CTS: A Cellular-based Trajectory Tracking System with GPS-level Accuracy

XINGYU HUANG, YONG LI, and YUE WANG, Tsinghua University, China
XINLEI CHEN, Carnegie Mellon University, USA
YU XIAO, Aalto University, Finland
LIN ZHANG, Tsinghua-Berkeley Shenzhen Institute, China

GPS has been widely used for locating mobile devices on the road map. Due to its high power consumption and poor signal penetration, GPS is unfortunately unsuitable to be used for continuously tracking low-power devices. Compared with GPS-based positioning, cellular-infrastructure-based positioning consumes much less energy, and works in any place covered by the cellular networks. However, the challenges of cellular positioning come from the relatively low accuracy and sampling rate. In this paper, we propose a novel cellular-based trajectory tracking system, namely *CTS*. It achieves GPS-level accuracy by combining trilateration-based cellular positioning, stationary state detection, and Hidden-Markov-Model-based path recovery. In particular, *CTS* utilizes basic characteristics of cellular sectors to produce more credible inferences for device locations.

To evaluate the performance of *CTS*, we collaborated with a mobile operator and deployed the system the city of Urumchi, Xinjiang Province of China. We collected the location data of 489, 032 anonymous mobile subscribers from cellular networks during 24 hours, and retrieved 201 corresponding GPS trajectories. Our experimental results show that *CTS* achieves GPS-level accuracy in 95.7% of cases, which significantly outperforms the state-of-the-art solutions.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**;

Additional Key Words and Phrases: Cellular networks, cellular positioning, trajectory tracking, system implementation

ACM Reference Format:
Xingyu Huang, Yong Li, Yue Wang, Xinlei Chen, Yu Xiao, and Lin Zhang. 2017. CTS: A Cellular-based Trajectory Tracking System with GPS-level Accuracy. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 140 (December 2017), 29 pages. https://doi.org/10.1145/3161185

Authors' addresses: Xingyu Huang; Yong Li; Yue Wang, Tsinghua University, Department of Electronic Engineering, Beijing, 100084, China, huangxy14@mails.tsinghua.edu.cn; Xinlei Chen, Carnegie Mellon University, Department of Electrical and Computer Engineering, Pittsburgh, 15213, USA; Yu Xiao, Aalto University, Department of Communications and Networking, Espoo, 02150, Finland; Lin Zhang, Tsinghua-Berkeley Shenzhen Institute, Department of Electronic Engineering, Beijing, 100084, China.

2474-9567/2017/12-ART140 $15.00
https://doi.org/10.1145/3161185

1 INTRODUCTION

Tracking the locations of massive mobile devices is required by many Internet-of-things (IoT) applications [1][2], such as goods tracking and bike sharing. Accurate trajectory can be utilized, for example, for optimizing the urban traffic scheduling [3], allocation of bike-sharing stations [4], and etc. Depending on the application, the accuracy requirement for outdoor positioning and tracking varies from meters to tens of meters. So far, GPS is considered as the most accurate positioning solution in outdoor environments, whereas it suffers from rapid battery depletion and poor signal penetration [5]. Meeting the requirements of IoT applications for low power and passive tracking is still a challenging while open research problem.

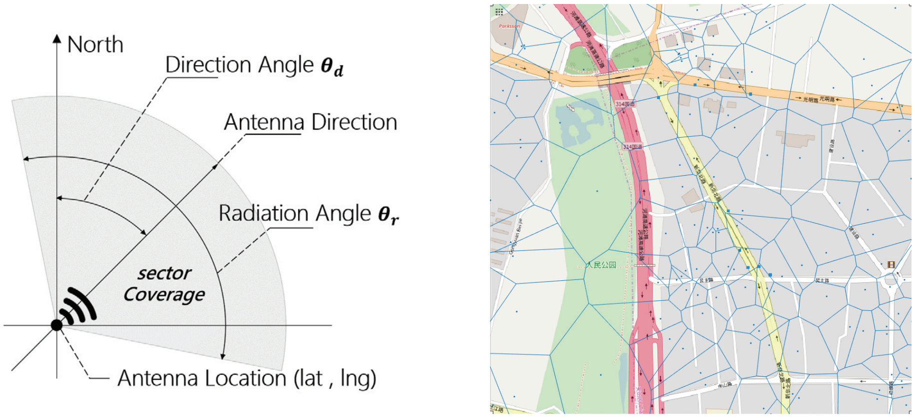
Concerning the wide coverage of cellular networks, we propose to utilize cellular infrastructure to identify the outdoor locations of mobile devices on the road map. The existing cellular positioning solutions only provide coarse locations, with the error varying with the density of base stations from hundreds of meters to over a kilometer [6]. In addition, the sampling intervals of positions range from seconds to minutes. By solving these problems, in this paper, we aim at developing a novel Cellular-based Trajectory Tracking System (CTS) that would track mobile devices on the road map with GPS-level accuracy. Towards that end, there are three critical issues to be solved.

- Large errors: the median error of trilateration-based cellular positioning is 140 meters, which is too large for identifying the exact road segment in the urban areas, where the density of road segments is high.
- Serious drifting: while a device stays in the same place, the locations provided by the cellular positioning system might drift for hundreds of meters, making it difficult to detect whether the device is moving or not.
- Low sampling rate: cellular positioning system updates device locations when the device requests services of voice, message or data connection, resulting in potentially large update intervals. Taking our test set as example, the average interval between two consecutive updates is over 4 minutes.

Our system calculates accurate trajectories and achieve passive tracking in the following three steps. Firstly, it obtains coarse locations of mobile devices based on the signal trilateration algorithm [7], and applies both speed-based and direction-based noise filtering algorithms to remove erroneous outputs. Secondly, we design a tri-state state machine to effectively detect whether a device moves or stays, which is a indispensable process when dealing with noisy positioning data. Thirdly, we invent a novel approach that utilizes the cellular sector parameters, such as orientation and radiation angles, to calculate the candidates of road segments corresponding to each device location. And we combine such approach with enhanced Hidden Markov Model (HMM) based algorithm to recover the most probable path that the device has traveled through.

We deployed our system in Urumchi City, China, and collected 489,032 mobile subscribers' cellular positioning data. Meanwhile, we retrieved 201 GPS trajectories by sniffing users' navigation apps. Using these GPS traces as reference, our system proves to provide GPS-level accuracy in over 95% cases, which significantly outperforms the state of the art. The key contributions of this work can be summarized as below.

- We build a real-time cellular-based trajectory tracking system (*CTS*), which achieves GPS-level accuracy according to the real-life experiments in Urumchi City, China.
- We design a state detection algorithm that can distinguish moving and stationary states by solving the interference issue caused by *drifting points*.
- We propose a HMM-based path recovering algorithm to track mobile devices on the road map. Our algorithm utilizes the operating characteristics of cellular sectors to promote the reliability of location


North
Direction Angle θ_d
Antenna Direction
Radiation Angle θ_r
sector
Coverage
Antenna Location (lat , lng)

(a) Sector characteristics.

(b) Voronoi diagram.

Fig. 1. (a) Several static parameters of a cellular sector. (b) A Voronoi diagram divides the map plane into irregular regions to model the rough coverage of cellular sectors. Each region contains one sector.

inference of mobile device. Additionally, to avoid *U-turn error* and *detour error*, we introduce two kinds of path selecting penalties into the calculation of transition probability.

The remainders of the paper are arranged as follows: in Sec. 2, we give an overview of the background and related works. In Sec. 3, we describe the details of system design. System implementation is introduced in Sec. 4. Parameter configuration and performance evaluation are presented in Sec. 5 and Sec. 6 respectively. Finally, we briefly summarize our work in Sec. 7.

2 BACKGROUND AND RELATED WORKS

In this section we first briefly introduce the cellular positioning technology and its main drawbacks compared with GPS-based position. After that, we present backgrounds with the remained challenges of tracking mobile devices.

2.1 Cellular Positioning

In cellular networks, a mobile device typically communicates with several cellular sectors. As shown in Fig. 1, each sector includes a set of parameters: $S_P = \{L, \theta_r, \theta_d, V_s\}$, where L refers to the geo-coordinates, θ_r and θ_d are the radiation angle and direction angle of the sector antenna, respectively. A smaller θ_r means the antenna is directed to a narrower region. θ_d measures the angle between the North axis and the bisector of θ_r, and V_s represents the Voronoi diagram[8] of all sectors in the city. It separates the map plane into thousands of cells, where each cell roughly models the coverage of the sector.

The classical forward-link-trilateration algorithm[9] calculates the location of a mobile device based on the time it takes for the signals to travel from each of the connected sectors to the device. As illustrated in Fig. 2, a **cellular trajectory**, which refers to a sequence of device locations, will be generated witm time. Note that in [10] and [11], a cellular trajectory refers to a sequence of cellular base stations to which a mobile phone connected over time.

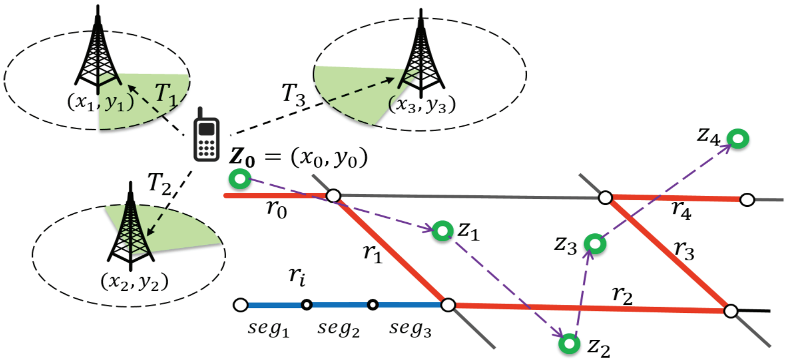


Fig. 2. Illustration of cellular positioning based on forward-link trilateration. The location of the mobile device $z_0 = (x_0, y_0)$ is calculated from T_1, T_2, T_3 that refer to the durations of sending signal from each of the connected cellular sectors to the device$(x_{1:3}, y_{1:3})$. A sequence of device locations $(z_1, ..., z_4)$ can be mapped to a sequence of road segments $(r_1, ..., r_4)$.

We define the cellular trajectory of a mobile device as Z, which consists of N points. Each point is described with the corresponding latitude, longitude, and time stamp. It can be expressed as follows,

$$Z = z_1, ..., z_t, ..., z_N, \tag{1}$$

where $z_t = (lat_t, lng_t, T_t), 1 < t < N$.

Compared with GPS-based positioning, cellular positioning provides lower spatial accuracy, and supports a lower sampling rate. In addition, it suffers from the problem of location drifting.

