ISPRS Int. J. Geo-Inf. **2015**, *4*, 515-534; doi:10.3390/ijgi4020515

OPEN ACCESS

ISPRS International Journal of
Geo-Information
ISSN 2220-9964
www.mdpi.com/journal/ijgi

Article

Moving Point Density Estimation Algorithm Based on a Generated Bayesian Prior

Akinori Asahara [1,*], Hideki Hayashi [1] and Takashi Kai [2]

[1] Research & Development Group, Center for Technology Innovation-Systems Engineering, Hitachi Ltd., 1-280, Higashi-koigakubo, Kokubunji-shi, 185-8601, Tokyo, Japan; E-Mail: hideki.hayashi.xu@hitachi.com

[2] Social Innovation Business Office, Hitachi Ltd., Hitachi Omori 2nd Building, 27-18, Minami-Oi 6-chome, Shinagawa-ku, 140-8572, Tokyo, Japan; E-Mail: takashi.kai.xc@hitachi.com

* Author to whom correspondence should be addressed; E-Mail: akinori.asahara.bq@hitachi.com; Tel.: +81-42-323-1111.

Academic Editor: Steve H.L. Liang, Mohamed Bakillah and Wolfgang Kainz

Received: 11 December 2014 / Accepted: 27 March 2015 / Published: 7 April 2015

Abstract: To improve decision making, real-time population density must be known. However, calculating the point density of a huge dataset in real time is impractical in terms of processing time. Accordingly, a fast algorithm for estimating the distribution of the density of moving points is proposed. The algorithm, which is based on variational Bayesian estimation, takes a parametric approach to speed up the estimation process. Although the parametric approach has a drawback, that is the processes to be carried out on the server are very slow, the proposed algorithm overcomes the drawback by using the result of an estimation of an adjacent past density distribution.

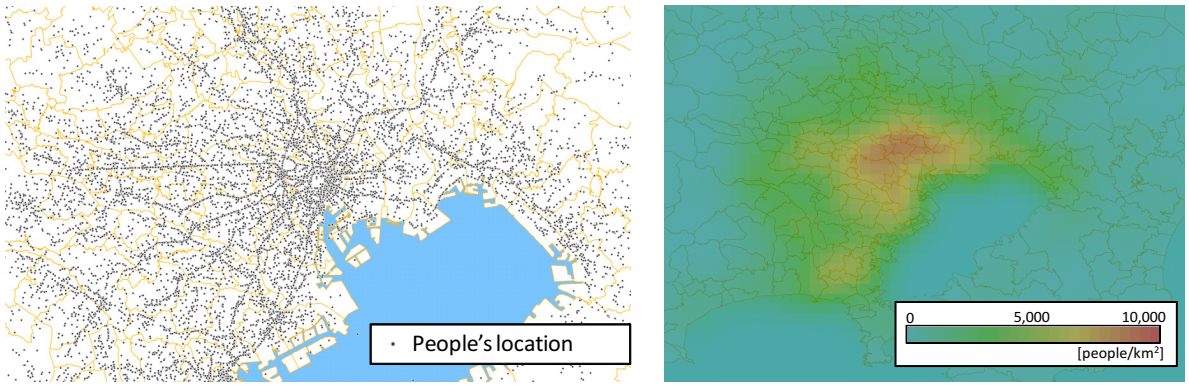
Keywords: variational Bayesian estimation; density estimation; moving features; Big Data

---

1. Introduction

Population density, namely the number of people per unit area, is one of the most effective factors in decision making involved in tasks, like infrastructure management, city development planning and merchandising [1]. For example, police officers can optimize their patrols in a city on the basis of

high-population density locations, because accidents often occur at those locations in cities. A dataset of people's positions, consisting of "moving points", as the example in Figure 1a, can be used to calculate the population density. Figure 1a was drawn by plotting the results of questionnaires about people's journeys (namely, data pairs consisting of origination and destination pairs of the travels) during a whole day on a base map [2]. The drawing thus represents people's positions at the moment in the past. In contrast, the people's positions measured in real time through mobile phones with GPS-like positioning functions [3,4] reflect people's movements. In that case, the people's positions are thus recorded in real time as moving points. Such massive data about people's positions should be commonly used, because they should be taken into account for many decisions, such as devising patrol plans for police officers. The OGC (Open Geospatial Consortium [5]), which is an international standard organization concerned with geospatial information systems, publishes standard specifications applicable to such data. A standard for encoding such massive moving-point data is defined as "OGC Moving Features encoding" [6], which is defined for exchanging massive amounts of moving-point data easily.



(a) (b)

Figure 1. Two types of population visualization. (**a**) Moving points; (**b**) density distribution. Background map: [2].

As mentioned above, real-time moving-point data are useful for many applications; especially, the distribution function of the density of moving points (hereafter "point density"), expressed as a function of position (x, y) and time t, is the most effective feature for describing the tendency of the moving points. However, handling a massive point dataset itself is not so easy, even if it is exchangeable. Therefore, a conversion from massive moving-point data to easily handleable data is needed.

"Coverage data," which describes the point density, are one of the most easily handleable kinds of data. The coverage can be encoded by various encoding standards, like OGC NetCDF [7] and GML [8]. Figure 1b shows a population density distribution visualized from the points shown in (a). Such an intelligible image colored according to density, often called a "heatmap," can be generated automatically (Figure 1b is generated by a quadratic kernel density estimator shown in Section 2.) After visualization, the heatmaps can be encoded by general image formats, like PNG and JPG. For distributing such images, an OGC WMS (Web Map Service) [9] interface is commonly used. Accordingly, a conversion from point data to a coverage would contribute to system interoperability.

Counting the number of the points is the most naive technique for estimating the density of stationary points. Namely, a space in which points exist is separated into small cells, and the number of points in each cell is counted. Such a method is called a histogram method [10]. However, the histogram method has several problems. One of the problems is the trade-off between the size of the cell ("cell size") and the number of data. Cell size can be adjusted in the case that the hierarchical cell structure, like a "quadtree" [11], is adapted. It should be smaller if the point density is required to be finer; however, statistical dispersion of the estimation results becomes higher if the cells are small, because smaller cells have fewer points. To estimate a finer point density, a huge dataset is therefore required. For that reason, several conventional algorithms for accurately estimating the point density, even with small cells, have been proposed. The reciprocal of the cell size is called "resolution"; that is, the resolution of a point density using small cells is "high."

Another problem with histograms arises in practical data-providing systems. An example of a client-server system using point density is shown in Figure 2. The data source periodically provides a real-time point dataset to the analysis server. The server estimates the point density and sends it to user-client systems A and B. The user clients use the point density by associating it with other information, such as temperature distribution and map illustrations. In this case, the server is basically a "high-spec" computer; in contrast, the user clients are generally "low-spec" computers. For such systems, OGC standards are available. Namely, OGC WMS [9] is a service interface for distributing map images, and OGC WCS (Web Coverage Service) [12] is a service interface for distributing coverage data. They provide web APIs to send the point density data. WMS-request data sent from user-client systems include the area boundary of the required map and the map-image size, which determine the image resolution. The resolution required by user-client A might be different from that required by user-client B; besides, the required resolution dynamically changes according to the user requirements. The point density must thus be dynamically calculated.

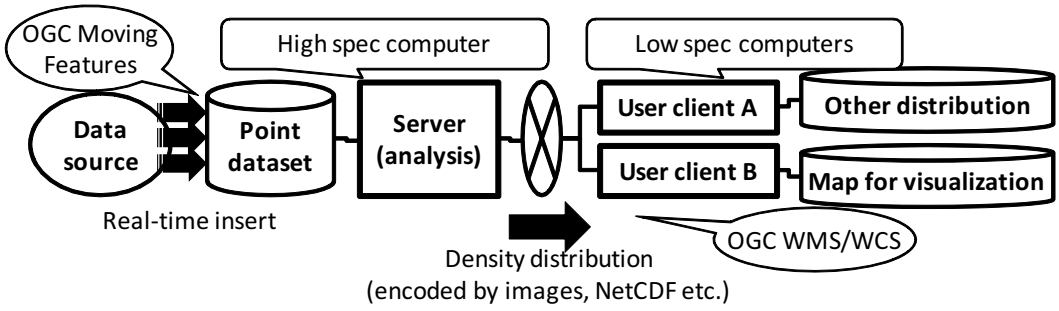


Figure 2. A system using point density.

