Online Map Matching With Route Prediction

Shun Taguchi, Satoshi Koide, and Takayoshi Yoshimura

***Abstract*—Map matching is a procedure that estimates the route traveled by vehicles or people by using observed coordinates. It is an important preprocessing procedure for location services based on global positioning system (GPS) data obtained from probe vehicles. One recently proposed major map matching approach is the hidden Markov model (HMM)-based method. However, HMM-based approaches suffer from latency, because they rely on the availability of future GPS points. This latency limits the ability of real-time traffic sensing and location services. This paper presents a novel online map matching algorithm that uses a probabilistic route prediction model instead of future GPS points. The probabilistic route prediction model can be trained by using historical trajectory data. Our experimental results show that the accuracy of the untrained proposed model is competitive with a naïve online HMM-based method without any latency. Moreover, the results show that the trained model obtains even higher accuracy. The experimental results also show that the proposed method is faster than the online HMM.**

***Index Terms*—Map matching, route prediction, Bayesian filtering, multiple hypothesis technique**

I. Introduction

MAP MATCHING is a procedure that estimates the route traveled by vehicles or people from their observed coordinates. It is an important preprocessing procedure for location services based on global positioning system (GPS) data and is becoming more popular as wireless communication spreads. In transport systems, map matching has long been studied for in-vehicle navigation systems, which must be able to determine on which road a vehicle is traveling in real time. Recently, however, map matching has become important not only for navigation systems, but also for various location-based services such as traffic sensing [1], [2], fleet management [3], and route recommendation [4], [5]. The main difference between recent and traditional map matching is the sparseness of GPS trajectories. Traditional map matching assumes in-vehicle processing, and the GPS data are obtained directly from sensors, meaning that the sampling rate is high (e.g., one sample per 5 s). By contrast, recent map matching assumes server-side processing, and therefore the GPS data are obtained through wireless communication. The sampling rate for recent map matching is low (e.g., one sample per minute), because the dense trajectories are costly with respect to live transmission and storage. These properties make map matching surprisingly difficult.

Manuscript received June 6, 2016; revised February 14, 2017, July 17, 2017, and December 25, 2017; accepted February 25, 2018. Date of publication March 19, 2018; date of current version December 21, 2018. The Associate Editor for this paper was Z. Ding. *(Corresponding author: Shun Taguchi.)*

The authors are with Toyota Central R&D Labs., Inc., Nagakute 480-1192, Japan (e-mail: s-taguchi@mosk.tytlabs.co.jp; koide@mosk.tytlabs.co.jp; yoshimura@mosk.tytlabs.co.jp).

Digital Object Identifier 10.1109/TITS.2018.2812147

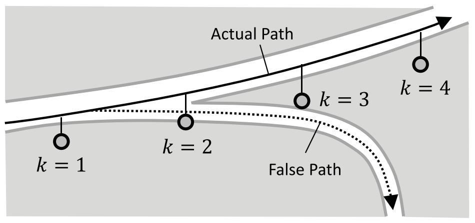

Fig. 1. Example situation that is difficult for online map matching. The gray points represent the obtained GPS points and k denotes the time step of the GPS data.

The main difficulty of map matching is caused by the tradeoff between choosing a route that is close to the GPS points and a route that is more likely to be traveled. Simple methods such as nearest neighbor matching often result in winding routes that are unlikely to be traveled. On the contrary, recent major map matching systems use hidden Markov model (HMM)-based approaches [6]–[13]. In these methods, a latent state sequence corresponds to the route and the observed variables correspond to the GPS trajectory. HMM-based methods estimate a route that maximizes the likelihood that contains both a route transition probability and an observation probability. These HMM-based approaches thus work well for *offline* map matching.

Map matching algorithms are divided into two types: *offline* and *online* algorithms. Offline algorithms batch process the entire input trajectory before generating the solution. Most HMM-based map matchings have been proposed as offline algorithms [7]–[11]. By contrast, online algorithms estimate the current road segment immediately a GPS data point is obtained. Online map matching is more difficult than offline map matching, and this difficulty arises from the unavailability of future GPS points. Fig. 1 shows a situation that becomes difficult for online map matching. In this case, offline map matching can estimate the correct route by using the GPS data at $k = 4$. However, online map matching at time steps $k = 2, 3$ is difficult because there is no information with which to confirm the route except the distance from the GPS point, which leads to a false path. Despite these difficulties, several location services require online map matching, such as real-time traffic sensing and services that use real-time traffic information. Recently, several researchers have extended HMM-based map matching to online algorithms [12], [13]. However, these HMM-based online approaches also suffer from time step latency because the method requires time to obtain the future GPS points used to indicate the final route. To remove this latency, we present a novel online map matching algorithm using route prediction instead of future GPS points.

The main idea of the proposed method is to replace future GPS points with a probabilistic route prediction model. The parameters of the probabilistic route prediction model are estimated from historical GPS trajectory data. Map matching based on route prediction can be implemented as a Bayesian filter [14]. The route prediction model obtains the prior probabilities of the routes and then the posterior probability is calculated from the priors and observed GPS data points by Bayesian filtering. Bayesian filtering is often used in map matching methods with data that have a high sampling rate [15]–[21]. However, these methods employ a vehicle dynamical model that does not work for data that have a low sampling rate (e.g., over 5 s). By contrast, the proposed method employs a route prediction model that can estimate the discrete route sequence even if the sampling rate is low. Therefore, it is robust against the sparseness of GPS trajectories.

The main contribution of this study is the application of a probabilistic route prediction model to the online map matching problem. This model significantly improves the accuracy of online map matching without any latency. The experimental results presented herein show that the proposed method has a similar accuracy to an online HMM method based on [8], even without training the parameters of the route prediction model. Furthermore, the simulation results show that parameter training significantly improves accuracy without any latency. These results indicate that better route prediction improves online map matching. The experimental results also show that the proposed method is faster than the online HMM method.

The remainder of this paper is organized as follows. In Section II, we review work related to this study. In Section III, we introduce the models used in the proposed map matching: the route prediction model and observation model. In Section IV, we present the implementation of the proposed map matching algorithm. In addition, we explain Bayesian filtering and a computation technique for the route prediction model. The experimental results are presented in Section V. In Section VI, we discuss about computational complexity of proposed method. Finally, we conclude the paper and discuss directions for future map matching research.

II. Related Work

A. Map Matching Before HMM

Initially, deterministic approaches (e.g., geometric or topological approaches) were proposed for map matching. A review of such algorithms can be found in [15]. A geometric map matching algorithm uses the geometric information of spatial road network data by considering only the shape of the segments; for example, point-to-point matching, point-to-curve matching, and curve-to-curve matching [22] are geometric approaches. As more advanced type of deterministic approach, topological approaches, which are based on the topology of networks [23] or the Frèchet distance [24], have been proposed. However, both deterministic approaches are sensitive to noise.

To achieve robustness, probabilistic map matching approaches became more popular as probabilistic positioning techniques based on Bayesian filtering [25] became more prevalent. Lamb and Thiebaux [16] proposed a method using the multiple hypothesis technique (MHT), which combines the Kalman filter and Markov model to eliminate movements between unconnected road segments. The MHT, which is also employed by our method, has been continuously studied for integrated in-vehicle positioning and map matching methods [17]–[21]. The concept presented by [16] is similar to our approach, but it is not robust to the sparseness of GPS trajectories. The idea of modeling road segment transitions was introduced into HMM-based map matching by Hummel [6]. Our approach is thus a different development from the HMM-based approaches based on [16].