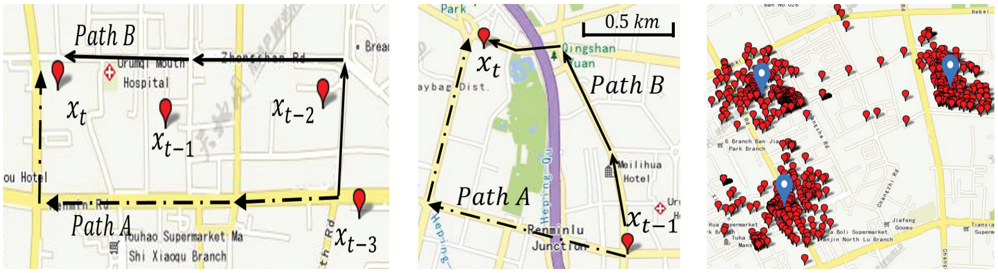
Low Spatial Accuracy: The accuracy of cellular positioning is greatly affected by the multi-path propagation of wireless signals, especially in the areas with high density of buildings. In case the closest cellular sectors are heavily loaded, a mobile device may be connected to other cellular sectors that are located far away. In this case, the error of cellular positioning becomes extremely large (Fig. 3(a)). According to our measurement, the median error of cellular positioning is 143 meters, with 90% of errors smaller than 215 meters.

Low Sampling Rate: The existing cellular positioning systems update the device locations when the device is initiating or terminating a voice, message or data service. In case the user has not used any of these services for a long time, the location information will be missing during a long period. As the result, there may be a large distance between two consecutive locations (Fig. 3(b)).

Location Drifting Issue: When a mobile device is stationary, the results of trilateration-based positioning over time do not remain the same. The estimated locations may randomly scatter within a circle with a diameter of hundreds of meters. We call the ground truth of the device location as *Stay Point* (three blue marks in Fig. 3(c)), and the estimated locations with random errors as *Drifting Points* (massive red marks in Fig. 3(c)). We can observe from Fig. 3(c) that the phone user has been to three different places, corresponding to three clusters of drifting points. The radius of such cluster can be several hundreds of meters according to our experiments.

2.2 Trajectory tracking

In digital road map, a road is defined as a sequence of **road segments**. For instance, road r_i in Fig. 2 consists of 3 segments, namely, seg_1, seg_2, seg_3. Accordingly, a device's **trajectory** on the road map



(a) Path A is the ground truth. Incorrect Path B was recovered due to large errors of cellular positioning.

(b) Difficulty in identifying the exact path due to the low sampling rate.

(c) Clusters of drifting points.

Fig. 3. Illustration of three major challenges.

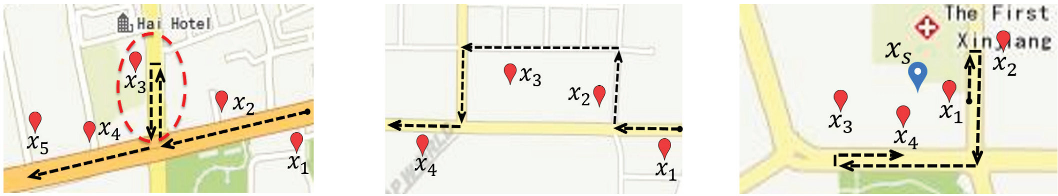
can be described as a sequence of road segments that the device has covered. Identifying such device trajectories on the road map is required by many location-aware and IoT applications.

The essential issue for trajectory tracking is matching the positioning points, e.g. GPS points, onto the road map, and recovering the covered road segments. This is also known as the *map matching problem*. During the last decade, various map matching algorithms have been developed for GPS data. These techniques can be categorized into three groups: topological [12, 13], geometric[14, 15], and probabilistic[16, 17] techniques.

However, when it comes to cellular positioning, the map matching approaches designed for GPS-based data often cause severe errors, due to the low accuracy, low sampling rate and the drifting issue. For example, when the density of roads is high, the errors of cellular positioning, up to hundreds of meters, may easily cause mismatch of road segments. Taking Fig. 3(a) as an example, *Path A* is the actual path the mobile device traveled through, but the estimated locations are actually closer to *Path B*. Due to the low sampling rate, it may also be hard to determine which route the device has taken when there are alternative routes with similar travel distances and duration. An example in Fig. 3(b) shows that the 1.5-kilometers-long gap between x_{t-1} and x_t makes it practical to travel through *either Path A or Path B*. We summarize the errors in path recovery as three typical categories as follows.

U-turn error: While a mobile device moves **straight** along a road, if one estimated position x_t diverges from the road (such as x_3 in Fig. 4(a)), an undesired U-turn would be included in the recovered route. We will discuss this issue in detail in Sec. 3.3.

Detour error: Showing in Fig. 4(b), the device moves along an arterial road. Because the estimated position x_3 is closer to a sideway, which is parallel to the arterial road, a detour error occurs.



(a) U-turn error.

(b) Detour error.

(c) Back and forth error.

Fig. 4. Three typical categories of errors in path recovery.

Back-and-forth error: In Fig. 4(c), a mobile device generates drifting points, $x_{1:4}$, when it remains at x_s. When recovering the path, it looks like the device is moving back and forth between these drifting points, whereas in fact the device is stationary.

2.3 Related Work and Motivation

In recent years, several solutions have been proposed to solve the challenges of cellular-based trajectory tracking. CTrack[18] fuses device's cellular fingerprints and additional sensor data to perform the tracking. However, it requires access to the sensor data collected by mobile devices. *Cell**[19] divides the digital map into grids with fixed size, calculates the probability of a user being in each grid, and incorporates A* algorithm to determine the most likely sequence of grids that the mobile device has traveled through. In order to model the coverage of cellular sectors in each grid, the system has to gather massive GPS location data as ground truth.

SnapNet [10] is a relatively thorough system for cellular-based trajectory tracking. It utilizes HMM-based algorithm to track the mobile device without any additional sensors. However, the system may not be practicable enough, since it only provides seven test cases. Further, solutions for common errors in path recovery, such as U-turn and detour errors, are not adequately considered in the system. As a geometric-based approach, *TPDA* [20] introduces Voronoi diagram to model the geometric relationships among cell towers, roads and mobile devices. The limitation of this algorithm is that it performs well only when the density of cell towers is high and the trajectories are rather straight.

As summarized in Table 1, *CTS* has overcome most drawbacks of cellular-infrastructure-based trajectory tracking. More specifically, we employ *speed-based* and *direction-based filter* to remove noisy points of the cellular positioning, and solve the drifting issue with *state detection algorithms*. Regarding path recovery, we introduce *U-turn and Direction changing penalties* into the HMM-based algorithm framework, and dramatically reduce three typical errors. Furthermore, we utilize the *sector characteristics* and *Voronoi diagram* to improve the accuracy when matching cellular positioning points to the road map. We will describe the details in Sec. 3.

Table 1. Comparison of cellular tracking systems

System / Features	Additional Sensors	Training Process	Massive Ground Truth	Curved Trajectories	Algorithm Practicability
CTrack[18]	√	√	×	√	×
Cell*[19]	×	×	√	√	√
SnapNet[10]	×	×	×	√	×
TPDA[20]	√	×	×	×	√
CTS	×	×	×	✓	✓

3 SYSTEM DESIGN

As illustrated in Fig. 5, *CTS* consists of three system modules: *cellular positioning, state detection,* and *path recovery*. In the first module, we adopt a widely used technique, trilateration positioning, to locate mobile devices, and focus on removing noisy positioning points through two noise filters. Then, a state detection process is proposed to solve the drifting issue. Finally in the third module, we design the HMM-based approach to recover the actual paths of cellular trajectories. Now, we present the details of these modules.

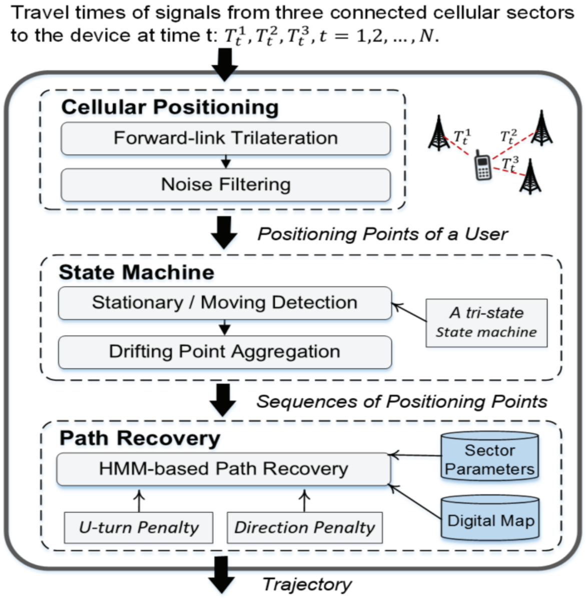
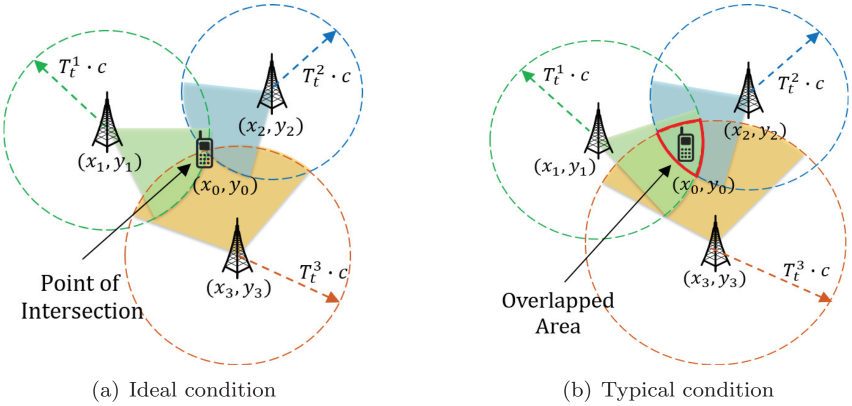

Travel times of signals from three connected cellular sectors to the device at time t: $T_t^1, T_t^2, T_t^3, t = 1,2,\dots,N$.
Cellular Positioning
Forward-link Trilateration
Noise Filtering
T_t^1
T_t^2
T_t^3
Positioning Points of a User
State Machine
Stationary / Moving Detection
Drifting Point Aggregation
A tri-state State machine
Sequences of Positioning Points
Path Recovery
HMM-based Path Recovery
Sector Parameters
Digital Map
U-turn Penalty
Direction Penalty
Trajectory

Fig. 5. System architecture of *CTS*.