The computational power of the user client system is generally weaker than that of the server. Furthermore, the point dataset itself may not be distributed, because the amount of data provided by the server should be small enough to be sent via a network with low throughput. Many calculations of point density with various resolutions should therefore be carried out on the server in real time. Accordingly, a fast on-line algorithm for calculating point densities is needed.

To satisfy the requirements mentioned above, in this work, a "point-density-estimation algorithm" for handling a moving-point dataset (described with OGC Moving Features) is proposed. The proposed

algorithm, which is based on Bayesian estimation, can estimate point density (distributed with OGC WMS or WCS) by using only a partial dataset; consequently, it works faster than conventional algorithms.

2. Related Works and Contributions

2.1. Trajectory Analysis and Density Estimators

An analysis of tracks of moving points is called "trajectory analysis", namely trajectory smoothing [13] (for reducing errors in positioning), trajectory clustering [14–17] (for retrieving similar trajectories from a database), prediction of movement [18,19], representative path extraction [15,17,20–22] and an algorithm for labeling positioning data [23].

These trajectory analyses have been developed for extracting information from stored trajectories, not points. Basically, the density of moving points does not depend on the movement of points; that is, the density can be calculated by using only a "snapshot" of the moving points. With the proposed algorithm, however, the density estimation is accelerated by using the "adjacent past density distribution", which is based on the similarity between the current point density and the most recent one.

Kernel-density estimators [24,25] are popular methods for estimating a point density. The density calculated by a kernel-density estimator is the summation of distributions around the points. The Gaussian kernel-density estimator, which is the most general type, estimates the point densities as a summation of Gaussians. The density $P(\mathbf{x})$ is given as:

$$P(\mathbf{x}) = \frac{1}{N}\sum_{n}^{N}\frac{1}{2\pi h^2}\exp\left(-\frac{|\mathbf{x}-\mathbf{x}_n|^2}{2h^2}\right) \tag{1}$$

where N points are denoted as $\{\mathbf{x}_n\}$ and h corresponds to the radius of each Gaussian. A quadratic kernel-density estimator, which estimates points densities as a summation of quadratic functions as:

$$P(\mathbf{x}) = \frac{1}{Nh^2}\sum_{n}^{N}Q\left(\frac{|\mathbf{x}-\mathbf{x}_n|}{h}\right), \qquad Q(r) \equiv \begin{cases} \frac{6}{\pi}(1-r)^2 & r<1 \\ 0 & r \geq 1 \end{cases} \tag{2}$$

is often used. A quadratic kernel estimator works faster than a Gaussian kernel estimator. On the other hand, the accuracy of a quadratic kernel estimator is generally less than that of a Gaussian kernel estimator. Accordingly, a quadratic kernel estimator is often applied to visualize density in the form of a heatmap. Several kernel-density estimators for summarizing the tendency of moving points have been proposed; as examples, the dual kernel-density estimator [26] is designed to show differences between two different point densities; and the directional kernel-density estimator [27] visualizes the tendency of the movement itself. These kernel-density estimators use all points to estimate the point density. Thus, the calculation of $P(\mathbf{x})$ in the density estimation becomes more complex in proportion to the number of points. Consequently, it is difficult to carry out the calculation in real time. This kind of estimator is called "nonparametric", because the parameters used in the model are not determined by the estimation process.

Another approach, called "parametric", is based on a specific functional formula. The parameters of the functional formula are optimized to make them consistent with the dataset. The calculation time taken

by the parametric approach is proportional to the complexity of the function. Thus, even if the number of points is large, the density estimation does not take a long time. However, optimization of the parameters takes a very long time. For example, a standard variational Bayesian estimation, which is based on a Gaussian mixture model (GMM), is often applied as a probability distribution of the point dataset. In the case of a variational Bayesian estimation, the distribution modeling is a convergence that makes parameters consistent with all point data; that is, the calculations are iterated. Although the distribution modeling can be carried out by the server, the processing time is too long for real-time calculation.

2.2. Contributions

Even though the proposed algorithm takes a parametric approach, its parameter optimization is very fast, because the point density is generated by updating the adjacent past point density. Measurement by GPS-like positioning functions is generally periodic, so the proposed algorithm handles discrete time (a period of which equals a time discretization unit). Indices of the discrete times are denoted by integers $\cdots t, t+1, \cdots$. Under the assumption that the point density at time t is expressed as the sum of Gaussian components, the point density at $t+1$ is considered as the sum of the Gaussian components similar to the Gaussian components at time t. The difference between the point density at t and that at $t+1$ is predictable even if the point dataset is partially used.

The proposed algorithm is based on the update of the Gaussian components by using a partial point dataset. As the mechanism for updating the Gaussian components, Bayesian estimation is applied. In the case of Bayesian estimation, a probability-distribution function is assumed *a priori* and revised by using the point dataset. Bayesian estimation thus seems applicable for updating the probability-distribution function; however, to apply Bayesian estimation, a trick to determine parameters used by the assumed probability-distribution function is needed. In the following section, the trick named "responsibility reduction" is described.

3. Proposed Algorithm

3.1. Variational Bayesian Estimation

Variational Bayesian estimation is often applied to estimate a multivariate probability-distribution function. An argument of the multivariate probability-distribution function is replaced by a pair of coordinate values of a point as (x, y). The N-points density, which is obtained from point probability distribution $P(x, y)$, is calculated as $NP(x, y)$.

A probability distribution of parameters used in the probability-distribution function is defined in the variational Bayesian estimation. The probability-distribution function of the parameter set is called the "prior probability-distribution function" ("prior"). The prior and additional observed points ("observation") derive the estimated "posterior probability distribution" ("posterior") in the variational Bayesian estimation. The probability-distribution function of the GMM is denoted by:

$$P(\mathbf{x}|\{\pi_k\}, \{\mu_k\}, \{\Lambda_k\}) = \sum_k \pi_k \mathcal{N}(\mathbf{x}|\Lambda_k, \mu_k) \tag{3}$$

where $\mathbf{x}$ is a vector consisting of an observation's coordinate values (*i.e.*, (x, y)), $\mathcal{N}(\mathbf{x}|\Lambda, \mu)$ is a two-dimensional Gaussian function with a correlation matrix denoted by Λ_k and an average denoted by μ_k, and π_k is the mixing ratio of the k-th component to the sum of all components. The prior for $\{\pi_k\}, \{\mu_k\}$ and $\{\Lambda_k\}$ is then defined as:

$$P(\{\pi_k\}, \{\mu_k\}, \{\Lambda_k\} | W_0, \nu_0, \beta_0) = \mathrm{Dir}(\{\pi_k\}|\alpha_0) \prod_k \mathcal{N}(\mu_k|(\beta_0\Lambda_k)^{-1}, \mathbf{m}_0)\mathcal{W}(\Lambda_k|W_0, \nu_0)$$

Here, α_0, W_0, ν_0 and β_0 are parameters, called "hyperparameters," of the prior. $\mathcal{W}(\cdot|W, \nu)$ is the Wishart distribution with deviation W and degrees of freedom ν, and $\mathrm{Dir}(\cdot|\alpha)$ is the Dirichlet distribution with parameter α. The prior is conjugate to the Gaussian mixture; that is, the posterior formula updated by observations is the same as that of the prior. Estimation of the posterior is therefore implemented as the update of hyperparameters $\alpha_0, \beta_0, \mathbf{m}_0$ and W_0 by using observations $\{\mathbf{x}_n\}$ as:

$$\alpha_k = \alpha_0 + N_k \tag{4}$$

$$\beta_k = \beta_0 + N_k \tag{5}$$

$$\mathbf{m}_k = \frac{1}{\beta_k}(\beta_0\mathbf{m}_0 + N_k\bar{\mathbf{x}}_k) \tag{6}$$

$$W_k^{-1} = W_0^{-1} + S_kN_k + \frac{N_k\beta_0}{N_k + \beta_0}(\mathbf{m}_0 - \bar{\mathbf{x}}_k)(\mathbf{m}_0 - \bar{\mathbf{x}}_k)^T \tag{7}$$

$$\nu_k = \nu_0 + N_k \tag{8}$$

N_k, $\mathbf{x}_k$ and S_k are respectively calculated as:

$$N_k = \sum_n r_{n,k}, \qquad \bar{\mathbf{x}}_k = \frac{1}{N_k}\sum_n r_{n,k}\mathbf{x}_n, \qquad S_k = \sum_n r_{n,k}(\mathbf{x}_n - \bar{\mathbf{x}}_k)(\mathbf{x}_n - \bar{\mathbf{x}}_k)^{\mathrm{T}} \tag{9}$$

$r_{n,k}$, "responsibility", is calculated as:

$$r_{n,k} = \frac{\rho_{n,k}}{\sum_n \rho_{n,k}}, \qquad \rho_{n,k} \equiv \tilde{\pi}_k|\tilde{\Lambda}_k|^{\frac{1}{2}} e^{\left[-\frac{\beta_k}{2} - \frac{1}{2}(\mathbf{x}_n - \mathbf{m}_k)^T W_k(\mathbf{x}_n - \mathbf{m}_k)\right]} \tag{10}$$

$$\ln\tilde{\pi}_k \equiv \ln\psi(\alpha_k) - \ln\psi(\sum_k \alpha_k), \qquad \ln|\tilde{\Lambda}_k| \equiv \ln\psi(\frac{\nu_k}{2}) + \ln\psi(\frac{\nu_k - 1}{2}) + \ln 4|W_k| \tag{11}$$

where $\psi(\cdot)$ is a digamma function. According to Equation (9), $r_{n,k}$ is interpreted as the contribution coming from the n-th observation to each component labeled with k.