B. HMM-Based Map Matching

The HMM-based approach was first proposed in [6], who introduced the idea of leveraging the transition probability of the Markov model in [16]. However, the naïve definition of the transition probability in [6] does not work well when the sampling interval is long, because this approach does not consider the transitions between road segments that are not directly connected but can be traveled during the sampling interval. Krumm *et al.* [7] used a probabilistic travel time constraint instead of pure topology. Newson and Krumm [8] and Lou *et al.* [9] introduced projection points as latent states instead of road segments, and took into account the corresponding travel distance. Despite its simplicity, the resulting HMM of [8] is surprisingly robust to noise and sparseness for offline map matching.

Recently, several researchers have extended the HMM for online map matching [12], [13]. Goh *et al.* [12] proposed a *variable sliding window* method to find the optimal solution to the tradeoff between accuracy and latency. However, this approach also had a latency of about 100 s. Jagadeesh and Srikanthan [13] proposed a *heuristic* 1*-lag* method that ignores dissimilar routes. Although this method is faster than that of [12] because it removes dissimilar candidates, it does not remove all of the time step latency.

As another recent trend, several researchers have introduced route features (e.g., number of signals, turns) into the HMM to improve accuracy [26], [27]. In these works, the weights of the route features are determined by experiments. Machine learning can help to determine the weights from datasets. In the next subsection, we review such machine learning based methods.

C. Machine Learning for Map Matching

To improve the accuracy of map matching, some research has introduced machine learning methods [11], [12], [28], [35]. For example, Goh *et al.* [12] proposed a method based on a support vector machine. Hunter *et al.* [28] proposed a method based on conditional random fields [29]. For offline methods, Osogami and Raymond [11] also proposed an inverse reinforcement learning approach [30]. These methods introduced several route features into the HMM and improved accuracy by deciding the weight by using machine learning approaches.

By contrast, the proposed method improves accuracy by learning a prediction model of road segment transitions. Unlike offline map matching, the online situation cannot look forward toward the GPS coordinates in the future, which makes the problem difficult. Our idea is to improve accuracy by using the prediction model instead of such unavailable future GPS coordinates. Moreover, this method does not require additional information on the map unlike feature-based methods. As discussed in [31] and [32], a simple map contributes to efficient map matching thanks to fast map loading. We therefore show the experimental results to compare with Hunter *et al.*'s [28] feature-based method in Section V-B.

Our proposed method learned the prediction model by the popularity of the road segment transition. A method based on the popularity of the route has already proposed by [35] and its effectiveness has been indicated. However, this method has only been applied to offline map matching. We instead present an online map matching method based on the occurrence frequency of road segment transitions.

III. Modeling for Map Matching

A. Problem Definition

Definition 1: A *trajectory* $g_{1:K}$ is a sequence of the K GPS data points collected by a vehicle. Each GPS data point g_k is a 4-tuple (g_k.t, g_k.lat, g_k.lng, g_k.v), indicating the time stamp, latitude, longitude, and velocity, respectively. In this study, it is assumed that velocities can be obtained from the vehicle. If a situation occurs in which velocities cannot be obtained, the values calculated from the differential of the GPS coordinates can be used.

Definition 2: A *road network* is a directed graph $G(V, E)$, where V is a set of vertices and E is a set of edges representing road segments. Each road segment $e \in E$ is a 4-tuple (e.id, e.l, e.start, e.end) consisting of its ID, length, start point, and end point, where $e.\text{start} \in V$ and $e.\text{end} \in V$.

Definition 3: A *route* $r_{1:K}$ is a sequence of K road segments corresponding to given trajectory $g_{1:K}$. Each road segment $r_k \in E$ represents the road segment on which the vehicle is traveling at time step k.

Problem Definition: The online map matching problem is defined as follows: *Given the road network $G(V, E)$, find a road segment r_k at every time step $1 \le k \le K$ using only past GPS data points $\{g_j | 1 \le j \le k\}$.*

B. Route Prediction Model

In the proposed method, candidates of r_k is predicted from previous route r_{k-1} by using a route prediction model, then r_k is updated and determined by using a observation model with a current observation g_k. In this section, we explain the prediction model and observation model for the proposed map-matching method.

This study proposes a route prediction model based on a Markov model that describes route transition probability $p(r_{k+1}|r_k)$. This transition probability can be transformed into

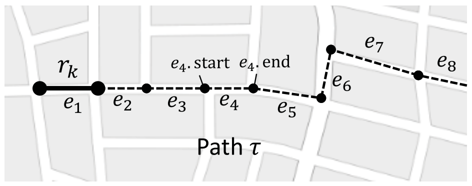

Fig. 2. Path τ is a path starting from r_k of infinite length, which is a sequence of connected road segments that starts at r_k, i.e., $\tau = \{e_1 \to e_2 \to \cdots\}$, where $e_1 = r_k$, e_i.end $= e_{i+1}$.start, $1 \le i < \infty$.

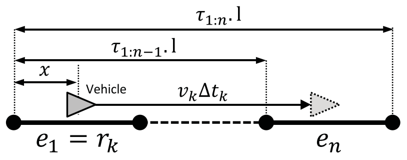

Fig. 3. Physical analysis of the transition of road segments during $[t_k, t_k + \Delta t_k]$. If movement distance $v_k \Delta t_k$ is between $\tau_{1:n-1}.\text{l} - x$ and $\tau_{1:n}.\text{l} - x$, a transition from r_k to $r_{k+1} = e_n$ occurs.

the following equation:

$$p(r_{k+1}|r_k) = \sum_{\tau} p(\tau) p(r_{k+1}|r_k, \tau), \tag{1}$$

where τ is a path starting from an r_k of infinite length. Equation (1) means that the road segment transition probability is calculated as an expected value of the conditional probability that the vehicle moves along each path. Here, τ is a sequence of connected road segments that starts at r_k, i.e., $\tau = \{e_1 \to e_2 \to \cdots\}$, where $e_1 = r_k$, e_i.end $= e_{i+1}$.start, $1 \le i < \infty$ (see Fig. 2). Therefore, $p(\tau)$ can be described by edge transition probability $p(e_i \to e_{i+1})$ as follows:

$$p(\tau) = \prod_{i=1}^{\infty} p(e_i \to e_{i+1}), \tag{2}$$

where an edge transition probability from e_i to e_j, $p(e_i \to e_{i+1})$, is modeled as a Markov model, which can be estimated from historical data as follows:

$$p(e_i \to e_j) = \frac{N(e_i \to e_j) + 1}{\sum_j N(e_i \to e_j) + N_j}, \tag{3}$$

where $N(e_i \to e_j)$ is the number of transitions from e_i to e_j in the training data and N_j is the number of road segments connected from e_i. This training procedure is called add-one smoothing [36]. This smoothing technique is useful because it can calculate the probability even if there are no training data.

