Incremental Route Refinement for GPS-enabled Cellular Phones

Naoharu Yamada[*,**], Yoshinori Isoda[***], Masateru Minami[****], and Hiroyuki Morikawa[*****]

[*]Service & Solution Development Department, NTT DOCOMO, INC.
3-6 Hikari-no-oka, Yokosuka-Shi, Kanagawa, 239-8536 Japan, yamadana@nttdocomo.co.jp
[**] RCAST, The University of Tokyo
[***] Service & Solution Development Department, NTT DOCOMO, INC., isoday@nttdocomo.co.jp
[****] RCAST, The University of Tokyo, minami@mlab.t.u-tokyo.ac.jp
[*****] RCAST, The University of Tokyo, mori@mlab.t.u-tokyo.ac.jp

ABSTRACT

This paper proposes significant route identification; the concept is to combine the locations acquired by the GPS module of the user's cellular phone. Significant routes are those that the user most frequently passes along. The proposal overcomes the two main weaknesses of cellular phones. One is that GPS data is acquired infrequently in order to reduce the phone's power consumption, and the other is the poor spatial accuracy of the cellular phone's commercial GPS module. In order to overcome these weaknesses, the proposed approach incrementally refines the stored routes whenever the user passes along them. In more detail, the system compares the newly acquired GPS points to the stored points and appends the new data so as to decrease route ambiguity. A experiment with 6 subjects over periods of 6-14 months demonstrates that the more often a user passes along a significant route, the more precisely the proposed approach can identify it.

Keywords: location-based service, GPS, significant routes, incremental route refinement

1 INTRODUCTION

Most modern cellular phones include a GPS module. This allows location-based services to enter a new era by supporting a user not on just his/her current location but also his/her "significant routes". The significant routes are the user's familiar routes that s/he frequently passes along. The location-based services alter their responses depending on whether a significant route is being use or not. For example, if the user is now passing along a significant route, the system can estimate the next area to be passed or visited and update information about the shops and restaurants in that area. Otherwise, the system supports the user by providing navigation services, and information about popular sightseeing spots, shops and restaurants in this unfamiliar area.

Much research has been conducted to identify users' "significant locations" [3][5]. They define significant locations as the places that the user often visits. Unfortunately, these approaches are insufficient since they fail to identify the significant location when users moves, i.e. significant routes.

The three requirements described below should be satisfied for implementing truly effective services based on significant route identification.

(1) **Little GPS data:** Some works apply the clustering approach for significant route identification, which requires a lot of GPS data [7][8]. However, these approaches suffer the cold-start problem in that the users cannot enjoy any location-based service until a lot of GPS data has been acquired. The system must provide some form of service even no or only a little GPS data has been acquired.
(2) **Infrequent GPS activation:** Continuously acquiring GPS data drains the cellular phone very quickly. Since the capacity of a cellular phone's battery is limited, GPS data acquisition must be restricted so as not to disturb the normal use of the cellular phone. This results in sparse GPS points. Therefore, the system must be able to overcome the gaps in the route between two consecutive GPS points.
(3) **GPS spatial error:** GPS accuracy can be poor since the GPS module on a cellular phone uses the location of cell towers in case of acquiring less than four GPS satellites. This results in incorrect routes. Therefore, the system has to be able to refine the route by reducing the errors in location accuracy.

This paper proposes a location-based system that satisfies the above three requirements. The system incrementally refines the estimated routes with new GPS data when the user passes along the same route again, which satisfies requirement (1). The system adds newly acquired GPS data to the stored data of the estimated route, which satisfies requirement (2). In addition, the system incrementally refines the GPS data used to estimate the significant routes so as to minimize route ambiguity, which satisfies requirement (3). The proposed approach can precisely identify the significant routes since more data becomes available with user movement.

The remainder of this paper is organized as follows. Section 2 presents related work and clarifies the

contribution of this paper. In Section 3, we describe the incremental route updating system in detail. Before concluding, we describe an experiment that demonstrates the abilities of our proposal.

2 RELATED WORK

This section reviews existing location-based systems and explains the techniques used to estimate significant locations; the discussion emphasizes the importance of identifying the significant routes. We clarify the drawbacks of conventional significant route identification schemes and the contribution of our work.

Location-based systems are one part of the all-encompassing concept of pervasive computing introduced by Mark Weiser[9]. Location-based systems are able to respond to user context without explicit user intervention and thus aim at increasing usability and effectiveness by taking account of user location [2]. Early work on location-based systems attempted to identify just the user's current location [1][2]. While services based on this information are useful, many more attractive services could be created by identifying the user's movement tracks, his significant routes.

Most work attempts to identify the user's significant locations. comMotion[10] constantly acquires GPS data, and interprets a loss of signal as a significant location, i.e. the user has entered some building. Ashbrook et al.[5] use the K-means clustering algorithm, and Kang et al.[6] use a time-based clustering algorithm to identify areas in which the user spends a lot of time.

With these approaches, the system can identify the places where the user often stays such as *home*, *office*, and *favorite shops*. However, these approaches are insufficient since they cannot identify the user's significant routes. Examples of these routes include *commuter train*, *school route*, and *the road to the local shopping mall*.

