SensTrack: Energy-Efficient Location Tracking With Smartphone Sensors

Lei Zhang, *Student Member, IEEE*, Jiangchuan Liu, *Senior Member, IEEE*, Hongbo Jiang, *Member, IEEE*, and Yong Guan, *Member, IEEE*

***Abstract*— Nowadays, as smartphones are becoming more and more powerful, applications providing location based services have been increasingly popular. Many, if not all, smartphones are equipped with a powerful sensor set (GPS, WiFi, the acceleration sensor, the orientation sensor, etc.), which makes them capable of accomplishing complicated tasks. Unfortunately, as the core enabler of most location tracking applications on smartphones, GPS incurs an unacceptable energy cost that can cause the complete battery drain within a few hours. Although GPS is often preferred over its alternatives, the coverage areas of GPS are still limited (GPS typically cannot function indoors). To this end, our goal in this paper is to improve the energy-efficiency of traditional location tracking service as well as to expand its coverage areas. In this paper, we introduce SensTrack, a location tracking service that leverages the sensor hints on the smartphone to reduce the usage of GPS. SensTrack selectively executes a GPS sampling using the information from the acceleration and orientation sensors and switches to the alternate location sensing method based on WiFi when users move indoors. A machine learning technique, Gaussian process regression, is then employed to reconstruct the trajectory from the recorded location samples. We implemented a prototype on an Android smartphone that can sample the related sensors during the user's movement and collect the sensor data for further processing on PCs. Evaluation on traces from real users demonstrates that SensTrack can significantly reduce the usage of GPS and still achieve a high tracking accuracy.**

***Index Terms*— Location tracking, smartphone, sensor.**

I. Introduction

UNDERSTANDING human mobility in daily life is a fundamental resource for broad-domain applications, especially for the applications that provide location based services. With the increasing pervasiveness of smartphones over the past few years, many emerging location based applications are adopted by mobile users. Consumer and advertiser expenditure on location based services is expected to approach \$ 10 billion by 2016 [1]. The reason that location based applications become so popular is two-fold. First, location based services rely on the knowledge about the user's geographical location to obtain relevant information on the spot, and thus offer the user a plethora of options to satisfy his/her needs under that particular context. Second, a typical modern mobile device usually has the ability to locate or estimate its current position. The localization technologies used today mainly based on Global Positioning System (GPS), other technologies also obtain assistance from WiFi and GSM, each of which can vary widely in energy consumption and localization accuracy. As it is known to be more accurate, GPS is often preferred on mobile platforms over its alternatives such as GSM/WiFi based positioning systems.

Although smartphones today are capable to accomplish complicated tasks such as localization, we still face problems. The demand of computing and storage capability on mobile devices is rapidly increasing in recent years, whereas the battery manufacturing industry moves forward slowly (battery capacity grows by only 5% annually [2]). In spite of the increase in processing power, feature-set, and sensing capabilities, the smartphones continue to suffer from limited battery life. Unfortunately, it is also well-known that GPS, the core enabler of many location-based applications, is power-hungry. The aggressive usage of GPS can cause the battery to completely drain within a few hours [3], [4]. Location based applications still cannot assume continuous and ubiquitous location access in their design because of the high energy expense for localization. Even within the limited hours of being activated, GPS may not function well all the time, especially when the mobile user is under the shelter of buildings due to the signal loss under indoor environment [5]. When GPS is unavailable, alternate location sensing techniques must be used to obtain the approximated location. The variability in accuracy provided by various location sensing technologies and the limits on their coverage areas pose additional challenges for application developers [6]. Using multiple location sensors simultaneously to make up for this variability in accuracy would further increase energy cost.

In this paper, we present the design of SensTrack, a location tracking service that provides user's moving trajectory while reducing its impact on the devices's battery life. By applying different localization technologies, we expand the coverage area compared to the traditional approach that only uses GPS. In addition, the sensor hints from the smartphone itself can help us make decisions about adaptive sampling. SensTrack

Manuscript received January 28, 2013; revised June 9, 2013; accepted July 1, 2013. This work was supported in part by a Canada NSERC Discovery Grant, an Engage Grant, a MITACS Intership Grant, and a China NSF international collaboration under Grant 61120106008. The associate editor coordinating the review of this paper and approving it for publication was Dr. Mahmoud Daneshmand.

L. Zhang and J. Liu are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: lza70@sfu.ca; jcliu@cs.sfu.ca).

H. Jiang is with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: hongbojiang@hust.edu.cn).

Y. Guan is with the College of Information Engineering, Capital Normal University, Beijing 10048, China (e-mail: guanyong@cnu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSEN.2013.2274074

smartly selects the location sensing methods between WiFi and GPS, and reduces the sampling rate by utilizing the information from acceleration sensor and orientation sensor, two of the most common sensors found on smartphones today. We have implemented a prototype on the Google Nexus S phone, which continuously collects data from the acceleration sensor and the orientation sensor, and records the location samples from GPS and WiFi. Experiments have been conducted on a real world path while the phone was carried by a mobile user in a region of our university campus. The collected data is further analyzed and filtered on computers. To predict the user's original trajectory, a track reconstruction algorithm based on a machine learning technique is also implemented on the server side. Performance evaluation on the real data sets shows that SensTrack only needs 7% GPS samples of the naive approach and saves nearly 90% GPS activated time. Meanwhile, SensTrack reconstructs the user's trajectory with high accuracy and better coverage.

The main contributions of this paper are listed as follows:

- We identify the problems of traditional location tracking service including limited availability of GPS and unnecessary GPS samplings. The opportunities of energy-efficiency improvements by utilizing the assistance from sensors on smartphones are discussed.
- We present the detailed design of an energy-efficient location tracking service, SensTrack. As the main component, a track reconstruction algorithm based on Gaussian Process Regression is proposed. Other mechanisms for making smart adaptive sampling decisions are also discussed.
- We implement a prototype of SensTrack, and evaluate the proposed system through real-world experiments.

This paper is organized as follows. In Section II we review the related work on energy-efficient location sensing. Section III presents our observations on the defects of tradition location based applications based on GPS, and discusses the opportunities of improvements. The detailed design of SensTrack is proposed in Section IV. We evaluate our proposal in Section V and analyze the performance improvement. Further considerations are discussed in Section VI. Section VII concludes the paper and outlines the future work.

II. Related Work

To track the users' locations, many energy-efficient sensing approaches with adaptive sensing policies have been proposed to minimize the energy consumption [3], [7]–[9]. With the objective of minimizing the location error for a given energy budget, EnLoc [3], an energy-efficient localization framework, includes a heuristic with a local mobility tree to predict the next sensing time by utilizing the dynamic programming technique. Jigsaw [8] uses the information obtained from the acceleration sensor and the microphone to continuously monitor human activities and environmental context. According to the user's mobility patterns, a discrete-time Markov Decision Process is employed to learn the optimal GPS duty cycle schedule with a given energy budget.

There are also works based on the observation that the required localization accuracy varies with locations. An adaptive location service for mobile devices, a-Loc [7] uses a Bayesian estimation framework to determine the dynamic accuracy requirement, and tunes the energy expenditure accordingly. It argued in [9] that given the less accuracy of GPS in urban areas, it suffices to turn on GPS adaptively to achieve this accuracy. The rate-adaptive positioning system for smartphone applications (RAPS) was then proposed to minimize energy consumption with given accuracy threshold by using the information of moving distance, space-time history, and cell tower-based blacklisting.

Smartphones' energy consumption has been a major concern in research for a long time, and a number of studies have been done to improve the energy efficiency of mobile devices. In order to understand where and how the energy is used, A. Carroll *et al.* [10] measured the power consumption of a modern mobile device (the Openmoko Neo Freerunner mobile phone), broken down to the devices major subsystems (CPU, memory, touchscreen, graphics hardware, audio, storage, and various networking interfaces), under a wide range of realistic usage scenarios. M. Ra *et al.* [11] proposed the Stable and adaptive link selection algorithm (SALSA), an optimal online algorithm for energy-delay tradeoff based on the Lyapunov optimization framework. SALSA defers the transmissions of delay-tolerant applications until a less energy-consuming WiFi connection becomes available.

Utilizing the sensing power of smartphones is not a new topic in literature. M. Keally *et al.* [12] presented the design of Practical Body Networking (PBN) system to provide practical activity recognition with mobile devices, which combines the sensing power of on-body wireless sensors with the additional sensing power, computational resources, and user-friendly interface of an Android smartphone through the unification of TinyOS motes and Android smartphones. Another interesting ongoing work discusses how to fuse information from Microsoft Kinect's tracking with the smartphone's sensor readings to improve Kinect gaming experience [13].

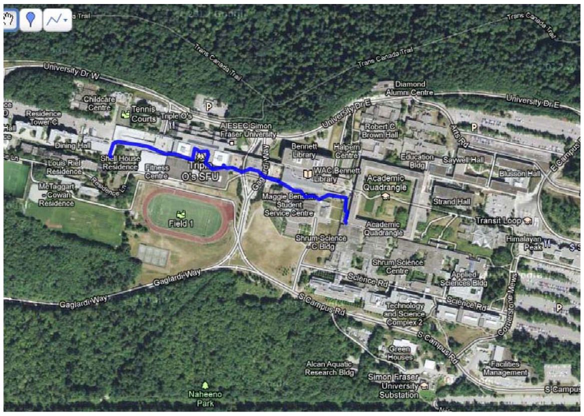
Inspired by many existing studies, in this paper we take efforts to achieve a high energy efficiency by reducing the sampling rate of sensing users' locations. However, our work uses a novel approach by utilizing the acceleration sensors and the orientation sensors on smartphones to capture the geometric features of users' moving trajectories. We will further explain the difference between SensTrack and existing works in the following sections.

III. Challenges and Opportunities

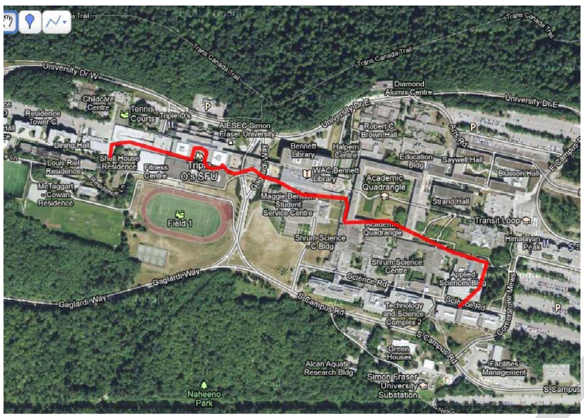
In this section, we start by describing the defects of typical location-based applications that utilize GPS, including limited availability and unnecessary samples. We then discuss the opportunities for making improvements.

A. Limited Availability of GPS Versus Multiple Location Sensing Methods

It should be noted that traditional GPS cannot work properly under the indoor environment. The standard GPS



(a)



(b)

Fig. 1. Tracking results when $T = 5$ s, $\theta = 45°$, $D = 100$ m, $\upsilon = 8$ m/s. (a) Track recorded by the naive approach. (b) Track reconstructed by SensTrack.

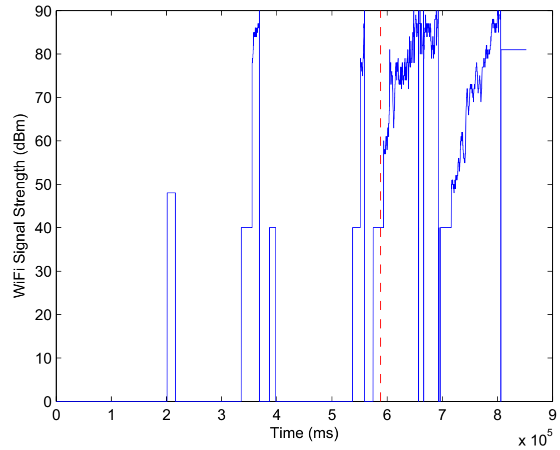


Fig. 2. WiFi signal strength along the track.

receiver requires signals from at least 4 satellites simultaneously to calculate and output 3-dimensional locations and velocity information [5]. Therefore, the mobile devices need to be in line-of-sight contact with the GPS satellites, which significantly limits the usage of typical location based applications.