3.1 Cellular Positioning

3.1.1 Forward-link Trilateration Positioning: The forward-link trilateration is a time-based positioning technique, which is widely implemented in cellular networks[7]. It measures the signal travel times, $T_t^i (i = 1, 2, 3, t = 1, 2, ..., N)$, from each of the three connected sectors to the mobile device at time t. Then, the distances between the device and the sectors are calculated by $d_t^i = c \cdot T_t^i$, where c is the velocity of electromagnetic wave. The exact locations of the sectors, denoted as (x_i, y_i), are obtained from the set of sector parameters S_P. Ideally, as illustrated in Fig. 6(a), we could calculate a unique solution of the device location (x_0, y_0), by solving:

$$(T_t^i \cdot c)^2 = (x_i - x_0)^2 + (y_i - x_0)^2, i = 1, 2, 3. \qquad (2)$$



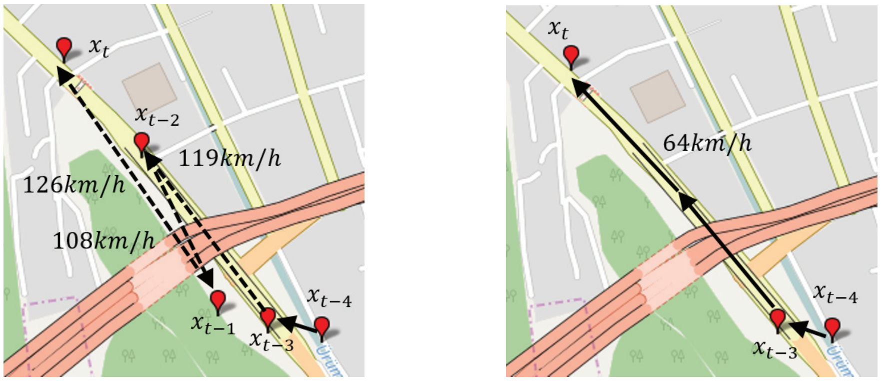
(a) Ideal condition

(b) Typical condition

Fig. 6. Forward-link Trilateration Method.

The multi-path propagation and time synchronization error might cause the equations over-constrained. [21] applies least-square approach to solve it. As shown in Fig. 6(b) the estimated locations typically fall in the overlapped area of the sectors. In this way, a sequence of locations of a mobile device are estimated along the time.

3.1.2 Speed-based Noise Filtering: As mentioned in Sec. 2, the error of trilateration could reach hundreds of meters. Consequently, the calculated speeds of the device may become abnormally high when traveling between erroneous positioning points. As illustrated in Fig. 7(a), the device travels from x_{t-4} towards x_t. Because the signal is blocked by the overpass, the cellular positioning system generates x_{t-2} and x_{t-1} with extremely large errors. This causes over-estimation of the speeds over $\overrightarrow{x_{t-3}x_{t-2}}$, $\overrightarrow{x_{t-2}x_{t-1}}$, and $\overrightarrow{x_{t-1}x_t}$.



(a) Over-estimation of speeds caused by noisy points.

(b) After the noisy points are removed by the speed-based filter.

Fig. 7. Illustration for speed-based noise filter.

Two speed thresholds, th_1 and th_2, are defined for every road, according to the distribution of users speed on that road. As explained in Sec. 5.2, th_2 is slightly higher than the speed limit of the road, in order to tolerate acceptable positioning errors. Yet th_1 is about twice the value as the speed limit according the data analysis in Sec. 5.2. In practice, we use a sliding window with size 3 to filter out noisy points based on speed information. Each time, we calculate the speed between point A and B, and the speed between point B and C. We denote the speeds by S_{AB} and S_{BC}, respectively. If any of the following two criteria is satisfied, the middle point B would be filtered out.

$$\begin{cases} \exists\ s \in \{S_{AB}, S_{BC}\},\ \ s > th_1; & \text{(3a)} \\ \forall\ s \in \{S_{AB}, S_{BC}\},\ \ s > th_2. & \text{(3b)} \end{cases}$$

As shown in Fig. 7(b), after we remove x_{t-2} and x_{t-1}, the recovered path no longer needs to travel back and forth, thus the speed between x_{t-3} and x_t returns to a normal level. Note that, since the positioning points have not been matched onto the road map, we assign each point to the nearest road. This approximation makes sense because the attribute of different roads in a local area are typically similar, meaning that there is little difference between the speed limits.

3.1.3 Direction-based Noise Filtering: After removing the noisy positioning points based on speed information, we detect the state of the device in order to identify the drifting points. When the device remains in a stay point (the blue-hollow point in Fig. 8(a)) or move in a small range, it is in the **stationary state**. Otherwise, it is in a **moving state** or in a **transient state**. As mentioned above, when the device remains stationary, it produces *drifting points* (the red points in Fig. 8(a)), which cause severe disturbance in path recovery.

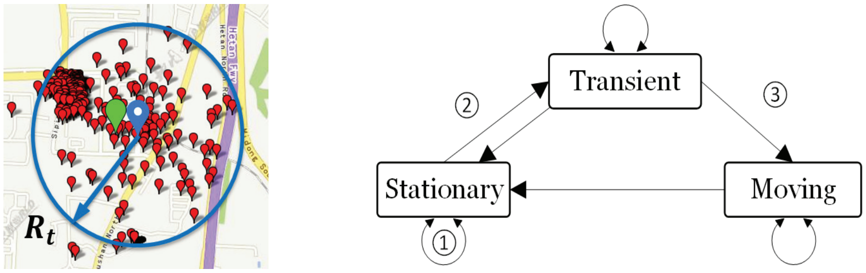
As the drifting points can cause the back-and-forth errors, they are supposed to be removed from trajectories. The drifting points can be identified by detecting the stationary state, if the user has stayed in the same place for a long time. But in case the stay is too short to be detected, we propose to apply a *direction filter* to eliminate the back-and-forth errors as follows. If the angle between the moving directions at two consecutive positioning points exceeds a threshold α_d, the former point will be removed from the trajectory. With a smaller α_d, more back-and-forth errors can be removed, whereas some reasonable turnings may be misclassified as errors. This direction-based filter will be disabled by setting α_d as a flat angle, and the optimal value of α_d is estimated in Sec. 5.3.

3.2 Stationary State Detection

We design a state detection algorithm to solve the drifting issue, and its basic ideas are as follow:

- If a series of positioning points of a device can be included in a small circle, the user tends to remain stationary;
- If some of the new generated points fall out of the circle, either *the device is moving* or *these points are seriously influenced by noise*. Hence, we have to observe some more points to make sure the device is actually moving.
- Similarly, when the device is considered to be traveling, in order to determine that the device turns stationary again, we have to wait for adequate new generated points to gather in a small circle.
- The radius of the circle increases along the time in a certain period, because the longer a device **stays** in the same place, the more likely it is to generate positioning points far away from each other. The reason is that a device in the stationary state may connect to different sets of sectors due to the cellular network's load-balancing strategy and time-varying signal channels. Thus the positioning points produced by different sectors can be far away.

Suppose that at time t, we have collected a sequence of positioning points since time i, denoted as $X_t = \{x_i, x_{i+1}, ..., x_t\}$. Then we draw the aforesaid small circle, C_t, whose center is set as the geometric



(a) Drifting points.

(b) The state machine implements the state detection algorithm.

Fig. 8. Illustration of drifting issue and state machine

center of X_t. The radius of C_t varies as the following expression:

$$R_t = \begin{cases} = R_1 + t * R_{step} & \text{,if } t_0 \leq t \leq t_n; \\ = R_1 + t_n * R_{step} & \text{,if } \quad t > t_n, \end{cases} \tag{4}$$

where R_t increases approximately linearly over time when $1 \leq t \leq t_p$, and almost remains invariant if the device keeps stationary when $t > t_p$. We will verify it in Sec. 5.4.

We utilize a *tri-state state machine* to implemented the algorithm, as shown in Fig. 8(b). Given the input data, i.e. a sequence of positioning points, the state machine detects the stationary period of the device, and aggregates the drifting points to avoid the back-and-forth error in path recovery. The specifics of the algorithm are presented in **Algorithm 1**.

3.3 HMM Based Path Recovery

With the noisy points filtered and drifting points aggregated in the above modules, now we obtain a clean location sequence of a mobile device. The next step is to track the device on the road map by recovering its traveled paths.

In terms of *Path Recovery* procedure, the most critical issue is the tradeoff between the paths indicated by the positioning data and the actual applicability of the paths. In our case, this tradeoff is even more vital than previous GPS-based scenarios, because cellular positioning data has significantly larger error. That means, a path which is closest to a cellular positioning point might not be on the real route of the phone user.

Thus, our *CTS* system employs an HMM-based algorithm to make correct matches between noisy positioning data and the roads. Our major contributions in this algorithm are summarized as follows.

- Improving the precision of observation probability by utilizing sector parameters to indicate the more probable areas where the mobile device might appear. This enables the system to infer the traveled road segments with higher accuracy.
- Introducing turning and heading penalties into the calculation of transition probability, which reduces the appearances of U-turn and detour errors.

3.3.1 *Important Notations of the Model.*

Hidden States: Roads that the mobile user actually traveled are corresponding to *hidden states* of HMM. They are called *hidden* because we do not directly know the route that the user went through. We define them as $R = \{r_1, ..., r_t, ..., r_N\}$.

Observations: The locations measured by *CTS*, i.e., points in cellular-based trajectories, correspond to the *observations* of hidden states. For example, if $X = \{x_1, ..., x_N\}$ represents a trajectory, then $x_t, 1 \leq t \leq N$, are HMM observations.

State Candidates: The ground truth of a location point x_t may belong to any road segment close to x_t, due to the random error of cellular positioning. All the road segments within the error range are defined as *state candidates* of x_t. The set of state candidates is denoted by $S_t = \{s_{t,1}, s_{t,2}, ..., s_{t,N}\}$, where $s_{t,i}$ represents the ith road segment to which x_t might be mapped.

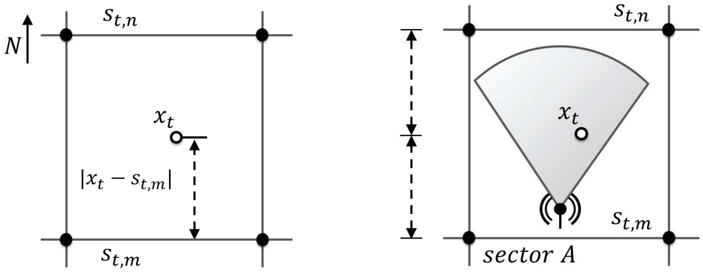
Observation Probability: $P[x_t|s_{t,m}]$, the *observation probability*, represents the probability that x_t can be observed when the hidden state is $s_{t,m}$. That means, when the phone user travels along a road $s_{t,m}$, he would be observed to be in the location x_t with a probability $P[x_t|s_{t,m}]$.

Transition Probability: $P[s_{t+1,m}|s_{t,m}]$, the *transition probability*, represents the probability of the state shifting from $s_{t,m}$ to $s_{t+1,m}$, i.e., the probability of the phone user moving into road $s_{t+1,m}$ at time $t+1$ after leaving road $s_{t,m}$.

