Computer Communications 50 (2014) 29–40

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Energy-efficient automatic location-triggered applications on smartphones

Yemao Man [a], Edith C.-H. Ngai [b,*]

[a] Department of Shipping and Marine Technology, Chalmers University of Technology, Sweden
[b] Department of Information Technology, Uppsala University, Sweden

ARTICLE INFO

Article history:
Available online 29 March 2014

Keywords:
Energy-efficiency
Location-based services
Data collection
Mobile applications

ABSTRACT

With the prevalence of localization techniques in smartphones, location-based applications on mobiles have become increasingly popular. However, only minorities of applications can be triggered automatically by the predefined locations of interest without any human interaction. One reason is that the inevitable operation of location detection by GPS is power-intensive. While existing work has focused on energy efficiency in continuous location tracking, energy-efficient location detection for matching predefined location of interest remains to be further explored. This paper proposes a unified framework that supports energy-efficient location detection for automatic location-triggered applications. Our framework triggers desired events only when the user is approaching the predefined locations of interest. Besides the efforts we make to reduce the number of GPS updates by cooperating with other types of on-device sensors, the framework also aims to coordinate multiple location-triggered applications to further reduce energy consumption on location updates. We implemented our framework as a middleware in the Android operating system and conducted extensive real experiments. The experimental results demonstrate that our framework can reduce the number of GPS requests and low the energy consumption of the smartphones significantly.

1. Introduction

With the popularity of mobile Internet, people have exploded intelligent mobile services and innovative applications on smartphones. The convenience of the handheld equipment and powerful functions of the sensors on device also brings the new era of the localization applications in our daily life. The usages of location-based applications (LBAs) are generally categorized into venues check-in [1], social connection [2] and local information exploration [3,4]. Several applications are trying to combine localization, mobility and social network into one integrated service solution [5]. Nevertheless, the increasing computation capability of smartphones has drastically reduced the lifetimes of the devices. Designing mobile applications is no longer just about perfecting its computing power, communication, and user interface, but about providing all these functions with less power. Most smartphones contain Bluetooth, Wi-Fi, and GPS radios inside, and in many instances these components operate simultaneously. The GPS radio, in particular, is a notorious battery killer. You can see the battery bar getting shorter as you run your navigation application. Development of energy-efficient LBAs is important to prolong the lifetime of smartphones and maintain a green network for a sustainable society.

There exists a few popular and successful location-trigger applications in either Android or iOS platform. The applications need information about a user's current position to provide better services, such as uploading data and fetching available service nearby. However, the power-intensive GPS sensor reduces the phone's battery life to less than nine hours [6,7]. In order to save energy, it turns out to be the user's responsibility to launch the application at location of interests. This will arouse problem if a user fails to start the application because of individual reasons or surrounded circumstance, such as in disaster scenarios like earthquake. Therefore our work tends to introduce a unified framework for designing energy-efficient and automatic location-triggered applications.

One motivation for us to research on location-triggered application design is that there are many LBAs in the market, but very few of them are based on automatic location detection to trigger actions. Most of the existing location based applications adopt naïve GPS updates. Other applications rely on WiFi and cell towers for localization, but they can only support urban areas that are fully covered by WiFi and cellular networks. For example, a very popular

* Corresponding author. Tel.: +46 70 167 9360.
E-mail addresses: yemao.man@chalmers.se (Y. Man), edith.ngai@it.uu.se (E.C.-H. Ngai).

http://dx.doi.org/10.1016/j.comcom.2014.03.023
0140-3664/© 2014 Elsevier B.V. All rights reserved.

location-triggered application, called Llama [8], supports users only in populated areas with full cellular tower coverage for localization. Nowadays, almost every smartphone has GPS module, but it has huge energy concern. Can we support location-triggered tasks energy-efficiently with on-device GPS when people are in the suburbs? This motivates us to focus more on the GPS performance and its energy saving issue in this paper.

Considering the diversified information and possible services associated with locations, it is crucial to be able to trigger services automatically on the mobile phones with minimum human interference. To achieve this, the mobile application needs to maintain and keep track of the actions that it wants to perform at predefined locations. Location-triggered applications for smartphones ought to maintain database for storing locations of interest and their associated actions. We need an energy-efficient and adaptive scheme when coming to using GPS to detect if the device is at any of the locations of interest, and such framework should be independent from the specific business needs, i.e., different customized actions from different applications we see in the markets.