Figure 1(a) shows one track that we took using GPS on a mobile device. Although we did not stop recording, the track ends once it entered the building (the Academic Quadrangle in our campus), which indicates the performance of GPS largely depends on the working condition. The signals from GPS satellites can be blocked not only by buildings but also by canyon walls, trees, and even thick clouds. When the user walks through buildings, GPS equipped by a normal smartphone cannot function since the lack of satellite signals. Even worse, GPS units may consume more energy than the normal situation when there is no satellite signals [14].

Besides GPS, there also exist alternate location sensing technologies. For example, Android OS provides a network-based localization mechanism, which exploits GSM footprints from cell towers and WiFi signals to obtain an approximate location. Although the network-based location sensing is not as accurate as GPS, it provides the possibility to keep tracking inside a building since it mainly relies on the WiFi connection, in which case GPS units can be deactivated to save battery. For the scenarios like university campus, hotels or hospitals, we can always assume persistent wireless local network access, which implies that other location sensing methods may provide us valid options when GPS is out of use.

Figure 2 shows the received WiFi signal strength along the track presented in Figure 1(a). The dash line indicates the time stamp (588 s) at which the user entered the Academic Quadrangle. There are some spikes before 588 s (201 s$\sim$216 s, 335 s$\sim$368 s, 387 s$\sim$398 s, 537 s$\sim$558 s), which means that the user can receive some WiFi signal for a short time when passing by buildings. After entering the building at 588 s, the received WiFi signal stayed at a relatively high level since the WiFi connection is assured in teaching areas of the university campus. This figure can support our argument that, when the user is inside a building, WiFi signal is usually relatively strong. Therefore, the network-based localization can be a valid choice under the indoor environment where GPS is no longer available. The idea is to use the GPS satellite signal and the wireless network connection as indicators for switching between GPS and the network-based location sensing method.

B. Unnecessary GPS Samplings Versus Adaptive Sampling

The GPS sensor can sample the user's location at a relatively high rate. However, it is not ideal to record every location update since the error for each location sample varies. To make the path more smooth and fit the real trajectory, a typical location based application usually updates the user's location only if the distance to the last valid location sample is larger than a certain threshold [15]. Therefore, with a fixed and frequent GPS location sampling policy, it probably introduces a significant amount of unnecessary GPS samples.

To demonstrate this, we collect the system log of an Android application, *My Tracks* [16], which uses the GPS sensor in mobile devices to record the paths that users take while hiking, cycling, running, or participating in other activities. Figure 3 shows part of the system log, demonstrating its executing history in one run. As shown in the figure, the application usually takes several GPS samples to get one valid location update, in which case the threshold is 5 meters. Our experimental result in this case shows that up to 79%

```
04-16 01:25:35.535 D/MyTracks(18469): Not recording. Distance to last
recorded point (3.257859 m) is less than 5 m.
04-16 01:25:36.566 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:36.574 D/MyTracks(18469): Not recording. Distance to last
recorded point (4.096747 m) is less than 5 m.
04-16 01:25:37.519 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:37.527 D/MyTracks(18469): Not recording. Distance to last
recorded point (4.636137 m) is less than 5 m.
04-16 01:25:38.515 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:38.527 D/MyTracksLib(18469):
MyTracksProviderUtilsImpl.insertTrackPoint
04-16 01:25:38.527 D/MyTracksProvider(18469): MyTracksProvider.insert
```

Fig. 3. A part of system log when running a location-based application.

location samples of *My Tracks* are unnecessary. Since many of the samples are discarded, these invalid location measurements cause unnecessary energy consumption.

C. Assistance From Other Sensors

Nowadays smartphones become more and more powerful in terms of hardware, which usually contains various sensors. As an example, iPhone 4 is equipped with several environmental sensors, including an ambient light sensor, a magnetic compass, a proximity sensor, an accelerometer, and a three-axis gyroscope [17]. Android 4.0 (API Level 14) also supports up to 13 kinds of sensors [18], even though the sensors' availability varies from device to device. The supported list of sensors in a Google Nexus S phone consists of: one KR3DM 3-axis Accelerometer, one AK8973 3-axis Magnetic field sensor, one AK8973 Orientation sensor, one GP2A Proximity sensor, one GP2A Light sensor, one Linear Acceleration Sensor, one Rotation Vector Sensor, one K3G Gyroscope sensor, and one Gravity Sensor [19].

To reduce unnecessary GPS samples, adaptive sampling is proposed in many existing works [3], [7]–[9]. Usually we need additional information to make adaptive sampling decisions, which may include the location history, the speed history, the distance information, remaining battery power, the accuracy requirement, etc. In this paper, we utilize the powerful sensors equipped by smartphones to obtain the information about changes of the orientation, moving speed, and traveled distance. Based on these useful information, we are able to make smart adaptive sampling decisions. The detailed design is described in the following section.

IV. SensTrack: Design Details

A. Overview

To reduce the frequency of location sensing, SensTrack periodically collects data from the corresponding sensor to detect a turning point or estimate current speed and the distance from the last recorded location. The high energy efficiency of this approach is supported by the fact that the GPS sensor consumes much more energy than the acceleration sensor and the orientation sensor [9], [20]. When the GPS satellite signal is not available and the WiFi connection is active, SensTrack switches to the network-based location sensing method to obtain the raw coordinates. The last step of SensTrack is to upload the coordinates of sampled locations to an online server that uses a machine learning algorithm to reconstruct a smooth and accurate trajectory.

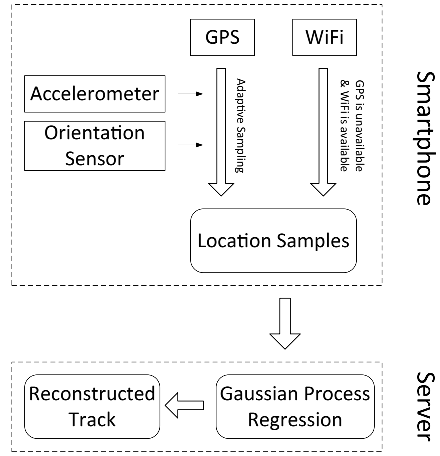


Fig. 4. The system architecture.

Figure 4 demonstrates the SensTrack's system architecture. The service consists of two stages: the first is to collect the location samples; and the second is to reconstruct the original trajectory. Given the working conditions, SensTrack switches between the GPS-based and the network-based localization methods using the GPS or WiFi sensors, respectively. By utilizing the sensor hints from the acceleration sensor and the orientation sensor, SensTrack is able to make smart adaptive sampling decisions in the GPS mode. For example, when the smartphone detects a turning point or if it estimates a unreasonable speed or a unexpected large traveling distance, it uses GPS to record the current location. After the server side receives all the collected location samples, a Gaussian Process Regression algorithm is then employed to predict the trajectory that the user has taken.

B. Track Reconstruction: Gaussian Process Regression

Once the collection of location samples is finished, it is not ideal to simply connect all the recorded locations, since the distances between any two successive locations may not be the same. For some parts of a trajectory, the recorded locations can be very sparse, while for other parts, the location samples may be relatively intensive. If we simply connect the location samples, the resultant trajectory can be very abstract. Therefore, uploading the collected data to the online server either by a wireless or wired connection to reconstruct the trajectory is our last stage. We adopt the Gaussian Process Regression (GPR), a machine learning technique to perform the interpolation. The training set of the algorithm is the recorded critical locations decided by the sensor hints which capture most of key features of a trajectory. And the testing set is the predicted locations between the successive but far-away location samples. Combing both input and output gives us the final trajectory. We next detailed describe GPR and how the user's trajectory can be reconstructed by using GPR.

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution, and is fully specified by a mean function and a covariance function [21]. The inference of continuous values with a Gaussian process prior is known as Gaussian Process

Regression. Consider x as a general random variable. We define the mean function $m(x)$ and the covariance function $k(x, x')$ of a real process $f(x)$ as

$$m(x) = E[f(x)],$$
$$k(x, x') = E[(f(x) - m(x))(f(x') - m(x'))],$$

and can write the Gaussian process as

$$f(x) \sim gp(m(x), k(x, x')).$$

For notational simplicity the mean function is usually set to be zero. In our method the covariance function will be the squared exponential covariance function, although other covariance functions may also be useful. Assuming that observations are noise-free, the covariance function specifies the covariance between pairs of random variables

$$cov(f(x_p), f(x_q)) = k(x_p, x_q) = exp(-\frac{1}{2}|x_p - x_q|^2). \quad (1)$$

For a estimate data set X_*, we can generate a random Gaussian vector f_* for target values with the covariance matrix calculated from Equation 1

$$f_* \sim N(0, K(X_*, X_*)).$$

Therefore, the joint distribution of the training outputs f and the test outputs f_* according to the prior is

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(K_*, X) & K(X_*, X_*) \end{bmatrix}\right). \quad (2)$$

If X contains n training points and X_* contains n_* test points, then $K(X, X_*)$ is the $n \times n_*$ matrix of the covariances evaluated at all pairs of training and test points. And the other entries $K(X, X)$,$K(X_*, X)$,and $K(X_*, X_*)$ are similar.

If observations are noisy, we can write $y = f(x)+\varepsilon$. Assuming additive independent identically distributed Gaussian noise ε with variance σ^2, we have the prior as

$$cov(y_p, y_q) = k(x_p, x_q) + \sigma_n^2 \delta_{pq}$$

or

$$cov(\mathrm{y}) = K(X, X) + \sigma_n^2 I,$$

where δ_{pq} is a Kronecker delta which is one when $p = q$ and zero otherwise. Introducing the noise in Equation 2, the joint distribution of the observed target values and the function values at test points according to the prior will be

$$\begin{bmatrix} \mathrm{y} \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(K_*, X) & K(X_*, X_*) \end{bmatrix}\right). \quad (3)$$

The posterior distribution over functions can be obtained by restricting the joint prior distribution on the observations. Then we arrive at the key predictive equations for GPR

$$f_*|X, \mathrm{y}, X_* \sim N(\overline{f_*}, cov(f_*)), \text{ where} \quad (4)$$
$$\overline{f_*} = E[f_*|X, \mathrm{y}, X_*] = K(X_*, X) \times [K(X, X) + \sigma_n^2 I]^{-1}\mathrm{y}, \quad (5)$$
$$cov(f_*) = K(X_*, X_*) - K(X_*, X) \times [K(X, X) + \sigma_n^2 I]^{-1} K(X, X_*). \quad (6)$$

Algorithm 1 Predictions(X,y,k,σ_n^2,x_*)

1: $L = \text{cholesky}(K + \sigma_n^2 I)$
2: $\alpha = L^\top \backslash (L \backslash \mathrm{y})$
3: $\overline{f_*} = k_*^\top \alpha$
4: $\upsilon = L \backslash k_*$
5: $V[f_*] = k(x_*, x_*) - \upsilon^\top \upsilon$
6: $\log p(\mathrm{y}|X) = -\frac{1}{2} y^\top \alpha - \sum_i \log L_{ii} - \frac{n}{2} \log 2\pi$
7: return $(\overline{f_*}, V[f_*], \log p(\mathrm{y}|X))$

We then focus on explaining how to use GPR with given location samples to reconstructed the estimated trajectory. A trajectory can be considered as the path that the user follows through space as a function of time. Specifically, we have n location samples from x_1 to x_n, each of which can be represented by a two-dimensional points $\mathrm{x}_i = (x_i, y_i)$. Then X is the sampled date set for all (x_i, y_i) s. According to what we have explained, the user's track can be represented by generated GPR functions which is determined by a covariance function and a mean function. In the case that there is only one test point x_*, we let $k(\mathrm{x}_*) = k_*$ denote the vector of covariances between the test point and the n training points. Then for a single test pointx_*, Equation 5 and 6 can be reduced to

$$\overline{f_*} = k_*^\top (K + \sigma_n^2 I)^{-1} \mathrm{y}, \quad (7)$$
$$V(f_*) = k(X_*, X_*) - k_*^\top (K + \sigma_n^2 I)^{-1} k_*. \quad (8)$$

On obtaining Equation 7 and 8, we further propose the following Algorithm 1 for a single test case, in which cholesky $(K + \sigma_n^2 I)$ is the Cholesky decomposition on the matrix of $K + \sigma_n^2 I$. The implementation addresses the matrix inversion required by Equation 7 and 8 using Cholesky factorization. For multiple test cases lines $3 \sim 6$ are repeated. In our case, X is time space of the training set, y is the set of observed target values (location samples), k is the covariance function, $\sigma_n^2 I$ is the noise, and x_* is the testing data. The outputs are as follows. $\overline{f_*}$ is the mean predicted value (predicted location of x_*), $V[f_*]$ is its variance, and $\log p(\mathrm{y}|X)$ is the marginal likelihood. A more detailed explanation can be referred to our previous work [22].