Some work has tackled the identification of significant routes. Zhou et al.[3][4] use a density and join-based clustering algorithm to identify significant routes. Liao et al.[7][8] use a hierarchical Markov model and hierarchical Conditional Random Fields. However, all clustering approaches require a significant quantity of GPS data in advance, which is inconvenient for the user since s/he cannot enjoy the advanced services until enough GPS data is acquired. The system encounters three difficulties when trying to identify significant routes in the absence of a lot of GPS data. The first is that GPS data is acquired infrequently. Second, the user's movements mean that it is rare for two GPS observations to occur at the same location. Third is the inaccuracy inherent in the cellular phone's commercial GPS modules.

This paper proposes to precisely identify the significant routes by supplementing the route data with additional GPS data acquired when the user passed along again. The proposed system can roughly identify significant routes even if the stored GPS data is sparse.

3 INCREMENTAL ROUTE REFINEMENT SYSTEM

3.1 System Component

Figure 1 shows the incremental route refinement system. This system consists of an incremental route refinement server and GPS-enabled cellular phones. The GPS module periodically acquires 4-tuple data set of latitude, longitude, error, and time as a GPS point. The incremental route refinement server has three functions: estimating the current route from GPS data, identifying past routes match the current route, and using new GPS data to refine, if possible, past routes.

Figure 2 shows the incremental route refining procedure. First, the GPS module acquires GPS data at fixed intervals. The interval should be long (for example, every five minutes) in order to reduce the power consumption of the cellular phone. The cellular phone sends the GPS data, in real time, to the incremental route refinement server via the cellular network. Second, the incremental route refinement server then estimates the current route from the GPS data received in real time while the user is moving. Third, if the user stays at a certain area for a certain period, the incremental route refinement server completes current route estimation. Finally, if a known route matches the current route, the incremental route refinement server uses the GPS data of the current route to refine, if possible, the known route. Otherwise, the incremental route refinement server stores the current route in the route DB.

The remainder of this section describes the three functions of the incremental route refinement server in detail.

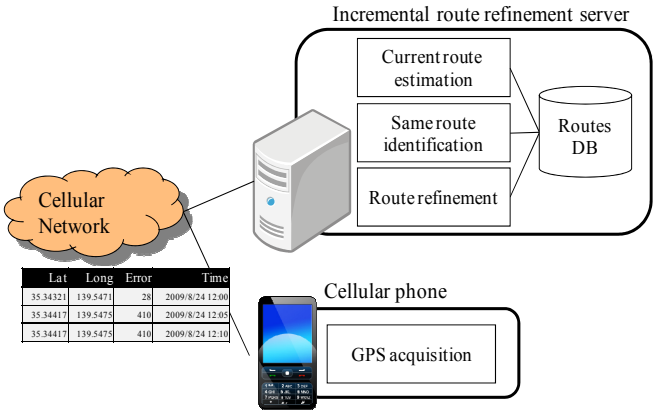


Figure 1: Incremental route refinement system

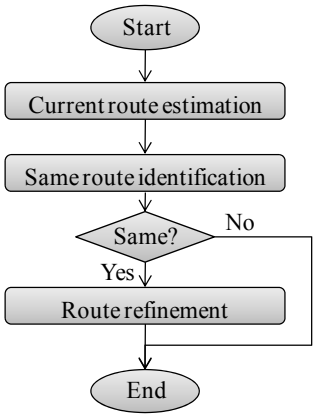


Figure 2: The incremental route refinement procedure

3.2 Current route estimation

The system needs to "fill-in" the route between sequential GPS points. One solution is for the system to plot an ellipse that contains the two points. However, this is computationally expensive.

Our solution is based on the use of rectangles. In detail, this paper denotes a GPS point in terms of position p_i and acquisition time t_i. Position p_i constitutes a square whose center is the latitude and longitude and whose sides are equal to twice the positioning error (Figure 3 (a)). Error is calculated based on the horizontal dilution of precision (HDOP). The route r_j ($1 \leqq j \leqq J$) is denoted by

$$r_j = \{((^j p_i, ^j t_i), (^j p_{i+1}, ^j t_{i+1}), m_{i-i+1}) \mid 1 \leqq i \leqq I\text{-}1\}, \quad (1)$$

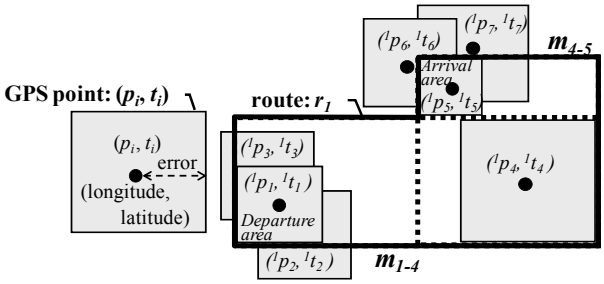
where m_{i-i+1} is the minimum bounding rectangle of $^j p_i$ and $^j p_{i+1}$, and $\{^j p_i \mid 1 \leqq i \leqq I\}$ are acquired while the user moves on r_j. The status of move and stay is identified by whether $N_{m\text{-}s}$ consecutive GPS points overlap or not. If $N_{m\text{-}s}$ consecutive GPS points overlap, the system defines the user as stationary, otherwise, moving. The algorithm of the current route estimation is described in Algorithm 1. Here, $^j p_d$ is the departure area of r_j, $^j p_a$ is the arrival area of r_j, and switch ($^j p_i$, $^j p_{i+1}$) is replacing $^j p_i$ with $^j p_{i+1}$ if the error of $^j p_{i+1}$ is smaller than $^j p_i$.