We next focus on conditional road segment transition probability $p(r_{k+1}|r_k, \tau)$ on the right-hand side of (1), which can be calculated as follows:

$$p(r_{k+1}|r_k, \tau) = \int p(x) \int p(v_k) p(r_{k+1}|r_k, \tau, x, v_k) dv_k dx, \tag{4}$$

where x is the vehicle position on road segment r_k and v_k is the estimated velocity of the vehicle. The proposed method does not estimate position x on road segment r_k. This position is assumed to be a roughly uniform distribution as follows:

$$p(x) = \begin{cases} \frac{1}{r_k.\mathrm{l}} & \text{if } 0 \le x \le r_k.\mathrm{l}, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Moreover, velocity υ_k is assumed to be a normal distribution as follows:

$$p(\upsilon_k) = \frac{1}{\sqrt{2\pi\sigma_{\upsilon,k}^2}} \exp\left(-\frac{(\upsilon_k - \bar{\upsilon}_k)}{2\sigma_{\upsilon,k}^2}\right), \tag{6}$$

where $\bar{\upsilon}_k$ is the mean value of υ_k and $\sigma_{\upsilon,k}^2$ is the variance of υ_k. Both $\bar{\upsilon}_k$ and $\sigma_{\upsilon,k}^2$ are estimated by $g_k.\mathrm{v}$ using the Kalman filter, which is shown in Appendix VII. Here, $p(r_{k+1}|r_k, \tau, x, \upsilon_k)$ in (4) can be determined by a physical analysis that is shown in Fig. 3 and calculated as follows:

$$\begin{aligned} p(r_{k+1}&|r_k, \tau, x, \upsilon_k) \\ &= \sum_{n=1}^{\infty} p(r_{k+1} = e_n|r_k, \tau, x, \upsilon_k)\delta_{r_{k+1},e_n} \\ &= \sum_{n=1}^{\infty} p(\tau_{1:n-1}.\mathrm{l} - x \le \upsilon_k \Delta t_k < \tau_{1:n}.\mathrm{l} - x)\delta_{r_{k+1},e_n}, \end{aligned} \tag{7}$$

where Δt_k is the time between sampling steps k and $k+1$, $\tau_{1:n}.\mathrm{l}$ is the length of the subpath of τ, which is from e_1 to e_n, i.e., $\tau_{1:n}.\mathrm{l} = e_1.\mathrm{l} + \cdots + e_n.\mathrm{l}$, and δ_{r_{k+1},e_n} is the Kronecker delta. Equation (7) uses a constant velocity model to predict vehicle movement, which means that if the predicted vehicle position is confined to e_n, a transition from r_k to $r_{k+1} = e_n$ occurs. Here, it is assumed that if $\upsilon_k \Delta t_k < 0$, the vehicle remains on the same path, i.e., $r_{k+1} = r_k$. Substituting (5)–(7) into (4) leads to

$$p(r_{k+1}|r_k, \tau) = \sum_{n=1}^{\infty} \frac{\delta_{r_{k+1},e_n}}{r_k.\mathrm{l}} \int_0^{r_k.\mathrm{l}} \int_{\frac{\tau_{1:n-1}.\mathrm{l}-x}{\Delta t_k}}^{\frac{\tau_{1:n}.\mathrm{l}-x}{\Delta t_k}} p(\upsilon_k) d\upsilon_k dx. \tag{8}$$

By substituting (8) into (1), we have

$$\begin{aligned} p(r_{k+1}&|r_k) \\ &= \sum_{\tau} p(\tau) \sum_{n=1}^{\infty} \frac{\delta_{r_{k+1},e_n}}{r_k.\mathrm{l}} \int_0^{r_k.\mathrm{l}} \int_{\frac{\tau_{1:n-1}.\mathrm{l}-x}{\Delta t_k}}^{\frac{\tau_{1:n}.\mathrm{l}-x}{\Delta t_k}} p(\upsilon_k) d\upsilon_k dx. \end{aligned} \tag{9}$$

The calculation of (9) is shown in Section IV.

C. Observation Model

GPS observation probability $p(g_k|r_k)$ is modeled as follows:

$$p(g_k|r_k) = \int_0^{r_k.\mathrm{l}} \frac{1}{\sqrt{2\pi\sigma_g}} \exp\left(\frac{dist(g_k, x)^2}{2\sigma_g}\right) p(x) dx. \tag{10}$$

where $dist(g_k, x)$ is the Euclidian distance between g_k and position x on road segment r_k and σ_g is the variance of the observation error (see Fig. 4). The HMM-based map matching algorithm uses a normal distribution based on the distance between observed data g_k and nearest point x' on road segment r_k [6]–[13], [28]. However, the nearest point assumption can generate unrealistic matching results, as pointed out by [28]. Therefore, the proposed method uses a marginalized value around position x on r_k. This is an integral of the normal distribution, which is difficult to calculate analytically. The approximation used in this study is shown in Appendix A.

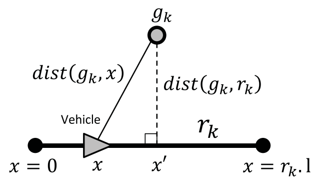

Fig. 4. Observation probability $p(g_k|r_k)$ is calculated by integrating the observation probability, which is defined as a normal distribution at each position x on road segment r_k.

IV. Map Matching Algorithm

A. Bayesian Filtering

As mentioned in the problem definition, online map matching can be implemented by using Bayesian filtering, which estimates conditional posterior probability $p(r_k|g_{1:k})$ for each sampling step k. We can calculate $p(r_k|g_{1:k})$ by using Bayes' theorem as follows:

$$p(r_k|g_{1:k}) = \frac{p(g_k|r_k, g_{1:k-1})p(r_k|g_{1:k-1})}{p(g_k|g_{1:k-1})}. \tag{11}$$

Assuming conditional independence, $p(g_k|r_k, g_{1:k}) = p(g_k|r_k)$ leads to

$$\begin{aligned} p(r_k|g_{1:k}) &\propto p(g_k|r_k)p(r_k|g_{1:k-1}), \\ &\propto p(g_k|r_k) \sum_{r_{k-1}} p(r_k|r_{k-1})p(r_{k-1}|g_{1:k-1}), \end{aligned} \tag{12}$$

where $p(r_k|r_{k-1})$ is the route prediction probability, which is described in Section III-B, and $p(r_{k-1}|g_{1:k-1})$ is the posterior probability of the previous step. The procedure used to calculate prior probability $p(r_k|g_{1:k-1})$ in (12) is called *prediction*. The procedure used to calculate posterior probability $p(r_k|g_{1:k})$ is called *filtering*. Bayesian filtering-based map matching is therefore a repetition of *prediction* and *filtering*.

B. MHT

The proposed method employs the MHT, which is a Bayesian filtering algorithm that represents $p(r_k|g_{1:k})$ as follows:

$$p(r_k|g_{1:k}) = \sum_{i=1}^{N_C} c_i.\mathrm{w} \cdot \delta_{r_k,c_i.e}, \tag{13}$$

where N_C is the number of road segment candidates, c_i is the ith candidate, which is a pair consisting of the road segment and its probability, i.e., $(c_i.e, c_i.\mathrm{w})$, and $\delta_{r_k,c_i.e}$ is the Kronecker delta. In the MHT, the probability density function is divided into the probability of each candidate. When using the MHT, the *prediction* and *filtering* calculations simplify because they are performed by calculating each candidate. The idea that the complex probability distribution is represented by a finite number of candidates is similar to the particle filter approach [14]. However, the MHT differs from a particle filter in that it does not use random sampling to predict states. In the MHT, each candidate produces new candidates for all states that can be transited and then candidates that represent the same states are integrated by summing their probabilities. For a

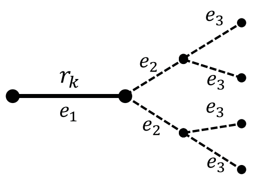

Fig. 5. The paths from r_k can be enumerated along each road connection.

system based on discrete transitions between a finite number of states, the MHT is more suitable than the particle filter because its calculation cost is an order of magnitude lower and it does not cause degeneracy. In the MHT, the number of candidates becomes large during *prediction*; therefore, we remove candidates that have a probability below a threshold value L_u after *filtering*.