ALGORITHM 1: Stationary/moving state detection.

```
Input: $X^{in} = \{x_1, ..., x_m, ..., x_n, ..., x_p, ..., x_q, ..., x_N\}$;
(A set of cellular positioning points when $1 < t < N$);
Output: $X^{out} = \{x_1, ..., x_{Agg}^{m:n}, ..., x_{Agg}^{p:q} ..., x_N\}$ ;
(The drifting points of $X^{in}$, $x_{m:n}$ and $x_{p:q}$, are merged respectively to $x_{Agg}^{m:n}$ and $x_{Agg}^{p:q}$);
Initialization, $ST_1 = stationary, t = 1, R_t = R_1, c_t = x_1, x_i = x_1$;
for $t = 2; t < N; t = t + 1$ do
    if $ST_t == stationary$ then
        $X_t^{in} = \{x_i, x_{i+1}, ..., x_t\}$;
        Calculate the geometric center, $c_t$, of $X_t^{in}$;
        Draw circle $C_t$, whose center is $c_t$, radius is $R_t$;
        if the new coming point $x_{t+1}$ falls in $C_n$ then
            $ST_t = stationary$, (notation ①);
        end
        else
            $ST_t = transient$, (notation ②);
        end
    end
    else if $ST_t == transient$ then
        set a sliding window $W$ with $width = Wid$;
        while $W$ is not full do
            load $x_t$ into $W$;
            $t = t + 1$;
        end
        if $Wid * 0.9$ or more pts in $W$ are out of $C_t$ then
            $ST_t = moving$, notation ③);
            Regard $X_t^{in}$ as drifting points;
            Aggregate $X_t^{in}$ to $x_{Agg}^{i:t}$, where $x_{Agg}^{i:t} = c_t$;
        end
        else
            $ST_t = stationary$
        end
    end
    else if $ST_t == moving$ then
        Determine whether the device still moves, or turns stationary.
    end
end
```

Formulation of the Path Recovery Problem: With the notations above, the path recovery problem can be mathematically defined as follows. Given an observation sequence (i.e., a cellular positioning trajectory) $X = \{x_1, ..., x_t, ..., x_N\}$, and sectors parameters S_P, our goal is to define the most probable state sequence $Y = \{y_1, ..., y_t, ..., y_N\}$ as the recovered path, where $y_t \in S_t, 1 \leq t \leq N$. Then, the sequence Y would be the closest approximation of hidden states R, which is the actual but unknown roads traveled by the user.



(a) Definition of $|x_t - s_{t,m}|$

(b) Situation where the phone is connected to sector A

Fig. 9. Illustrations of notations related to observation probability

3.3.2 Observation Probability. The observation probability $P[x_t|s_{t,m}]$ gives a measurement of how possible it is to observe x_t when the phone user is moving on the road segment $s_{t,m}$. It is reasonable to consider that the shorter the distance between road $s_{t,m}$ and location x_t is, the larger the probability of observing x_t from $s_{t,m}$ is. Meanwhile, the error of cellular positioning causes the mismatch between a positioning point and the actual traveled road segment. Here, we suppose the positioning error to follow Gaussian distribution, with standard deviation of σ_x. Then, we are able to temporarily model the observation as:

$$p(x_t|s_{t,m}) = \frac{1}{\sqrt{2\pi}\sigma_x} \cdot e^{-0.5(\frac{|x_t - s_{t,m}|}{\sigma_x})^2}, \tag{5}$$

where the measurement $|x_t - s_{t,m}|$ is the spherical distance between x_t and $s_{t,m}$, as shown in Fig. 9(a).

In Fig. 9(b), the observation x_t has equal distances to $s_{t,m}$ and to $s_{t,n}$, i.e., $|x_t - s_{t,m}| = |x_t - s_{t,n}|$. In *CTS* system, with the knowledge of sector parameters $S_P = \{\theta_r, \theta_d, V_s\}$, we can figure out which one of $s_{t,m}$ or $s_{t,n}$ is the more probable state. Suppose that sector A is deployed near $s_{t,m}$, with its direction angle to be $\theta_d = 0°$ (North oriented) and its radiation angle to be θ_r. Apparently, sector A is directed towards $s_{t,n}$, and back against $s_{t,m}$. If a phone is connected to sector A and is estimated to locate at x_t, the phone is more likely staying on road $s_{t,n}$. This situation is modeled as:

$$p(x_t, S_A|s_{t,m}) = p(S_A|s_{t,m}) \cdot p(x_t|s_{t,m}, S_A), \tag{6}$$

where S_A represents the sector to which the phone is connected. Since being connected to S_A is the prerequisite to sampling the location x_t, we rewrite (6) as:

$$p(x_t|s_{t,m}, S_A) = \frac{p(x_t|s_{t,m})}{p(S_A|s_{t,m})}, \tag{7}$$

where $p(x_t|s_{t,m}, S_A) \geq p(x_t|s_{t,m})$ since $p(S_A|s_{t,m}) \leq 1$.

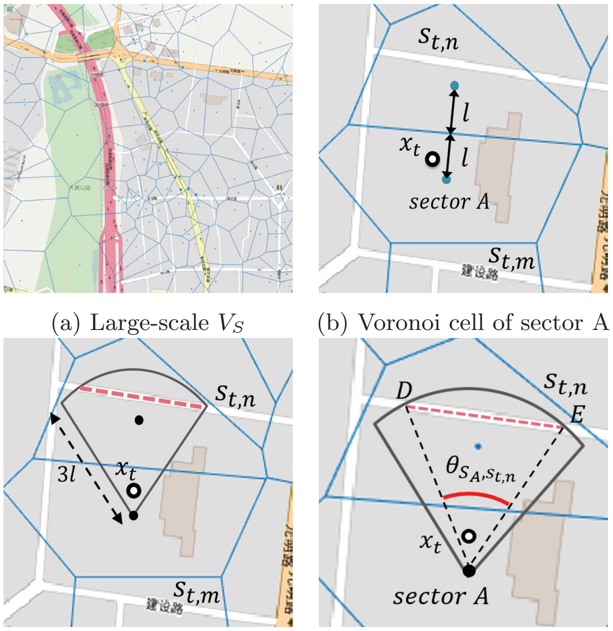
We mark $p(x_t|s_{t,m}, S_A)$ as $p^*(x_t|s_{t,m})$, and denote $p(S_A|s_{t,m})$ by factor $\boldsymbol{f_{S_A}}$, which measures the influence of sector A. Then, we rewrite (7) as:

$$p^*(x_t|s_{t,m}) = p(x_t|s_{t,m}) \cdot f_{S_A}, \quad f_{S_A} \geq 1. \tag{8}$$

Here, we derive the final observation probability $p^*(x_t|s_{t,m})$, and calculate $p(x_t|s_{t,m})$ based on (5). Regarding f_{S_A}, we propose a rule-based method to compute it.

Rule-based Method for Calculating $\boldsymbol{f_{S_A}}$: To calculate f_{S_A}, we use a two-step method.

First, we determine if x_t and $s_{t,n}$ are covered by sector S_A. We introduce Voronoi Diagram V_s to separate the map plane into thousands of *cells*, as illustrated in Fig. 10(a). The center of every cell is a *cellular sector*. For example, the blue solid dots in Fig. 10(b) represent two sectors. The lower



(a) Large-scale V_S (b) Voronoi cell of sector A

(c) Coverage of sector A (d) Chord angle θ_A

Fig. 10. Illustration of the method for calculating f_{S_A}

one is S_A. The distance between S_A and the cell border above it is $\boldsymbol{l}$. The simplest way to model the sector coverage is to regard the whole cell as the coverage range, but usually cellular sectors are not omnidirectional ($\theta_r \neq 360°$). Therefore, we draw a fan-shape, whose radius is $3l$, and refer other parameters to $S_P = \{\theta_r, \theta_d, V_s\}$ (Fig. 10(c)).

Second, we determine the formula of calculating f_{S_A} according to whether $s_{t,n}$ is included by the fan-shape. The road segment $s_{t,n}$ is considered to be included *in* the fan-shape when they have intersecting parts. If either x_t or $s_{t,n}$ is out of the fan-shape, f_{S_A} is set to be 1. Otherwise, if the road segment $s_{t,n}$ intersects the fan-shape at chord DE (Fig. 10(d)), then the chord angle, notated as θ_A ($\angle DAE$), ranges from 0 to θ_r. Assume that θ_A is positively correlated with f_{S_A}, f_{S_A} can be calculated as below.

$$f_{s_A} = 1 + \delta(x_t, s_{t,n}) * s * (1 - e^{\theta_A/\theta_r}), \tag{9}$$

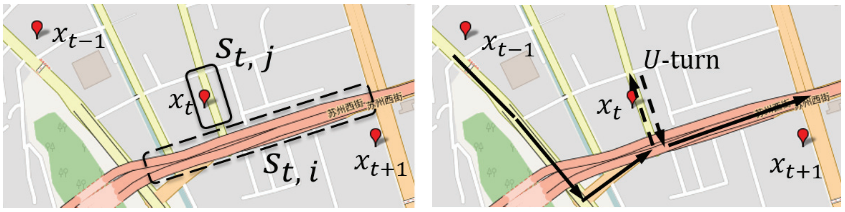
where $\theta_A \in [0, \theta_r]$, and s is the scalar parameter. In addition, $\delta(x_t, s_{t,n}) = 1$ when x_t and $s_{t,n}$ are both covered by the sector, and otherwise equals to 0.

Substituting (9) into (8), we obtain the modified observation probability as follow:

$$p^*(x_t|s_{t,m}) = \frac{1}{\sqrt{2\pi}\sigma_x} \cdot e^{-0.5(\frac{|x_t - s_{t,m}|}{\sigma_x})^2} * [1 + \delta(x_t, s_{t,n}) * s * (1 - e^{\theta_A/\theta_r})], \tag{10}$$

and the related parameters in (9) are estimated in Sec. 5.5.

3.3.3 Transition Probability. Suppose the mobile user travels in road segment $s_{t-1,m}$ at time $t-1$. At time t, he might move to one of the state candidates in $s_{t,n}$. We use *transition probability* to measure its



(a) Erroneous positioning point x_t

(b) U-turn error

Fig. 11. An example scenario of U-turn error.

possibility[17]:

$$p(s_{t,n}|s_{t-1,m}) = 1/\beta \cdot e^{-d_t/\beta}, \tag{11}$$

where d_t is the difference of the distance the phone user have to travel from $s_{t-1,m}$ to $s_{t,n}$, and the straight-line distance between the corresponding location x_{t-1} and x_t. β is estimated according to the dataset (discussed in Sec. 5.6). C. Goh [22] proposed a *distance discrepancy function*, where the historical average velocity of each road segment was taken into consideration. This function could only be applied when we know the transportation mode of the phone user, whereas there is no such information in most daily applications. Thus, in *CTS* system, we modify (11) into (12), taking into account two penalties: (a) *U-turn penalty* λ_U; (b)*direction changing penalty* λ_{DC},

$$p(s_{t,n}|s_{t-1,m}) = \lambda_U \cdot \lambda_{DC} \cdot \frac{1}{\beta} \cdot e^{-d_t/\beta}. \tag{12}$$

The estimations of λ_U and λ_{DC} are conducted in Sec. 5.6. **U-turn Penalty** As mentioned in Sec. 2.2, the errors of cellular positioning points can be as large as hundreds of meters. Given two candidates $s_{t,i}$ and $s_{t,j}$, although $s_{t,i}$ is the actual road segment the user travels in, $s_{t,j}$ which is closer to x_t would earn higher transition probability than $s_{t,i}$. As illustrated in Fig. 11 (a), the wrong state $s_{t,j}$ is included in the recovered path sequence, leading to a U-turn error (Fig. 11(b)).

