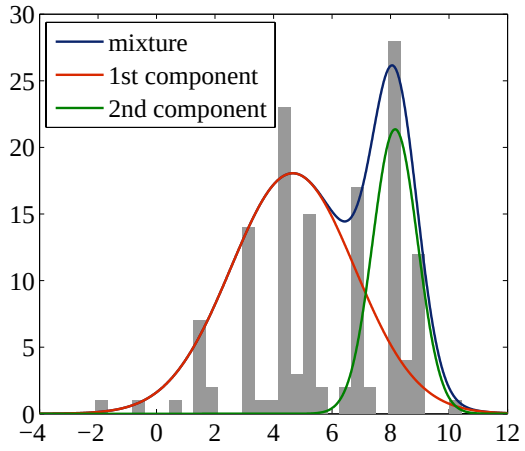
We show an example in Fig. 5 to illustrate the difference between the proposed restricted GMM and the generic GMM discussed in Section 3.2. In this example, we fit these two GMMs, both with $k = 2$, to a sample from two lanes, containing 137 data points. For the generic GMM shown in Fig. 5(a), the estimated parameters are $\hat{w}_1 = 0.7$, $\hat{w}_2 = 0.3$, $\hat{\mu}_1 = 4.7$, $\hat{\mu}_2 = 8.2$, $\hat{\sigma}_1^2 = 4.5$, and $\hat{\sigma}_2^2 = 0.6$. For the restricted GMM shown in Fig. 5(b), the estimated parameters are $\hat{w}_1 = 0.4$, $\hat{w}_2 = 0.6$, $\hat{\mu}_1 = 3.5$, $\hat{\mu}_2 = 7.5$, and $\hat{\sigma}_1^2 = \hat{\sigma}_2^2 = 2.1$. The restricted GMM matches better with our intuition about traffic lanes: the traffic is more balanced and the variances are equal. Moreover, training of the restricted GMM was much faster: the EM algorithm terminated after 17 iterations, while for training the generic GMM, the EM algorithm terminated after 136 iterations.

3.4 Model Selection

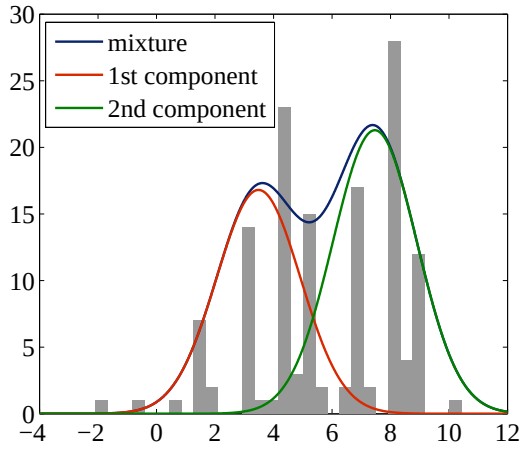
How to select the number of components for a GMM belongs to the topic of model selection. For us, this corresponds to

[1] To find a meaningful estimate of the parameters, we use the EM algorithm merely to search for a well-behaved local maximum of the likelihood function.

[2] The uniform prior on μ is an improper prior since $\mu \in \mathbb{R}$.



(a)



(b)

Figure 5: We trained a generic GMM and a restricted GMM, both with $k = 2$, on a sample from two lanes, containing 137 data points. The densities of these two GMMs and their individual components, multiplied by the number of data points, are shown in (a) and (b), respectively. In the background is the histogram of the sample. The parameters of the generic GMM were learned by MLE via EM. The parameters of the restricted GMM were learned by MAP via EM; for the hyperparameters, we chose $\nu = 3$, $\varsigma^2 = 4$, $\eta = 4$ and $\kappa = 100$.

automatically finding the number of lanes. Let $\mathcal{D}$ denote the observed data $\{x_i\}_{i=1}^n$, and let $\theta_k \triangleq \{w_i, \mu_i, \sigma_i^2\}_{i=1}^k$. A common practice is to estimate θ_k for a set of k's and then select the k that minimizes the following cost function:

$$-\frac{1}{n}\sum_{i=1}^{n}\log p(x_i \,|\, \theta_k) + \lambda R(\mathcal{D}, \theta_k). \tag{7}$$

The first term in (7) is the negative mean log-likelihood, which assesses how well the model fits the observed data. In the second term, $R(\mathcal{D}, \theta_k)$ is a regularization term that penalizes complex models, and $\lambda > 0$ is the regularization parameter. By minimizing (7), we seek a trade-off between model fitness and model complexity in order to achieve good generalization.