At the beginning of the algorithm, the set of candidates C is initialized as the set whose distance from the first observed data point g_1 is smaller than a predefined distance L_c. Each candidate's weight is initialized by its road segment length $c_i.e.l$ because the prior probability of each candidate is expected to be proportional to its road segment's length. The posterior probability is then calculated in *filtering*. The output of each time step is the candidate with the highest probability, i.e., argmax c_i.w.

C. Route Prediction Calculation

Route prediction requires the transition probabilities for all paths to be summed. However, we cannot calculate the probabilities for all paths because the number of paths is boundless. Therefore, we introduce an algorithm that calculates the transition probabilities forward along the road connections.

First, (9) is written as follows:

$$p(r_{k+1}|r_k) = \sum_{\tau} p(\tau) \sum_{n=1}^{\infty} f(\tau_{1:n})\delta_{r_{k+1},e_n},$$
$$f(\tau_{1:n}) = \frac{1}{r_k.l}\int_0^{r_k.l}\int_{\frac{\tau_{1:n-1}.l-x}{\Delta t_k}}^{\frac{\tau_{1:n}.l-x}{\Delta t_k}} p(v_k)dv_k dx, \tag{14}$$

where $\tau_{1:n}$ is a subpath of τ, which starts at e_1 and ends at e_n, i.e., $\{e_1 \to \cdots \to e_n\}$. Then, τ is divided into two paths, $\tau_{1:n}$ and $\tau_{n:\infty}$, as follows:

$$\begin{aligned} p(r_{k+1}|r_k) &= \sum_{n=1}^{\infty}\sum_{\tau_{1:n}}\sum_{\tau_{n:\infty}} p(\tau_{1:n})p(\tau_{n:\infty})f(\tau_{1:n})\delta_{r_{k+1},e_n}, \\ &= \sum_{n=1}^{\infty}\sum_{\tau_{1:n}}\left(\sum_{\tau_{n:\infty}} p(\tau_{n:\infty})\right)p(\tau_{1:n})f(\tau_{1:n})\delta_{r_{k+1},e_n}, \\ &= \sum_{n=1}^{\infty}\sum_{\tau_{1:n}} p(\tau_{1:n})f(\tau_{1:n})\delta_{r_{k+1},e_n}. \end{aligned} \tag{15}$$

Here, the sum of paths can be written as follows:

$$\sum_{\tau_{1:n}} p(\tau_{1:n}) = \sum_{e_1}\cdots\sum_{e_n}\prod_{i=1}^{n-1} p(e_i \to e_{i+1}). \tag{16}$$

From this equation, we can enumerate the paths at n forward along the road connections (see Fig. 5). Substituting (16) into (15) leads to

$$p(r_{k+1}|r_k) = \sum_{n=1}^{\infty}\sum_{e_1}\cdots\sum_{e_n}\prod_{i=1}^{n-1} p(e_i \to e_{i+1})f(\tau_{1:n})\delta_{r_{k+1},e_n}. \tag{17}$$

Here, $f(\tau_{1:n})$ can be transformed as follows:

$$\begin{aligned} f(\tau_{1:n}) &= h(\tau_{1:n-1}, \bar{v}_k, \sigma_{v,k}, \Delta t_k) \\ &\quad - , h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k), \\ h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k) &= \frac{1}{e_1.l}\int_0^{e_1.l}\int_{\frac{\tau_{1:n}.l-x}{\Delta t_k}}^{\infty} p(v_k)dv_k dx, \end{aligned} \tag{18}$$

where $h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k)$ denotes the probability of the vehicle traveling further than e_n. Function h is a double integral of a normal distribution, which is difficult to calculate analytically; therefore, a numerical calculation or an analytical approximation is needed. An analytical approximation is used in this study, as detailed in Appendix VII.

The route prediction calculation can be approximated by calculating the probability of road segments forward along the road connections until the point at which the probability that the vehicle goes further is smaller than predefined probability L_p, i.e., $c_i.\mathrm{w} \cdot p(\tau) \cdot h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k) < L_p$. If L_p is sufficiently small, the route transition probability calculated by adopting this procedure becomes a good approximation of (1). This procedure is implemented by using the recursive function shown in Algorithm 2.

D. Complete Algorithm

Summarizing the calculation methods presented in the previous sections, the concrete procedure of the proposed algorithm is described in Algorithm 1. In this algorithm, τ.last is the last road segment of τ, i.e., $\tau_{1:n}$.last $= e_n$, p_τ is $p(\tau)$, and h_{pre} and h_{new} are $h(\tau_{1:n-1}, \bar{v}_k, \sigma_{v,k}, \Delta t_k)$ and $h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k)$ in (18), respectively. Function *Observe* calculates (10), and *KalmanFilter* is a general Kalman filter calculation with a constant velocity model that is described in Appendix VII. The *Find* method on line 15 in Algorithm 2 determines the first candidate whose edge is τ.last from C_{new}. In this case, each candidate in set C_{new} is unique to the edge.

V. Experimental Results

In this study, we performed two evaluations. The first was an evaluation of map matching performance (accuracy and efficiency) from real driving data, the route of which was already known. The second evaluated the improvement in accuracy gained by parameter training by using simulation data. By using a simulation, we can create a large amount of historical data for parameter training for the probabilistic route prediction model.

In both evaluations, the parameters of the proposed algorithm were set as follows: Velocity model system error $\sigma_a = 3.0$ m/s; GPS observation error $\sigma_g = 5.0$ m; Velocity observation error $\sigma_{\text{v}} = 30.0$ km/h; Range of the initial segment

Algorithm 1 Online Map Matching Algorithm

Input: Road network $G(V, E)$, trajectory $g_{1:K}$, and sampling interval $\Delta t_{1:K}$
Output: Route $r_{1:K}$

```
1: /* Initialization */
2: $C \leftarrow \{c|c.\mathrm{w} = c.e.\mathrm{l}, c.e \in E, dist(g_1, c.e) < L_c\}$
3: $\bar{v} \leftarrow g_1.\mathrm{v}$
4: $\sigma_v \leftarrow \sigma_\mathrm{v}$
5: /* Main Loop */
6: for $k = 1$ to $K$ do
7:   if $k > 1$ then
8:     /* prediction */
9:     $C_{\mathrm{new}} \leftarrow \emptyset$
10:    for all $c \in C$ do
11:      $\tau \leftarrow \{c.e\}$
12:      $p_\tau \leftarrow 1$
13:      $h_{\mathrm{pre}} \leftarrow 1$
14:      /* New candidates connected from this edge */
15:      $C_{\mathrm{new}} \leftarrow Recur(C_{\mathrm{new}}, c.\mathrm{w}, \tau, p_\tau, \bar{v}, \sigma_v, \Delta t_{k-1}, h_{\mathrm{pre}})$
16:    end for
17:    $C \leftarrow C_{\mathrm{new}}$
18:    /* Kalman filtering for the velocity */
19:    $\bar{v}, \sigma_v \leftarrow KalmanFilter(\bar{v}, \sigma_v, g_k.\mathrm{v})$
20:  end if
21:  /* filtering */
22:  for all $c \in C$ do
23:    $c.\mathrm{w} \leftarrow \frac{c.\mathrm{w}\cdot Observe(g_k, c)}{\sum_{c' \in C} c'.\mathrm{w}\cdot Observe(g_k, c')}$
24:  end for
25:  $C \leftarrow \{c|c \in C, c.w > L_u\}$
26:  $r_k \leftarrow \left(\underset{c \in C}{\mathrm{argmax}}\, c.\mathrm{w}\right).e$
27:  output $r_k$
28: end for
```