When calculating $p(s_{t,n}|s_{t-1,m})$, we use the Dijkstra shortest path algorithms to find the optimal path from $s_{t-1,m}$ to $s_{t,n}$, represented as $P_{t,t-1} = \{r_{t,1}, ..., r_{t,M}\}$. Since these road segments $\{r_{t,1}, ..., r_{t,M}\}$ are connected one by one, if any two consecutive segments, say $r_{t,i-1}$ and $r_{t,i}$, are pointing to opposite directions, we add a penalty λ_U to $p(s_{t,n}|s_{t-1,m})$ in order to eliminate the U-turn error.

Direction Changing Penalty Aside from U-turn error, large positioning errors could cause *detour error* as well, as shown in Fig. 4(b). A practical way to eliminate the detour error is to add *direction changing penalty* λ_{DC} to path $P_{t,t-1} = \{r_{t,1}, ..., r_{t,M}\}$ which includes too many turns, i.e., changing direction too frequently. In this way, the detour error could be excluded by Viterbi Algorithm[23] while finding the most likely states sequence, i.e., the most likely sequence of road segments the user traveled.

$$\lambda_{DC} = b_{DC}^{-\Sigma\theta_i/(0.5\pi)}, \tag{13}$$

where $\theta_i(1 \leq i \leq N_{turn})$ is the angle of each turn on path $P_{t,t-1}$, and N_{turn} is the number of turns included in path $P_{t,t-1}$.

Finally, with the emission probability and transition probability, expressed by (10) and (12), the Viterbi algorithm can efficiently calculate the optimal route on the digital map for a trajectory.

In summary, *CTS* locates a mobile device in three steps. Firstly, locations with low accuracy are estimated using the trilateration method. Secondly, we utilize a state machine to detect stationary points and aggregate drifting points. Finally, an HMM-based path recovery algorithm is applied to accurately locate the mobile device on the road map.

4 SYSTEM IMPLEMENTATION AND DETAILS

We implemented *CTS* based on the cellular infrastructure owned by a mobile operator in China. The system was deployed and tested in Urumchi city, the capital city of Xinjiang province of China. The cellular positioning module is built based on the CDMA cellular infrastructure[24], while the state detection and path recovery algorithm are deployed in a central server of the mobile operator. The database of sector parameters is maintained by the mobile operate. Besides the cellular positioning data, we also collected GPS traces by sniffing and resolving the anonymous subscribers' internet accessing logs in the cellular network[1].

4.1 Architecture of Implemented System

To provide an overview, we draw the system of the implemented system in Fig. 12. The left part of the figure illustrates the cellular positioning system, while the right half is the rest parts of *CTS*.

As shown in the figure, when a mobile station(MS) communicates with the base station subsystem(BSS), the measurement data for trilateration positioning is sent to mobile service center(MSC). Then, the mobile positioning center(MPC) sends such measurements to the position determining entity(PDE), who calculate the position of the mobile phone by trilateration algorithm. Next, the positions are transmitted as trajectory data to the central server of our system. There, the algorithm components, including filters, state detection, and path recovery, are utilized to track the mobile phone in the road network. Finally, the tracking results can be used by various location-based services in the form of location coordinates.

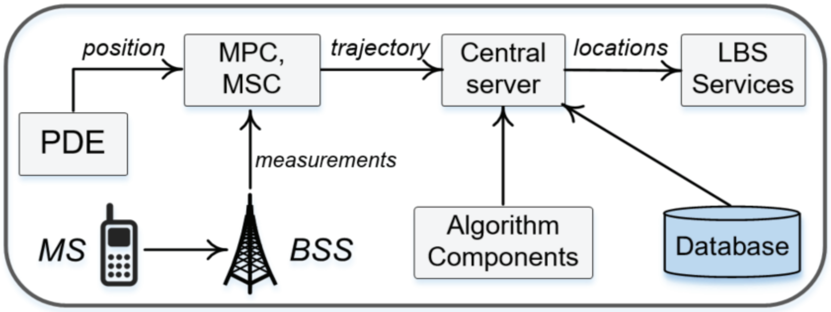


Fig. 12. Architecture of the implemented system

4.2 Collecting Cellular Positioning data

Our cellular positioning data was collected during one day, from 23:40 October 9th, 2016 to 23:40 October 10th, 2016 in Urumchi. Only at the moment when a subscriber launches or terminates a message, voice or data traffic service, the cellular-network-side locates the subscriber based on trilateration. Afterwards, the cellular network outputs a piece of positioning data, represented by a tuple, to a database:

$$Positioning\ Data = \{UserID,\ Location,\ Time,\ SectorID\},$$

where the *Location* is in the format of $(latitude, longitude)$, and $SectorID$ refers to the IDs of sectors that accomplished the trilateration positioning.

[1] All the sniffed packets are anonymous, i.e., we analyze the packets without knowing to whom they belong

Table 2. Detailed Description of Cellular Positioning Data

Item	Description
Location	Urumchi, the capital of Xinjiang Province, China
Time Period	23:40 October 9th 2016 to 23:40 October 10th 2016 (24 hours)
Triggers of Data Sampling	Voice Service, Message Service, Data Traffic Service
Amount of Subscribers	489,032
Percentage of Different Subscribers	2G users: 7%, 3G users: 93%
Average Amount of Data each User Generates	276 pieces / 24hours
Median Amount of Data each User Generates	134 pieces / 24hours

Among anonymous 489,032 subscribers in Urumchi, we recorded 135 million pieces of positioning data. Each subscriber generated on average in 24 hours 276 pieces of data. The median during the 24 hours is 134. If we look into the time period from 8:00 AM to 8:00 PM during which users are active, we only get a median of 11.2 pieces of positioning data per hour for each user, i.e., the median sampling rate of our dataset is 11.2 samples per hour at best. This rate is extremely low compared with that of the GPS's, which is generally at least 60 samples per hour. But even worse, the positioning accuracy is also much lower than GPS's. As mentioned before, the median error of cellular positioning is 143 meters, and 90% of errors are less than 215 meters. More details and system parameters are summarized in Table 2.

4.3 Sector Parameters Database

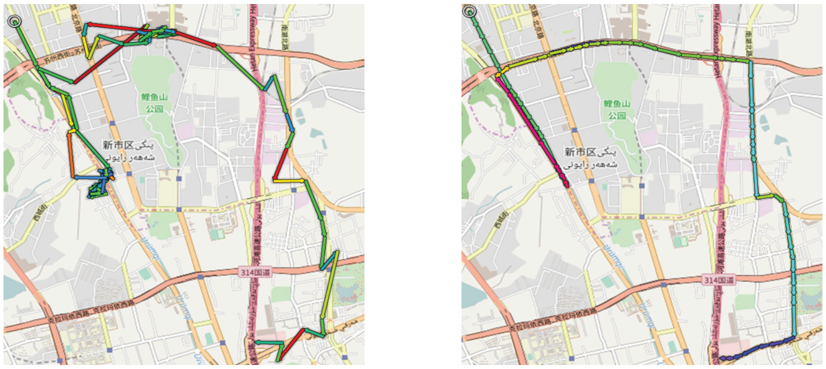
We are authorized to access a database that stores the information of all the 5,367 sectors that possessed by the mobile operator in Urumchi. From this database, we can acquire precise locations of all sectors, in the form of $Loc = (latitude, longitude)$, and part of their operating parameters, including orientation (Direction angle) and opening angle (beam width). These parameters are denoted as $S_P = \{L, \theta_o, \theta_d, V_s\}$, as introduced in Sec. 2.1.

4.4 Ground Truth Trajectories

Ground truth trajectories are the paths/routes that the phone users actually traveled when the cellular positioning data is generated, which are needed for system performance evaluation. Since the ground truth path is spatially continuous, the data providers [19] usually sample GPS points sequence as the substitution. The ground truth GPS points are typically collected through Apps installed in appointed mobile devices.

The installation process and data uploading requires extra resources. Thus, we design an alternative method to obtain the ground truth trajectories, mainly by sniffing and resolving data packets of users':

- *Sniff* the data packets when users send *GET* requests from their Apps. Usually, a *GET* request contains a Uniform Resource Locator (URL) that the user tends to visit. Some of such URLs are generated by Apps providing Location-based Services(LBS), such as Google Map and Uber. These URLs may contain latitude and longitude coordinates in them.
- *Extract* Latitude and Longitude coordinates from these URLs.
- *Sort* these coordinates by user's ID, and *assemble* the coordinates of each user into GPS trajectories.
- *Match* these GPS trajectories with the cellular positioning trajectories by user's ID.



(a) Cellular trajectory (b) Resolved GPS trajectory

Fig. 13. An example of a cellular trajectory and its ground truth GPS trace.

By this means, we resolved 201 GPS traces, whose total distance is 1696 kilometers. An example of a cellular positioning trajectory and its corresponding GPS ground truth trajectory are shown in Figure 13.

5 PARAMETER SELECTION

In this section, we estimate the parameters of all three system components, either by utilizing the statistical characteristics of the trajectory data or by conducting numerical experiments on the dataset. Because of the system includes multiple parameters, it is critical to make the relationships of their estimation processes clear in Table 3.

Table 3. Relationships among estimations of multiple parameters.

Estimation method:	numerical	statistical	dependencies
a. $\{th_1, th_2\}$ in Speed-Based Noise Filter	×	√	×
b. α_d in Direction-based Noise Filter	√	×	a,c
c. $\{R_1, R_{step}, t_0, t_n\}$ in State Detection	×	√	×
d. $\{\sigma_x, \beta\}$ in HMM	×	√	×
e. Sector Factor f_{S_A}	√	×	a,b,c
f. U-turn Penalty λ_U	√	×	a,b,c,d,f
g. Direction Changing Penalty λ_{DC}	√	×	a,b,c,d,e

Note that, since the system is application-oriented, it is acceptable to perform coarse-grained numerical calculation to obtain suboptimal parameters, considering the time cost. More precise values could be calculated with higher time cost if necessary.

5.1 Metrics of System Performance

Before conducting the experiments in this section and evaluations in 6, we define two metrics of system performance to make the following discussion clear. We define the *overall relative accuracy of distance* as:

$$a_d = 1 - \frac{\Sigma_i |d_r^i - d_g^i|}{\Sigma_i d_g^i}, 1 \leq i \leq N_T, \tag{14}$$

where d_r^i refers to the distance of the path recovered from the ith trajectory, d_g^i is the distance of the ith ground truth path, and N_T is the total number of the tested trajectories.

Similarly, the *overall relative accuracy of number of road segments* is defined as:

$$a_s = 1 - \frac{\Sigma_i |n_r^i - n_g^i|}{\Sigma_i n_g^i}, 1 \leq i \leq N_T, \tag{15}$$

where n_r^i represents the number of road segments of path that recovered from the ith trajectory, and n_g^i refers to the number of road segments of the ith ground truth path.