In addition, energy efficiency is a major concern for mobile devices. Although many approaches that are only related to the smartphones' intrinsic properties have been proposed to trade-off accuracy with energy consumption, like using accelerometer to estimate movements [9,10], the calculations are not targeted for locating predefined locations as destinations when scheduling the sensors in mobile phones. This motivates us to delve deeper in energy efficiency for mobile location-trigger applications. By taking the advantage of geometrical composition of destinations in the map, we investigate intelligent sensor scheduling to duty-cycle GPS for saving energy.

Another inevitable motivation for the research comes from the fact that almost no application keeps its eye on the power-intensive sensing behaviors from other simultaneously running applications on the same device. Without a doubt, it will arouse the problem of redundant GPS request, when one application requests location update with GPS in its own business flow and only 10 s later, another application requests location update with GPS again. Obviously the user is still at the same place but the expensive price for the second GPS calling has been paid. From a business implementation perspective, it is not necessary and expensive to monitor other applications' behavior. Programmers from different companies need only focus on their individual business and the users will just need to set up their private preferences. Therefore we need to investigate how to design the framework as one middleware to support multiple location-triggered applications installed on the same phone to benefit from each other for energy concern.

Our work in this paper makes the following contributions: (1) we present a unified framework to support location-triggered service by maintaining the geo-fenced areas of interest with associated actions. (2) We propose an energy-efficient approach that intelligently duty-cycles GPS to detect locations. The principle is to trade-off accuracy for energy efficiency by location estimations and multiple sensors scheduling. (3) We extend the framework further to coordinate multiple location-triggered applications with different business flows to achieve the same energy efficiency goal. (4) We implemented our proposed framework on Android smartphones for experiments in real-world applications. The results demonstrate that the proposed approach used in location-triggered scenarios could greatly reduce the usage of GPS for location updates.

2. Related work

Localization and trajectory tracking have attracted much research attention for location-based services for mobile devices. The simplest implementation is for the mobile devices to fetch the GPS readings and report them to the server. The request of GPS readings can be done periodically by the clients or upon request from the server [11]. Unfortunately, making frequent GPS requests can consume a lot of energy on the mobiles. Different from existing research on distance calculation for moving object and target [12,13], our work takes the advantage of utilizing alternative and more light-weight sensors to perform localization. It also considers the coordination among multiple location-triggered applications.

Technologies like Wi-Fi, cell-tower triangulation [14], Bluetooth [10] have been considered as alternatives to GPS when it comes to designing proactive energy-efficient LBS. WiFi and GSM can last as long as 40 and 60 h [15] compared to GPS request. Such solutions usually have clear constraints that the network should be available all the time. Dhondge et al. [16] proposes the system that facilitates mobile devices with estimated locations using WiFi in cooperation with a few available GPS broadcasting devices. SensLoc [17] also investigated localization techniques based on WiFi and GSM to improve battery life in expense of localization accuracy. Bareth [18] researches location-triggered concepts as we do by proposing the reverse cellular positioning architecture and WiFi to detect fine-grained target areas. Even though it is the closest research to our work from the perspective of spatial relation between a user and a target, the system cannot work without the support from cell tower.

Recently the newly developed "proximity beacons" from Qualcomm [19] and iBeacon from Apple [20] have drawn enormous attention due to their energy-efficient solutions that are based on Bluetooth. However, compared with the existing location-based applications, there are still barriers to make the beacon technology widely adopted today. Firstly, the beacon technology is specifically designed for indoor micro-location monitoring scenarios, such as in a big shopping mall. It requires installation of extra hardware (e.g. the "proximity beacons" from Qualcomm) to be installed at predefined locations. They are usually being attached to the walls or ceilings to transmit signals via Bluetooth connection. When Bob wants to be reminded to buy milk when passing by a small grocery store, or to fetch his ironed suit when passing by another small local clothing store on his way home, he must make sure his places of interest all have beacons installed. In such common scenarios, beacon applications are not only less convenient than most LBAs on the end-user devices, but also face the practical hurdles for wide adoption. Secondly, even some specific iOS devices that support Bluetooth Low Energy can be turned into an iBeacon [20], they always requires the user to turn on Bluetooth and establish connection with the beacon through a specified application. It might decrease the usability in many scenarios. In addition, the security-oriented approach does not necessarily protect the privacy of users. For example, the user may expose his physical locations to beacons of external parties, while his locations could actually be obtained by the GPS on his mobile phone privately. Thirdly, existing beacon applications are generally not supporting Android. A study from International Data Corporation (IDC) discovered that Android dominates 81% of world smartphone market during the third quarter of 2013 [21]. With the popularity of Android smartphones, there is a gap for beacon technology to bridge as it has been developed mainly based on iOS. For example, the proximity features from Qualcomm are only available on iOS [19]. Similarly, the iBeacon technology is only available on specific iOS devices [20].