Algorithm 2 Recur

Input: C_{new}, w, τ, p_τ, $\bar{v}$, σ_v, Δt_{k-1}, and h_{pre}
Output: C_{new}

```
1: /* Calculate the probability */
2: $h_{\mathrm{new}} \leftarrow h(\tau, \bar{v}, \sigma_v, \Delta t_{k-1})$
3: if $w \cdot p_\tau \cdot h_{\mathrm{new}} > L_p$ then
4:   $\mathcal{E}_{\mathrm{next}} \leftarrow \{e_{\mathrm{next}}|e_{\mathrm{next}} \in E, e_{\mathrm{next}}.\mathrm{start} = \tau.\mathrm{last.end}\}$
5:   for all $e_{\mathrm{next}} \in \mathcal{E}_{\mathrm{next}}$ do
6:     $\tau' \leftarrow \{\tau \rightarrow e_{\mathrm{next}}\}$
7:     $p'_\tau \leftarrow p_\tau \cdot p(\tau.\mathrm{last} \rightarrow e_{\mathrm{next}})$
8:     $h'_{\mathrm{pre}} \leftarrow h_{\mathrm{new}}$
9:     /* New candidates connected from this edge */
10:    $C_{\mathrm{new}} \leftarrow Recur(C_{\mathrm{new}}, w, \tau', p'_\tau, \bar{v}, \sigma_v, \Delta t_{k-1}, h'_{\mathrm{pre}})$
11:  end for
12: end if
13: $w' \leftarrow w \cdot p_\tau \cdot (h_{\mathrm{pre}} - h_{\mathrm{new}})$
14: /* Add the probability to new set of candidates */
15: $c_{\mathrm{new}} \leftarrow C_{\mathrm{new}}.Find(c_{\mathrm{new}}|c_{\mathrm{new}}.e = \tau.\mathrm{last})\}$
16: if $c_{\mathrm{new}} \neq null$ then
17:   $c_{\mathrm{new}}.\mathrm{w} = c_{\mathrm{new}}.\mathrm{w} + w'$
18: else
19:   $C_{\mathrm{new}} \leftarrow C_{\mathrm{new}} \cup (\tau.\mathrm{last}, w')$
20: end if
21: return $C_{\mathrm{new}}$
```

search $L_c = 30.0$ m; Prediction threshold $L_p = 0.00001$; and Update threshold $L_u = 0.001$.

In the evaluations, the proposed method was compared with the nearest match (point-to-curve match), online and offline HMM, and the path inference filter (PIF) [28]. The HMM methods were based on the Newson and Krumm method [8], which is a major HMM-based method. The online HMM method evaluated in this study did not perform backward smoothing and output the estimation immediately. By contrast, the offline HMM method output the estimation after obtaining the entire trajectory. The transition probabilities of these HMM methods are as follows:

$$p(r_k = e_i|r_{k-1} = e_j) = \frac{1}{\beta_k}\exp\left(\frac{-d_k}{\beta_k}\right)$$
$$d_k = \left|dist(g_k, g_{k-1}) - \|h_{k,i^*} - h_{k-1,j^*}\|_{route}\right|, \quad (19)$$

where h_{k,i^*} and h_{k-1,j^*} are the projection points of g_k and g_{k-1} onto e_i and e_j, respectively, and $\|h_{k,i^*} - h_{k-1,j^*}\|_{route}$ indicates the shortest path length between them. In this experiment, parameter β_k was set as $\beta_k = 0.5\Delta t_{k-1}$, where the unit of Δt_{k-1} is seconds and that of d_k is meters. Because d_k is expected to be proportional to Δt_{k-1}, we assumed β_k is also proportional to Δt_{k-1} for the normalization. The PIF is a conditional random field-based method, which is a state-of-the-art method based on route features. With reference to feature-based methods [11], [12], [26]–[28], we chose four of the features available from our experimental dataset: the square of the distance between the GPS and segmnet $dist(g_k, r_k)^2$, the difference between the GPS travel distance and route length d_k, the number of signals on the route, and the sum of the turn angles of the route. The first two features were the same as those of the Newson and Krumm method. The PIF was evaluated only on the simulation data because this method assumes its parameters are already trained.

Performance was evaluated with respect to both accuracy and efficiency. Accuracy in this study was the point assignment, which is the most important accuracy metric for online map matching methods. Efficiency is the number of data points that can be processed in 1 millisecond, which is also important when handling massive amounts of data on a cloud server. Many researchers have proposed methods to increase efficiency [31]–[34]. The calculation time also causes latency; therefore, efficiency is important for online map matching methods. All algorithms were implemented in C# on the Visual Studio platform, and the results of the experiments were calculated on a computer with an Intel Core i7-4770K CPU 3.50 GHz and 32 GB RAM. The map data are stored in memory, with spatial indexing based on R-tree [39]. For the HMM and PIF methods, the A* algorithm [40] was used to find the shortest path.

A. Evaluation of Real Driving Data

1) Experimental Datasets: The performance of the methods was evaluated by using real driving data collected in

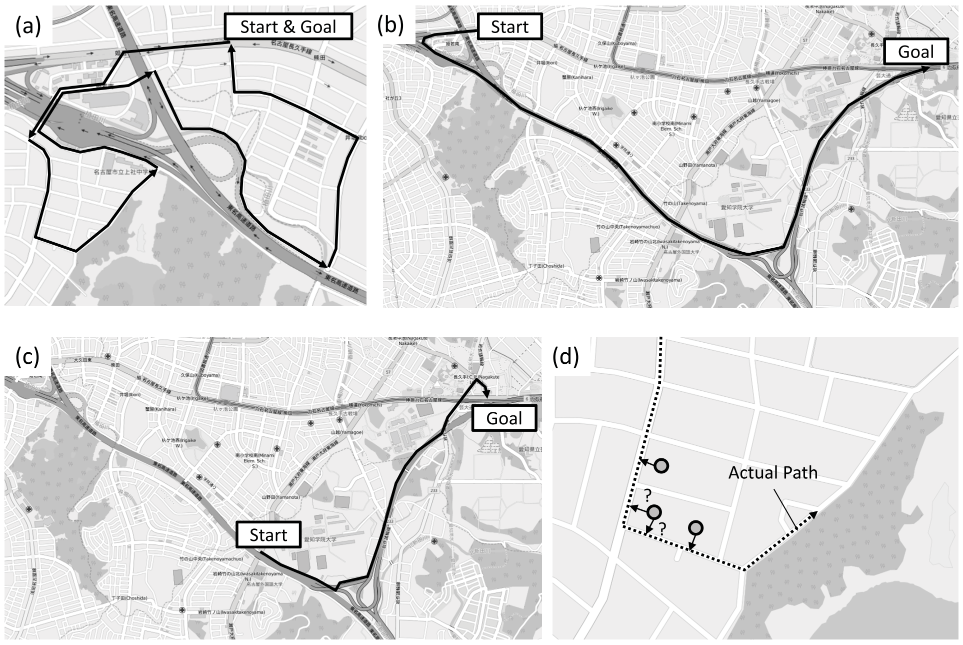

Fig. 6. Routes of the real driving data: (a) side road that is near an elevated road, (b) freeway, (c) under an elevated road, and (d) situation difficult to annotate.