For simplicity, a_d and a_s are called **accuracy of distance** and **accuracy of segment** in the rest of the paper.

5.2 Speed-Based Noise Filter

In our system, we utilize the speed-based noise filter to remove points with abnormally high speeds, where the speed thresholds, $th_{1:2}$, in (3a) and (3b) need to be configured. In the digital map, roads are labeled with different category tags, and each type of road has its max-speed key, indicating the maximum legal driving speed on the road. In order to determine the value of $th_{1:2}$, we analyze the points with speeds **higher** than max-speed on each type of road, and draw the box-plots in Fig. 14 to demonstrate their distributions. Note that, since the positioning points have not been matched onto the road map, we assign each point to the nearest road. This approximation makes sense because the attribute of different roads in a local area are typically similar, meaning that there is little difference between the speed limits.

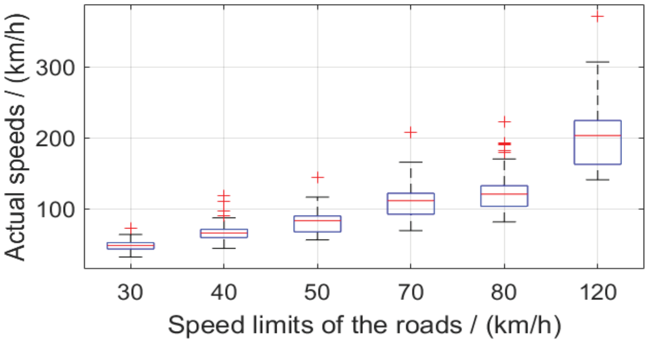


Fig. 14. Actual speeds of users on roads with different speed limits.

We observe that the variance of speeds increases with the speed limit. The highway, with the speed limit of 120km/h, has the largest variation and upper extreme, possibly due to the sparsity of cellular infrastructures in highway.

For th_2 in (3b), in order to tolerate acceptable positioning errors, the value should be slightly higher than speed limit, so that we adopt the median of each box. (3a) provide a more decisive rule, therefore, the value of th_1 ought to be high enough to detect the noise points. Hence, we use upper extreme of each box as the threshold for (3a). Medians and upper extremes of different types of roads are listed in Table 4.

Table 4. Statistics of roads with different speed limits.

Road Limit Speed (km/h):	30	40	50	70	80	120
Median (km/h)	48	66	83	111	121	203
Upper Extreme (km/h)	64	87	116	165	170	307

5.3 Direction-based Noise Filter

Direction-based noise filter is introduced in our system to eliminate the back-and-forth errors. We tune the value of α_d with the step of $10°$ to obtain a suboptimum with low time cost. Note that, when α_d is set as a flat angle, the filter is not active, because any angle of direction changing is less than $180°$. As shown in Fig. 15, when α_d is in the range of $[140°, 160°]$, the accuracy of the recovered paths is much higher. We set α_d as $150°$ in the following evaluations.

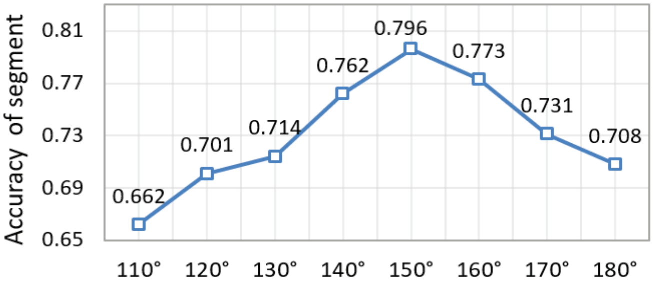


Fig. 15. Overall accuracy of segment with different α_d.

5.4 State Detection

In our system, the detection of stationary state is accomplished by the state machine, in which parameters $\{R_1, R_{step}, t_0, t_n\}$ need to be determined. When a user is in the *stationary* state, we find that his or her cellular positioning points spread into a wider range along the time. We adopt the *standard deviational ellipse*[25] to measure such spreading trend.

As shown in Fig. 16, a standard deviational ellipse measures the geographic distribution of the spatial data, with its long axis pointing to the direction that the data shows maximum deviation. Thus, the long axis diameter indicates the range the cluster covers. We calculate ellipses of 200 drifting points clusters extracted from cellular trajectories, and study the variation of the long axis diameters over time.

Fig. 17 shows that the median of the long axis diameters increases from 330 meters, and tends to be stable at 800 meters when the user stays for more than 2 hours. Moreover, the median of diameters rises approximately linearly, with the slope of 320 meters per hour. Therefore, in (4), the initial radius R_1 equals to $330/2 = 165$ meters, the increasing rate, R_{step}, is set as 320 meters/hour, and the diameter-varying time window $[t_0, t_n]$ should be in the range of $[0.5hour, 1.75hour]$.

Recall the cellular positioning method discussed in Sec. 3.1, the mobile phone is located by three cellular sectors. However, when phone user remains stationary or moves only in a narrow range for a

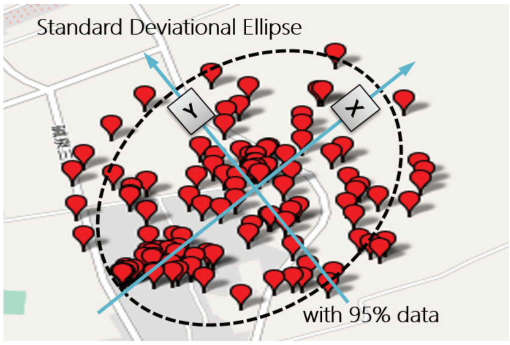


Fig. 16. Standard deviational ellipse of a drifting points cluster.

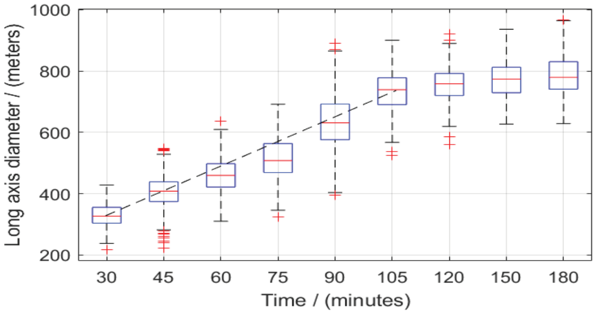


Fig. 17. The variation trend of long-axis diameters with time.

certain period of time, the cellular-side positioning system might allocates another set of sectors to perform the positioning process. As a result, the new generated positioning points would show a new spatial distribution. Further, if the user stays for several hours, he might be located by all possible sets of nearby sectors. Then, the distribution of the positioning points keeps stable.

5.5 Observation Probability

When calculating the observation probability of HMM-based path recovery algorithm, the standard deviation of the cellular positioning error σ_x in (5), and the sector factor f_{s_A} in (9) are to be estimated. We calculate the former based on its definition, and it equals to 130 in our system.

The f_{S_A} is employed to adjust the influence of sector information on path recovery. According to (9), we need to estimate the scalar parameter s, and then calculate f_{S_A} using that equation. In order to speed up the estimation process, multi-round process is recommended, during which the step size decreases with rounds. Here, we perform 2-round estimation.

In the first round, the initial value is set as $s = 10^{-1}$, and the value rises tenfold in each step, up to $s = 10^3$. Dotted line in Fig. 18 shows that the peak falls when $s = 10^1$. We perform the second round to further determine that $s = 10^{0.9}$ is the optimum when the step size is 0.1 in logarithm. In the following evaluations, we set $s = 10^{0.9} \approx 8$.

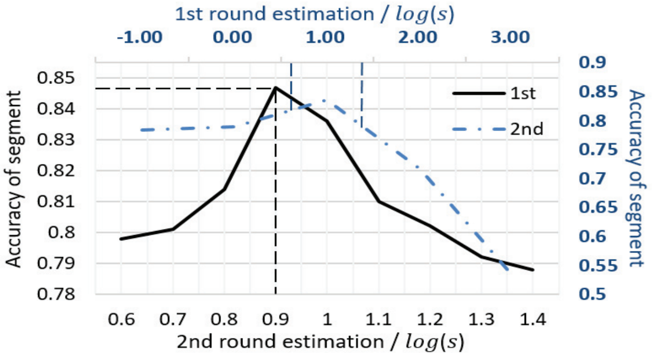


Fig. 18. The 2-round estimation of scalar factor s.

5.6 Transition Probability

When calculating the transition probability for path recovery, we need to determine the values of U-turn penalty λ_U, direction changing penalties λ_{DC}, and the normalization factor β in (12). β is estimated according to the method proposed by P. Newson[17], and it is estimated to be 200 meters in our system.

The other two parameters, λ_U and λ_{DC} are introduced to eliminate the back-and-forth error and the detour error, respectively. To reduce the computational cost, we separately estimate them coarsely in the first place, and then conduct joint-estimation with a finer granularity. Note that, for λ_{DC}, we should estimate its base number b_{DC}, and then calculate it based on (13).

In Fig. 19, we observe that the accuracy varies slightly with the U-turn penalty λ_U, and the optimum falls in $[-1, -2.33]$ in logarithm. While $[0.6, 0.9]$ is the preferred range of b_{DC}, which is the base number of λ_{DC}, the accuracy drops down sharply when b_{DC} approaches zero.

These trends are consistent with our expectation. If b_{DC} is too small, in order to avoid the direction changing punishment, the path recovery process would choose the route that has as few turnings as possible, regardless other regulations. Moreover, when the λ_U approaches zero, the accuracy drops slightly, because in cellular trajectories there are few cases of actual U-turns, and the drop-down of performance comes from misjudging the U-turns as wrong recoveries.

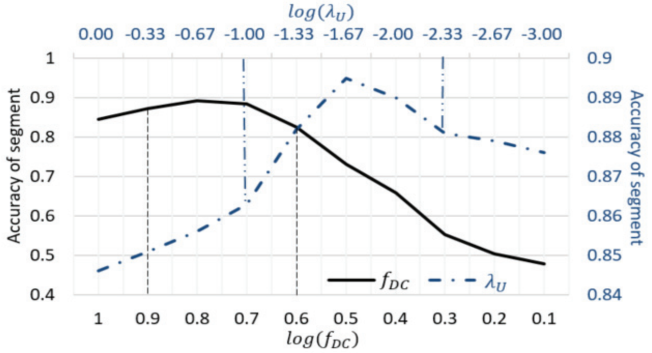


Fig. 19. The preliminary estimation of λ_U and b_{DC}.

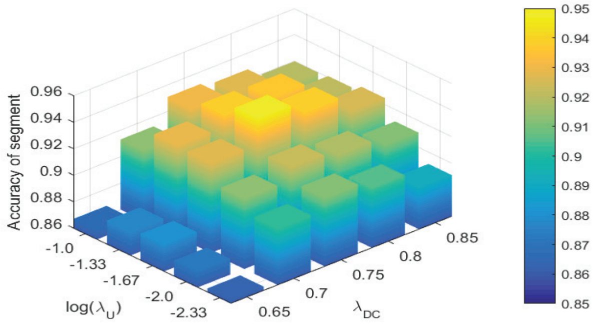


Fig. 20. The joint estimation of λ_U and b_{DC}.