C. Switching Location Sensing Methods

As mentioned, it is well-known that GPS cannot function properly indoors. To expand the coverage areas, SensTrack switches between GPS and the network-based localization through the wireless connection. Basically, we want to use GPS outdoors and the network-based localization indoors, and thus it is important to decide when to switch. Initially, SensTrack starts in the GPS mode and periodically executes a WiFi scan. When it detects the GPS signal loss as well as an active wireless network connection, SensTrack turns into the WiFi mode. If GPS becomes available again, and the phone loses the WiFi connection or the accuracy of location samples provided by the network decreases significantly, SensTrack switches back into the GPS mode.

We note that there are two conditions satisfied to switch the location sensing method: the current method fails to obtain location samples, and the other method is guaranteed to work, which prevents from switching between the two modes too often. Frequently changing location sensing mechanism can be very energy consuming, because the high-power components associated with both location providers need to be active. In some cases, both of the two methods are available when the user passing by some buildings. According to our rules, we should not change SensTrack's working mode, since in these situations the wireless connection tends to be unstable and short. In other cases, none of the two methods are available if we simply lose the GPS satellite signal outdoors. Our rules can also avoid the unnecessary switching in these cases. It is also worth mentioning that SensTrack stops collecting the sensor hints when it switches into the WiFi mode. In another word, we passively receive location updates in this mode. The reason is that, unlike GPS, when we request the location information, the WiFi localization technology cannot respond within a tolerable delay. It means that even if we apply the sensor hints to sense the location adaptively, we cannot obtain a location sample timely in the WiFi mode. Therefore, considering the WiFi localization updates the location less frequently than GPS, we decided not to waste energy on the acceleration sensor and the orientation sensor.

D. Utilizing Sensor Hints

1) Orientation: SensTrack employs the orientation sensor as a detector of turning points when the user is moving. The idea is that there is no need to record the user's location if he/she is in a steady movement without changing direction. For a sliding window of size T, SensTrack collects the readings of the orientation sensor, and computes the changes in direction. If user's moving direction changes dramatically (greater than the threshold θ), a location sensing of the user's current location is executed. Considering the readings from the orientation sensor is approximately continuous, the window size T should be larger enough to observe the potential direction changes. Table I shows the effect of the window size T. In our experiments, T was set to be 5 s because it would lose some turns of the trajectory for smaller window size. On the other hand, a larger window size is not necessary as it requires more memory and computation, which in turn requires more powerful hardware. The user can also decide the threshold θ, the other key parameter, according to their expectations on accuracy. Table II presents the number of missing turning points for different values of θ. Roughly speaking, SensTrack is more sensitive with a smaller θ. However, a too small θ may cause redundant detections of the trajectory's turns (false positives) if we consider the noises in the readings from the sensor, which potentially wastes energy in sensing locations at those false turning points.

2) Acceleration: The acceleration sensor in a mobile device has been widely used in many existing location sensing systems, in which it acts as a binary sensor to detect user movement or non-movement. We notice that distance is theoretically a simple integral of speed, which in turn is an integral of acceleration. Unlike most prior works, we do not limit the acceleration sensor just to be the user's movement detector,

TABLE I
EFFECT OF WINDOW SIZE T

$T =$	$1s$	$3s$	$5s$	$7s$
key turning points	4 misses	1 miss	0 miss	0 miss

TABLE II
EFFECT OF THRESHOLD θ

$\theta =$	$45°$	$60°$	$75°$	$90°$
key turning points	0 miss	1 miss	3 misses	4 misses

rather explore the possibility of calculating the distance that the user has traveled and the speed that the user is moving at.

It should be noted that the readings of the acceleration sensor on a moving device are usually noisy, especially when the user is walking. Activities with higher speed, like biking and driving, actually are more stable, whereas the movement of a pedestrian is always fluctuating. It often overestimates distance when the user is holding the phone in his/her hands, and underestimates distance when sitting quietly on a cushioned car seat [9]. When calculating the integrals, errors caused by the noise in the sensing data are accumulated. However, we argue that the estimated distance and speed obtained as integrals of acceleration are still useful even if they are inaccurate, because the location and velocity information provided by GPS can help us to calibrate the calculation. Once the estimated distance or the estimated speed exceeds the thresholds, specifically D and v, SensTrack activates GPS to sense the current location and speed. The thresholds can be set based on the accuracy requirement or the user's moving patterns. For example, for a pedestrian, usually the moving speed can be no more than 10 m/s and should not be negative, and the accuracy requirement is usually higher. Moreover, the calibration of calculating the integrals can also be done when GPS is activated at the turning points.

V. EVALUATION

A. Data Collection and Methodology

We evaluated SensTrack using a real data set collected from a Google Nexus S phone carried by a mobile user walking in our university campus. The phone is equipped with an integrated GPS, an WiFi sensor, an accelerometer, and an orientation sensor. We implemented a SensTrack prototype on Android 4.0 (API level 14). During its runtime, the prototype continuously collects data from the acceleration sensor and the orientation sensor at default rate of the system service (SENSOR_DELAY_NORMAL) in Android OS. When the GPS signal is available, a location listener is registered to request location updates from GPS periodically. Meanwhile, the prototype always tries to initiate and maintain a WiFi connection, which can be used to record the location updates from the network-based location provider. In our experiments, a PC server was used to further analyze the data collected by the smartphone and filter the GPS and WiFi location samples with the given parameters. The trajectory reconstruction algorithm based on GRP was also implemented on the server side, which uses the filtered and valid location samples to predicted

TABLE III

AVERAGE ERROR OF PREDICTED LOCATIONS

	recorded locations	predicted locations	average error
SensTrack	38 samples	24 predictions	3.128m
GPS trace	568 samples	0	0

the original trajectory. For most of the presented results, our settings were $T = 5$ s, $\theta = 45°$, $D = 100$ m, $\upsilon = 8$ m/s, and a prediction was made if the time gap between two successive GPS samples is greater than 15 s.

We also compared SensTrack with the naive approach, in which GPS is the only way to obtain location information and the GPS sensor is kept to be activated during the whole tracking period. Unlike SensTrack, which samples the GPS location actively, the naive approach is a passive method that records all the valid location updates from GPS. We conducted the experiments on the same real path for several times, which started from outdoor environment, came into a building, and then ended indoors. The total length of the path is around 1.1 km. The results show that, without significantly losing the accuracy of tracking, SensTrack effectively reduce the number of GPS samples and the time that the GPS sensor needs to be turned on.

B. Accuracy

We first present the tracking results by SensTrack and the naive approach. Despite the tracking service maintained, the trajectory shown in Figure 1(a) ended once the user entered the building since the signals from GPS satellites were blocked by the building, which indicates the performance of GPS largely depends on the working condition. Compared to the naive approach, SensTrack demonstrates a reasonably better performance. Figure 1(b) shows that the trajectory reconstructed by SensTrack has a similar outdoor part, meanwhile it has the indoor part that the original one does not have. Although the indoor part of the second trajectory may be not that accurate given the limitation of WiFi localization technology, it is still good to have a approximate trajectory.

As previously stated, the resulting trajectory generated by SensTrack consists of two kinds of points: the sampled locations and the predicted locations. To evaluate the accuracy of SensTrack, we took the GPS trace as the ground truth and calculated the average error of the predicted locations. For every prediction, we computed the difference between the predicted location and the real location in the GPS trace at the same time. The result shown in Table III proves that SensTrack can achieve a high accuracy. The average error of the predictions is 3.128 meters, which is quite acceptable (GPS can achieve an accuracy of 5 meters in good signal conditions). It should be noted that even the GPS trace may not be the real path that the user has taken, because the performance of GPS depends on a number of factors such as the user's position, time, surroundings, weather, etc, which means that the GPS trace itself can be inaccurate. Another result from Table III is that the naive approach recorded 568 samples over the testing path, although some of them may be unnecessary as discussed earlier. It is worth mentioning that, whether a sample is necessary should be decided case by case. For different scenarios, the ideal minimal distance (threshold) between two valid samples can vary significantly. We can adjust the number of necessary samples by setting the granularity between successive samples and filtering the recorded samples accordingly. In our experiments, the number of necessary samples does not affect the total number of GPS samples as the naive approach passively received every sample, and the granularity between successive samples cannot reflect the error of reconstructed trajectory.

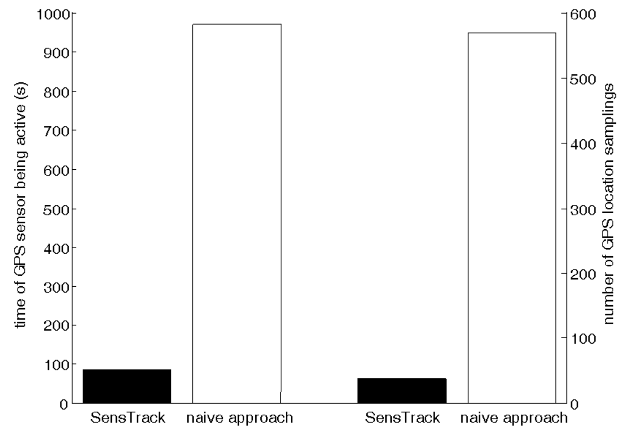


Fig. 5. Comparison of the energy efficiency.

C. Energy Efficiency

In modern mobile devices, the GPS receiver usually consume much more power than the accelerometer and the digital compass. For example, our testing device, a Google Nexus S phone, is equipped with a BCM4751 integrated GPS receiver (produced by Broadcom), a KR3DM 3-axis accelerometer (produced by STMicroelectronics), and an AK8973 3-axis electronic compass (produced by Asahi Kasei Microdevices). With the battery supply (3.7 volt), the power consumption (in terms of current) of the accelerometer is 0.23 mA; and the current consumption of the compass is 6.8 mA; however, the current consumption of the GPS receiver can be as much as 80 mA. To demonstrate the energy efficiency of SensTrack, we present that SensTrack can significantly reduce the number of needed GPS samples and the time that the GPS sensor needs to be activated. We did not measure the actual energy consumption of SensTrack, since we thought it is unnecessary. For different hardware, the power consumption varies, and thus the energy consumption of SensTrack on a specific hardware model only provides limited information. Therefore, it is convincing and sufficient for us to show the relative energy efficiency of SensTrack to the naive approach by comparing the number of required sampling and the activated time of the GPS receiver.

Figure 5 shows that compared to the naive approach, SensTrack only needs 7% GPS samples for the described path, and the time of the GPS sensor being active is decreased by nearly 90%. The naive approach almost updated the user's location every second, and the GPS sensor was kept to be activated even when the user entered the building and lost the

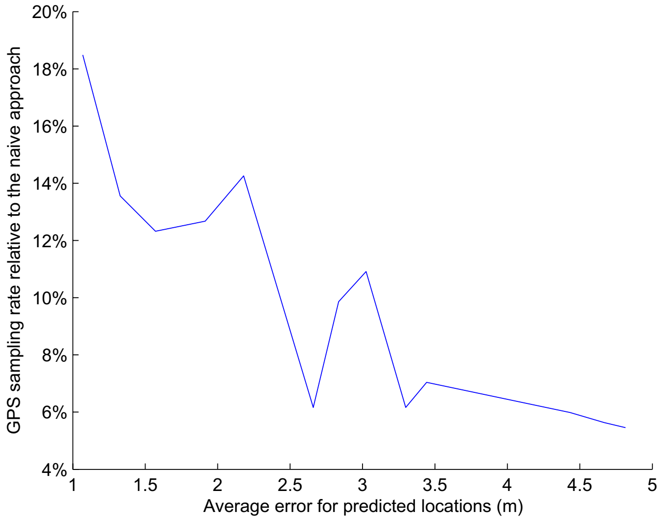


Fig. 6. Tradeoff between sampling rate and accuracy.