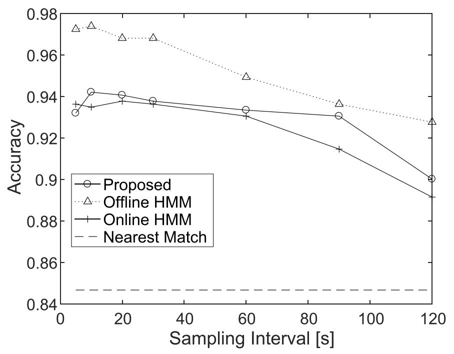

Fig. 7. Accuracy of map matching methods on real driving data in complex environments.

Nagakute, Japan. These driving data were collected in intricate environments that are difficult for map matching, such as at junctions and under elevated roads (see Figs. 6 (a)–(c)). Each route was traveled twice and the total travel time was about 2 h. The sampling interval of GPS data was about 5 seconds and velocity was measured in addition to latitude and longitude. In the following evaluation, these GPS data are subsampled to evaluate accuracy and efficiency for various sampling intervals. The map data used in this evaluation were taken from OpenStreetMap [37] and the ground truth road segments were annotated manually, which was easy because the driving course was known. However, some situations, such as that shown in Fig. 6 (d), are difficult to annotate because there are multiple segments close to a GPS point on the actual path. In this case, both segments were treated as correct.

2) Evaluation Results: Fig. 7 shows the accuracy of the methods for various sampling intervals. The horizontal axis represents the sampling interval and the vertical axis represents accuracy. Fig. 7 shows that the proposed method has much higher accuracy than the nearest match method and similar accuracy to the online HMM method, even without parameter training. Note that the proposed method and online HMM methods are similar, except that the former uses a prediction model, whereas the latter use the transition probability. This result shows that there is little difference between the performance of the untrained proposed prediction model and the transition probability in (19). Fig. 7 also shows that the accuracy of the proposed and online HMM methods is 3% less than that of the offline HMM method. This difference is caused by the availability of the future GPS points.

Meanwhile, Fig. 8 shows the efficiency of the methods for various sampling intervals. The horizontal axis represents the sampling interval and the vertical axis represents efficiency [points/ms]. Fig. 8 shows that the proposed method is faster than the HMM methods. In this case, the A* algorithm often cannot find the path because the candidates include both an elevated road and a route under an elevated road, which causes the HMM calculation cost to become high. By contrast, this does not affect the proposed method. Furthermore, for short-interval data, the proposed method is faster than the nearest match method because it does not search the set of all segments for candidate segments around the GPS points. The proposed method picks up candidate segnmets by searching along the segment connections, which is faster than searching all segments in short-interval situations.

B. Improvement With Parameter Training

1) Experimental Dataset: Here, we evaluate the improvement in accuracy gained by parameter training. In this study, we used a traffic simulation dataset because it was difficult to obtain a large real (and public) dataset that was sufficient to

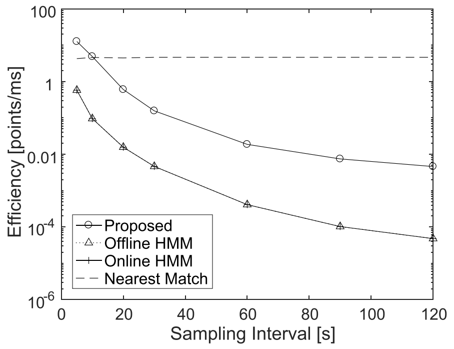

Fig. 8. Efficiency of map matching methods on real driving data in complex environments.

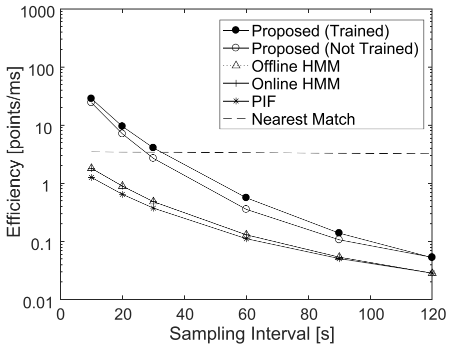

Fig. 10. Efficiency of the map matching methods with parameters trained on the simulation data.

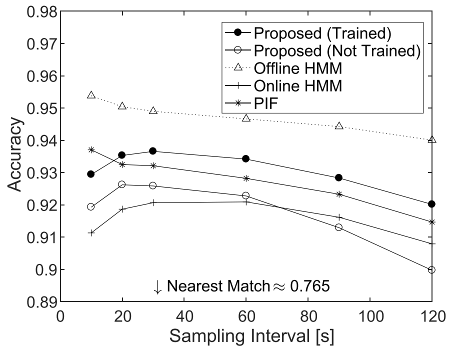

Fig. 9. Accuracy of the map matching methods with parameters trained on the simulation data.

train the model. The simulator used for this evaluation was the Traffic Simulation Framework (TSF), which is publicly available at the IEEE ICDM Contest 2010 website [38].

The evaluation was performed by using 1-h simulation data. In addition, we prepared 10-h simulation data for training. All the parameters of the simulation were set to their default values except for *limit*, which defines the length of the simulation.

The sampling interval of the GPS data was 10 s. Data were obtained from about 1% of the vehicles. The data contained the correct road IDs that the vehicles traversed. The number of evaluation data points was approximately 160,000 and the data produced by the simulator had no noise. To simulate the GPS data, we added independent Gaussian noise that had a standard deviation of about 5.0 m.

In this evaluation, the PIF online map matching method [28], trained by using the same 10-h data as for the proposed method, was also evaluated.

2) Evaluation Results: Fig. 9 shows the accuracy of each method for data with various sampling intervals. The horizontal axis represents the sampling interval and the vertical axis represents accuracy. In Fig. 9, the accuracy of the nearest match method is much less than that of the other methods; therefore, it is out of range ($\approx$ 0.765). Fig. 9 shows that parameter training reduced the error of the proposed method by about 20%. This improvement is larger than the difference between the PIF and online HMM methods except when the sampling interval is 10s. This result indicates that our idea: the prediction model helps to improve the accuracy of the online map matching is correct.

Note that the proposed method has advantages in its learning procedure. The PIF assumes that the sampling interval is constant. In the case of variable sampling intervals, however, the parameters must be estimated for each sampling interval (10 s, 20 s, 30 s, etc.). By contrast, the proposed method requires training parameters that are independent of the sampling interval. Therefore, it can be easily applied for non-constant sampling interval data.

Although the accuracy of the proposed method is high, it does not reach that of the offline HMM method. The probabilistic route prediction model employed in this study is a simple one and therefore prediction accuracy is not notably high, as shown by [42]. Hence, there is potential for improvement by replacing the prediction model with a more sophisticated model. The development of a better route prediction model would also contribute to the improvement of online map matching accuracy.

Meanwhile, Fig. 10 shows the efficiency [points/ms] of each method for data with various sampling intervals. The horizontal axis represents the sampling interval and the vertical axis represents efficiency. In Fig. 10, the difference is less than that in the evaluation for the real data case (Fig. 8), although the proposed method is faster than the HMM methods, even in general environments that are not as complex as Fig. 6. The effects of the training can also be seen. Surprisingly, efficiency improves after training in the proposed method because parameter training causes a non-uniform road transition probability, which prunes unlikely routes.