Algorithm 1. Current route estimation

1. $j = 1$.
2. $i = 1$, r_j = null, $k = 0$.
3. if $^j p_i$ overlaps $^j p_{i+1}$, then $k := k + 1$, switch($^j p_i$, $^j p_{i+1}$).
4. if $k \geqq N_{m\text{-}s}$,
5. if r_j == null, then $^j p_d = {}^j p_i$, $i := i + 1$, goto 3.
6. else $^j p_a = {}^j p_i$, $j := j + 1$, goto 2.
7. else $i := i + 1$, goto 3.
8. else $r_j := \{r_j, ((^j p_i, ^j t_i), (^j p_{i+1}, ^j t_{i+1}), m_{i-i+1})\}$,
9. $k := 0$, $i := i + 1$, goto 3.

Figure 3 (b) shows an example of estimating current route r_1 from $\{(^1 p_i, ^1 t_i) \mid 1 \leqq i \leqq 7\}$ ($N_{m\text{-}s} = 2$). When the system receives ($^1 p_1$, $^1 t_1$), ($^1 p_2$, $^1 t_2$), and ($^1 p_3$, $^1 t_3$), the system identifies $^1 p_1$ that is the smallest error, as the departure area. Then, receiving ($^1 p_4$, $^1 t_4$), the system estimates r_1 as (($^1 p_1$, $^1 t_1$), ($^1 p_4$, $^1 t_4$), $m_{1\text{-}4}$). $m_{1\text{-}4}$ is the MBR containing $^1 p_1$ and $^1 p_4$. The system then estimates r_1 as {(($^1 p_1$, $^1 t_1$), ($^1 p_4$, $^1 t_4$), $m_{1\text{-}4}$), (($^1 p_4$, $^1 t_4$), ($^1 p_5$, $^1 t_5$), $m_{4\text{-}5}$)} when the system receives ($^1 p_5$, $^1 t_5$), ($^1 p_6$, $^1 t_6$), and ($^1 p_7$, $^1 t_7$). In this case, the arrival area is $^1 p_5$ that has smallest error among $^1 p_5$, $^1 p_6$, and $^1 p_7$.

Since routes are defined by MBRs, the system can fill-in the route between dispersed GPS points at small computational cost.



(a) Representation of GPS point: (p_i, t_i)

(b) Route estimation from $\{(^1 p_i, ^1 t_i) \mid 1 \leqq i \leqq 7\}$ ($N_{m\text{-}s} = 2$)

Figure 3: Route estimation

3.3 Same route identification

The system proposed here matches the current route r_c against stored routes r_s. If a match is found, the system attempts to reduce the route ambiguity.

In order to determine whether the current route matches a stored route, the system detects the overlap between them. Here, the system deals with no overlap due to larger error of GPS than it is and passing along the slightly different route such as a little side trip or bypass. Specifically, the system retrieves the past routes from the route DB whose departure and arrival areas overlap those of the current route. Then, the system identifies the past route is different from the current route if the $N_{s\text{-}d}$ consecutive GPS points of a route do not overlap the other route. Otherwise, the system identifies the past route is same as the current route. The algorithm of the same route identification is described in Algorithm 2. Here, $^s p_i$ is the GPS points of the past route r_s and $^c p_i$ is the GPS points of the current route r_c.

Figure 4 shows the example of current route r_{t1}, r_{t2} and stored route r_1 ($N_{s\text{-}d} = 2$). As for r_{t1}, each GPS points of r_{t1}: $^{t1} p_1$, $^{t1} p_2$, $^{t1} p_3$, $^{t1} p_4$, $^{t1} p_5$ overlaps r_1, and each GPS points of r_1: $^1 p_1$, $^1 p_4$, $^1 p_5$ also overlaps r_{t1}. Therefore, the system identifies r_{t1} as matching r_1. On the other hand, the system identifies r_{t2} as not matching r_1 since the two GPS points of r_{t2}: $^{t2} p_2$ and $^{t2} p_3$ do not overlap r_1.