GPS satellite signals. SensTrack on the contrary only selectively activated the GPS sensor at some separate locations, and turned the GPS sensor off once the device lost the satellite signals and had an active WiFi connection. It should be pointed out that the energy efficiency of SensTrack depends on the user's movements and the path that the user takes. If the user's movement is very unstable and the direction changes frequently, SensTrack inevitably activates the GPS sensor more frequently, and thus consumes more energy.

D. Energy-Accuracy Tradeoff

By intelligently managing the energy and localization accuracy trade-off, the battery life of a mobile device can be significantly extended, which is of great importance for the smartphone users. Since the required localization accuracy varies with locations, there is significant potential to trade-off the accuracy and the energy consumption based on the application's needs and different working scenarios.

As mentioned before, we take the GPS sampling rate as a representative of SensTrack's power consumption. Figure 6 demonstrates the trade-off between sampling rate and accuracy, which SensTrack presents under different configurations. Even though there exists some bias, we can observe a clear trend that a higher accuracy requires a higher GPS sampling rate, which means more power consumption. On the other hand, Figure 6 does not present a strict monotonicity. A higher energy consumption does not necessarily indicate a higher accuracy. For example, it only requires 6% samples to achieve a higher accuracy (average error is 2.66 m), whereas 11% samples are needed to produce a relatively lower accuracy (average error is 3.02 m). This is because the error of one prediction not only depends on the GPS sampling rate but also depends on the performance of the reconstruction algorithm. For GPR in our case, if the location samples have higher covariances between each other and are uniformly distributed on the path in time space, the algorithm can produce better results and achieve a higher accuracy. Therefore, besides the sampling rate, the actual samples themselves collected by

TABLE IV
AVERAGE WiFi TRAFFIC

	received	transmitted
Baseline	0.38 packet/s	0.88 packet/s
SensTrack	0.94 packet/s	0.81 packet/s

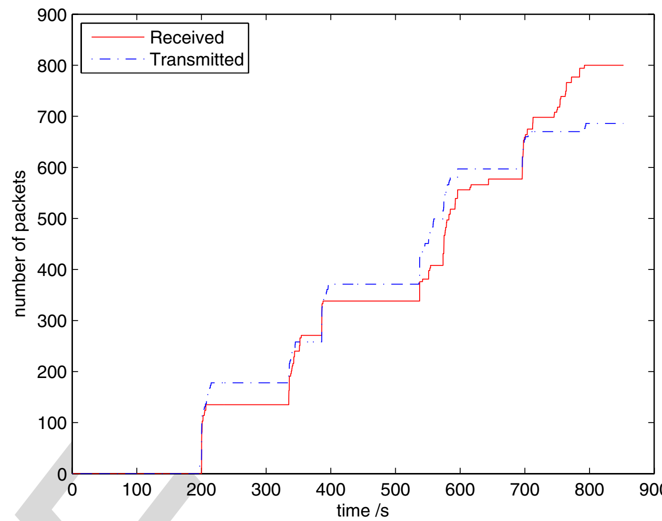


Fig. 7. WiFi traffic of SensTrack.

the system have a huge impact on the results. The samples that have similar covariances between every two successive samples are more likely to produce highly accurate predictions.

E. Transmission Overhead

There is no doubt that exploiting network-based localization technology to obtain approximate locations would incur some extra network transmissions. To measure the extra traffic, we recorded the traffic loads of SensTrack and the baseline. As the baseline, there only maintains a valid wireless network connection. To be clear, we did not include the uploading of location samples into the transmission overhead, because unlike the indoor location sensing, the uploading process does not need to be done in real time.

Table IV presents the average numbers of the received and transmitted packets during the tracking process. For both SensTrack and the baseline, the average numbers of the transmitted packets were close. Although SensTrack theoretically should transmit more packets as it requests location information through the wireless link, the result is within a normal error range. On the other hand, SensTrack received more than twice as many packets as the baseline did. We argue that even if the number of received packets increases, the total transmission overhead may not be intolerable, because the size of received packets that contains only the location information should be small. Moreover, since the WiFi connection is usually free, there is no need to worry about the wireless network traffic. Another point is that communicating with the access points consumes less energy than communicating with the GPS satellites. Figure 7 further shows SensTrack's traffic pattern, which matches the result in Figure 2. SensTrack had WiFi traffic in the time intervals of strong WiFi signals

(201 s ∼ 216 s, 335 s ∼ 368 s, 387 s ∼ 398 s, 537 s ∼ 558 s). After entering the building at 588 s, SensTrack continuously transmitted and received packets.

VI. Further Discussion

A. Multiple Mobility Patterns

Although our work focuses on the pedestrians, it can be easily extended on multiple mobility patterns, such as running, biking, driving, etc, which are often with higher speeds. Intuitively these movements are more stable, and thus the trajectories are likely less complex, and thus the sensors on smartphones can easily capture the features of the path. Therefore, our approach at least paves the road of designing the efficient tracking service for multiple mobility patterns. However, given the characteristics of different movements, modifications should be carefully considered.

B. Energy Consumption of Accelerometer and Orientation Sensor

In this paper, to make our point clear, we assume a continuous sampling of the acceleration sensor and the orientation sensor, which may cause unnecessary energy cost. It is not necessarily the case. Given that the energy-efficiency is a major goal of our design, users can further employ a low duty cycle on the usage of the acceleration sensor and the orientation sensor. Since the high speed movements are more stable, a low duty cycle can still allow the sensors to capture the features of the users' movements.

C. Other Indoor Localization Technologies

Our work chose the network-based method, which is mainly based on the WiFi positioning system, as our indoor localization approach. The primary reason is that the implementation of this method is already provided as APIs in Android platforms (since API level 1). Other methods for the indoor localization can also be employed such as the specialized real-time locating systems (RTLS) [23] or the inertial measurement unit (IMU)-based navigation systems [24]. However, many of these methods also require a costly infrastructure or additional hardware, which hardly satisfy the need for a cost-effective solution. On the other hand, indoor localization is not our main concern in this paper, rather it is a supplementary of GPS to extended the coverage of SensTrack.

VII. Conclusion

In this paper, we have proposed a novel location tracking service, SensTrack. We first discussed the limitations of the traditional GPS-based approach and opportunities of improvements. Next, the detailed design of SensTrack was presented including: the trajectory reconstruction algorithm based on the Gaussian Process Regression, the rules of switching between two location sensing methods, and the principles for exploiting the sensor hints. We then used the real traces to evaluate the performance of SensTrack, which shows that SensTrack can significantly reduce the usage of GPS and generate accurate tracking results. The design of SensTrack and evaluation presented above reveal several interesting challenges which remain for future work including resilient accelerometer data processing, tracking for multiple mobility patterns, and joint optimization of energy and accuracy.

References

[1] N. Patel, "The $10 b rule: Location, location, location," Strategy Anal., Newton, MA, USA, Tech. Rep., May 2011. AQ:1

[2] S. Robinson, "Cellphone energy gap: Desperately seeking solutions," Strategy Anal., Newton, MA, USA, Tech. Rep., Mar. 2009.

[3] I. Constandache, S. Gaonkar, M. Sayler, R. Choudhury, and L. Cox, "EnLoc: Energy-efficient localization for mobile phones," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2716–2720.

[4] Z. Zhuang, K. Kim, and J. Singh, "Improving energy efficiency of location sensing on smartphones," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 315–330.

[5] E. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications*. Norwood, MA, USA: Artech House, 2006.

[6] *Android Developers Reference: Location Strategies* [Online]. Available: http://developer.android.com/guide/topics/location/strategies.html AQ:2

[7] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 285–298.

[8] H. Lu, J. Yang, Z. Liu, N. Lane, T. Choudhury, and A. Campbell, "The Jigsaw continuous sensing engine for mobile phone applications," in *Proc. 8th ACM Conf. Embedded Netw. Sensor Syst.*, 2010, pp. 71–84.

[9] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive GPS-based positioning for smartphones," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 299–314.

[10] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in *Proc. USENIX Conf. Annu. Tech. Conf.*, 2010, p. 21.

[11] M. Ra, J. Paek, A. Sharma, R. Govindan, M. Krieger, and M. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 255–270.

[12] M. Keally, G. Zhou, G. Xing, J. Wu, and A. Pyles, "PBN: Towards practical activity recognition using smartphone-based body sensor networks," in *Proc. 9th ACM Conf. Embedded Netw. Sensor Syst.*, 2011, pp. 246–259.

[13] R. Meng, J. Isenhower, C. Qin, and S. Nelakuditi, "Can smartphone sensors enhance kinect experience?" in *Proc. 13th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2012, pp. 265–266.

[14] Y. Chon, E. Talipov, H. Shin, and H. Cha, "Mobility prediction-based smartphone energy optimization for everyday location monitoring," in *Proc. 9th ACM Conf. Embedded Netw. Sensor Syst.*, 2011, pp. 82–95.

[15] I. Akyildiz, J. Ho, and Y. Lin, "Movement-based location update and selective paging for PCS networks," *IEEE/ACM Trans. Netw.*, vol. 4, no. 4, pp. 629–638, Aug. 1996.

[16] *My Tracks* [Online]. Available: http://code.google.com/p/mytracks/

[17] *Case Design Guidelines for Apple Devices* [Online]. Available: https://developer.apple.com/resources/cases/Case-Design-Guidelines.pdf

[18] *Android Developers Reference: Location and Sensors* [Online]. Available: http://developer.android.com/guide/topics/sensors/index.html

[19] (2012). *Android Analyzer Report* [Online]. Available: http://androidfragmentation.com/database/l/manufacturer/samsung/deviceModel/Nexus

[20] M. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, "Entracked: Energy-efficient robust position tracking for mobile devices," in *Proc. 7th Int. Conf. Mobile Syst., Appl., Services*, 2009, pp. 221–234.

[21] C. Rasmussen, "Gaussian processes in machine learning," *Adv. Lect. Mach. Learn.*, vol. 14, no. 2, pp. 63–71, Apr. 2004.

[22] L. Ma, J. Liu, L. Sun, and O. Karimi, "The trajectory exposure problem in location-aware mobile networking," in *Proc. IEEE 8th Int. Conf. MASS*, Oct. 2011, pp. 7–12.

[23] A. Cavanaugh, M. Lowe, D. Cyganski, and R. Duckworth, "WPI precision personnel location system: Rapid deployment antenna system and sensor fusion for 3D precision location," in *Proc. Int. Tech. Meeting Inst. Navigat.*, Jan. 2010, pp. 214–219.

[24] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proc. 10th Int. Conf. Ubiquitous Comput.*, Sep. 2008, pp. 114–123.

Lei Zhang (S'12) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2011. He is currently pursuing the master's degree at the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. His current research interests include cloud computing, wireless networks, and multimedia communications.

Jiangchuan Liu (S'01–M'03–SM'08) received the B.Eng. degree (*cum laude*) from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2003, both in computer science. He is a recipient of the Microsoft Research Fellowship in 2000, the Hong Kong Young Scientist Award in 2003, and the Canada NSERC DAS Award in 2009. He is a co-recipient of the IEEE IWQoS in 2008 and the IEEE/ACM IWQoS in 2012 Best Student Paper Awards, the IEEE Communications Society Best Paper Award on Multimedia Communications in 2009, the IEEE Globecom Best Paper Award in 2011, and the ACM Multimedia Best Paper Award in 2012.

He is currently an Associate Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada, and was an Assistant Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, from 2003 to 2004. His current research interests include networking, in particular, multimedia communications, peer-to-peer networking, cloud computing, social networking, and wireless sensor/mesh networking. He is a member of Sigma Xi. He is an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA, and an Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS.

Hongbo Jiang (M'08) received the B.S. and M.S. degrees from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from Case Western Reserve University, Cleveland, OH, USA, in 2008.

He is an Associate Professor with the Faculty with the Huazhong University of Science and Technology. His current research interests include computer networking, especially algorithms and architectures for wireless and high-performance networks.

Yong Guan received the Ph.D. degree in computer science from the China University of Mining and Technology, Beijing, China, in 2004. He is currently a Professor with Capital Normal University, Beijing. His current research interests include formal verification, PHM for power, and embedded system design. He is a member of the Chinese Institute of Electronics Embedded Expert Committee and the Beijing Institute of Electronics Professional Education Committee, and Standing Council Member of the Beijing Society for Information Technology in Agriculture.