VI. Discussion

In our experiments, we have shown that our method is faster as compared with the HMM-based methods. Herein, we analyze the computational complexity of the proposed method. The time complexity of the *prediction* step depends on the prediction tree (see Fig. 5). The complexity is $O(b^{d_p})$, where b is the branching factor (the average number of successors per road segment) and d_p is the maximum depth of the prediction tree. The complexity of the *update* step is also $O(b^{d_p})$; it depends on the number of the predicted candidates, and in the worst case, it equals b^{d_p}. Therefore, the time complexity of the proposed method is $O(b^{d_p})$.

TABLE I
COMPUTATIONAL COMPLEXITY

	Computational Complexity
Proposed method	$O(b^{d_p})$
Nearest match	C_{Rtree}
HMM-based methods	$O(b^{d_p}) + C_{\mathrm{Rtree}}$

In contrast, the complexity of the HMM-based methods mainly comprises the A* algorithm and R-tree search. In the worst case, the time complexity of the A* algorithm is $O(b^d)$, where d is the depth of solution [41]. In the A* algorithm of the HMM-based methods, it is sufficient to traverse road segments within a reachable range, i.e., $d \leq d_p$. Therefore, the time complexity of the A* algorithm in HMM-based methods is $O(b^{d_p})$. The time complexity of the R-tree search is denoted as C_{Rtree}. The R-tree is not possible to guarantee better worst case performance than $O(|E|)$; its complexity depends on the range of a query and the road network size $|E|$. The complexity of the nearest match comprises only the R-tree search. These are summarized in Table I. The branching factor b of road network in Nagakute and TSF are 1.40 and 1.39 respectively. The depth of the prediction tree is calculated as $d_p \approx \hat{v}\Delta t / l$, where $\hat{v}$ is the highest velocity whose cumulative distribution function is less than $1 - L_p$ and l is the average length of the road segments. The average length of the road segments l of road network in Nagakute and TSF are 35.0[m] and 94.0[m], respectively.

The proposed method is fastest for short sampling interval data because $O(b^{d_p})$ is smaller than C_{Rtree} for such data. As the sampling interval increases, the computational time of the proposed method increases exponentially similar to the HMM-based methods. This is confirmed in Fig. 8 and Fig. 10.

VII. CONCLUSION

We proposed a novel online map matching method with route prediction. The proposed map matching method predicts future routes by using a probabilistic route prediction model and the estimated route is then updated by the observed trajectory. This can improve accuracy without any time step latency, unlike the HMM-based approaches that use future coordinates, which is a notable advantage for real-time applications. The experimental results showed that the proposed method had accuracy as high as the naïve online HMM method, even without parameter training (Fig. 7). Furthermore, it is faster than the online HMM method (Fig. 8). Moreover, parameter training based on historical trajectory data improved the accuracy of the proposed method significantly, well above that of the naïve online HMM method (Fig. 9). This improvement was larger than that of the feature-based method (i.e., the PIF).

The improvement in accuracy in this study was achieved by using the probabilistic route prediction model. This result suggests that better route prediction improves map matching. The probabilistic route prediction model employed in this study is a simple one. Hence, replacing this prediction model with a more sophisticated model such as that presented by [42] is expected to lead to further improvements in future studies.

APPENDIX A
APPROXIMATION OF THE INTEGRAL FOR NORMAL DISTRIBUTIONS

In the proposed route prediction method, it is necessary to calculate $h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k)$ in (16), which is an integral of a normal distribution. However, this integral is difficult to calculate analytically. Therefore, we approximate the integral of a normal distribution by the following logistic function:

$$\begin{aligned} &h(\tau_{1:n}, \bar{v}_k, \sigma_{v,k}, \Delta t_k) \\ &\approx \frac{1}{e_1.\mathrm{l}} \int_0^{e_1.\mathrm{l}} \frac{1}{1 + \exp\left(-\frac{\pi\ (v_k - \bar{v}_k)}{\sqrt{3}\sigma_{v,k}}\right)} \Bigg|_{v_k = \frac{\tau_{1:n}.\mathrm{l} - x}{\Delta t_k}}^{\infty} dx \\ &= -\frac{1}{e_1.\mathrm{l}} \left(s \log\left(\exp\left(\frac{\bar{v}_k \Delta t_k - (\tau_{1:n}.\mathrm{l} - x)}{\sqrt{3}\sigma_{v,k}\Delta t_k/\pi} \right) + 1 \right) \right) \Bigg|_{x=0}^{e_1.\mathrm{l}} \end{aligned} \tag{20}$$

Observation probability $p(g_k|r_k)$ is calculated from (10); however, it is also an integral of a normal distribution. Therefore, we also approximate it by the following logistic function:

$$p(g_k|r_k) \approx \frac{1}{r_k.\mathrm{l}} \exp\left(\frac{dist(g_k, r_k)^2}{2\sigma_g} \right) \xi, \tag{21}$$

$$\xi = \left(\frac{1}{1 + \exp\left(-\frac{\pi(x - x')}{\sqrt{3}\sigma_g} \right)} \right) \Bigg|_{x=0}^{r_k.\mathrm{l}},$$

where $dist(g_k, r_k)$ is the Euclidean distance between g_k and r_k.

APPENDIX B
KALMAN FILTER FOR VEHICLE VELOCITY ESTIMATION

We employ a Kalman filter to estimate the distribution of vehicle velocity v_k. The system model of velocity is as follows:

$$\begin{aligned} v_k &= v_{k-1} + a_k \Delta t_{k-1}, \\ a_k &\sim \mathcal{N}(0, \sigma_a^2), \end{aligned} \tag{22}$$

where a_k is the system error, which is the acceleration of the vehicle, and this is assumed to be a normal error with variance σ_a^2. This is a common modeling approach used in positioning. The feature variables $\bar{v}_k$ and $\sigma_{v,k}$ are calculated in two steps: *prediction* and *filtering*. In *prediction*, they are predicted as follows:

$$\begin{aligned} \bar{v}_k &= \bar{v}_{k-1}, \\ \sigma_{v,k} &= \sqrt{\sigma_{v,k-1}^2 + \sigma_a^2 \Delta t_{k-1}^2}. \end{aligned} \tag{23}$$

In *filtering*, they are updated as follows:

$$\begin{aligned} \bar{v}_k &= \frac{\sigma_{\mathrm{v}}^2 \bar{v}_k + \sigma_{v,k}^2 g_k.\mathrm{v}}{\sigma_{\mathrm{v}}^2 + \sigma_{v,k}^2}, \\ \sigma_{v,k} &= \sqrt{\left(\sigma_{\mathrm{v}}^{-2} + \sigma_{v,k}^{-2} \right)^{-1}}. \end{aligned} \tag{24}$$

By using these equations, $\bar{v}_k$ and $\sigma_{v,k}$ are estimated incrementally. Function $KalmanFilter$ in Algorithm 1 calculates both (23) and (24).

REFERENCES

[1] Z. Li, Y. Zhu, H. Zhu, and M. Li, "Compressive sensing approach to urban traffic sensing," in *Proc. 31st Int. IEEE Conf. Distrib. Comput. Syst.*, Jul. 2011, pp. 889–898.

[2] D. B. Work, O.-P. Tossavainen, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton, "An ensemble Kalman filtering approach to highway traffic estimation using GPS enabled mobile devices," in *Proc. 47th IEEE Conf. Decision Control*, Dec. 2008, pp. 5062–5068.

[3] Z. Liao, "Real-time taxi dispatching using global positioning systems," *Commun. ACM*, vol. 46, no. 5, pp. 81–83, May 2003.