After the coarse and separate estimations, we perform 2-D numerical calculation in a finer granularity. The step size can be smaller if necessary. According to Fig. 20, the optimum is near $(\lambda_U, b_{DC}) = (10^{-1.67}, 0.75)$. Thus, we set $(\lambda_U, b_{DC}) = (0.02, 0.75)$ in the following evaluations.

6 EVALUATION

We conduct comprehensive experiments to evaluate the proposed system. We first compare our system with five baseline algorithms, and analyze the effectiveness of each component of the algorithm. Second, we evaluate the impact of two external factors, road density and sampling rate, on the system performance. All the experiments are based on the test dataset[2], which consists of cellular trajectories and their ground truth GPS trajectories. The specifics of the test dataset are illustrated in Table 5. The average sample rate of GPS trajectories is three times higher than that of cellular trajectories, as well as useful points. The total distances of the two kinds of trajectories are basically equal.

Table 5. Details of Test Data.

Item	**Cellular Trajectories**	**GPS Trajectories**
Average sample rate	18.2 Pts/hour	81.3 Pts/hour
Total positioning points	19,828 Points	87,277 Points
Removed drifting points	12,452 Points	42,662 Points
Useful points	7,376 Points	44,615 Points
Total distance	1637 kilometers	1696 kilometers

6.1 Tracking Performance Evaluation

In this subsection, we firstly evaluate the overall performance of *CTS* against several baseline algorithms, and then examine how much accuracy promotion each meta-component of *CTS* brings about.

[2] We explained in Sec. 4.2 that the ground truth of cellular trajectories are resolved from sniffed data, but only part of the trajectories have ID-matched sniffed data.

6.1.1 Performance Against Baselines. To demonstrate *CTS* is more suitable for tracking the mobile device, we compare it with Newson's *Simple HMM* algorithms[17], and Mohamed's *SnapNet* system[10] in terms of accuracy of segment and accuracy of distance. Besides, three weakened versions of *CTS*, *CTS-1,CTS-2, and CTS-3*, are used to test the effects of each system module. *CTS-1, and CTS-2* represent *CTS* without applying noise filters and the state detection algorithms, respectively. *CTS-3* is the system without applying the improvement of HMM-based path recovery, including sector factor, U-turn penalty and direction changing penalty.

As shown in Fig. 21, a_s and a_d of *CTS* are 94.7% and 95.7%, respectively, which demonstrate that our system achieves GPS-level accuracy for mobile device tracking. The accuracy of *CTS* is 30% higher than *SnapNet* and 50% higher than simple HMM algorithm by absolute percentage.

SnapNet is designed for tracking task based on cellular base-station footprints. The system only knows the base stations that the device connected, and the accuracy is intrinsically influenced by the spatial density of base stations. Simple HMM model is appropriate for recovering the path of GPS positioning trajectories. But it is not endurable for positioning points with large errors, which are the normal cases in cellular positioning system.

CTS improves the accuracy through successful combination of cellular positioning and several supporting algorithms. More specifically, the noisy positioning points are filtered out by speed-based and direction-based noise filters. The state detection process effectively solves the drifting issues. Low sampling rate causes long spatial interval between consecutive positioning points. This challenge is overcome by our HMM-based path recovery algorithm. To further improve the performance, we also utilize sector information and design penalties to eliminate U-turn and detour errors.

Note that the accuracy of distance a_d is slightly lower than that of segment in case of *CTS*. This is because mismatches of road segments sometimes occur at the starting and the ending points of a trajectory, due to the inevitable positioning errors. Each mismatch only counts once when calculating a_s, but a single mismatch may significantly decrease a_d. However, a_d is much higher than a_s in cases of simple HMM, *Snapnet* and *CTS-2*. It happens when the lengths of the mismatched segments are close to those of the correct segments. Fig. 21 also shows that a_d and a_s drops by around 10% when disabling noise filters or the improvements of HMM-based path recovery, and by at least 15% with the state detection module removed.

The results above show that *CTS* outperforms all the five baselines in both metrics, which proves our system is the most appropriate one for cellular trajectory tracking.

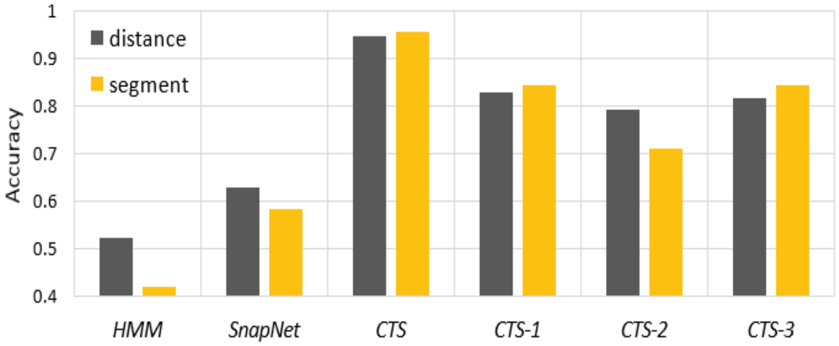


Fig. 21. Overall performance against baselines.

6.1.2 Effects of Components of Each Module. The importance of the state detection module has been discussed before, and now we evaluate the effects of the other two modules in detail.

First, we disable the speed-based noise filter and the direction-based noise filter, respectively, and keep the rest of the system unchanged while measuring the tracking accuracy. We learn from Fig. 22 that a_s and a_d both decrease by 9% by disabling speed-based noise filter. This makes sense because the median positioning error is up to 143 meters in cellular system, making the abnormal speeds appear frequently. Contrarily, the performance decline is rather small, which is around 3%. The reason is in the path recovery process, the U-turn penalty could help eliminate some incorrect matches caused by back-and-forth errors.

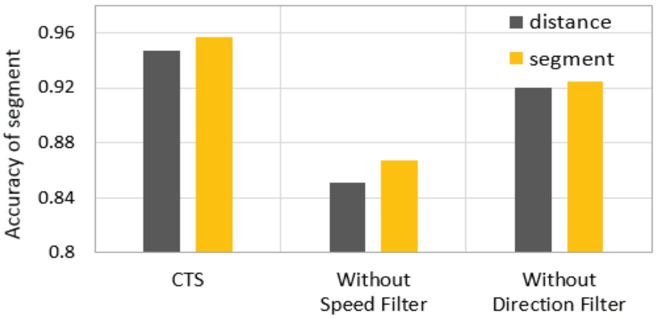


Fig. 22. Overall performance with and without filters.

Second, we disable three components that improves the path recovery algorithm, including sector factor, U-turn penalty and direction changing penalty, and show the results in Fig. 23. From the results, we observe that sector information provides more accuracy advancement than direction changing penalty, though they are both expected to reduce detour errors. U-turn penalty proved to be the most effective component among the three, with the increase of about 6%. Note that the U-turn penalty and direction-based noise filter are both designed for eliminating back-and-forth errors, but the former one performs better. Recall the situation shown in Fig. 11, x_t generates a U-turn error without causing a directional problem, because the sampling rate is too low to capture the details of direction error. Such error could be removed by U-turn penalty.

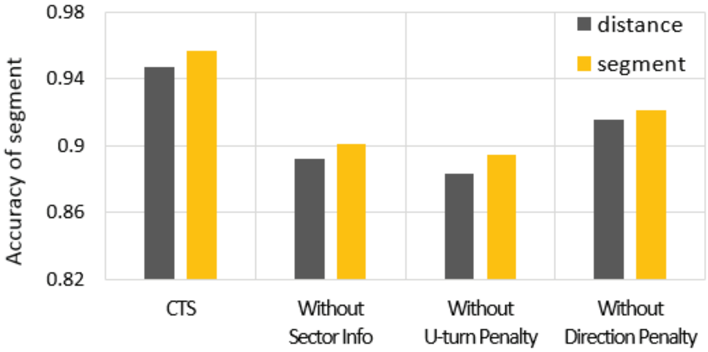


Fig. 23. Overall performance with and without improvements of HMM-based path recovery.

To clearly demonstrate the impact of each component of the algorithm, we list the accuracy advancement by each of them in Table 6. The performances are evaluated by only disabling one of the components while keeping the others functioning normally.

Table 6. Accuracy advancement of each Component.

Advancement of:	Accuracy of segment	Accuracy of distance
State detection	24.6%	15.5%
Speed filter	9.0%	9.5%
U-turn penalty	6.4%	6.2%
Sector info	5.5%	5.8%
Direction penalty	3.1%	3.5%
Direction filter	2.7%	3.2%

6.2 Influence of External Factors

The density of road networks and the positioning sample rate are two important external factors that might influence the system performance. In this section, we investigate these related issues.

6.2.1 Density of Road Networks. Intuitively, the difficulties of trajectory tracking varies with density of road networks. When the roads are sparse, the task is easier because there are only few alternative routes. On the contrary, if the roads are dense, the positioning errors are more probable to cause confusions in path recovery. In the *downtown, ordinary city area,* and *suburb* of Urumchi City, we select three representative $4km \times 4km$-sized square areas. where the roads are dense, medium dense, and sparse, respectively. Then we calculate the density of road and cellular sectors, and show the results in Fig. 24.

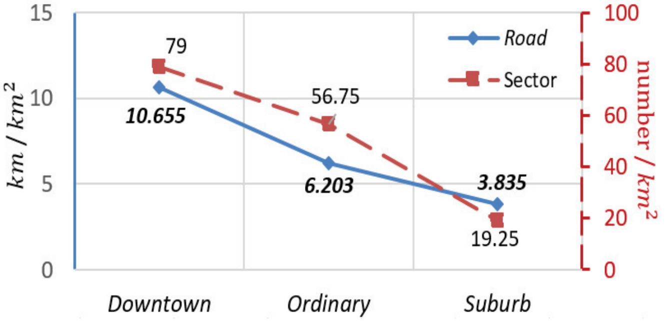


Fig. 24. Relevance between road density and sector density.

We observe from the results that the densities of road networks and cellular sectors are positively correlated. The downtown area has $10.66km$ roads and 79 sectors in average per square kilometer, while that of the suburb area are both low. We extract trajectories that pass through these areas, and evaluate the performance of *CTS* and *CTS* without the sector factor with those trajectories. The results are shown in Fig. 25. The result shows that the best tracking performance can be achieved in suburb area, which verifies our intuition. The system performance is rarely influenced by the sector information, because the regulation provided by road network structure is sufficient. While in the downtown area, the system performs worst. However, the sector information helps *CTS* achieves 7% higher accuracy in the downtown. Because when the roads are dense, the sectors are dense as well, and the positive effect of sector information is significant helpful where the road network is dense.