Algorithm 2. Same route identification

1. $\boldsymbol{R}_s = \{ \boldsymbol{r}_s \mid {}^s\boldsymbol{p}_d$ overlaps ${}^c\boldsymbol{p}_d$ && ${}^s\boldsymbol{p}_a$ overlaps ${}^c\boldsymbol{p}_a \}$.
2. for each $\boldsymbol{r}_s \in \boldsymbol{R}_s$, $\boldsymbol{k} = 0$.
3. for each ${}^c\boldsymbol{p}_i \in \boldsymbol{r}_c$,
4. if ${}^c\boldsymbol{p}_i$ does not overlap $\boldsymbol{r}_s$, then $\boldsymbol{k} := \boldsymbol{k} + 1$.
5. if $\boldsymbol{k} \geqq N_{s\text{-}d}$, then $\boldsymbol{r}_c$ is different from $\boldsymbol{r}_s$, goto 2.
6. else if ${}^c\boldsymbol{p}_i == {}^c\boldsymbol{p}_a$, then $\boldsymbol{n} = 0$.
7. for each ${}^s\boldsymbol{p}_m \in \boldsymbol{r}_s$,
8. if ${}^s\boldsymbol{p}_m$ does not overlap $\boldsymbol{r}_c$, then $\boldsymbol{n} := \boldsymbol{n} + 1$.
9. if $\boldsymbol{n} \geqq N_{s\text{-}d}$, then $\boldsymbol{r}_s$ is different from $\boldsymbol{r}_c$, goto 2.
11. else if ${}^s\boldsymbol{p}_m == {}^s\boldsymbol{p}_a$, then $\boldsymbol{r}_s$ is same as $\boldsymbol{r}_c$, goto 2.

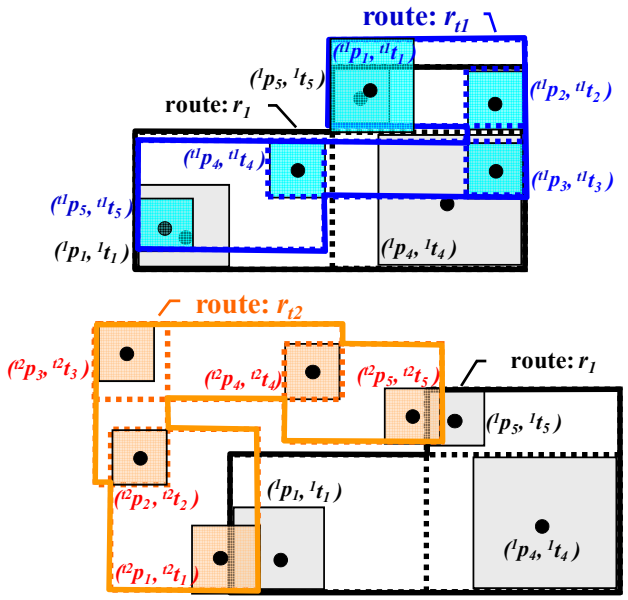


Figure 4: Same route identification: r_{t1} is identified as the same route as r_1 while r_{t2} is identified as the different route from r_1 since ${}^{t2}p_2$, ${}^{t2}p_3$ don't overlap r_1.

3.4 Route refinement

If the system identifies that the current route matches a stored route, the system refines the latter by two procedures: switching the GPS points of the past route and those of the current route, and appending the GPS points of the current route to the past route. The switching and appending procedures are specifically described below.

(1) Switching GPS points

The system switches the GPS point of the stored route ${}^s\boldsymbol{p}_m$ and that of the current route ${}^c\boldsymbol{p}_i$ if ${}^c\boldsymbol{p}_i$ overlaps ${}^s\boldsymbol{p}_m$, and the error of ${}^c\boldsymbol{p}_i$ is smaller than that of ${}^s\boldsymbol{p}_m$. In this case, the system associates ${}^s\boldsymbol{t}_m$ as well as ${}^c\boldsymbol{t}_i$ with ${}^c\boldsymbol{p}_i$ so that the system refines the stored route while retaining the time at which the user passed through the point.

Figure 5 shows an example of updating $\boldsymbol{r}_1$ with $\boldsymbol{r}_{t1}$. In this example, the system acquired ${}^{t1}\boldsymbol{p}_1$, ${}^{t1}\boldsymbol{p}_2$, ${}^{t1}\boldsymbol{p}_3$, ${}^{t1}\boldsymbol{p}_4$, and ${}^{t1}\boldsymbol{p}_5$ when the user passed along route $\boldsymbol{r}_1$ again. Here, ${}^{t1}\boldsymbol{p}_1$, ${}^{t1}\boldsymbol{p}_3$, ${}^{t1}\boldsymbol{p}_5$ overlap ${}^1\boldsymbol{p}_5$, ${}^1\boldsymbol{p}_4$, ${}^1\boldsymbol{p}_1$, respectively. Therefore, the system replaces ${}^1\boldsymbol{p}_4$ with ${}^{t1}\boldsymbol{p}_3$, and ${}^1\boldsymbol{p}_1$ with ${}^{t1}\boldsymbol{p}_5$ because ${}^1\boldsymbol{p}_1$, ${}^1\boldsymbol{p}_4$ have larger error. The system appends times ${}^{t1}\boldsymbol{t}_1$, ${}^1\boldsymbol{t}_4$, ${}^1\boldsymbol{t}_1$ to ${}^1\boldsymbol{p}_5$, ${}^{t1}\boldsymbol{p}_3$, ${}^{t1}\boldsymbol{p}_5$. In this way, the system can refine the route by replacing GPS points with low accuracy with those of high accuracy.