[4] Y. Zheng and X. Xie, "Learning travel recommendations from user-generated GPS traces," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 1, 2011, Art. no. 2.

[5] J. Yuan *et al.*, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2010, pp. 99–108.

[6] B. Hummel, "Map matching for vehicle guidance," in *Dynamic and Mobile GIS: Investigating Space and Time*, J. Drummond and R. Billen, Eds. Boca Raton, FL, USA: CRC Press, 2006.

[7] J. Krumm, E. Horvitz, and J. Letchner, "Map matching with travel time constraints," in *Proc. Soc. Autom. Eng. World. Congr.*, Apr. 2007, paper 2007-01-1102.

[8] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2009, pp. 336–343.

[9] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang, "Map-matching for low-sampling-rate GPS trajectories," in *Proc. ACM SIGSPATIAL Int. Conf. Geograph. Inf. Syst.*, 2009, pp. 352–361.

[10] J. Yuan, Y. Zheng, C. Zhang, X. Xie, and G.-Z. Sun, "An interactive-voting based map matching algorithm," in *Proc. 11th Int. IEEE Conf. Mobile Data Manage.*, May 2010, pp. 43–52.

[11] T. Osogami and R. Raymond, "Map matching with inverse reinforcement learning," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2012, pp. 2547–2553.

[12] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet, "Online map-matching based on Hidden Markov model for real-time traffic sensing applications," in *Proc. 15th Int. IEEE Conf. Intell. Transp. Syst.*, Sep. 2012, pp. 776–781.

[13] G. R. Jagadeesh and T. Srikanthan, "Robust real-time route inference from sparse vehicle position data," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 296–301.

[14] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[15] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: State-of-the art and future research directions," *Transp. Res. C, Emerg. Technol.*, vol. 15, no. 5, pp. 312–328, 2007.

[16] P. Lamb and S. Thiebaux, "Avoiding explicit map-matching in vehicle location," in *Proc. 6th World Congr. Intell. Transp. Syst.*, 1999.

[17] J.-S. Pyo, D.-H. Shin, and T.-K. Sung, "Development of a map matching method using the multiple hypothesis technique," in *Proc. Int. IEEE Conf. Intell. Transp. Syst.*, Aug. 2001, pp. 23–27.

[18] N. Schuessler and K. W. Axhausen, "Map-matching of GPS traces on high-resolution navigation networks using the multiple hypothesis technique (MHT)," *Arbeitsberichte Verkehrsund Raumplanung*, vol. 568, pp. 1–22, 2009.

[19] M. Jabbour, P. Bonnifait, and V. Cherfaoui, "Map-matching integrity using multihypothesis road-tracking," *J. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 189–201, 2008.

[20] P. Bonnifait, J. Laneurit, C. Fouque, and G. Dherbomez, "Multi-hypothesis map-matching using particle filtering," in *Proc. 16th World Congr. Syst. Services*, 2009, pp. 1–8.

[21] C. Fouque and P. Bonnifait, "Matching raw GPS measurements on a navigable map without computing a global position," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 887–898, Jun. 2012.

[22] C. E. White, D. Bernstein, and A. L. Kornhauser, "Some map matching algorithms for personal navigation assistants," *Transp. Res. C, Emerg. Technol.*, vol. 8, nos. 1–6, pp. 91–108, 2000.

[23] J. S. Greenfeld, "Matching GPS observations to locations on a digital map," in *Proc. 81st Annu. Meeting Transp. Res. Board*, 2002, vol. 1. no. 3, pp. 164–173.

[24] S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk, "On map-matching vehicle tracking data," in *Proc. 31st Int. Conf. Very Large Data Bases*, 2005, pp. 853–864.

[25] F. Gustafsson *et al.*, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, Feb. 2002.

[26] Y. Zheng and M. A. Quddus, "Weight-based shortest-path aided map-matching algorithm for low-frequency positioning data," in *Proc. 90th Annu. Meeting Transp. Res. Board*, 2011, p. 20.

[27] M. Rahmani and H. N. Koutsopoulos, "Path inference from sparse floating car data for urban networks," *Transp. Res. C, Emerg. Technol.*, vol. 30, pp. 41–54, May 2013.

[28] T. Hunter, P. Abbeel, and A. Bayen, "The path inference filter: Model-based low-latency map matching of probe vehicle data," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 507–529, Apr. 2014.

[29] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th ICML*, 2001, pp. 282–289.

[30] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI*, 2008, pp. 1433–1438.

[31] K. Liu, Y. Li, F. He, J. Xu, and Z. Ding, "Effective map-matching on the most simplified road network," in *Proc. 20th Int. Conf. Adv. Geograph. Inf. Syst.*, 2012, pp. 609–612.

[32] Y. Li, C. Liu, K. Liu, J. Xu, F. He, and Z. Ding, "On efficient map-matching according to intersections you pass by," in *Proc. Int. Conf. Database Expert Syst. Appl.*, Berlin, Germany, 2013, pp. 42–56.

[33] Y. Tang, A. D. Zhu, and X. Xiao, "An efficient algorithm for mapping vehicle trajectories onto road networks," in *Proc. 20th Int. Conf. Adv. Geograph. Inf. Syst.*, 2012, pp. 601–604.

[34] R. Song, W. Lu, W. Sun, Y. Huang, and C. Chen, "Quick map matching using multi-core CPUs," in *Proc. 20th Int. Conf. Adv. Geograph. Inf. Syst.*, 2012, pp. 605–608.

[35] K. Zheng, Y. Zheng, X. Xie, and X. Zhou, "Reducing uncertainty of low-sampling-rate trajectories," in *Proc. IEEE 28th Int. Conf. Data Eng.*, Apr. 2012, pp. 1144–1155.

[36] C. D. Manning, P. Raghavan, and H. Schuetze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[37] Accessed: Jul. 2, 2015. [Online]. Available: http://www.openstreetmap.org/

[38] Accessed: Mar. 20, 2015. [Online]. Available: http://www.tunedit.org/challenge/IEEE-ICDM-2010

[39] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *Proc. ACM Int. Conf. Manage. Data (SIGMOD)*, 1984, pp. 47–57.

[40] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.

[41] R. E. Korf and M. Reid, "Complexity analysis of admissible heuristic search," in *Proc. AAAI/IAAI*, 1998, pp. 305–310.

[42] J. Krumm, "A Markov model for driver turn prediction," in *Proc. Soc. Autom. Eng. World. Congr.*, Apr. 2008, paper 2008-01-0195.

Shun Taguchi received the B.E. and M.E. degrees from Nagoya University, Japan, in 2007 and 2009, respectively. In 2009, he joined Toyota Central R&D Labs., Inc., Japan. His current research interests include Bayesian filtering, data assimilation, and their application to ITS.

Satoshi Koide received the B.S. and M.S. degrees from University of Osaka, Japan, in 2004 and 2006, respectively. In 2006, he joined Toyota Central R&D Labs., Inc., Japan. His research interests include data structures, machine learning, optimization, and their application to ITS.

Takayoshi Yoshimura received the B.S. degree from the Department of Computer Science, Nagoya Institute of Technology (NIT), Japan, in 1997, and the M.Eng. and Dr.Eng. degrees from the Department of Electrical and Computer Engineering, NIT, in 1999 and 2002, respectively. He is currently a Senior Researcher with Toyota Central R&D Labs., Inc., Japan. His research interests include in-vehicle signal processing, data-driven stochastic modeling, and the application of these to in-vehicle information systems.