The calculation of the responsibility requires the hyperparameters; however, the estimation of the hyperparameters also requires the responsibility. Accordingly, to optimize the hyperparameters, the above-described processes are iterated; first, the hyperparameters are randomly initialized; second, the responsibilities are calculated with the initial hyperparameters; third, the hyperparameters are updated with the calculated responsibility. After iterating the calculation, the optimized hyperparameters are obtained.

3.2. KL Divergence as a Criterion for Convergence

In the variational Bayesian estimation, the initial hyperparameters α_0, β_0, $\mathbf{m}_0$ and W_0 are given *a priori*. In other words, the initial hyperparameters, which specify the prior, can be determined to make

the convergence of the estimation results earlier. If the prior is set to a similar function as the posterior, most of the processes of the point density estimation are unnecessary. Accordingly, most of the point dataset has little effect on the point density; that is, most of the point dataset can be ignored. Moreover, the prior is accurately predictable as the adjacent past posterior, which was estimated at the previous iteration, because the point density of moving points does not change so much in a short period. The prior should therefore be set to the same function as the adjacent past posterior in order to stop the point density estimation early.

Kullback–Leibler (KL) divergence, which behaves like the distance between distribution functions, is applicable as a criterion for judging whether the calculation can be stopped or not.

Figure 3 illustrates such a judgment process. The point dataset is separated into small blocks in advance, where the block size is a constant parameter during the process. Note that the block size should be large enough to detect statistical significance. In Step 1, the prior is generated from the adjacent past posterior. In Step 2, to estimate the posterior, one of the small blocks (labeled "1") is input into the prior. In Step 3, the next prior, which is generated by the estimated posterior in Step 2, is updated with the block labeled "2". In this step, KL divergence from the posterior generated in Step 2 to the posterior generated in Step 3 is calculated. If the KL divergence is small enough, additional blocks are judged unnecessary. If not, the posterior is similarly calculated, and so on. In the evaluation of KL-divergence, the number of the input data is controlled; that is, few data are used when the point density changes a little, but many data are used when the point density changes drastically.

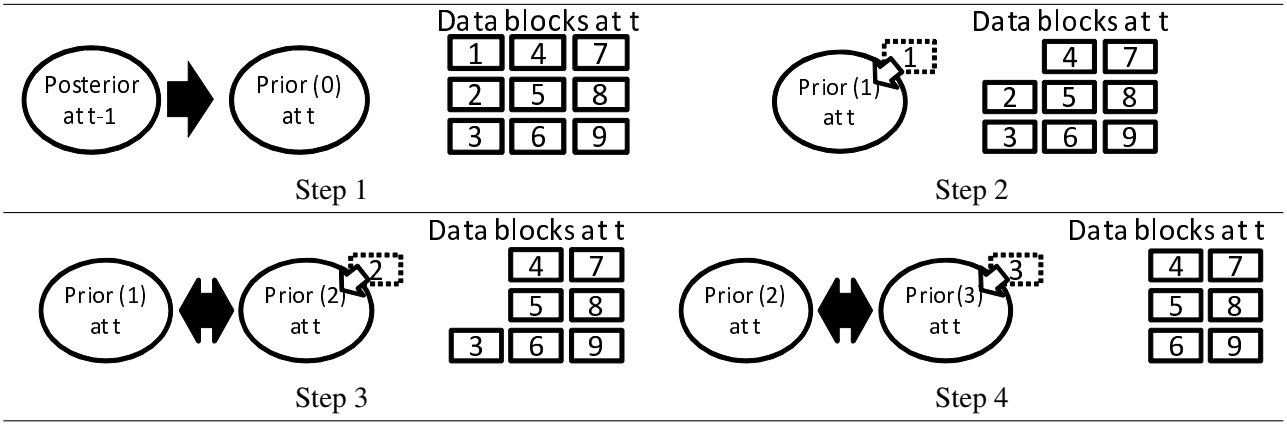


Figure 3. Serial intromission.

The KL divergence from $p(x)$ to $q(x)$ $KL(p(x)||q(x))$ is defined as:

$$KL(p(x)||q(x)) = \int p(x) \ln \frac{q(x)}{p(x)} \mathrm{d}x \tag{12}$$

The proposed algorithm requires the KL-divergence from the prior to the posterior. That is,

$$K \equiv KL(p(\{\theta_i\}|\{\mathbf{x}_n\}_{m+1})||p(\{\theta_i\}|\{\mathbf{x}_n\}_m)) \tag{13}$$

where $\{\theta_i\} = \{\{\pi_k\}, \{\Lambda_k\}, \{\mu_k\}\}$, and $\{\mathbf{x}_n\}_m$ is the m-th data block. If K is small enough, the estimation result is considered to be converged. $p(\{\theta_i\}|\{\mathbf{x}_n\}_m)$ can be replaced by $q(\{\theta_i\}|\{\mathbf{x}_n\}_m) =$

$q(\{\pi_k, \Lambda_k, \mu_k\}|\{\alpha_k, \beta_k, \mathbf{m}_k, \nu_k, W_k\})$, because it is the result of the variational Bayesian estimation. K is thus transformed to:

$$\begin{aligned} K &= \int q(\{\theta_i\}|\{\alpha_k, \beta_k, \mathbf{m}_k, \nu_k, W_k\}) \ln q(\{\theta_i\}|\{\alpha'_k, \beta'_k, \mathbf{m}'_k, \nu_k, W'_k\}) \prod_i \mathrm{d}\theta_i \\ &= \ln \alpha_0 + \sum (\alpha_0 - 1) \ln \tilde{\pi}_k - \ln \beta'_k - \frac{\beta'_k}{2} \left(\frac{2}{\beta_k} + \nu_k (\mathbf{m}_k - \mathbf{m}'_k)^{\mathrm{T}} W_k (\mathbf{m}_k - \mathbf{m}'_k) \right) \\ &+ \ln B(W'_k, \nu'_k) + \frac{1}{2}(\nu'_k - 1) \ln |\tilde{\Lambda}_k| - \sum_{i,j} \frac{W'^{-1}_{i,j}}{W^{-1}_{i,j}} \end{aligned} \tag{14}$$

where $\{\alpha_k, \beta_k, \mathbf{m}_k, \nu_k, W_k\}$ is the set of hyperparameters updated by $\{\mathbf{x}_n\}_m$ and $\{\alpha'_k, \beta'_k, \mathbf{m}'_k, \nu_k, W'_k\}$ is the set of hyperparameters updated by $\{\mathbf{x}'_n\}_{m+1}$. Here, $B(\cdot)$ is defined as:

$$B(W, \nu) \equiv |W|^{-\frac{1}{2}} \left(2^{\nu} \pi^{\frac{1}{2}} \Gamma(\frac{\nu}{2}) \Gamma(\frac{\nu - 1}{2}) \right)^{-1} \tag{15}$$

3.3. Responsibility Reduction

With the proposed method, a trick named "responsibility reduction" is applied to determine the hyperparameters. For example, as shown in Equation (5), $\beta_{t,k} = \beta_0 + N_{t,k}$, where $N_{t,k}$ is the sum of the responsibilities, which are contributed by the k-th component of the GMMat time t. Thus, $N_{t,k}$ corresponds to the number of points belonging to the k-th component, because responsibility $r_{n,k}$ indicates the i-th point's "belongingness" to the k-th component. Then, $\beta_{t,k}$ is updated with $N_{t+1,k}$ as $\beta_{t+1,k} = \beta_0 + N_{t,k} + N_{t+1,k}$, where $\beta_{t+1,k}$ is a hyperparameter of a posterior at $t + 1$. Similarly,

$$\beta_{t+n,k} = \beta_0 + N_{t,k} + N_{t+1,k} + \cdots + N_{t+n,k} \tag{16}$$

The hyperparameter at time $t + n$ thus gets larger if n gets larger. Consequently, the contribution of the newest point dataset, $N_{t+n,k}$, is relatively smaller than the contribution of the prior, β_0, because the prior includes the sum of the all past responsibilities. However, the contribution of the newest point dataset should be greater than the others.