Mobility studies of the mobile users have also been introduced to meet the needs of proactive position tracking to save energy. Ryder et al. concentrated on contextual information collection and mobility prediction using decision model [6]. Chon et al. [22] proposed a Markov decision process-based adaptive duty cycling scheme to user's mobility. The mobility studies are also combining the technologies such as radio beacons to reach higher precision percentage [23]. Adaptive estimation and prediction for movements has been suggested using on-device sensors [24], primarily accelerometer [9,25]. Other approaches include those considering location-time history [10], constructing mobility tree [7], and utilizing the adaption technique [26]. Nevertheless, the above work has been focusing on continuous location tracking for the mobile user. Location-based services that are triggered automatically by the areas of interest remain to be further explored.

Energy efficiency has also been investigated from other technical angles such as monitor the battery level [27] and provide adaptive GPS-positioning [10]. Kjaergaard et al. [28] proposed the joint trajectory and position tracking to help the device to decide when to perform sampling. It uses the prescribed error threshold to decide how long the GPS sensor should sleep. Jurdak et al. [29,30] used empirical GPS and radio contact data to model node mobility and duty cycle strategies. Similar to our work, Kjaergaard et al. [13] also investigated on-device sensor management strategy to reduce the number of GPS updates. Nevertheless, none of the above work has considered coordination among multiple location-triggered processes. On the contrary, our framework can schedule the GPS and other on-device sensors by coordinating the requests from multiple processes for saving more energy.

In terms of application-oriented design, Cai et al. [31] presents the design, analysis, and implementation of a specific real-time location-based supported system. Similar to the geo-fenced area we define in our work, the real-time location-based supported system allows each client to define a capable zone. We bind specific actions to different locations to generate the associate policy pair. Analogously, each zone is associated with each piece of information. It does not require periodic location updates from mobile. When a client moves, it monitors movement and requests for location update only when its capable zone is no longer contained by its resident domain. Different from existing work, we focus on location detection with reference to the area of interests for the users rather than merely positioning a mobile device. Our approach supports automatically location-triggered services without any user interference, with the usability keeping in mind.

A layer between the GPS module and the applications could be built to coordinate the power-intensive sensing requests. In this approach, a smartphone can avoid unnecessary registration and calling operation toward the GPS module in the hardware level. However, the techniques for multiple location-triggered processes to coordinate their location requests have barely been discussed in previous research. Bareth [18] has only mentioned it in his future work. The solution from Zhenyun et al. [27] oriented towards conventional LBA synchronization based on user-triggered scenarios rather than location-triggered scenarios. In the latter case, the location-triggered applications need to focus on the distances between the current location and the interested locations. Unfortunately, their solution in user-triggered scenarios was not designed for the location-triggered features. Different from the above work, our work focuses on location-triggered services. Another difference of our approaches exists in the deployment and implementation of the services. In their solution, users must wipe the original Android and upgrade to their ROM, which brings hurdles to the Android users in practice.

The issue of redundant sensing among multiple applications has recently been explored in the newest version of Android 4.4 KitKat, which introduces a new platform to support hardware sensor batching [32]. However, the implementation of sensor batching is closely linked to the hardware, so the function is currently only available in Nexus 5. In contrast, we design and implement a generic middleware solution that is applicable for a wide range of Android smartphones in this work.