AUTHOR QUERIES

AQ:1 = Please provide the report no. for ref. [1]–[2].
AQ:2 = Please provide the year for ref. [6], [16]–[18].

SensTrack: Energy-Efficient Location Tracking With Smartphone Sensors

Lei Zhang, *Student Member, IEEE*, Jiangchuan Liu, *Senior Member, IEEE*, Hongbo Jiang, *Member, IEEE*, and Yong Guan, *Member, IEEE*

***Abstract*— Nowadays, as smartphones are becoming more and more powerful, applications providing location based services have been increasingly popular. Many, if not all, smartphones are equipped with a powerful sensor set (GPS, WiFi, the acceleration sensor, the orientation sensor, etc.), which makes them capable of accomplishing complicated tasks. Unfortunately, as the core enabler of most location tracking applications on smartphones, GPS incurs an unacceptable energy cost that can cause the complete battery drain within a few hours. Although GPS is often preferred over its alternatives, the coverage areas of GPS are still limited (GPS typically cannot function indoors). To this end, our goal in this paper is to improve the energy-efficiency of traditional location tracking service as well as to expand its coverage areas. In this paper, we introduce SensTrack, a location tracking service that leverages the sensor hints on the smartphone to reduce the usage of GPS. SensTrack selectively executes a GPS sampling using the information from the acceleration and orientation sensors and switches to the alternate location sensing method based on WiFi when users move indoors. A machine learning technique, Gaussian process regression, is then employed to reconstruct the trajectory from the recorded location samples. We implemented a prototype on an Android smartphone that can sample the related sensors during the user's movement and collect the sensor data for further processing on PCs. Evaluation on traces from real users demonstrates that SensTrack can significantly reduce the usage of GPS and still achieve a high tracking accuracy.**

***Index Terms*— Location tracking, smartphone, sensor.**

I. Introduction

UNDERSTANDING human mobility in daily life is a fundamental resource for broad-domain applications, especially for the applications that provide location based services. With the increasing pervasiveness of smartphones over the past few years, many emerging location based applications are adopted by mobile users. Consumer and advertiser expenditure on location based services is expected to approach $ 10 billion by 2016 [1]. The reason that location based applications become so popular is two-fold. First, location based services rely on the knowledge about the user's geographical location to obtain relevant information on the spot, and thus offer the user a plethora of options to satisfy his/her needs under that particular context. Second, a typical modern mobile device usually has the ability to locate or estimate its current position. The localization technologies used today mainly based on Global Positioning System (GPS), other technologies also obtain assistance from WiFi and GSM, each of which can vary widely in energy consumption and localization accuracy. As it is known to be more accurate, GPS is often preferred on mobile platforms over its alternatives such as GSM/WiFi based positioning systems.

Although smartphones today are capable to accomplish complicated tasks such as localization, we still face problems. The demand of computing and storage capability on mobile devices is rapidly increasing in recent years, whereas the battery manufacturing industry moves forward slowly (battery capacity grows by only 5% annually [2]). In spite of the increase in processing power, feature-set, and sensing capabilities, the smartphones continue to suffer from limited battery life. Unfortunately, it is also well-known that GPS, the core enabler of many location-based applications, is power-hungry. The aggressive usage of GPS can cause the battery to completely drain within a few hours [3], [4]. Location based applications still cannot assume continuous and ubiquitous location access in their design because of the high energy expense for localization. Even within the limited hours of being activated, GPS may not function well all the time, especially when the mobile user is under the shelter of buildings due to the signal loss under indoor environment [5]. When GPS is unavailable, alternate location sensing techniques must be used to obtain the approximated location. The variability in accuracy provided by various location sensing technologies and the limits on their coverage areas pose additional challenges for application developers [6]. Using multiple location sensors simultaneously to make up for this variability in accuracy would further increase energy cost.

In this paper, we present the design of SensTrack, a location tracking service that provides user's moving trajectory while reducing its impact on the devices's battery life. By applying different localization technologies, we expand the coverage area compared to the traditional approach that only uses GPS. In addition, the sensor hints from the smartphone itself can help us make decisions about adaptive sampling. SensTrack

Manuscript received January 28, 2013; revised June 9, 2013; accepted July 1, 2013. This work was supported in part by a Canada NSERC Discovery Grant, an Engage Grant, a MITACS Intership Grant, and a China NSF international collaboration under Grant 61120106008. The associate editor coordinating the review of this paper and approving it for publication was Dr. Mahmoud Daneshmand.

L. Zhang and J. Liu are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: lza70@sfu.ca; jcliu@cs.sfu.ca).

H. Jiang is with the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: hongbojiang@hust.edu.cn).

Y. Guan is with the College of Information Engineering, Capital Normal University, Beijing 10048, China (e-mail: guanyong@cnu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSEN.2013.2274074

smartly selects the location sensing methods between WiFi and GPS, and reduces the sampling rate by utilizing the information from acceleration sensor and orientation sensor, two of the most common sensors found on smartphones today. We have implemented a prototype on the Google Nexus S phone, which continuously collects data from the acceleration sensor and the orientation sensor, and records the location samples from GPS and WiFi. Experiments have been conducted on a real world path while the phone was carried by a mobile user in a region of our university campus. The collected data is further analyzed and filtered on computers. To predict the user's original trajectory, a track reconstruction algorithm based on a machine learning technique is also implemented on the server side. Performance evaluation on the real data sets shows that SensTrack only needs 7% GPS samples of the naive approach and saves nearly 90% GPS activated time. Meanwhile, SensTrack reconstructs the user's trajectory with high accuracy and better coverage.

The main contributions of this paper are listed as follows:

- We identify the problems of traditional location tracking service including limited availability of GPS and unnecessary GPS samplings. The opportunities of energy-efficiency improvements by utilizing the assistance from sensors on smartphones are discussed.
- We present the detailed design of an energy-efficient location tracking service, SensTrack. As the main component, a track reconstruction algorithm based on Gaussian Process Regression is proposed. Other mechanisms for making smart adaptive sampling decisions are also discussed.
- We implement a prototype of SensTrack, and evaluate the proposed system through real-world experiments.

This paper is organized as follows. In Section II we review the related work on energy-efficient location sensing. Section III presents our observations on the defects of tradition location based applications based on GPS, and discusses the opportunities of improvements. The detailed design of SensTrack is proposed in Section IV. We evaluate our proposal in Section V and analyze the performance improvement. Further considerations are discussed in Section VI. Section VII concludes the paper and outlines the future work.

II. Related Work

To track the users' locations, many energy-efficient sensing approaches with adaptive sensing policies have been proposed to minimize the energy consumption [3], [7]–[9]. With the objective of minimizing the location error for a given energy budget, EnLoc [3], an energy-efficient localization framework, includes a heuristic with a local mobility tree to predict the next sensing time by utilizing the dynamic programming technique. Jigsaw [8] uses the information obtained from the acceleration sensor and the microphone to continuously monitor human activities and environmental context. According to the user's mobility patterns, a discrete-time Markov Decision Process is employed to learn the optimal GPS duty cycle schedule with a given energy budget.

There are also works based on the observation that the required localization accuracy varies with locations. An adaptive location service for mobile devices, a-Loc [7] uses a Bayesian estimation framework to determine the dynamic accuracy requirement, and tunes the energy expenditure accordingly. It argued in [9] that given the less accuracy of GPS in urban areas, it suffices to turn on GPS adaptively to achieve this accuracy. The rate-adaptive positioning system for smartphone applications (RAPS) was then proposed to minimize energy consumption with given accuracy threshold by using the information of moving distance, space-time history, and cell tower-based blacklisting.

Smartphones' energy consumption has been a major concern in research for a long time, and a number of studies have been done to improve the energy efficiency of mobile devices. In order to understand where and how the energy is used, A. Carroll *et al.* [10] measured the power consumption of a modern mobile device (the Openmoko Neo Freerunner mobile phone), broken down to the devices major subsystems (CPU, memory, touchscreen, graphics hardware, audio, storage, and various networking interfaces), under a wide range of realistic usage scenarios. M. Ra *et al.* [11] proposed the Stable and adaptive link selection algorithm (SALSA), an optimal online algorithm for energy-delay tradeoff based on the Lyapunov optimization framework. SALSA defers the transmissions of delay-tolerant applications until a less energy-consuming WiFi connection becomes available.

Utilizing the sensing power of smartphones is not a new topic in literature. M. Keally *et al.* [12] presented the design of Practical Body Networking (PBN) system to provide practical activity recognition with mobile devices, which combines the sensing power of on-body wireless sensors with the additional sensing power, computational resources, and user-friendly interface of an Android smartphone through the unification of TinyOS motes and Android smartphones. Another interesting ongoing work discusses how to fuse information from Microsoft Kinect's tracking with the smartphone's sensor readings to improve Kinect gaming experience [13].

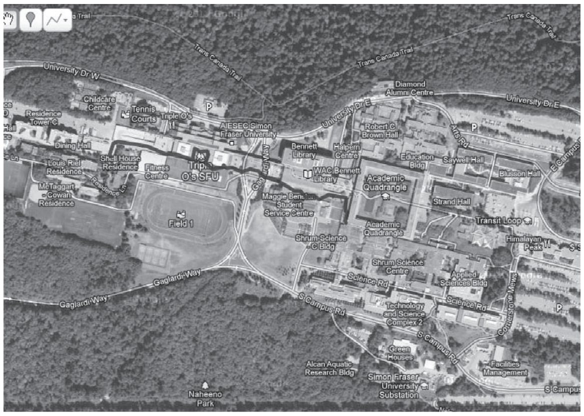
Inspired by many existing studies, in this paper we take efforts to achieve a high energy efficiency by reducing the sampling rate of sensing users' locations. However, our work uses a novel approach by utilizing the acceleration sensors and the orientation sensors on smartphones to capture the geometric features of users' moving trajectories. We will further explain the difference between SensTrack and existing works in the following sections.

III. Challenges and Opportunities

In this section, we start by describing the defects of typical location-based applications that utilize GPS, including limited availability and unnecessary samples. We then discuss the opportunities for making improvements.

A. Limited Availability of GPS Versus Multiple Location Sensing Methods

It should be noted that traditional GPS cannot work properly under the indoor environment. The standard GPS



(a)

(b)

Fig. 1. Tracking results when $T = 5$ s, $\theta = 45°$, $D = 100$ m, $\upsilon = 8$ m/s. (a) Track recorded by the naive approach. (b) Track reconstructed by SensTrack.

receiver requires signals from at least 4 satellites simultaneously to calculate and output 3-dimensional locations and velocity information [5]. Therefore, the mobile devices need to be in line-of-sight contact with the GPS satellites, which significantly limits the usage of typical location based applications.

Figure 1(a) shows one track that we took using GPS on a mobile device. Although we did not stop recording, the track ends once it entered the building (the Academic Quadrangle in our campus), which indicates the performance of GPS largely depends on the working condition. The signals from GPS satellites can be blocked not only by buildings but also by canyon walls, trees, and even thick clouds. When the user walks through buildings, GPS equipped by a normal smartphone cannot function since the lack of satellite signals. Even worse, GPS units may consume more energy than the normal situation when there is no satellite signals [14].

Besides GPS, there also exist alternate location sensing technologies. For example, Android OS provides a network-based localization mechanism, which exploits GSM footprints from cell towers and WiFi signals to obtain an approximate location. Although the network-based location sensing is not as accurate as GPS, it provides the possibility to keep tracking inside a building since it mainly relies on the WiFi connection, in which case GPS units can be deactivated to save battery.

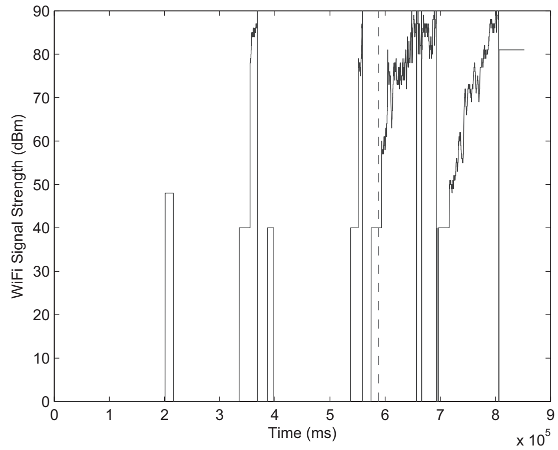


Fig. 2. WiFi signal strength along the track.