The problem described in the preceding paragraph comes from confusion between point datasets. The new point density is different from the past point density; therefore, the past points and new points should not be equally used. Accordingly, the prior should be generated from the past posterior, but the prior should not be equal to the posterior. The effects of past points on the hyperparameters should thus be removed. If the effects from the hyperparameters are equal to the effects produced by the new points, the effects of points in the hyperparameters are kept constant. The dataset at time t is separated into M blocks containing enough points for inputting sequentially, where the number of points in the blocks are denoted by $n_t^{(1)}, n_t^{(2)}, \cdots, n_t^{(i)}, \cdots, n_t^{(M)}$. These blocks are input sequentially to update the posterior. When the j-th block is input, the ratio of the number of input points to the total number of points is calculated as $\frac{\sum_i^j n_t^{(i)}}{\sum_i^M n_t^{(i)}}$. The reduction rate of responsibilities $\gamma^{(j)}(t)$ is then defined as:

$$\gamma_t^{(j)} = 1 - \frac{\sum_i^j n_t^{(i)}}{\sum_i^M n_t^{(i)}} \tag{17}$$

The sum of the past responsibility $N_{t,k}$ is multiplied by the reduction rate. By this process, the total number of points contributing to the hyperparameters is adjusted and kept constant.

The updating mechanism is figuratively drawn as an "evaporating pot" in Figure 4. In Step (a), the pot is filled by a liquid, labeled t. The content of the pot is then evaporated so that half of its capacity remains. Next, a new liquid, labeled $t+1$, is poured into the pot. The volume of the poured liquid is half of the pot capacity. In step (b), the contents of the pot are again evaporated to half its capacity. The liquid labeled t and the liquid labeled $t+1$ are reduced equally to half of their initial volumes. That is, the volumes of the liquids labeled t and $t+1$ become 1/4 of the pot capacity. Then, repeatedly, a new liquid, labeled by $t+2$, is poured into the pot, which is half full of the liquid. After repeating the process, as shown in Steps (c) and (d), the ingredients in the pot are a mixture of liquids labeled t, $t+1, \cdots$. The volume of liquid labeled t' is therefore $\left(\frac{1}{2}\right)^{t'-t+1} V$, where V is the pot capacity. Thus, the volume of liquid will be decreased exponentially.

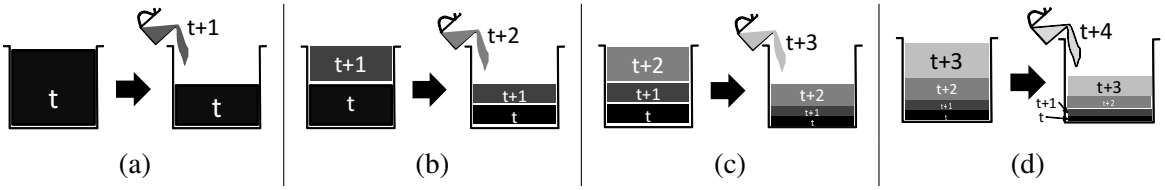


Figure 4. Evaporating pot model. (**a**) First pouring at t; (**b**) pouring at $t+1$; (**c**) pouring at $t+2$; (**d**) pouring at $t+3$.

The liquid in Figure 4 corresponds to the sum of responsibilities; in other words, evaporation corresponds to multiplication to the sum of responsibilities by the reduction rate. Accordingly, the contribution from the old dataset is decreased exponentially by multiplying it by the reduction rate, while the sum of all responsibilities is kept constant. By the multiplication, the hyperparameters in the prior at time $t+1$ given by the posterior at time t are derived as follows.

$$\alpha_{0,t+1}^{(0)} = \alpha_0 + \gamma_t^{(0)} N_{t,k} \tag{18}$$

$$\beta_{0,t+1}^{(0)} = \beta_0 + \gamma_t^{(0)} N_{t,k} \tag{19}$$

$$\mathbf{m}_{0,t+1}^{(0)} = \frac{1}{\beta_{0,t+1}} \left(\beta_0 \mathbf{m}_0 + \gamma_t^{(0)} N_{t,k} \bar{\mathbf{x}}_{t,k} \right) \tag{20}$$

$$W_{0,t+1}^{-1(0)} = W_0^{-1} + \gamma_t^{(0)} S_{t,k} N_{t,k} + \frac{\gamma_t^{(0)} N_{t,k} \beta_0}{\gamma_t^{(0)} N_{t,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})^{\mathrm{T}} \tag{21}$$

$$\nu_{0,t+1}^{(0)} = \nu_0 + \gamma_t^{(0)} N_{t,k} \tag{22}$$

Here, $N_{t,k}, \bar{\mathbf{x}}_{t,k}$ and $S_{t,k}$ at time t are given by Equation (9). The hyperparameters in the posterior at time $t+1$ are given by:

$$\alpha_{t+1,k}^{(0)} = \alpha_0 + \gamma_t^{(0)} N_{t,k} + n_{t,k}^{(0)} \tag{23}$$

$$\beta_{t+1,k}^{(0)} = \beta_0 + \gamma_t^{(0)} N_{t,k} + n_{t,k}^{(0)} \tag{24}$$

$$\mathbf{m}_{t+1,k}^{(0)} = \frac{1}{\beta_{t+1,k}^{(0)}} \left(\beta_0 \mathbf{m}_0 + \gamma_t^{(0)} N_{t,k} \bar{\mathbf{x}}_{t,k} + n_{t,k}^{(0)} \bar{\mathbf{x}}_{t,k}^{(0)} \right) \tag{25}$$

$$\begin{aligned} W_{t+1,k}^{-1(0)} = {} & W_0^{-1} + \gamma_t^{(0)} S_{t,k} N_{t,k} + s_{t,k}^{(0)} n_{t,k}^{(0)} + \frac{\gamma_t^{(0)} N_{t,k} \beta_0}{\gamma_t^{(0)} N_{t,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})^{\mathrm{T}} \\ & + \frac{n_{t,k}^{(0)} \beta_0^{(0)}}{n_{t,k}^{(0)} + \beta_0^{(0)}} (\mathbf{m}_{0,t}^{(0)} - \bar{\mathbf{x}}_{t,k}^{(0)})(\mathbf{m}_{0,t}^{(0)} - \bar{\mathbf{x}}_{t,k}^{(0)})^{\mathrm{T}} \end{aligned} \tag{26}$$

$$\nu_{t+1,k}^{(0)} = \nu_0 + \gamma_t^{(0)} N_{t,k} + n_{t,k}^{(0)} \tag{27}$$

where:

$$n_{t,k}^{(0)} = \sum_n r_{t,n,k}^{(0)}, \quad \bar{\mathbf{x}}_{t,k}^{(0)} = \frac{1}{n_{t,k}^{(0)}} \sum_n r_{t,n,k} \mathbf{x}_{t,n}^{(0)}, \quad s_{t,k}^{(0)} = \sum_n r_{t,n,k}^{(0)} (\mathbf{x}_{t,n}^{(0)} - \bar{\mathbf{x}}_{t,k}^{(0)})(\mathbf{x}_{t,n}^{(0)} - \bar{\mathbf{x}}_{t,k}^{(0)})^{\mathrm{T}} \tag{28}$$

$\mathbf{x}_{t,n}^{(0)}$ and $r_{t,n,k}^{(0)}$ are points in the n-th block and their responsibilities, respectively.