Akaike information criterion (AIC) and Bayesian information criterion (BIC) are two popular criteria for model selection [5]; they both have the form (7), with $\lambda = 1$ but different regularizers. For AIC, we have

$$R_{\mathrm{AIC}}(\mathcal{D}, \theta_k) = \frac{d}{n},$$

where d is the number of free parameters in the model. For the generic GMM, $d = 3k - 1$; for the restricted GMM, $d = k + 2$, $k \geq 2$. For BIC, we have

$$R_{\mathrm{BIC}}(\mathcal{D}, \theta_k) = \frac{\log n}{2n} d,$$

which leads to a heavier penalty for complex models than AIC.

We found through experimentation that AIC and BIC did not work well for our problem. However, we observe that the total spread of the data, which will be defined below, often correlates with the number of lanes on that sampling line. Based on this observation, we propose the following regularizer for our particular problem:

$$R_{\mathrm{LS}}(\mathcal{D}, \theta_k) = \left(\frac{S(\mathcal{D})}{k} - \delta\right)^2,$$

where $S(\mathcal{D})$ is the total spread of the data, and δ is a constant equal to the expected data spread on a single lane, termed *lane spread* (LS). We calculate $S(\mathcal{D})$ as follows: first, we sort all the data points in ascending order according to their distances from the median; then we choose the first 95% of the sorted data points, and $S(\mathcal{D})$ is the difference between the maximum and minimum of that 95%. These two steps are designed to avoid outliers.

4. EXPERIMENTAL RESULTS

For lane modeling, we compared the generic GMM, trained using both MLE and MAP, and the proposed restricted GMM on GPS data from the roads around the three intersections shown in Fig. 1. For each intersection, we collected traces in a circle with radius 120 meters centered on the intersection. We generated a sampling line every 5 meters along the roads' centerline,[3] and each sampling line includes two samples,[4] one from traffic in each direction. The number of data points in each sample ranges from 30 to 300. As to the hyperparameters in MAP estimation for the generic GMM, we chose $\nu = 3$ as recommended in [10], $\varsigma^2 = 4$ as the expected value of σ_i^2, $\eta = \frac{1}{n}\sum_{i=1}^n x_i$ as the expected value of μ_i, and $\kappa = 0.01$ for a flat prior on $\mu_i \,|\, \sigma_i^2$; for the restricted GMM, we chose the same ν and ς^2, but let $\eta = 4$ as the expected value of $\Delta\mu$, and $\kappa = 100$ for a concentrated prior on $\Delta\mu \,|\, \sigma^2$.

Although we do not have true probabilistic models of GPS traces to compare to, we evaluated the three GMM fitting methods on two related tasks. The first is lane num-

[3]We got the centerlines from Microsoft Bing Maps [2] through its API.

[4]By "sample," we mean a set of one-dimensional data points where the GPS traces pierce the sampling line across multiple lanes.

ber identification,[5] or more precisely, to identify the correct k from the set $\{1, 2, \ldots, 5\}$. We compared two regularizers, $R_{\mathrm{AIC}}(\mathcal{D}, \theta_k)$ and $R_{\mathrm{LS}}(\mathcal{D}, \theta_k)$, for model selection. As a heuristic, we set the lane spread parameter $\delta = 5$ for the proposed regularizer $R_{\mathrm{LS}}(\mathcal{D}, \theta_k)$. We did not fix the regularization parameter λ; instead, we chose λ by cross-validation, and it was cross-validated from the following set:

$$\lambda \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}.$$

We randomly selected 80% of the data for cross-validating λ and used the remaining 20% for testing. The process was repeated for 20 random partitions of the data. Table 1 shows the average error rates along with the standard deviations over the 20 runs on all the samples and samples from Intersection 3 only. The total number of samples is 270, and the number of samples from Intersection 3 only is 78.