For the scenarios like university campus, hotels or hospitals, we can always assume persistent wireless local network access, which implies that other location sensing methods may provide us valid options when GPS is out of use.

Figure 2 shows the received WiFi signal strength along the track presented in Figure 1(a). The dash line indicates the time stamp (588 s) at which the user entered the Academic Quadrangle. There are some spikes before 588 s (201 s $\sim$ 216 s, 335 s $\sim$ 368 s, 387 s $\sim$ 398 s, 537 s $\sim$ 558 s), which means that the user can receive some WiFi signal for a short time when passing by buildings. After entering the building at 588 s, the received WiFi signal stayed at a relatively high level since the WiFi connection is assured in teaching areas of the university campus. This figure can support our argument that, when the user is inside a building, WiFi signal is usually relatively strong. Therefore, the network-based localization can be a valid choice under the indoor environment where GPS is no longer available. The idea is to use the GPS satellite signal and the wireless network connection as indicators for switching between GPS and the network-based location sensing method.

B. Unnecessary GPS Samplings Versus Adaptive Sampling

The GPS sensor can sample the user's location at a relatively high rate. However, it is not ideal to record every location update since the error for each location sample varies. To make the path more smooth and fit the real trajectory, a typical location based application usually updates the user's location only if the distance to the last valid location sample is larger than a certain threshold [15]. Therefore, with a fixed and frequent GPS location sampling policy, it probably introduces a significant amount of unnecessary GPS samples.

To demonstrate this, we collect the system log of an Android application, *My Tracks* [16], which uses the GPS sensor in mobile devices to record the paths that users take while hiking, cycling, running, or participating in other activities. Figure 3 shows part of the system log, demonstrating its executing history in one run. As shown in the figure, the application usually takes several GPS samples to get one valid location update, in which case the threshold is 5 meters. Our experimental result in this case shows that up to 79%

```
04-16 01:25:35.535 D/MyTracks(18469): Not recording. Distance to last
recorded point (3.257859 m) is less than 5 m.
04-16 01:25:36.566 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:36.574 D/MyTracks(18469): Not recording. Distance to last
recorded point (4.096747 m) is less than 5 m.
04-16 01:25:37.519 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:37.527 D/MyTracks(18469): Not recording. Distance to last
recorded point (4.636137 m) is less than 5 m.
04-16 01:25:38.515 D/MyTracks(18469):
TrackRecordingService.onLocationChanged
04-16 01:25:38.527 D/MyTracksLib(18469):
MyTracksProviderUtilsImpl.insertTrackPoint
04-16 01:25:38.527 D/MyTracksProvider(18469): MyTracksProvider.insert
```

Fig. 3. A part of system log when running a location-based application.

location samples of *My Tracks* are unnecessary. Since many of the samples are discarded, these invalid location measurements cause unnecessary energy consumption.

C. Assistance From Other Sensors

Nowadays smartphones become more and more powerful in terms of hardware, which usually contains various sensors. As an example, iPhone 4 is equipped with several environmental sensors, including an ambient light sensor, a magnetic compass, a proximity sensor, an accelerometer, and a three-axis gyroscope [17]. Android 4.0 (API Level 14) also supports up to 13 kinds of sensors [18], even though the sensors' availability varies from device to device. The supported list of sensors in a Google Nexus S phone consists of: one KR3DM 3-axis Accelerometer, one AK8973 3-axis Magnetic field sensor, one AK8973 Orientation sensor, one GP2A Proximity sensor, one GP2A Light sensor, one Linear Acceleration Sensor, one Rotation Vector Sensor, one K3G Gyroscope sensor, and one Gravity Sensor [19].

To reduce unnecessary GPS samples, adaptive sampling is proposed in many existing works [3], [7]–[9]. Usually we need additional information to make adaptive sampling decisions, which may include the location history, the speed history, the distance information, remaining battery power, the accuracy requirement, etc. In this paper, we utilize the powerful sensors equipped by smartphones to obtain the information about changes of the orientation, moving speed, and traveled distance. Based on these useful information, we are able to make smart adaptive sampling decisions. The detailed design is described in the following section.

IV. SensTrack: Design Details

A. Overview

To reduce the frequency of location sensing, SensTrack periodically collects data from the corresponding sensor to detect a turning point or estimate current speed and the distance from the last recorded location. The high energy efficiency of this approach is supported by the fact that the GPS sensor consumes much more energy than the acceleration sensor and the orientation sensor [9], [20]. When the GPS satellite signal is not available and the WiFi connection is active, SensTrack switches to the network-based location sensing method to obtain the raw coordinates. The last step of SensTrack is to upload the coordinates of sampled locations to an online server that uses a machine learning algorithm to reconstruct a smooth and accurate trajectory.

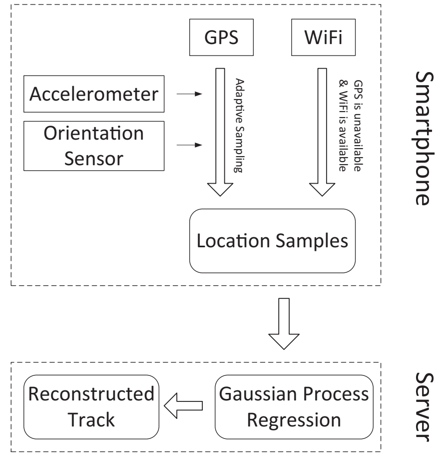


Fig. 4. The system architecture.

Figure 4 demonstrates the SensTrack's system architecture. The service consists of two stages: the first is to collect the location samples; and the second is to reconstruct the original trajectory. Given the working conditions, SensTrack switches between the GPS-based and the network-based localization methods using the GPS or WiFi sensors, respectively. By utilizing the sensor hints from the acceleration sensor and the orientation sensor, SensTrack is able to make smart adaptive sampling decisions in the GPS mode. For example, when the smartphone detects a turning point or if it estimates a unreasonable speed or a unexpected large traveling distance, it uses GPS to record the current location. After the server side receives all the collected location samples, a Gaussian Process Regression algorithm is then employed to predict the trajectory that the user has taken.

B. Track Reconstruction: Gaussian Process Regression

Once the collection of location samples is finished, it is not ideal to simply connect all the recorded locations, since the distances between any two successive locations may not be the same. For some parts of a trajectory, the recorded locations can be very sparse, while for other parts, the location samples may be relatively intensive. If we simply connect the location samples, the resultant trajectory can be very abstract. Therefore, uploading the collected data to the online server either by a wireless or wired connection to reconstruct the trajectory is our last stage. We adopt the Gaussian Process Regression (GPR), a machine learning technique to perform the interpolation. The training set of the algorithm is the recorded critical locations decided by the sensor hints which capture most of key features of a trajectory. And the testing set is the predicted locations between the successive but far-away location samples. Combing both input and output gives us the final trajectory. We next detailed describe GPR and how the user's trajectory can be reconstructed by using GPR.

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution, and is fully specified by a mean function and a covariance function [21]. The inference of continuous values with a Gaussian process prior is known as Gaussian Process

Regression. Consider x as a general random variable. We define the mean function $m(x)$ and the covariance function $k(x, x')$ of a real process $f(x)$ as

$$m(x) = E[f(x)],$$
$$k(x, x') = E[(f(x) - m(x))(f(x') - m(x'))],$$

and can write the Gaussian process as

$$f(x) \sim gp(m(x), k(x, x')).$$

For notational simplicity the mean function is usually set to be zero. In our method the covariance function will be the squared exponential covariance function, although other covariance functions may also be useful. Assuming that observations are noise-free, the covariance function specifies the covariance between pairs of random variables

$$cov(f(x_p), f(x_q)) = k(x_p, x_q) = exp(-\frac{1}{2}|x_p - x_q|^2). \quad (1)$$

For a estimate data set X_*, we can generate a random Gaussian vector f_* for target values with the covariance matrix calculated from Equation 1

$$f_* \sim N(0, K(X_*, X_*)).$$

Therefore, the joint distribution of the training outputs f and the test outputs f_* according to the prior is

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(K_*,X) & K(X_*,X_*) \end{bmatrix}\right). \quad (2)$$

If X contains n training points and X_* contains n_* test points, then $K(X, X_*)$ is the $n \times n_*$ matrix of the covariances evaluated at all pairs of training and test points. And the other entries $K(X, X)$,$K(X_*, X)$,and $K(X_*, X_*)$ are similar.

If observations are noisy, we can write $y = f(x)+\varepsilon$. Assuming additive independent identically distributed Gaussian noise ε with variance σ^2, we have the prior as

$$cov(y_p, y_q) = k(x_p, x_q) + \sigma_n^2 \delta_{pq}$$

or

$$cov(\mathrm{y}) = K(X, X) + \sigma_n^2 I,$$

where δ_{pq} is a Kronecker delta which is one when $p = q$ and zero otherwise. Introducing the noise in Equation 2, the joint distribution of the observed target values and the function values at test points according to the prior will be

$$\begin{bmatrix} \mathrm{y} \\ f_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X)+\sigma_n^2 I & K(X,X_*) \\ K(K_*,X) & K(X_*,X_*) \end{bmatrix}\right). \quad (3)$$

The posterior distribution over functions can be obtained by restricting the joint prior distribution on the observations. Then we arrive at the key predictive equations for GPR

$$f_*|X, \mathrm{y}, X_* \sim N(\overline{f_*}, cov(f_*)), \text{ where} \quad (4)$$
$$\overline{f_*} = E[f_*|X, \mathrm{y}, X_*] = K(X_*, X) \times [K(X,X) + \sigma_n^2 I]^{-1}\mathrm{y}, \quad (5)$$
$$cov(f_*) = K(X_*, X_*) - K(X_*, X) \times [K(X,X) + \sigma_n^2 I]^{-1} K(X, X_*). \quad (6)$$

Algorithm 1 Predictions(X,y,k,σ_n^2,x_*)

1: $L = \text{cholesky}(K + \sigma_n^2 I)$
2: $\alpha = L^\top \backslash (L \backslash \mathrm{y})$
3: $\overline{f_*} = k_*^\top \alpha$
4: $\upsilon = L \backslash k_*$
5: $V[f_*] = k(x_*, x_*) - \upsilon^\top \upsilon$
6: $\log p(\mathrm{y}|X) = -\frac{1}{2} y^\top \alpha - \sum_i \log L_{ii} - \frac{n}{2} \log 2\pi$
7: return $(\overline{f_*}, V[f_*], \log p(\mathrm{y}|X))$

We then focus on explaining how to use GPR with given location samples to reconstructed the estimated trajectory. A trajectory can be considered as the path that the user follows through space as a function of time. Specifically, we have n location samples from x_1 to x_n, each of which can be represented by a two-dimensional points $\mathrm{x}_i = (x_i, y_i)$. Then X is the sampled date set for all (x_i, y_i) s. According to what we have explained, the user's track can be represented by generated GPR functions which is determined by a covariance function and a mean function. In the case that there is only one test point x_*, we let $k(\mathrm{x}_*) = k_*$ denote the vector of covariances between the test point and the n training points. Then for a single test pointx_*, Equation 5 and 6 can be reduced to

$$\overline{f_*} = k_*^\top (K + \sigma_n^2 I)^{-1} \mathrm{y}, \quad (7)$$
$$V(f_*) = k(X_*, X_*) - k_*^\top (K + \sigma_n^2 I)^{-1} k_*. \quad (8)$$

On obtaining Equation 7 and 8, we further propose the following Algorithm 1 for a single test case, in which cholesky $(K + \sigma_n^2 I)$ is the Cholesky decomposition on the matrix of $K + \sigma_n^2 I$. The implementation addresses the matrix inversion required by Equation 7 and 8 using Cholesky factorization. For multiple test cases lines $3 \sim 6$ are repeated. In our case, X is time space of the training set, y is the set of observed target values (location samples), k is the covariance function, $\sigma_n^2 I$ is the noise, and x_* is the testing data. The outputs are as follows. $\overline{f_*}$ is the mean predicted value (predicted location of x_*), $V[f_*]$ is its variance, and $\log p(\mathrm{y}|X)$ is the marginal likelihood. A more detailed explanation can be referred to our previous work [22].