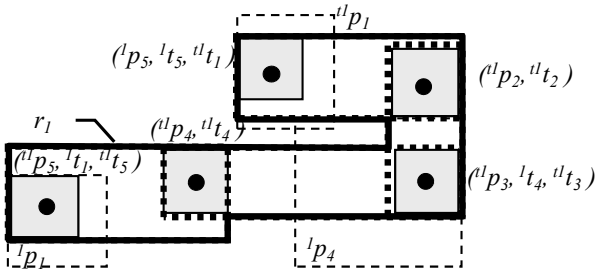


Figure 5: Route Refinement: Example of refining route r_1 with $({}^{t1}p_1, {}^{t1}t_1)$, $({}^{t1}p_2, {}^{t1}t_2)$, $({}^{t1}p_3, {}^{t1}t_3)$, $({}^{t1}p_4, {}^{t1}t_4)$, and $({}^{t1}p_5, {}^{t1}t_5)$ comprised in r_{t1}

(2) Attaching GPS points

The system attaches the GPS point of the current route ${}^c\boldsymbol{p}_i$ to the stored route $\boldsymbol{r}_s$ if ${}^c\boldsymbol{p}_i$ does not overlap any of the GPS points of $\boldsymbol{r}_s$. Here, the system needs to determine where to attach ${}^c\boldsymbol{p}_i$ in the GPS point sequence of $\boldsymbol{r}_s$.

The simple approach is to attach ${}^c\boldsymbol{p}_i$ between the 2-nearest neighboring GPS points of $\boldsymbol{r}_s$. However, this approach cannot handle curved routes. In Figure 5, for example, the nearest neighbor GPS point of ${}^{t1}\boldsymbol{p}_4$ is ${}^1\boldsymbol{p}_5$ so that the system attaches ${}^{t1}\boldsymbol{p}_4$ to ${}^1\boldsymbol{p}_5$ and ${}^{t1}\boldsymbol{p}_2$, which results in an incorrect route.

In general, the refined route does not significantly change from the stored route since the system has identified that the current route matches the stored route. Our solution is to attach ${}^c\boldsymbol{p}_i$ to the GPS point sequence of $\boldsymbol{r}_s$ so as to minimize the difference in route length between the pre-attached route and the post-attached route. Specifically, the system identifies the candidate GPS points $\{{}^{s\text{-}c}\boldsymbol{p}_n \mid 1 \leqq \boldsymbol{n} \leqq N\}$ of $\boldsymbol{r}_s$ within the range of $\boldsymbol{D}$ meters from ${}^c\boldsymbol{p}_i$. Then, the system attaches ${}^c\boldsymbol{p}_i$ to ${}^{s\text{-}c}\boldsymbol{p}_n$ and ${}^{s\text{-}c}\boldsymbol{p}_a$ since this minimizes equation (2). Here, $\{{}^{s\text{-}c}\boldsymbol{p}_a \mid 1 \leqq \boldsymbol{a} \leqq 2\}$ denotes the neighboring GPS points of ${}^{s\text{-}c}\boldsymbol{p}_n$, and $\mathbf{dis}({}^{s\text{-}c}\boldsymbol{p}_n, {}^{s\text{-}c}\boldsymbol{p}_a)$ denotes the distance between ${}^{s\text{-}c}\boldsymbol{p}_n$ and ${}^{s\text{-}c}\boldsymbol{p}_a$. $\mathbf{dis}({}^c\boldsymbol{p}_i, {}^{s\text{-}c}\boldsymbol{p}_n) + \mathbf{dis}({}^c\boldsymbol{p}_i, {}^{s\text{-}c}\boldsymbol{p}_a)$ in equation (2) indicates post-attached length while $\mathbf{dis}({}^{s\text{-}c}\boldsymbol{p}_n, {}^{s\text{-}c}\boldsymbol{p}_a)$ indicates pre-attached length.

$$\text{dis-variation}({}^c p_i, {}^{s\text{-}c}p_n, {}^{s\text{-}c}p_a) = \text{dis}({}^c p_i, {}^{s\text{-}c}p_n) + \text{dis}({}^c p_i, {}^{s\text{-}c}p_a) - \text{dis}({}^{s\text{-}c}p_n, {}^{s\text{-}c}p_a) \cdots (2)$$

In Figure 5, for example, let $\{{}^{1}p_5, {}^{t1}p_5\}$ be the candidate GPS points of ${}^{t1}p_4$. The system calculates **dis-variation**$({}^{t1}p_4, {}^{1}p_5, {}^{t1}p_2)$ and **dis-variation**$({}^{t1}p_4, {}^{t1}p_5, {}^{t1}p_3)$. This is illustrated in Figure 6. **dis**$({}^{c}p_i, {}^{s\text{-}c}p_n)$ + **dis**$({}^{c}p_i, {}^{s\text{-}c}p_a)$ in equation (2) indicates the length of the solid line while **dis**$({}^{s\text{-}c}p_n, {}^{s\text{-}c}p_a)$ indicates the length of the dotted line. In this example, ${}^{t1}p_5$ minimizes equation (2) so the system attaches ${}^{t1}p_4$ to ${}^{t1}p_5$ and ${}^{t1}p_3$.