Another block of the new point dataset is input if the KL-divergence between the posteriors is large. The prior is updated with the i-th block by the following formulas:

$$\alpha_{0,t+1}^{(i)} = \alpha_0 + \gamma_t^{(i)} N_{t,k} + \sum_j^{i-1} n_{t,k}^{(j)} \tag{29}$$

$$\beta_{0,t+1}^{(i)} = \beta_0 + \gamma_t^{(i)} N_{t,k} + \sum_j^{i-1} n_{t,k}^{(j)} \tag{30}$$

$$\mathbf{m}_{0,t+1}^{(i)} = \frac{1}{\beta_{0,t+1}^{(i)}} \left(\beta_0 \mathbf{m}_0 + \gamma_t^{(i)} N_{t,k} \bar{\mathbf{x}}_{t,k} + \sum_j^{i-1} n_{t,k}^{(j)} \bar{\mathbf{x}}_{t+1,k}^{(j)} \right) \tag{31}$$

$$\begin{aligned} W_{0,t+1}^{-1(i)} = {} & W_0^{-1} + \gamma_t^{(i)} S_k N_k + \sum_j^{i-1} s_{t+1,k}^{(j)} n_{t+1,k}^{(j)} + \frac{\gamma_t^{(i)} N_{t,k} \beta_0}{\gamma_t^{(i)} N_{t,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})^{\mathrm{T}} \\ & + \sum_j^{i-1} \frac{n_{t+1,k} \beta_{0,t+1}^{(j)}}{n_{t+1,k} + \beta_{0,t+1}^{(j)}} (\mathbf{m}_{0,t+1}^{(j)} - \bar{\mathbf{x}}_{t+1,k})(\mathbf{m}_{0,t+1}^{(j)} - \bar{\mathbf{x}}_{t+1,k})^{\mathrm{T}} \end{aligned} \tag{32}$$

$$\nu_{0,t+1}^{(i)} = \nu_0 + \gamma_t^{(i)} N_{t,k} + \sum_j^{i-1} n_{t,k}^{(j)} \tag{33}$$

To obtain the recursive formula, each side of the $i+1$-th formula is subtracted by each side of the i-th formula as follows.

$$\beta_{0,t+1}^{(i+1)} = (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} + \beta_{0,t+1}^{(i)} + n_{t,k}^{(i)} \tag{34}$$

Additionally, $\beta_k^{(i)} = \beta_0^{(i)} + n_k^{(i)}$, so:

$$\beta_0^{(i+1)} = \beta_{t,k}^{(i)} + (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} \tag{35}$$

Similarly, the following formulas are derived.

$$\alpha_{0,t+1}^{(i)} = \alpha_{t,k}^{(i)} + (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} \tag{36}$$

$$\mathbf{m}_{0,t+1}^{(i)} = \frac{1}{\beta_{0,t+1}^{(i)}} \left(\beta_{t+1,k}^{(i)} \mathbf{m}_{t+1,k}^{(i)} + (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} \bar{\mathbf{x}}_{t,k} \right) \tag{37}$$

$$\begin{aligned} W_{0,t+1}^{-1(i)} &= W_{t,k}^{-1(i)} + (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} S_{t,k} \\ &+ \left(\frac{\gamma_t^{(i+1)} N_{t,k} \beta_0}{\gamma_t^{(i+1)} N_{t,k} + \beta_0} - \frac{\gamma_t^{(i)} N_{t,k} \beta_0}{\gamma_t^{(i)} N_{t,k} + \beta_0} \right) (\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})^{\mathrm{T}} \end{aligned} \tag{38}$$

$$\nu_{0,t+1}^{(i)} = \nu_{t,k}^{(i)} + (\gamma_t^{(i+1)} - \gamma_t^{(i)}) N_{t,k} \tag{39}$$

The $i+1$-th prior can therefore be calculated from $\alpha_{t,k}^{(i)}$, $\beta_{0,t+1}^{(i)}$, $\mathbf{m}_{t+1,k}^{(i)}$, $W_{t,k}^{-1(i)}$, $\nu_{t,k}^{(i)}$, $\gamma_t^{(i+1)}$ and $\gamma_t^{(i)}$. After iterating this update computation until KL divergence becomes small enough, the estimated hyperparameters at time $t+1$ is obtained. When KL divergence is converged at $i = l$, the hyperparameters of the next prior are written as:

$$\alpha_{0,t+1,k} = \alpha_0 + N_{t+1,k} \tag{40}$$

$$\beta_{0,t+1,k} = \beta_0 + N_{t+1,k} \tag{41}$$

$$\mathbf{m}_{0,t+1,k} = \frac{1}{\beta_{0,t+1,k}} (\beta_0 \mathbf{m}_0 + N_{t+1,k} \bar{\mathbf{x}}_{t+1,k}) \tag{42}$$

$$W_{0,t+1,k}^{-1} = W_0^{-1} + S_{t+1,k} N_{t+1,k}, + \frac{N_{t+1,k} \beta_0}{N_{t+1,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})^{\mathrm{T}} \tag{43}$$

$$\nu_{0,t+1,k} = \nu_0 + N_{t+1,k} \tag{44}$$

by $N_{t+1,k}, \bar{\mathbf{x}}_{t+1,k}$ and $S_{t+1,k}$, which are defined as:

$$N_{t+1,k} = \alpha_{t,k}^{(l)} - \alpha_0 = \beta_{t,k}^{(l)} - \beta_0 \tag{45}$$

$$\bar{\mathbf{x}}_{t+1,k} = \frac{1}{N_{t+1,k}} \left(\beta_{t+1,k}^{(l)} \mathbf{m}_{t+1,k}^{(l)} - \beta_0 \mathbf{m}_0 \right) \tag{46}$$

$$S_{t+1,k} = \frac{1}{N_{t+1,k}} \left(W_{t,k}^{-1(l)} - W_0^{-1} - \frac{N_{t+1,k} \beta_0}{N_{t+1,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})^{\mathrm{T}} \right) \tag{47}$$

The updated formula at $t+2$ is similarly derived. Note that $N_{t+1,k}, \bar{\mathbf{x}}_{t+1,k}, S_{t+1,k}$ can be converted to the following formulas.

$$N_{t+1,k} = \gamma_t^{(l)} N_{t,k} + \sum_j^l n_{t+1,k}^{(j)} \tag{48}$$

$$\bar{\mathbf{x}}_{t+1,k} = \gamma_t^{(l)} N_{t,k} \bar{\mathbf{x}}_{t,k} + \sum_j^l n_{t+1,k}^{(j)} \bar{\mathbf{x}}_{t+1,k}^{(j)} \tag{49}$$

$$S_{t+1,k} = \gamma_t^{(l)} S_{t,k} N_{t,k} + \sum_j^l s_{t+1,k}^{(j)} n_{t+1,k}^{(j)} + \delta W \tag{50}$$

δW is defined as:

$$\delta W = \frac{\gamma_t^{(i)} N_{t,k} \beta_0}{\gamma_t^{(i)} N_{t,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t,k})^{\mathrm{T}} - \frac{N_{t+1,k} \beta_0}{N_{t+1,k} + \beta_0} (\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})(\mathbf{m}_0 - \bar{\mathbf{x}}_{t+1,k})^{\mathrm{T}}$$
$$+ \sum_j^l \frac{n_{t+1,k} \beta_{0,t+1}^{(j)}}{n_{t+1,k} + \beta_{0,t+1}^{(j)}} (\mathbf{m}_{0,t+1}^{(j)} - \bar{\mathbf{x}}_{t+1,k})(\mathbf{m}_{0,t+1}^{(j)} - \bar{\mathbf{x}}_{t+1,k})^{\mathrm{T}} \quad (51)$$

In comparison to Equation (9), the formulas for N and $\bar{\mathbf{x}}$ are natural; that is, the stack of the contributions from the past data is reduced and added to the current contribution. However, S seems similar to the other hyperparameters, except δW. δW represents the error caused by sequential estimation, so the effect of δW on estimation accuracy is quite small if N and n are large enough. Accordingly, in the case of the proposed algorithm, δW is ignored.

The proposed algorithm is shown in Figure 5. A normal variational Bayesian estimation is carried out first. The estimated posterior at time t is used to generate the prior at the next time $t + 1$. Each dataset is separated into many blocks, and the blocks are sequentially input to update the prior generated by the last estimated posterior. The block intromission is stopped when the KL divergence between the prior and the posterior is small enough. When that condition is met, the estimation of the point density at time t is finished. After the estimation, time t is incremented into $t+1$, and the estimation of the point density at time $t + 1$ is started. These processes are carried out iteratively to estimate the point density.