C. Switching Location Sensing Methods

As mentioned, it is well-known that GPS cannot function properly indoors. To expand the coverage areas, SensTrack switches between GPS and the network-based localization through the wireless connection. Basically, we want to use GPS outdoors and the network-based localization indoors, and thus it is important to decide when to switch. Initially, SensTrack starts in the GPS mode and periodically executes a WiFi scan. When it detects the GPS signal loss as well as an active wireless network connection, SensTrack turns into the WiFi mode. If GPS becomes available again, and the phone loses the WiFi connection or the accuracy of location samples provided by the network decreases significantly, SensTrack switches back into the GPS mode.

We note that there are two conditions satisfied to switch the location sensing method: the current method fails to obtain location samples, and the other method is guaranteed to work, which prevents from switching between the two modes too often. Frequently changing location sensing mechanism can be very energy consuming, because the high-power components associated with both location providers need to be active. In some cases, both of the two methods are available when the user passing by some buildings. According to our rules, we should not change SensTrack's working mode, since in these situations the wireless connection tends to be unstable and short. In other cases, none of the two methods are available if we simply lose the GPS satellite signal outdoors. Our rules can also avoid the unnecessary switching in these cases. It is also worth mentioning that SensTrack stops collecting the sensor hints when it switches into the WiFi mode. In another word, we passively receive location updates in this mode. The reason is that, unlike GPS, when we request the location information, the WiFi localization technology cannot respond within a tolerable delay. It means that even if we apply the sensor hints to sense the location adaptively, we cannot obtain a location sample timely in the WiFi mode. Therefore, considering the WiFi localization updates the location less frequently than GPS, we decided not to waste energy on the acceleration sensor and the orientation sensor.

D. Utilizing Sensor Hints

1) Orientation: SensTrack employs the orientation sensor as a detector of turning points when the user is moving. The idea is that there is no need to record the user's location if he/she is in a steady movement without changing direction. For a sliding window of size T, SensTrack collects the readings of the orientation sensor, and computes the changes in direction. If user's moving direction changes dramatically (greater than the threshold θ), a location sensing of the user's current location is executed. Considering the readings from the orientation sensor is approximately continuous, the window size T should be larger enough to observe the potential direction changes. Table I shows the effect of the window size T. In our experiments, T was set to be 5 s because it would lose some turns of the trajectory for smaller window size. On the other hand, a larger window size is not necessary as it requires more memory and computation, which in turn requires more powerful hardware. The user can also decide the threshold θ, the other key parameter, according to their expectations on accuracy. Table II presents the number of missing turning points for different values of θ. Roughly speaking, SensTrack is more sensitive with a smaller θ. However, a too small θ may cause redundant detections of the trajectory's turns (false positives) if we consider the noises in the readings from the sensor, which potentially wastes energy in sensing locations at those false turning points.

2) Acceleration: The acceleration sensor in a mobile device has been widely used in many existing location sensing systems, in which it acts as a binary sensor to detect user movement or non-movement. We notice that distance is theoretically a simple integral of speed, which in turn is an integral of acceleration. Unlike most prior works, we do not limit the acceleration sensor just to be the user's movement detector, rather explore the possibility of calculating the distance that the user has traveled and the speed that the user is moving at.

TABLE I

EFFECT OF WINDOW SIZE T

$T =$	$1s$	$3s$	$5s$	$7s$
key turning points	4 misses	1 miss	0 miss	0 miss

TABLE II

EFFECT OF THRESHOLD θ

$\theta =$	$45°$	$60°$	$75°$	$90°$
key turning points	0 miss	1 miss	3 misses	4 misses

It should be noted that the readings of the acceleration sensor on a moving device are usually noisy, especially when the user is walking. Activities with higher speed, like biking and driving, actually are more stable, whereas the movement of a pedestrian is always fluctuating. It often overestimates distance when the user is holding the phone in his/her hands, and underestimates distance when sitting quietly on a cushioned car seat [9]. When calculating the integrals, errors caused by the noise in the sensing data are accumulated. However, we argue that the estimated distance and speed obtained as integrals of acceleration are still useful even if they are inaccurate, because the location and velocity information provided by GPS can help us to calibrate the calculation. Once the estimated distance or the estimated speed exceeds the thresholds, specifically D and v, SensTrack activates GPS to sense the current location and speed. The thresholds can be set based on the accuracy requirement or the user's moving patterns. For example, for a pedestrian, usually the moving speed can be no more than 10 m/s and should not be negative, and the accuracy requirement is usually higher. Moreover, the calibration of calculating the integrals can also be done when GPS is activated at the turning points.

V. EVALUATION

A. Data Collection and Methodology

We evaluated SensTrack using a real data set collected from a Google Nexus S phone carried by a mobile user walking in our university campus. The phone is equipped with an integrated GPS, an WiFi sensor, an accelerometer, and an orientation sensor. We implemented a SensTrack prototype on Android 4.0 (API level 14). During its runtime, the prototype continuously collects data from the acceleration sensor and the orientation sensor at default rate of the system service (SENSOR_DELAY_NORMAL) in Android OS. When the GPS signal is available, a location listener is registered to request location updates from GPS periodically. Meanwhile, the prototype always tries to initiate and maintain a WiFi connection, which can be used to record the location updates from the network-based location provider. In our experiments, a PC server was used to further analyze the data collected by the smartphone and filter the GPS and WiFi location samples with the given parameters. The trajectory reconstruction algorithm based on GRP was also implemented on the server side, which uses the filtered and valid location samples to predicted

TABLE III
AVERAGE ERROR OF PREDICTED LOCATIONS

	recorded locations	predicted locations	average error
SensTrack	38 samples	24 predictions	3.128m
GPS trace	568 samples	0	0

the original trajectory. For most of the presented results, our settings were $T = 5$ s, $\theta = 45°$, $D = 100$ m, $\upsilon = 8$ m/s, and a prediction was made if the time gap between two successive GPS samples is greater than 15 s.

We also compared SensTrack with the naive approach, in which GPS is the only way to obtain location information and the GPS sensor is kept to be activated during the whole tracking period. Unlike SensTrack, which samples the GPS location actively, the naive approach is a passive method that records all the valid location updates from GPS. We conducted the experiments on the same real path for several times, which started from outdoor environment, came into a building, and then ended indoors. The total length of the path is around 1.1 km. The results show that, without significantly losing the accuracy of tracking, SensTrack effectively reduce the number of GPS samples and the time that the GPS sensor needs to be turned on.

B. Accuracy

We first present the tracking results by SensTrack and the naive approach. Despite the tracking service maintained, the trajectory shown in Figure 1(a) ended once the user entered the building since the signals from GPS satellites were blocked by the building, which indicates the performance of GPS largely depends on the working condition. Compared to the naive approach, SensTrack demonstrates a reasonably better performance. Figure 1(b) shows that the trajectory reconstructed by SensTrack has a similar outdoor part, meanwhile it has the indoor part that the original one does not have. Although the indoor part of the second trajectory may be not that accurate given the limitation of WiFi localization technology, it is still good to have a approximate trajectory.

As previously stated, the resulting trajectory generated by SensTrack consists of two kinds of points: the sampled locations and the predicted locations. To evaluate the accuracy of SensTrack, we took the GPS trace as the ground truth and calculated the average error of the predicted locations. For every prediction, we computed the difference between the predicted location and the real location in the GPS trace at the same time. The result shown in Table III proves that SensTrack can achieve a high accuracy. The average error of the predictions is 3.128 meters, which is quite acceptable (GPS can achieve an accuracy of 5 meters in good signal conditions). It should be noted that even the GPS trace may not be the real path that the user has taken, because the performance of GPS depends on a number of factors such as the user's position, time, surroundings, weather, etc, which means that the GPS trace itself can be inaccurate. Another result from Table III is that the naive approach recorded 568 samples over the testing path, although some of them may be unnecessary as discussed earlier. It is worth mentioning that, whether a sample is necessary should be decided case by case. For different scenarios, the ideal minimal distance (threshold) between two valid samples can vary significantly. We can adjust the number of necessary samples by setting the granularity between successive samples and filtering the recorded samples accordingly. In our experiments, the number of necessary samples does not affect the total number of GPS samples as the naive approach passively received every sample, and the granularity between successive samples cannot reflect the error of reconstructed trajectory.

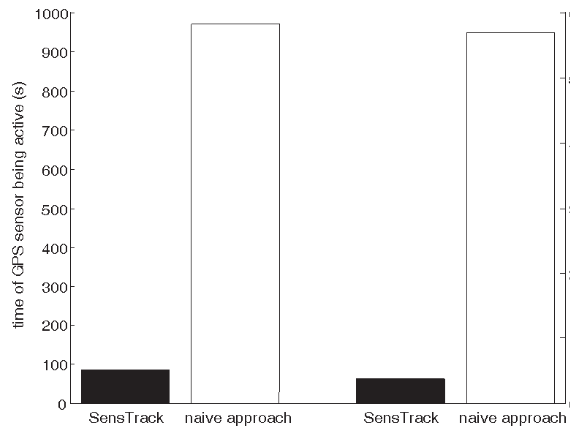


Fig. 5. Comparison of the energy efficiency.

C. Energy Efficiency

In modern mobile devices, the GPS receiver usually consume much more power than the accelerometer and the digital compass. For example, our testing device, a Google Nexus S phone, is equipped with a BCM4751 integrated GPS receiver (produced by Broadcom), a KR3DM 3-axis accelerometer (produced by STMicroelectronics), and an AK8973 3-axis electronic compass (produced by Asahi Kasei Microdevices). With the battery supply (3.7 volt), the power consumption (in terms of current) of the accelerometer is 0.23 mA; and the current consumption of the compass is 6.8 mA; however, the current consumption of the GPS receiver can be as much as 80 mA. To demonstrate the energy efficiency of SensTrack, we present that SensTrack can significantly reduce the number of needed GPS samples and the time that the GPS sensor needs to be activated. We did not measure the actual energy consumption of SensTrack, since we thought it is unnecessary. For different hardware, the power consumption varies, and thus the energy consumption of SensTrack on a specific hardware model only provides limited information. Therefore, it is convincing and sufficient for us to show the relative energy efficiency of SensTrack to the naive approach by comparing the number of required sampling and the activated time of the GPS receiver.

Figure 5 shows that compared to the naive approach, SensTrack only needs 7% GPS samples for the described path, and the time of the GPS sensor being active is decreased by nearly 90%. The naive approach almost updated the user's location every second, and the GPS sensor was kept to be activated even when the user entered the building and lost the

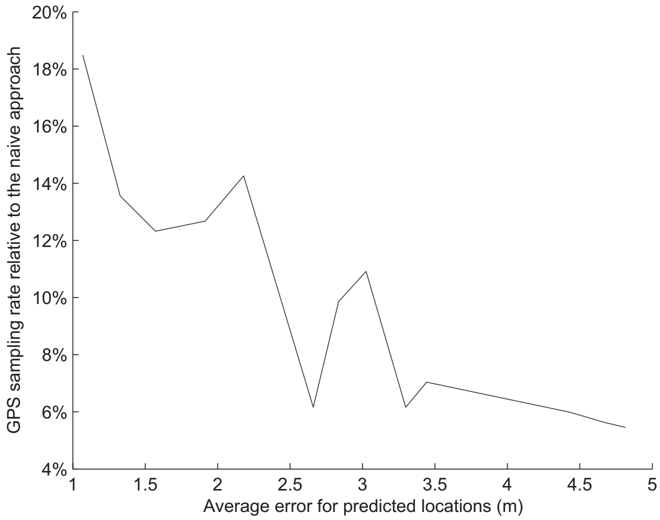


Fig. 6. Tradeoff between sampling rate and accuracy.

GPS satellite signals. SensTrack on the contrary only selectively activated the GPS sensor at some separate locations, and turned the GPS sensor off once the device lost the satellite signals and had an active WiFi connection. It should be pointed out that the energy efficiency of SensTrack depends on the user's movements and the path that the user takes. If the user's movement is very unstable and the direction changes frequently, SensTrack inevitably activates the GPS sensor more frequently, and thus consumes more energy.