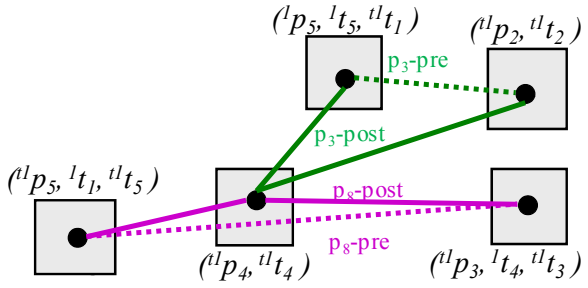


Figure 6: Attaching GPS point: Attaching ${}^{t1}p_4$ to ${}^{t1}p_5$ and ${}^{t1}p_3$ since this increases route length the least

The algorithm of the route refinement is described in Algorithm 3. Here, r_c and $({}^{c}p_i, {}^{c}t_i)$ denote the current route and its GPS points, r_s and $({}^{s}p_m, {}^{s}t_m)$ denote the past route and its GPS points, and D is the threshold to identify neighboring GPS points.

Algorithm 3. Route refinement

1. $R_s = \{ r_s \mid r_s$ is identified as the same route as $r_c\}$.
2. for each $r_s \in R_s$, ${}^{s}T_m = \{ {}^{s}t_m\}$.
3. for each ${}^{c}p_i \in r_c$,
4. if ${}^{c}p_i$ overlaps ${}^{s}p_m$, **//switching GPS points**
5. if the error of ${}^{c}p_i <$ the error of ${}^{s}p_m$,
6. then ${}^{s}p_m := {}^{c}p_i$, ${}^{s}T_m := \{ {}^{s}T_m, {}^{c}t_i\}$.
7. else ${}^{s}T_m := \{ {}^{s}T_m, {}^{c}t_i\}$.
8. else, **//attaching GPS points**
9. acquire $\{ {}^{s\text{-}c}p_n \mid \text{dis}({}^{s\text{-}c}p_n, {}^{c}p_i) \leqq D\}$.
10. $d = 0$. $d_t = 0$. $p_{t1} =$ null. $p_{t2} =$ null.
11. for each ${}^{s\text{-}c}p_n$,
12. for each ${}^{s\text{-}c}p_a$,
13. $d =$ dis-variation$({}^{c}p_i, {}^{s\text{-}c}p_n, {}^{s\text{-}c}p_a)$.
14. if $d_t > d$, then,
15. $d_t := d$, $p_{t1} := {}^{s\text{-}c}p_n$, $p_{t2} := {}^{s\text{-}c}p_a$.
16. add ${}^{c}p_i$ between p_{t1} and p_{t2}.

In this way, the system incrementally refines the stored routes by replacing and/or attaching newly acquired GPS points. This approach allows the system to precisely identify significant routes even if GPS acquisition is infrequent since the system can identify the routes more precisely as the user passes along them.

4 EXPERIMENTS

4.1 Evaluation

We evaluated the performance of the proposed approach in terms of three metrics: computational load, the performance of route complementation, and the performance of route refinement. Details of each are given below.

(1) Computational load

We demonstrate the superiority of MBR over ellipses by measuring the processing time needed to estimate the current route from 10,000 GPS points using MBRs and ellipses.

(2) The performance of route complementation

The number of GPS points in a route represents the degree of route complementation. Therefore, we counted the number of GPS points in each route with respect to each transit. We then calculated the increase rate of GPS points for each route r_j, and each transit k by using equation (3). Here, # denotes "the number of". Parameter values are shown in Table 1.

$$Increase\ rate(r_j, k) = \frac{\#GPS\ points\ of\ r_j\ after\ k\text{th}\ transit}{\#GPS\ points\ of\ r_j\ after\ 1\text{st}\ transit} \cdots (3)$$

(3) The performance of route refinement

The average error of GPS points in a route represents the accuracy of the estimated route. Therefore, we calculated the average error of GPS points $({}^{j}p_1, \ldots, {}^{j}p_I)$ in each route r_j with respect to each transit k by applying equation (4).

$$Average\ error(r_j, k) = \frac{1}{I}\sum_{i=1}^{I}(error\ \ of\ \ {}^{j}p_i) \cdots (4)$$

Table 1: Parameter values

Parameter	Value
$N_{m\text{-}s}$ for identifying status of stay and move	3
$N_{s\text{-}d}$ for identifying matching routes	3
D for identifying neighboring GPS points (m)	200