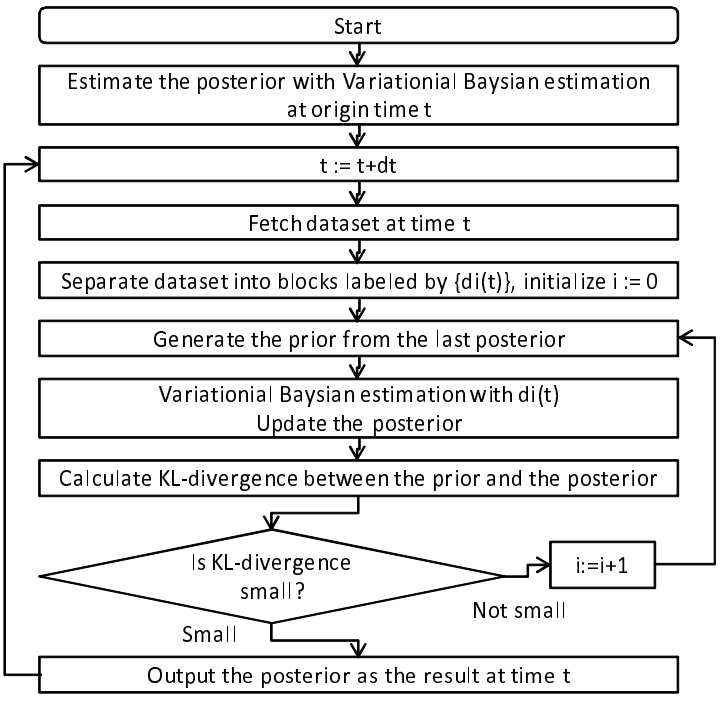


Figure 5. Proposed algorithm.

The parameters used for this algorithm are hyperparameters used by the variational Bayesian estimation and the number of blocks. They are determined manually in the same manner as a general variational Bayesian estimation. Note that the number of the blocks should be much smaller than the

size of the entire dataset in order to obtain blocks large enough for detecting statistical significance. Satisfying this condition is generally easy because the entire data size is extremely large.

4. Experiments

4.1. Experimental Settings and Criteria

The proposed algorithm was evaluated experimentally. The features of the point dataset used for this evaluation are listed in Table 1. The point datasets, named "people flow data" [28], describe the positions of people during a day. The point datasets were generated from the result of questionnaires about people's travels during a day, namely data pairs of the origination and destination of the journeys. Interpolations between the origination and the destination were carried out in order to generate point datasets every minute. Therefore, the interpolation points at the same time were collected as one dataset. The point density from 0:00 a.m. to 12:00 a.m. was estimated by the proposed algorithm and by two conventional kernel density estimators. The points, which are expressed as longitude and latitude coordinates, were converted to an orthogonal coordinate system of which the origin is the centroid of points at 0:00 a.m.. Moreover, the coordinate values of the points in the orthogonal coordinate system were scaled to adjust their maximal values to 10.0.

Table 1. PFLOWdataset summary.

Title	Value
Date	0:00 a.m.–12:00 a.m. in a day
Sampling rate	1 point per minute
Used dataset	600,000 trajectories

The dataset was separated into two blocks. One block was used to estimate the point density; the other was used for evaluating the accuracy criteria. This is generally called "holdout validation". Five thousand points included in the second block were actually used, because it would take too much time to carry out the evaluation if all of the points in the evaluation block were used.

Two criteria were evaluated: accuracy and processing time. As the accuracy criterion, Ev based on logarithmic likelihood, defined as:

$$\mathrm{Ev} = \frac{1}{N} \ln \prod_{n=0}^{N} P(\mathbf{x}_n) \tag{52}$$

was applied. Here, N is the number of points denoted by $\mathbf{x}_n$. Ev indicates the reproducibility of the point density of the entire dataset. As the processing-time criterion, the time for generating a 150-pixel $\times$ 100-pixel heat-map image was used; that is, the calculation was carried out 15,000 times. To evaluate the criteria accurately, the measurement of Ev and the processing time was carried out three times, and the average of the results was used as the criterion.

As the parameters used by the proposed algorithm, the number of mixed Gaussian distributions was set to 200; the block size for sequential intromission was 1,500; the initial hyperparameters were

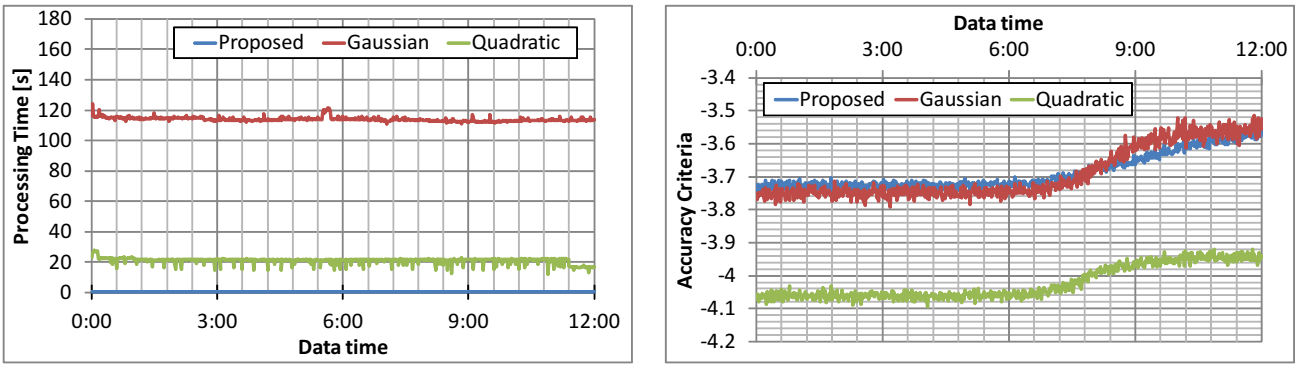
$\alpha_0 = 2.1$ and $\beta_0 = 1.0$; diagonal components of W_0 were set to 1.5; and non-diagonal components were set to 0.0 and $\nu_0 = 2.2$. In addition, 200 points were randomly selected from the initial point dataset as the $\mathbf{m}_0$. These parameters were determined manually to obtain the highest Ev.

As conventional methods, the Gaussian kernel method and the quadratic kernel method were similarly evaluated. Standard deviation h was set to 0.2 for the Gaussian kernel method; radius h was set to 3.0 for the quadratic kernel method in order to avoid the $P(\mathbf{x}_n) = 0$ (which gives $\mathrm{Ev} = -\infty$).

Incidentally, the program used for the evaluations was implemented with Java on a computer with an Intel Core i7-3980K 3.2 GHz and 32.0 GB RAM.

4.2. Results

The processing times for each method are plotted in Figure 6a. The people's positions at the time indicated by "dataset time" on the horizontal axis are included in the dataset used for the evaluation. Hereafter, "dataset" time is used in the same meaning. The processing time for the proposed algorithm was less than 1.0 s, that for the Gaussian kernel method around 250 s and that for the quadratic kernel method 80 s. The proposed algorithm is thus the fastest of the three methods. The accuracy criterion for each method is plotted in Figure 6b. The accuracy criterion for the proposed algorithm is close to that for the Gaussian-kernel density estimator, and that of the quadratic-kernel density estimator is lowest. These results indicate that the proposed algorithm is the fastest and accurate enough.



(a) (b)

Figure 6. Results of evaluations. (**a**) Processing-time comparison; (**b**) accuracy comparison.

Distributions obtained by the three methods are compared in Figure 7. The red regions in the images indicate high-density areas, the yellow regions medium density and the green regions sparse areas. Image (a), which represents the density distribution given by the proposed algorithm, is similar to that given by the Gaussian-kernel estimator, namely Image (b), but it is diffused a bit. Image (c), which shows the result of the quadratic kernel estimation, is diffused too much to describe the point density. This diffuseness is caused by large h; therefore, the image generated with the result of the quadratic kernel with $h = 1.0$ is similar to Images (a) and (b), as shown in Image (d). However, the accuracy criteria for the quadratic-kernel method with $h = 1.0$ is $-\infty$; that is, the distribution is not accurate at all. Accordingly, the proposed algorithm generates the most unblurred accurate heatmap in these three algorithms.