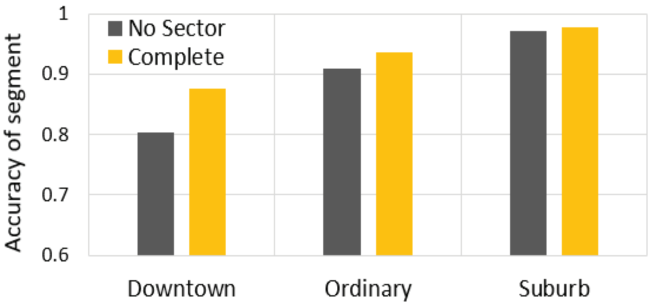


Fig. 25. Overall performance with and without sector information in different areas.

6.2.2 Sampling Interval. It is obvious that the sampling rates of trajectories are changing with time and users. We would like to investigate how the system performance varies when the sample rate changes with time and with users respectively. Note that for convenience in data analysis, we use the sampling interval, the interval time between two samples, to represent the sample rate.

First, we look into the distribution of sample intervals in different time of the day. In Fig. 26, we separate the sample intervals into six groups, and each group is denoted by a poly-line. Then we count how many points fall into each group in different periods of time in a day. For example, if a point is sampled 90 seconds later than the previous one, in 10:00 o'clock, then it falls into the range of $[09:00-11:59, 60-120sec]$. The figure shows that for all the groups, the positioning points collected in $00:00-08:59$ are much fewer than other periods, since most users are resting. In this figure, what we concern more is the fact that the proportion of each sample interval in every period is basically the **same**[3].

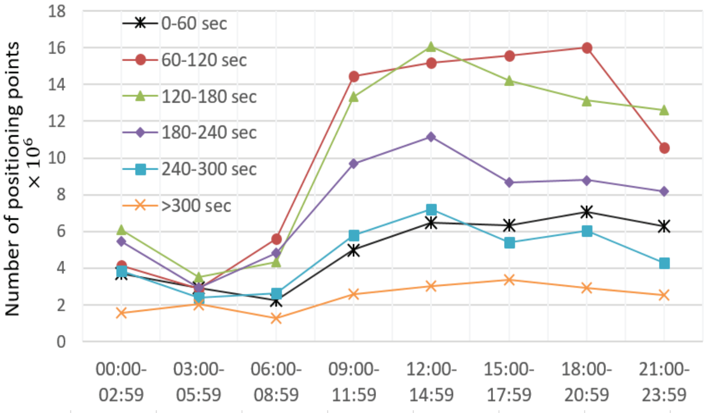


Fig. 26. Distribution of sampling intervals during 24 hours.

[3] Note that, here we discard trajectories that only consist of stay points, or are too short to be matched into the road map, making them irrelevant to this topic

Second, according to the result above, it is reasonable to infer that the system performance will not change sharply over time during a day. We conduct corresponding experiments, and the results are shown in Fig. 27. The figure shows that the best accuracy occurs during $15:00-17:59$, which is only higher than the worst case by 2.2%. This consequence is consistent with the data analysis.

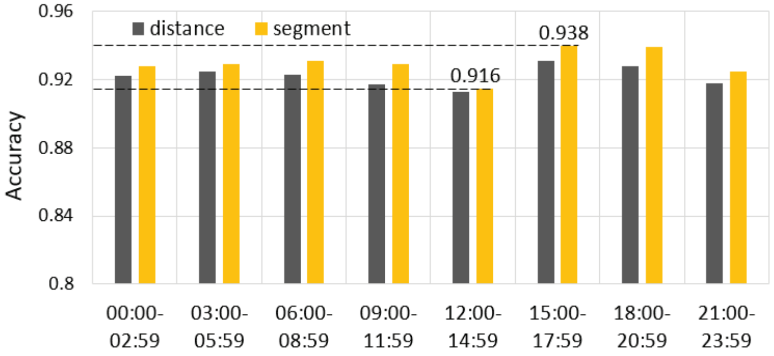


Fig. 27. Overall performance in different time of the day.

Third, we examine the distribution of users in term of sampling interval. We calculate the median sampling interval of each user during a day, and draw the bars in Fig. 28. From the figure we observe that the most of the positioning points are sampled with $60-240$ seconds interval, indicating that how well the system performs on these trajectories largely determines the overall performance.

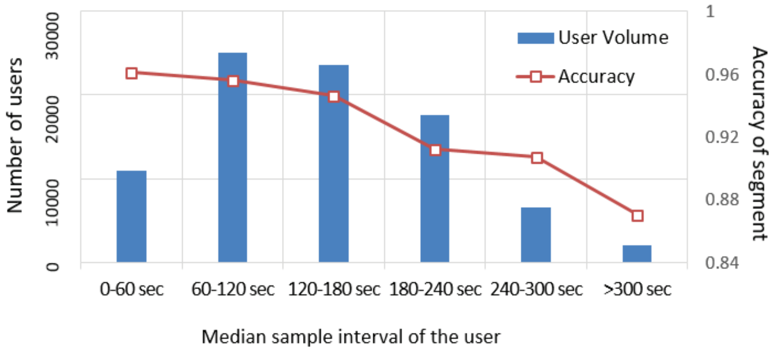


Fig. 28. Distribution of users in term of median sample rate during 24 hours.

Finally, we evaluate how the tracking accuracy varies when handling trajectories with different sampling rate, and the result is shown by the poly-line in Fig. 28. Consistent with our expectation, the group with shortest median sample interval performs best, yielding 96.1% accuracy of segment. Although the performance drops when the sampling interval increases, the accuracies remain at a relatively high level in the range of 60-240 seconds. As mentioned above, this ensures that the system functions well when dealing with most trajectories.

In summary, all the evaluation above shows that *CTS* outperforms its similar systems by at least 30% in both accuracy metrics. When working with dense road networks, our system performs well with the significant help of sector information. And sa long as the median sample interval of the trajectory is smaller than 300 seconds, *CTS* produces tracking accuracy higher than 90%. Thus, we believe this system is most applicable one for cellular trajectory tracking among other related works.

7 CONCLUSION

In this paper, we design and evaluate a cellular-based trajectory tracking system that achieves GPS-level accuracy. This system does not rely on input from mobile devices, and is suitable for tracking low-power sensing devices in wide areas covered by cellular networks. According to real life experiments, our system provides as accurate trajectory tracking as GPS-based solutions in 95.7% of cases. In summary, our work provides an efficient and practical way to track mobile devices using cellular infrastructures. With such accurate trajectories, we will extend this work in the future to enable automatic calibration of cellular positioning points.

REFERENCES

[1] M. Lin and W.-J. Hsu, "Mining gps data for mobility patterns: A survey," *Pervasive and Mobile Computing*, vol. 12, pp. 1–16, 2014.

[2] J. Wang, C. Jiang, L. Gao, S. Yu, Z. Han, and Y. Ren, "Complex network theoretical analysis on information dissemination over vehicular networks," in *IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.

[3] L. Stenneth, O. Wolfson, P. S. Yu, and B. Xu, "Transportation mode detection using mobile phones and gis information," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2011, pp. 54–63.

[4] E. Come, N. A. Randriamanamihaga, L. Oukhellou, and P. Aknin, "Spatio-temporal analysis of dynamic origin-destination data using latent dirichlet allocation: Application to vélib'bike sharing system of paris," in *TRB 93rd Annual meeting*. TRANSPORTATION RESEARCH BOARD, 2014, p. 19p.

[5] P. A. Zandbergen, "Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning," *Transactions in GIS*, vol. 13, no. s1, pp. 5–25, 2009.

[6] F. Alizadeh-Shabdiz, "Methods and systems for determining location using a cellular and wlan positioning system by selecting the best cellular positioning system solution," Apr. 10 2012, uS Patent 8,155,666.

[7] Y. Zhao, "Standardization of mobile phone positioning for 3g systems," *IEEE Communications Magazine*, vol. 40, no. 7, pp. 108–116, 2002.

[8] S. Wakamiya, R. Lee, and K. Sumiya, "Crowd-sourced cartography: Measuring socio-cognitive distance for urban areas based on crowd's movement," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, ser. UbiComp '12. New York, NY, USA: ACM, 2012, pp. 935–942. [Online]. Available: http://doi.acm.org/10.1145/2370216.2370424

[9] Y. Zhao, "Mobile phone location determination and its impact on intelligent transportation systems," *IEEE Transactions on intelligent transportation systems*, vol. 1, no. 1, pp. 55–64, 2000.

[10] R. Mohamed, H. Aly, and M. Youssef, "Accurate real-time map matching for challenging environments," *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[11] ——, "Accurate and efficient map matching for challenging environments," in *ACM Sigspatial International Conference on Advances in Geographic Information Systems*, 2014, pp. 401–404.

[12] C. E. White, D. Bernstein, and A. L. Kornhauser, "Some map matching algorithms for personal navigation assistants," *Transportation research part c: emerging technologies*, vol. 8, no. 1, pp. 91–108, 2000.

[13] W. Chen, M. Yu, Z. Li, and Y. Chen, "Integrated vehicle navigation system for urban applications," 2003.

[14] D. Bernstein and A. Kornhauser, "An introduction to map matching for personal navigation assistants," 1998.

[15] J. S. Greenfeld, "Matching gps observations to locations on a digital map," in *Transportation Research Board 81st Annual Meeting*, 2002.

[16] W. Y. Ochieng, M. A. Quddus, and R. B. Noland, "Map-matching in complex urban road networks," 2003.

[17] P. Newson and J. Krumm, "Hidden markov map matching through noise and sparseness," in *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. ACM, 2009, pp. 336–343.

[18] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, L. Girod *et al.*, "Accurate, low-energy trajectory mapping for mobile devices." in *NSDI*, 2011.

[19] I. Leontiadis, A. Lima, H. Kwak, R. Stanojevic, D. Wetherall, and K. Papagiannaki, “From cells to streets: Estimating mobile paths with cellular-side data,” in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 2014, pp. 121–132.

[20] K. Perera, T. Bhattacharya, L. Kulik, and J. Bailey, “Trajectory inference for mobile devices using connected cell towers,” in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2015, p. 23.

[21] F. Izquierdo, M. Ciurana, F. Barceló, J. Paradells, and E. Zola, “Performance evaluation of a toa-based trilateration method to locate terminals in wlan,” in *Wireless Pervasive Computing, 2006 1st International Symposium on*. IEEE, 2006, pp. 1–6.

[22] C. Y. Goh, J. Dauwels, N. Mitrovic, M. Asif, A. Oran, and P. Jaillet, “Online map-matching based on hidden markov model for real-time traffic sensing applications,” in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE, 2012, pp. 776–781.

[23] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[24] E. Hepsaydir, “Mobile positioning in cdma cellular networks,” in *Vehicular Technology Conference, 1999. VTC 1999-Fall. IEEE VTS 50th*, vol. 2. IEEE, 1999, pp. 795–799.

[25] D. W. Lefever, “Measuring geographic concentration by means of the standard deviational ellipse,” *American Journal of Sociology*, vol. 32, no. 1, pp. 88–94, 1926.

Received May 2017; revised August 2017; accepted October 2017