In the second experiment, we fit GMMs to each sample with k equal to the true number of lanes, and then calculated over all the samples the average distance between adjacent μ_j's and the average standard deviation of each Gaussian component. The results are also shown in Table 1.

From Table 1, we can see that the proposed regularizer $R_{\mathrm{LS}}(\mathcal{D}, \theta_k)$ performs much better than $R_{\mathrm{AIC}}(\mathcal{D}, \theta_k)$ in lane number identification, and when $R_{\mathrm{LS}}(\mathcal{D}, \theta_k)$ is used, the error rates of the three GMM fitting methods on all the samples are about the same with the proposed restricted GMM slightly better than the other two. However, also shown in Table 1 is that regardless of using MLE or MAP, the $\Delta\mu$ and σ of the generic GMM have very high variance, which fails to reflect the underlying structure of our GPS data, whereas the restricted GMM yields consistent estimates across components. Moreover, when focusing on lane number identification, the generic GMM can overfit the data. For example, on the samples from Intersection 3 only, the generic GMM using MLE achieves much lower error rate than the other two, but its $\Delta\mu$ still exhibits very high variance, which casts serious doubt on the quality of the model.

In summary, our regularizer performed much better than the standard AIC regularizer for counting the number of lanes, and our restricted GMM produces much more consistent results across the entire data set.

5. CONCLUSIONS AND FUTURE WORK

We have described a new approach to inferring the lane structure of roads from GPS data. Our technique fits a mixture of Gaussians to GPS traces, with one Gaussian for each lane. We derived a new way to fit the GMM that enforces constant lane width and constant GPS variance from lane to lane. We also introduced a new regularization term that is sensitive to the spread of the data across the road. Our experiments on real GPS data show that our new formulation is better at counting lanes and also gives more consistent results across our data set.

In addition to its ability to count lanes, our probabilistic approach naturally models the inherent noise in GPS data. In future work, preserving the resulting uncertainty will be important as models like these can be extended beyond one-dimensional slices across the road to full two-dimensional models that extend across and along the road. We envision an approach that extracts the continuous lane structure along the road, trading off localized uncertainty in across-the-road lane counts with probabilistic constraints on how lanes are added and subtracted along the road relatively infrequently. One possible direction to extend this work is to build a hidden Markov model (HMM) for each road segment connecting two intersections. Such HMM will model the sequence of observed intersection points on all the sampling lines in one road segment. Each (hidden) state is the number of lanes on the corresponding sampling line, the proposed restricted GMM can be used to model the observed intersection points given the state, and learning the transition probabilities between the states will naturally take into account the fact that the number of lanes on adjacent sampling lines is not likely to change abruptly.

[5]We obtained the ground truth by examining the satellite images.

6. REFERENCES

[1] Google Maps. `http://maps.google.com/`.

[2] Microsoft Bing Maps. `http://www.bing.com/maps/`.

[3] TomTom Map Share™. `http://www.tomtom.com/page/mapshare`.

[4] WikiMapia. `http://wikimapia.org/`.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, 2006.

[6] R. Brüntrup, S. Edelkamp, S. Jabbar, and B. Scholz. Incremental map generation with GPS traces. In *Proceedings of the 8th IEEE International Conference on Intelligent Transportation Systems*, pages 574–579, 2005.

[7] L. Cao 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*, pages 3–12, 2009.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–38, 1977.

[9] S. Edelkamp and S. Schrödl. Route planning and map inference with global positioning traces. In *Computer Science in Perspective: Essays Dedicated to Thomas Ottmann*, Lecture Notes in Computer Science, pages 128–151. Springer, 2003.

[10] C. Fraley and A. E. Raftery. Bayesian regularization for normal mixture estimation and model-based clustering. *Journal of Classification*, 24(2):155–181, 2007.

[11] M. Haklay and P. Weber. OpenStreetMap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008.

[12] 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, 2009.

[13] S. Rogers, P. Langley, and C. Wilson. Mining GPS data to augment road models. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discoverty and Data Mining*, pages 104–113, 1999.