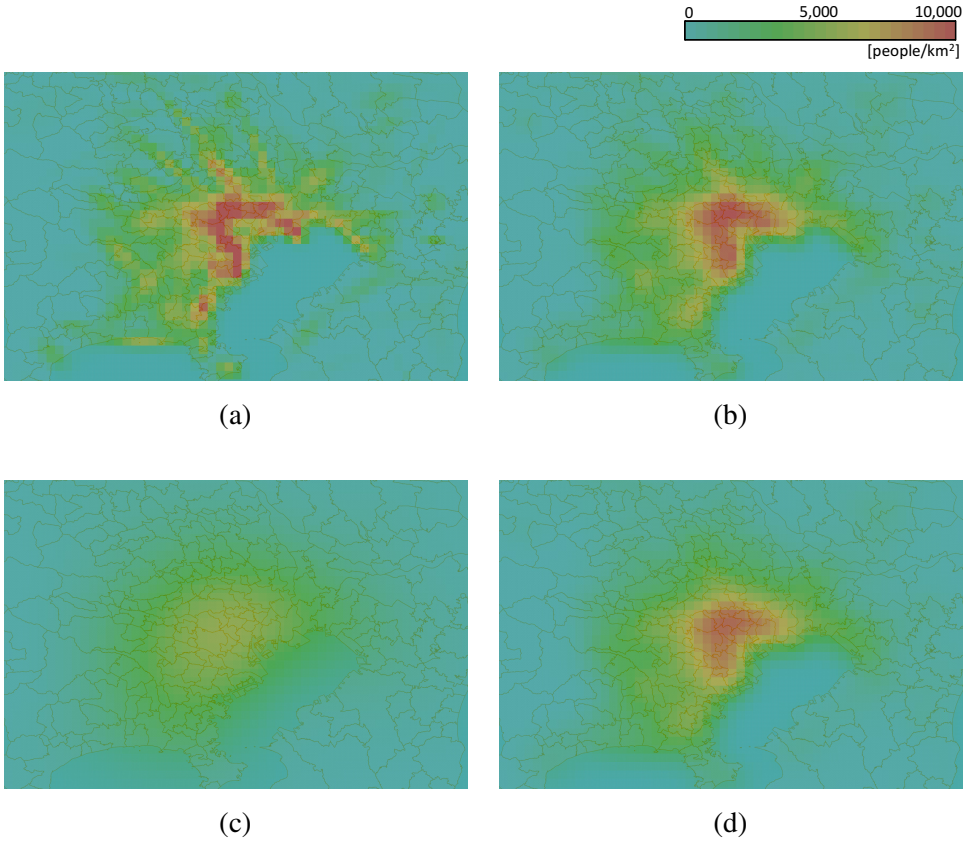
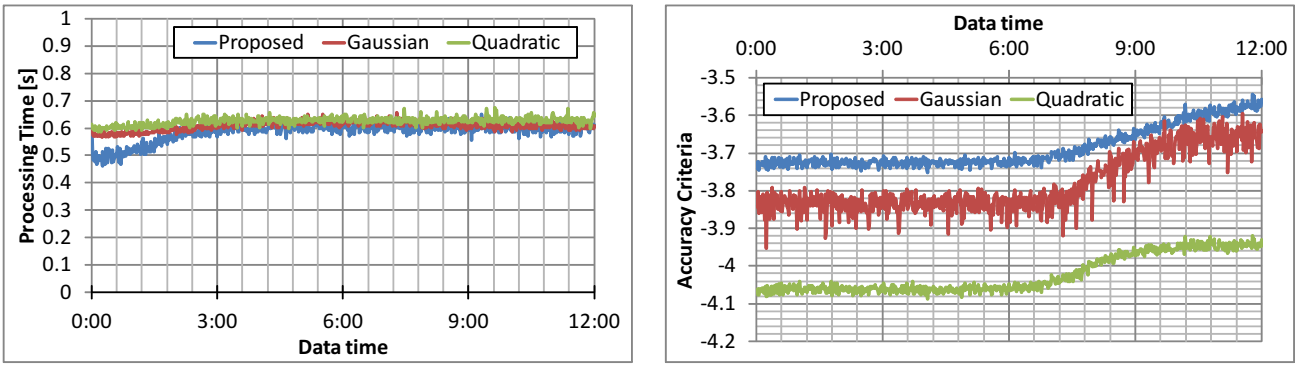


Figure 7. Visualized distributions (7:00 a.m.). (**a**) Proposed algorithm; (**b**) Gaussian-kernel estimator; (**c**) quadratic-kernel estimator ($h = 3.0$); (**d**) quadratic-kernel estimator ($h = 1.0$).

4.3. Discussion

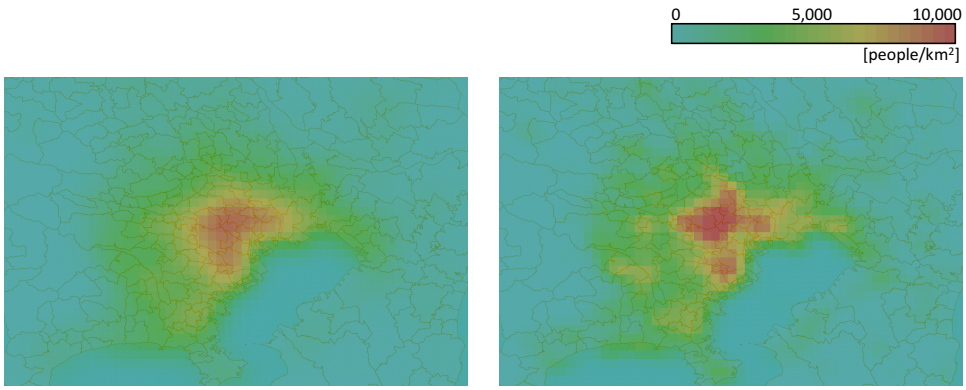
Sub-sampling is one of the most popular methods for speeding up density estimation processes. Accordingly, a sub-sampled dataset was also evaluated. Figure 8a plots a series of processing times after the point dataset was randomly sub-sampled to 2300 points for the Gaussian-kernel estimator ($h = 3.0$) and 17,000 points for the quadratic-kernel estimator. Both the Gaussian-kernel estimator and the quadratic-kernel estimator took 0.6 s under this condition. These times are comparable with that taken by the proposed algorithm; however, the accuracies of the two former estimators is lower, as shown in Figure 8b. It is thus concluded that the proposed algorithm maintains high accuracy even when the processing time was reduced.



(a) (b)

Figure 8. Comparison with sub-sampling. (**a**) Processing-time comparison; (**b**) accuracy comparison.

Heatmaps generated from sub-sampled datasets are shown in Figure 9a,b. The heatmap given by the quadratic-kernel estimator (Figure 9a) is very similar to the heatmap obtained without sub-sampling, which is shown in Figure 7c, because the sub-sampled dataset is large enough. However, the heatmap given by the Gaussian-kernel estimator (Figure 9b) is quite different from Figure 7b. This implies that the sub-sampled dataset used by the Gaussian-kernel estimator is too small to generate a heatmap. As shown in Figure 10, to compare heatmaps at 7:01 a.m., 7:02 a.m., 7:03 a.m. and 7:04 a.m., a generated heatmap depends on points chosen at random, that is the heatmap is changed randomly. Because the Gaussian-kernel estimator is slower than the quadratic-kernel estimator, fewer points were used. That means the quadratic-kernel estimator is more feasible for generating heatmaps quickly than the Gaussian-kernel estimator. However, as shown in Figure 8, the proposed algorithm can generate more accurate heatmaps than that generated by the quadratic kernel estimator.



(a) (b)

Figure 9. Visualized distributions after sub-sampling (7:00 a.m.). (**a**) Quadratic-kernel estimator (h = 1.0); (**b**) Gaussian-kernel estimator.

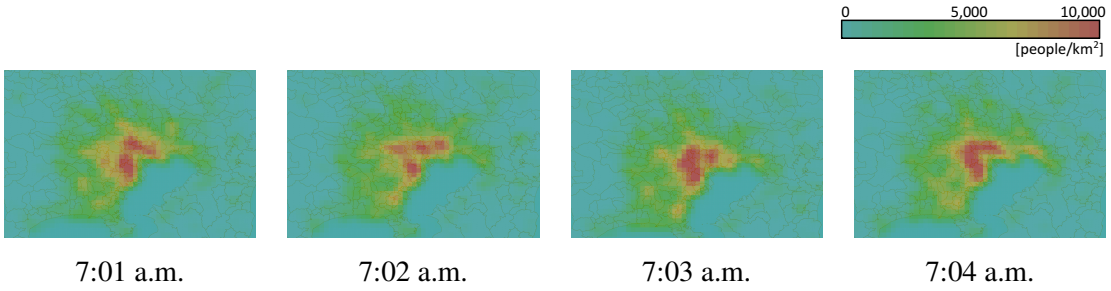


Figure 10. Visualized distributions by the Gaussian-kernel estimator after subsampling.