D. Energy-Accuracy Tradeoff

By intelligently managing the energy and localization accuracy trade-off, the battery life of a mobile device can be significantly extended, which is of great importance for the smartphone users. Since the required localization accuracy varies with locations, there is significant potential to trade-off the accuracy and the energy consumption based on the application's needs and different working scenarios.

As mentioned before, we take the GPS sampling rate as a representative of SensTrack's power consumption. Figure 6 demonstrates the trade-off between sampling rate and accuracy, which SensTrack presents under different configurations. Even though there exists some bias, we can observe a clear trend that a higher accuracy requires a higher GPS sampling rate, which means more power consumption. On the other hand, Figure 6 does not present a strict monotonicity. A higher energy consumption does not necessarily indicate a higher accuracy. For example, it only requires 6% samples to achieve a higher accuracy (average error is 2.66 m), whereas 11% samples are needed to produce a relatively lower accuracy (average error is 3.02 m). This is because the error of one prediction not only depends on the GPS sampling rate but also depends on the performance of the reconstruction algorithm. For GPR in our case, if the location samples have higher covariances between each other and are uniformly distributed on the path in time space, the algorithm can produce better results and achieve a higher accuracy. Therefore, besides the sampling rate, the actual samples themselves collected by the system have a huge impact on the results. The samples that have similar covariances between every two successive samples are more likely to produce highly accurate predictions.

TABLE IV
AVERAGE WiFi TRAFFIC

	received	transmitted
Baseline	0.38 packet/s	0.88 packet/s
SensTrack	0.94 packet/s	0.81 packet/s

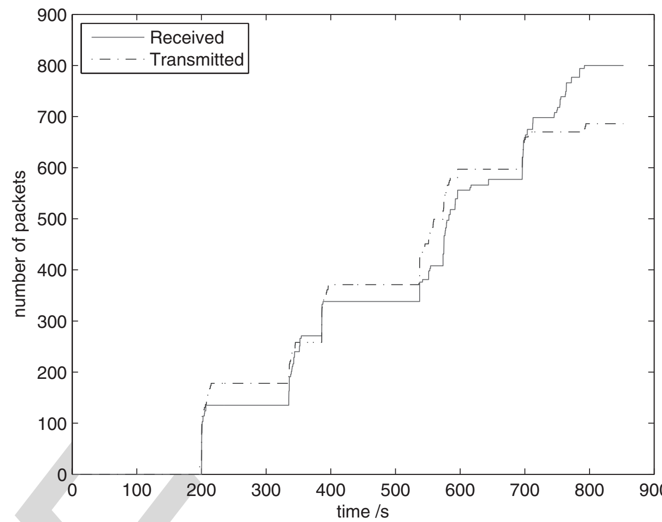


Fig. 7. WiFi traffic of SensTrack.

E. Transmission Overhead

There is no doubt that exploiting network-based localization technology to obtain approximate locations would incur some extra network transmissions. To measure the extra traffic, we recorded the traffic loads of SensTrack and the baseline. As the baseline, there only maintains a valid wireless network connection. To be clear, we did not include the uploading of location samples into the transmission overhead, because unlike the indoor location sensing, the uploading process does not need to be done in real time.

Table IV presents the average numbers of the received and transmitted packets during the tracking process. For both SensTrack and the baseline, the average numbers of the transmitted packets were close. Although SensTrack theoretically should transmit more packets as it requests location information through the wireless link, the result is within a normal error range. On the other hand, SensTrack received more than twice as many packets as the baseline did. We argue that even if the number of received packets increases, the total transmission overhead may not be intolerable, because the size of received packets that contains only the location information should be small. Moreover, since the WiFi connection is usually free, there is no need to worry about the wireless network traffic. Another point is that communicating with the access points consumes less energy than communicating with the GPS satellites. Figure 7 further shows SensTrack's traffic pattern, which matches the result in Figure 2. SensTrack had WiFi traffic in the time intervals of strong WiFi signals

(201 s ∼ 216 s, 335 s ∼ 368 s, 387 s ∼ 398 s, 537 s ∼ 558 s). After entering the building at 588 s, SensTrack continuously transmitted and received packets.

VI. Further Discussion

A. Multiple Mobility Patterns

Although our work focuses on the pedestrians, it can be easily extended on multiple mobility patterns, such as running, biking, driving, etc, which are often with higher speeds. Intuitively these movements are more stable, and thus the trajectories are likely less complex, and thus the sensors on smartphones can easily capture the features of the path. Therefore, our approach at least paves the road of designing the efficient tracking service for multiple mobility patterns. However, given the characteristics of different movements, modifications should be carefully considered.

B. Energy Consumption of Accelerometer and Orientation Sensor

In this paper, to make our point clear, we assume a continuous sampling of the acceleration sensor and the orientation sensor, which may cause unnecessary energy cost. It is not necessarily the case. Given that the energy-efficiency is a major goal of our design, users can further employ a low duty cycle on the usage of the acceleration sensor and the orientation sensor. Since the high speed movements are more stable, a low duty cycle can still allow the sensors to capture the features of the users' movements.

C. Other Indoor Localization Technologies

Our work chose the network-based method, which is mainly based on the WiFi positioning system, as our indoor localization approach. The primary reason is that the implementation of this method is already provided as APIs in Android platforms (since API level 1). Other methods for the indoor localization can also be employed such as the specialized real-time locating systems (RTLS) [23] or the inertial measurement unit (IMU)-based navigation systems [24]. However, many of these methods also require a costly infrastructure or additional hardware, which hardly satisfy the need for a cost-effective solution. On the other hand, indoor localization is not our main concern in this paper, rather it is a supplementary of GPS to extended the coverage of SensTrack.

VII. Conclusion

In this paper, we have proposed a novel location tracking service, SensTrack. We first discussed the limitations of the traditional GPS-based approach and opportunities of improvements. Next, the detailed design of SensTrack was presented including: the trajectory reconstruction algorithm based on the Gaussian Process Regression, the rules of switching between two location sensing methods, and the principles for exploiting the sensor hints. We then used the real traces to evaluate the performance of SensTrack, which shows that SensTrack can significantly reduce the usage of GPS and generate accurate tracking results. The design of SensTrack and evaluation presented above reveal several interesting challenges which remain for future work including resilient accelerometer data processing, tracking for multiple mobility patterns, and joint optimization of energy and accuracy.

References

[1] N. Patel, "The $10 b rule: Location, location, location," Strategy Anal., Newton, MA, USA, Tech. Rep., May 2011. AQ:1

[2] S. Robinson, "Cellphone energy gap: Desperately seeking solutions," Strategy Anal., Newton, MA, USA, Tech. Rep., Mar. 2009.

[3] I. Constandache, S. Gaonkar, M. Sayler, R. Choudhury, and L. Cox, "EnLoc: Energy-efficient localization for mobile phones," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2716–2720.

[4] Z. Zhuang, K. Kim, and J. Singh, "Improving energy efficiency of location sensing on smartphones," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 315–330.

[5] E. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications*. Norwood, MA, USA: Artech House, 2006.

[6] *Android Developers Reference: Location Strategies* [Online]. Available: http://developer.android.com/guide/topics/location/strategies.html AQ:2

[7] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 285–298.

[8] H. Lu, J. Yang, Z. Liu, N. Lane, T. Choudhury, and A. Campbell, "The Jigsaw continuous sensing engine for mobile phone applications," in *Proc. 8th ACM Conf. Embedded Netw. Sensor Syst.*, 2010, pp. 71–84.

[9] J. Paek, J. Kim, and R. Govindan, "Energy-efficient rate-adaptive GPS-based positioning for smartphones," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 299–314.

[10] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in *Proc. USENIX Conf. Annu. Tech. Conf.*, 2010, p. 21.

[11] M. Ra, J. Paek, A. Sharma, R. Govindan, M. Krieger, and M. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services*, 2010, pp. 255–270.

[12] M. Keally, G. Zhou, G. Xing, J. Wu, and A. Pyles, "PBN: Towards practical activity recognition using smartphone-based body sensor networks," in *Proc. 9th ACM Conf. Embedded Netw. Sensor Syst.*, 2011, pp. 246–259.

[13] R. Meng, J. Isenhower, C. Qin, and S. Nelakuditi, "Can smartphone sensors enhance kinect experience?" in *Proc. 13th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2012, pp. 265–266.

[14] Y. Chon, E. Talipov, H. Shin, and H. Cha, "Mobility prediction-based smartphone energy optimization for everyday location monitoring," in *Proc. 9th ACM Conf. Embedded Netw. Sensor Syst.*, 2011, pp. 82–95.

[15] I. Akyildiz, J. Ho, and Y. Lin, "Movement-based location update and selective paging for PCS networks," *IEEE/ACM Trans. Netw.*, vol. 4, no. 4, pp. 629–638, Aug. 1996.

[16] *My Tracks* [Online]. Available: http://code.google.com/p/mytracks/

[17] *Case Design Guidelines for Apple Devices* [Online]. Available: https://developer.apple.com/resources/cases/Case-Design-Guidelines.pdf

[18] *Android Developers Reference: Location and Sensors* [Online]. Available: http://developer.android.com/guide/topics/sensors/index.html

[19] (2012). *Android Analyzer Report* [Online]. Available: http://androidfragmentation.com/database/l/manufacturer/samsung/deviceModel/Nexus

[20] M. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, "Entracked: Energy-efficient robust position tracking for mobile devices," in *Proc. 7th Int. Conf. Mobile Syst., Appl., Services*, 2009, pp. 221–234.

[21] C. Rasmussen, "Gaussian processes in machine learning," *Adv. Lect. Mach. Learn.*, vol. 14, no. 2, pp. 63–71, Apr. 2004.

[22] L. Ma, J. Liu, L. Sun, and O. Karimi, "The trajectory exposure problem in location-aware mobile networking," in *Proc. IEEE 8th Int. Conf. MASS*, Oct. 2011, pp. 7–12.

[23] A. Cavanaugh, M. Lowe, D. Cyganski, and R. Duckworth, "WPI precision personnel location system: Rapid deployment antenna system and sensor fusion for 3D precision location," in *Proc. Int. Tech. Meeting Inst. Navigat.*, Jan. 2010, pp. 214–219.

[24] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proc. 10th Int. Conf. Ubiquitous Comput.*, Sep. 2008, pp. 114–123.

Lei Zhang (S'12) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2011. He is currently pursuing the master's degree at the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. His current research interests include cloud computing, wireless networks, and multimedia communications.

Jiangchuan Liu (S'01–M'03–SM'08) received the B.Eng. degree (*cum laude*) from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology, Hong Kong, in 2003, both in computer science. He is a recipient of the Microsoft Research Fellowship in 2000, the Hong Kong Young Scientist Award in 2003, and the Canada NSERC DAS Award in 2009. He is a co-recipient of the IEEE IWQoS in 2008 and the IEEE/ACM IWQoS in 2012 Best Student Paper Awards, the IEEE Communications Society Best Paper Award on Multimedia Communications in 2009, the IEEE Globecom Best Paper Award in 2011, and the ACM Multimedia Best Paper Award in 2012.

He is currently an Associate Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada, and was an Assistant Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, from 2003 to 2004. His current research interests include networking, in particular, multimedia communications, peer-to-peer networking, cloud computing, social networking, and wireless sensor/mesh networking. He is a member of Sigma Xi. He is an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA, and an Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS.

Hongbo Jiang (M'08) received the B.S. and M.S. degrees from the Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from Case Western Reserve University, Cleveland, OH, USA, in 2008.

He is an Associate Professor with the Faculty with the Huazhong University of Science and Technology. His current research interests include computer networking, especially algorithms and architectures for wireless and high-performance networks.

Yong Guan received the Ph.D. degree in computer science from the China University of Mining and Technology, Beijing, China, in 2004. He is currently a Professor with Capital Normal University, Beijing. His current research interests include formal verification, PHM for power, and embedded system design. He is a member of the Chinese Institute of Electronics Embedded Expert Committee and the Beijing Institute of Electronics Professional Education Committee, and Standing Council Member of the Beijing Society for Information Technology in Agriculture.

AUTHOR QUERIES

AQ:1 = Please provide the report no. for ref. [1]–[2].
AQ:2 = Please provide the year for ref. [6], [16]–[18].