Table 1: Lane number identification error (%) for $R_{\mathrm{AIC}}(\mathcal{D}, \theta_k)$ and $R_{\mathrm{LS}}(\mathcal{D}, \theta_k)$ on the test set averaged over 20 random partitions. Also shown are the average distance between adjacent μ_j's and the average standard deviation of each Gaussian component. In the parentheses are the corresponding standard deviations.

Intersection 1, 2 & 3	E_{AIC}		E_{LS}		$\Delta\mu$		σ	
Generic GMM (MLE)	72.41	(4.73)	31.39	(5.62)	5.34	(5.62)	2.96	(2.53)
Generic GMM (MAP)	74.54	(5.16)	30.19	(5.54)	4.73	(4.67)	2.33	(1.67)
Restricted GMM	83.15	(2.40)	28.06	(4.39)	4.11	(0.16)	2.82	(0.78)
Intersection 3 only	E_{AIC}		E_{LS}		$\Delta\mu$		σ	
Generic GMM (MLE)	54.38	(9.53)	13.12	(6.38)	5.33	(5.16)	1.56	(0.86)
Generic GMM (MAP)	64.06	(9.48)	20.62	(7.61)	4.64	(2.61)	1.55	(0.96)
Restricted GMM	50.62	(10.32)	20.00	(7.48)	4.06	(0.07)	2.01	(0.68)

[14] S. Schroedl, K. Wagstaff, S. Rogers, P. Langley, and C. Wilson. Mining GPS traces for map refinement. *Data Mining and Knowledge Discovery*, 9(1):59–87, 2004.

[15] M. Tavakoli and A. Rosenfeld. Building and road extraction from aerial photographs. *IEEE Transactions on Systems, Man and Cybernetics*, 12(1):84–91, 1982.

[16] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of the 18th International Conference on Machine Learning*, pages 577–584, 2001.

[17] S. Worrall and E. Nebot. Automated process for generating digitised maps through GPS data compression. In *Proceedings of the 2007 Australasian Conference on Robotics & Automation*, 2007.

APPENDIX

A. EM FOR THE RESTRICTED GMM

For the restricted GMM with k components, the parameters to be estimated are $\theta \triangleq (w_1, \ldots, w_k, \mu, \Delta\mu, \sigma^2)$, and we denote the estimate at the mth iteration by $\theta^{(m)}$. Let

$$\mu_j^{(m)} = \mu^{(m)} + (j-1)\Delta\mu^{(m)}, \quad j = 1, \ldots, k.$$

For the E-step, we first compute

$$\gamma_{ij}^{(m)} = \frac{w_j^{(m)} \phi(x_i \,|\, \mu_j^{(m)}, \sigma^{(m)})}{\sum_{l=1}^{k} w_l^{(m)} \phi(x_i \,|\, \mu_l^{(m)}, \sigma^{(m)})},$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, k$, and next compute

$$n_j^{(m)} = \sum_{i=1}^{n} \gamma_{ij}^{(m)},$$

for $j = 1, \ldots, k$. Then we can derive the Q-function as follows,

$$\begin{aligned} Q(\theta \,|\, \theta^{(m)}) &= \sum_{i=1}^{n}\sum_{j=1}^{k} \gamma_{ij}^{(m)} \log\left(w_j \phi(x_i \,|\, \mu + (j-1)\Delta\mu, \sigma)\right) \\ &= \sum_{j=1}^{k} n_j^{(m)} \log w_j - \frac{n}{2}\log 2\pi - \frac{n}{2}\log\sigma^2 \\ &\quad - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\sum_{j=1}^{k} \gamma_{ij}^{(m)} \left(x_i - \mu - (j-1)\Delta\mu\right)^2. \end{aligned}$$