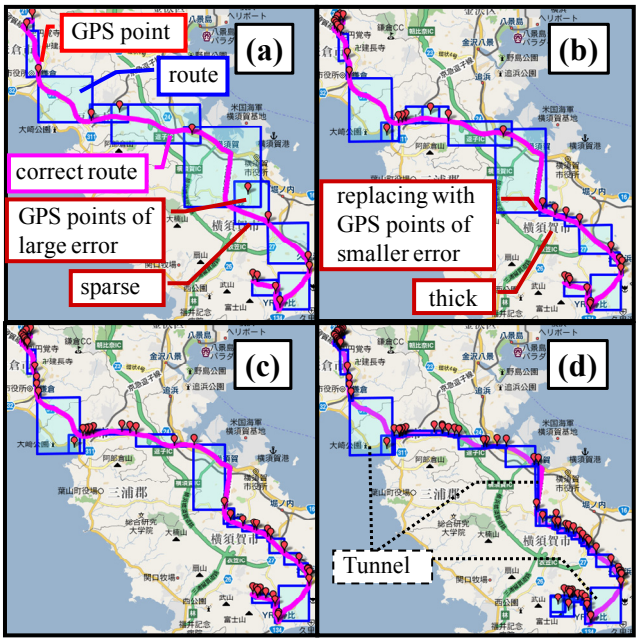
We implemented a prototype system to conduct an evaluation experiment. The system consisted of commercial GPS cellular phones sold by NTT docomo and a Core 2 Duo PC with 12GB memory as the incremental route refinement server. The three functions of the incremental route refinement server were implemented as Java programs.

The evaluation experiment had six subjects carry GPS phones for 6 – 14 months. The interval between acquiring GPS points affects the battery consumption of the cellular

phone. In a preliminarily experiment, we acquired GPS points every five minutes and measured the time from full charge to battery exhaustion; the results was that the cellular phone operated for two days and 10 hours. Since users generally charge batteries every day or every second day, we acquired GPS points every five minutes.

4.2 Results

Figure 7 shows the example of incremental refinement of a subject's commuting route. (a) shows the first estimation of a route; the user passed along this route for the first time. (b), (c), and (d) show the estimated route after the user passed along it six, ten, and twenty times respectively. Although the estimated route in (a) covers the correct route, the GPS points on route in (a) have large error and there are few of them, which result in ambiguous route. On the other hand, after six, ten, and twenty transits, the GPS points have less error, and there are many GPS points, which result in reducing ambiguity. The part of the route that remained sparsely covered is a tunnel in which GPS points could not be acquired.



(a) The estimated route after the first transit
(b) The estimated route after the 6th transit
(c) The estimated route after the 10th transit
(d) The estimated route after the 20th transit

Figure 7: Examples of incremental route refinement: Although the estimated route is ambiguous after the first transit, the more accurate the estimated route is, the more the user passes along it.

(1) The computational load

Figure 8 shows the processing time versus the number of GPS points. It demonstrates that the proposed approach of using MBRs can reduce the processing time by 88%. 10,000 GPS points equals the data captured by the user over a 35 day period. Increasing user number and the frequency of GPS point acquisition only strengths advantage. Supposing that NTT DOCOMO subscribers, 50 million users, use this system, the estimation using ellipse requires 4.67 hours par GPS point, while the estimation using MBR requires 33 minutes. While high-end server reduces the processing time, the hardware cost increases.

(2) The performance of route complementation

Figure 9 shows the increase rate in GPS points per route versus the number of transits. "x_n" in the graph legend represents "examinee id_route id". Since the 6 subjects yielded a lot of estimated routes, the figure shows only the top 10 routes in perspective of the number of transit. GPS points per route increased with transit number. The variation in increase rate is due to variation in subject movement speed. Figure 10 shows the increase rate in GPS points per route versus the movement speed. It shows that the higher the movement speed is, the fewer GPS points there are per route so newly acquired GPS points are more likely to be appended. In Figure 10, the subject passed along "d_20" on foot, while another user road a train while passing along "a_1".

(3) The performance of route refinement

Figure 11 shows the average error in GPS accuracy versus transit number. The figure also shows only the top 10 routes. The average error in GPS accuracy decreases as the transit number increases. The average error can momentarily increase since the newly appended GPS points have large error. Over time, however, more accurate points are added which replaced the inaccurate ones.

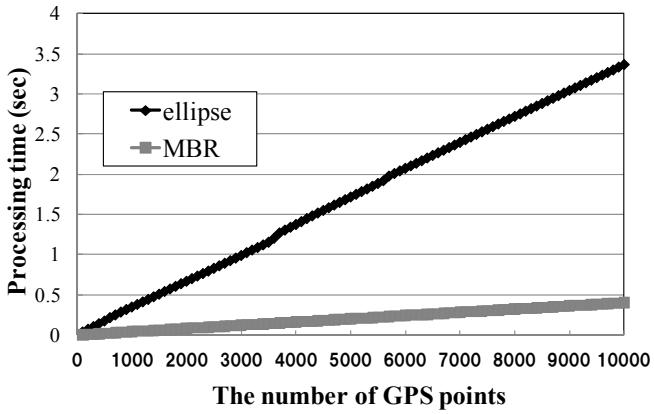


Figure 8: The processing time of route matching using MBRs and ellipses: The proposed approach using MBRs reduced the processing time by 88%