For ensuring these results, Ev's of the Gaussian-kernel estimator and the quadratic-kernel estimator under various h's are calculated with 10,000 points at 0:00 a.m.. In the case of the Gaussian-kernel estimator, $\mathrm{Ev} = -\infty$ at $h = 0.05$, $\mathrm{Ev} = -4.00$ at $h = 0.1$, $\mathrm{Ev} = -4.11$ at $h = 0.2$, $\mathrm{Ev} = -4.28$ at $h = 0.5$ and $\mathrm{Ev} = -4.41$ at $h = 1.0$. Similarly, in the case of the quadratic-kernel, $\mathrm{Ev} = -\infty$ at $h = 1.0$, $\mathrm{Ev} = -\infty$ at $h = 2.0$, $\mathrm{Ev} = -4.40$ at $h = 3.0$, $\mathrm{Ev} = -4.47$ at $h = 4.0$ and $\mathrm{Ev} = -4.53$ at $h = 5.0$. The Ev shown in the Figure 9b are comparable with these Ev. In addition, the proposed algorithm achieved $\mathrm{Ev} = -3.79$, which is also comparable with Ev shown in Figure 9b. That implies that the evaluation setting like h and the number of point data are not significant for the accuracy evaluation.

According to the results of the experiment described above, the proposed algorithm achieved faster performance than two existing algorithms, namely a quadratic-kernel density estimator and a Gaussian-kernel density estimator. Moreover, by subsampling, the kernel density estimators can be accelerated to speeds comparable with that of the proposed algorithm; however, they show lower accuracy. Therefore, it is confirmed that the proposed algorithm has advantages over the existing kernel-density estimators.

5. Concluding Remarks

A method for estimating the density of moving points, based on a variational Bayesian estimation, was proposed. The prior in the estimation is generated from the posterior given by the previous estimation. Responsibilities used in the posterior formula should be reduced in order to generate the prior; therefore, a trick named "responsibility reduction" was introduced. Moreover, the processing time and the estimation accuracy of the proposed algorithm were experimentally evaluated. The proposed algorithm was faster than two conventional methods; moreover, the estimation accuracy of the proposed algorithm is comparable with those of the conventional methods. These results demonstrate that the proposed algorithm is fast without loss of accuracy. Therefore, it is concluded that a massive moving-point dataset, which can be encoded with OGC Moving Features, can be converted to a heatmap, which is distributable with OGC WMS and WCS, in real time.

As future work, the proposed algorithm will be extended to handle the non-GMM distributions. The responsibility reduction is applicable to other distributions. The the extended algorithm will thus be further developed to speed up the point density estimation.

Author Contributions

Akinori Asahara invented the algorithm, conducted the experiments shown in this article and wrote this article. Hideki Hayashi collected the data used for the experiment. Takachi Kai reviewed and approved this article.

Conflicts of Interest

Akinori Asahara is the inventor stated in the patent application regarding the point density estimation algorithm disclosed in this article.

References

1. Ryosuke, S. OGC Standard for Moving Features; Requirements (12-117r1). Available online: https://portal.opengeospatial.org/files/?artifact_id=51623 (accessed on 26 February 2015).
2. Geospatial Information Authority of Japan. Fundamental Geospatial Data 1:25000. Available online: http://www.gsi.go.jp/kiban/index.html (accessed on 26 February 2015). (In Japanese)
3. Manandhar, D.; Kawaguchi, S.; Uchida, M.; Ishii, M.; Tomohiro, H. IMES for mobile users. Social implementation and experiments based on existing cellular phones for seamless positioning. In Proceedings of the International Symposium on GPS/GNSS 2008, Tokyo, Japan, 11–14 November 2008.
4. Loopt. Loopt®. Available online: http://www.loopt.com/ (accessed on 1 December 2012).
5. Open Geospatial Consortium. Available online: http://www.opengeospatial.org/ (accessed on 26 February 2015).
6. Open Geospatial Consortium. OGC(R) Moving Features. Available online: http://www.opengeospatial.org/standards/movingfeatures (accessed on 26 February 2015).
7. Open Geospatial Consortium. OGC Network Common Data Form (NetCDF) Core Encoding Standard Version 1.0 (10-090r3). Available online: http://www.opengeospatial.org/standards/netcdf (accessed on 26 February 2015).
8. Open Geospatial Consortium. OGC(R) GML Application Schema—Coverages (1.0.1) (09-146r2). Available online: http://www.opengeospatial.org/standards/gml (accessed on 26 February 2015).
9. Open Geospatial Consortium. OpenGIS Web Map Service (WMS) Implementation Specification 1.3.0 (06-042). Available online: http://www.opengeospatial.org/standards/wms (accessed on 26 February 2015).
10. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006.
11. Samet, H. The quadtree and related hierarchical data structures. *ACM Comput. Surv. (CSUR)* **1984**, *16*, 187–260.
12. Open Geospatial Consortium. OGC(R) WCS 2.0 Interface Standard—Core, Version 2.0.1 (09-110r4). Available online: http://www.opengeospatial.org/standards/wcs (accessed on 26 February 2015).

13. Chazal, F.; Chen, D.; Guibas, L.; Jiang, X.; Sommer, C. Data-driven trajectory smoothing. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '11), Chicago, IL, USA, 1–4 November 2011; ACM: New York, NY, USA, 2011; pp. 251–260.
14. Lei, C.; Tamer, O.M.; Vincent, O. Robust and fast similarity search for moving object trajectories. In Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, Baltimore, MD, USA, 13–17 June 2005; ACM: New York, NY, USA, 2005; pp. 491–502.
15. Chudova, D.; Gaffney, S.; Mjolsness, E.; Smyth, P. Translation-invariant mixture models for curve clustering. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 24–27 August 2003; ACM: New York, NY, USA, 2003; pp. 79–88.
16. Elias, F.; Kostas, G.; Yannis, T. Index-based most similar trajectory search. In Proceedings of the IEEE 23rd International Conference on Data Engineering, Istanbul, Turkey, 15–20 April 2007; pp. 816–825.
17. Trajcevski, G.; Ding, H.; Scheuermann, P.; Tamassia, R.; Vaccaro, D. Dynamics-aware similarity of moving objects trajectories. In Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems, Seattle, Washington, DC, USA, 7–9 November 2007; ACM: New York, NY, USA, 2007; pp. 1–8.
18. Asahara, A.; Maruyama, K.; Sato, A.; Seto, K. Pedestrian-movement prediction based on mixed Markov-chain model. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '11), Chicago, IL, USA, 1–4 November 2011; ACM: New York, NY, USA, 2011; pp. 25–33.
19. Asahara, A.; Maruyama, K.; Shibasaki, R. A mixed autoregressive hidden-markov-chain model applied to people's movements. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '12), Redondo Beach, CA, USA, 7–9 November 2012; ACM: New York, NY, USA, 2012; pp. 414–417.
20. Lee, J.G.; Han, J.; Whang, K.Y. Trajectory clustering: A partition-and-group framework. In Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, Beijing, China, 11–14 June 2007; ACM: New York, NY, USA, 2007; pp. 593–604.
21. Nirvana, M.; Rolf, A.D.B. Aggregation and comparison of trajectories. In Proceedings of the 10th ACM International Symposium on Advances in Geographic Information Systems, McLean, VA, USA, 8–9 November 2002; ACM: New York, NY, USA, 2002; pp. 49–54.
22. Van Kreveld, M.; Wiratma, L. Median trajectories using well-visited regions and shortest paths. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '11), Chicago, IL, USA, 1–4 November 2011; ACM: New York, NY, USA, 2011; pp. 241–250.
23. Asakura, Y.; Hato, E. Tracking survey for individual travel behaviour using mobile communication instruments. *Transp. Res. C Emerg. Technol.* **2004**, *12*, 273–291.
24. Silverman, B.W. *Density Estimation for Statistics and Data Analysis*; Chapman and Hall/CRC: London, UK, 1986.
25. Simonoff, J.S. *Smoothing Methods in Statistics*; Springer: Berlin, Germany, 1996.

26. Jansenberger, E.M.; Staufer-Steinnocher, P. Dual kernel density estimation as a method for describing spatio-temporal changes in the upper Austrian food retailing market. In Proceedings of the 7th AGILE Conference on Geographic Information Science, Heraklion, Greece, 29 April–1 May 2004; Volume 29, pp. 551–558.
27. Krisp, J.M.; Peters, S. Directed kernel density estimation (DKDE) for time series visualization. *Ann. GIS* **2011**, *17*, 155–162.
28. Center for Spatial Information Science, The University of Tokyo. People Flow Project (PFLOW), Collaborative Research: "Urban Space Information Platform Able to Assimilate Dynamic Data". Available online: http://pflow.csis.u-tokyo.ac.jp/ (accessed on 1 August 2014).