For MAP estimation, the M-step solves

$$\theta^{(m+1)} = \arg\max_{\theta} \Big(Q(\theta \,|\, \theta^{(m)}) + \log p(\theta) \Big), \tag{8}$$

where $p(\theta)$ is the prior of the parameters. For the prior proposed in Section 3.3, we have

$$p(\theta) \propto \left(\sigma^2\right)^{-\frac{\nu+3}{2}} \exp\left(-\frac{\varsigma^2 + \kappa(\Delta\mu - \eta)^2}{2\sigma^2}\right),$$

and thus

$$\begin{aligned} &Q(\theta \,|\, \theta^{(m)}) + \log p(\theta) \\ &\quad = \sum_{j=1}^{k} n_j^{(m)} \log w_j - \frac{n+\nu+3}{2}\log\sigma^2 - \frac{\varsigma^2 + \kappa(\Delta\mu - \eta)^2}{2\sigma^2} \\ &\qquad - \frac{1}{2\sigma^2}\sum_{i=1}^{n}\sum_{j=1}^{k} \gamma_{ij}^{(m)} \left(x_i - \mu - (j-1)\Delta\mu\right)^2 + C, \end{aligned}$$

where C is a constant that does not depend on θ. To solve (8) for the weights under the constraint $\sum_{j=1}^{k} w_j = 1$, we can use the Lagrange multiplier and easily get

$$w_j^{(m+1)} = \frac{n_j^{(m)}}{n}, \quad j = 1, \ldots, k.$$

To solve (8) for μ and $\Delta\mu$, we let

$$\frac{\partial}{\partial\mu}\Big(Q(\theta \,|\, \theta^{(m)}) + \log p(\theta) \Big) = 0, \tag{9}$$

and

$$\frac{\partial}{\partial\Delta\mu}\Big(Q(\theta \,|\, \theta^{(m)}) + \log p(\theta) \Big) = 0. \tag{10}$$

By combining (9) and (10), we obtain the following system of linear equations

$$A \begin{bmatrix} \mu \\ \Delta\mu \end{bmatrix} = b, \tag{11}$$

where $A = \begin{bmatrix} a_{ij} \end{bmatrix}_{2\times 2}$ with

$$\begin{aligned} a_{11} &= 1, \\ a_{12} &= a_{21} = \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j, \\ a_{22} &= \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j^2 + \frac{\kappa}{n}, \end{aligned}$$

and $b = \begin{bmatrix} b_1 & b_2 \end{bmatrix}^T$ with

$$\begin{aligned} b_1 &= \frac{1}{n} \sum_{i=1}^{n} x_i, \\ b_2 &= \frac{\kappa\eta}{n} + \frac{1}{n} \sum_{i=1}^{n} \sum_{j=2}^{k} \gamma_{ij}^{(m)} (j-1) x_i. \end{aligned}$$

We note that

$$\begin{aligned} \left(\sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j \right)^2 &= \left(\sum_{j=1}^{k-1} \sqrt{w_{j+1}^{(m+1)}} \sqrt{w_{j+1}^{(m+1)}} j^2 \right)^2 \\ &\overset{(a)}{\leq} \left(\sum_{j=1}^{k-1} w_{j+1}^{(m+1)} \right) \left(\sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j^2 \right) \\ &\overset{(b)}{\leq} \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j^2 \\ &\overset{(c)}{<} \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j^2 + \frac{\kappa}{n}, \end{aligned}$$

where (a) follows from the Cauchy-Schwarz inequality, (b) follows from $0 \leq \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} \leq 1$, and (c) follows from $\kappa > 0$. Therefore, we have

$$\begin{aligned} \det A &= a_{11} a_{22} - a_{12} a_{21} \\ &= \sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j^2 + \frac{\kappa}{n} - \left(\sum_{j=1}^{k-1} w_{j+1}^{(m+1)} j \right)^2 \\ &> 0, \end{aligned}$$

which indicates that (11) has a unique solution, and the solution gives us the new estimates

$$\mu^{(m+1)} = \frac{a_{22} b_1 - a_{12} b_2}{\det A},$$

and

$$\Delta\mu^{(m+1)} = \frac{a_{11} b_2 - a_{21} b_1}{\det A}.$$

Last, we let

$$\frac{\partial}{\partial \sigma^2} \left(Q(\theta \,|\, \theta^{(m)}) + \log p(\theta) \right) = 0,$$

and get

$$\sigma^{(m+1)} = \sqrt{\frac{\varsigma^2 + \kappa \left(\Delta\mu^{(m+1)} - \eta \right)^2 + \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij}^{(m)} \left(x_i - \mu_j^{(m+1)} \right)^2}{n + \nu + 3}}.$$

This ends the derivation of the EM algorithm for the proposed restricted GMM.