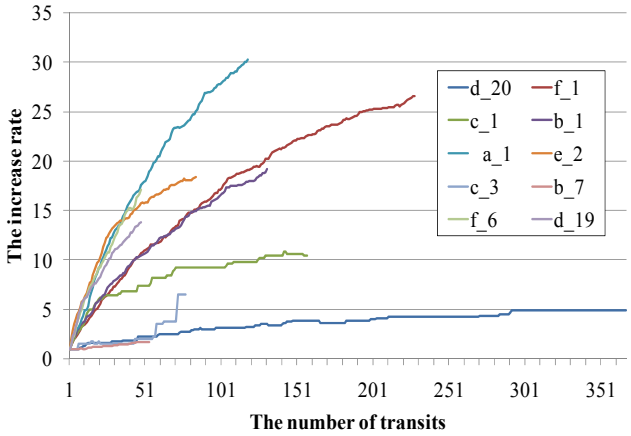


Figure 9: The performance of route complementation (the increase rate versus transit number)

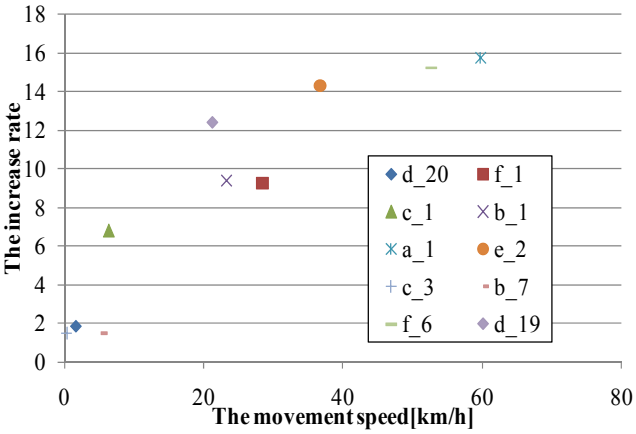


Figure 10: The performance of route complementation (the increase rate versus movement speed)

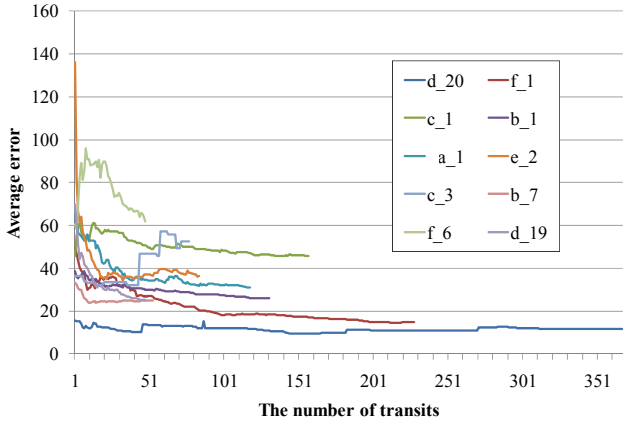


Figure 11: The performance of route refinement

5 CONCLUSION

This paper proposed to identify the user's significant routes from the GPS data acquired by the GPS-capable cellular phone. Significant routes are those routes that the user frequently passes along. The proposed approach precisely identifies the significant routes with infrequent GPS acquisition by incrementally refined the estimated route. A experiment using the data collected from 6 subjects over periods of 6 – 14 months demonstrated the superiority of the proposed approach.

Future work includes developing a system that can predict the areas that the user is likely to visit or pass through using the significant routes.

REFERENCES

[1] A. Ward, A. Jones, A. Hopper, A New Location Technique for the Active Office, IEEE Personal Communications, Vol. 4, No. 5, pp.42-47 (1997).

[2] B. Schilit, N. Adams, and R. Want, Context-Aware Computing Applications, Proc. of Intl. Workshop on mobile computing systems and applications, pp.85-90 (1994).

[3] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen, Discovering personally meaningful places: An interactive clustering approach, ACM Trans. Inf. Syst. vol.25, no.3 (2007).

[4] C. Zhou, S. Shekar, and L. Treveen, Discovering Personal Paths from Sparse GPS Traces, Proc. of the JCIS 2005 Workshop on Data Mining (2005).

[5] D. Ashbrook, T. Starner, Using gps to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing, Vol. 7, Issue. 5, pp. 275–286 (2003).

[6] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, Extracting places from traces of locations. In Proc. WMASH, pp.110-118 (2004).

[7] L. Liao, D. Fox, and H. Kautz, Learning and Inferring Transportation Routines. In Proc. of AAAI-04 (2004).

[8] L. Liao, D. Fox, and H. Kautz, Extracting Places and Activities from GPS Traces Using Hierarchical Conditional Random Fields, Int. J. Rob. Res. 26, 1, pp.119-134 (2007).

[9] M. Weiser, The Computer for the Twenty-First Century, Scientific American (1991).

[10] N. Marmasse and C. Schmandt, Location-aware information delivery with commotion. In Proc. HUC, pp.157-171 (2000).