3. Energy efficiency in location-based services

3.1. Emergence of auto-triggered LBS

Most of the existing location-based services are activated by the users when they are interested in the data provided by the corresponding applications. The user-triggered application scheme is a compromise between energy consumption and users friendliness, because the user has to know exactly when to turn on GPS to get service based on the location, so it can avoid excessive requesting readings from the power-intensive GPS. The users may lose the accessibility of data if they fail to turn on the applications, especially when it comes to the scenarios of natural disasters.

Confronted with such limitation from user-triggered LBS, auto-triggered LBS emerge with their intrinsic characteristics of demanding less human interaction and higher intelligence. It gives birth to the novel location-triggered applications in the markets. Location-triggered applications bring better user experiences by performing actions automatically when reaching predefined locations. It notifies the user relevant business events so the user will be involved in the business interactive flows as shown in Fig. 1.

3.2. Energy efficiency strategies of location-triggered applications

Nevertheless, there are still many challenges to provide ubiquitous energy-efficient automatically location-triggered service. First we need information about a user's current position in the application to activate further service. Although constantly and periodically requesting GPS updates can guarantee the accuracy of detecting the user's location, it inevitably brings severe battery draining problem caused by the power-intensive GPS sensor. While using WiFi and Cell-Id can save more energy than GPS, they have the limitation for positioning techniques, such as they cannot work when the user is moving in less populated area without WiFi. Besides, they might risk missing triggering events when user are near the predefined triggered area, considering the accuracy loss could be up to 50 m for WiFi and 5 km for Cell-Id [33]. Admittedly GPS has the indoor usage limitation but when it comes to the usual outdoor location-triggered application scenarios, like "Remind me to do something when I pass by that area", the accuracy loss of GPS positioning (up to 10 m) is the most important factor to guarantee the validity of such location-triggered software, though it can be done through different ways.

In order to have an overview of the current energy efficient strategies in the markets, we explore Google Play Markets to extract the most representative location-triggered LBS in Table 1. However, we could only explore their mechanisms by running their applications rather than obtaining and reading their codes for analysis. Due to the concurrent activities on the smartphone (e.g. ringing the alarm, uploading information to remote servers, etc.), it could be hard to evaluate their energy efficiency accurately.

Strategies have been discussed to switch on and off the GPS adaptively, such as disconnecting the GPS while the user stays indoors, and reconnecting it when the user leaves the building [41]. A major weakness of the mainstream LBS is the lack of intelligent strategy for requesting the user's location. The problem is even worse for GPS location updates due to its higher power consumption. Nevertheless, GPS update is important when the other location providers (e.g. Wifi and cellular networks) are not available. The application should aim for requesting location only when the user's current actual location is really close to the predefined geo-fenced area to avoid unnecessary positioning. Although energy saving is important, over reducing location requests pose the risk of missing the targeted areas. From a usability perspective, the software should not require manual request for localization like

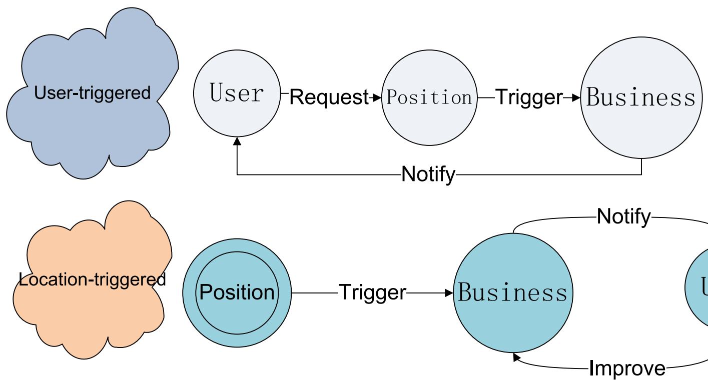

Fig. 1. User-triggered vs. location-triggered data access pattern.

Table 1
Published location-triggered Apps in Google Play Market.

Software	Main goals	Technology	Limitation	Strategy	Review
Llama [8]	Change the user's ringer and ringtones	Cell-Id	The validity depends on how many phone masts in user's area	Avoid using GPS at all	4.7 points of 10,105 reviews
Incoming GPS tracker [34]	Location tracking	GPS	Indoor	Periodic updates	4.3 points of 223 reviews
Call, GPS, SMS tracker [35]	Location tracking	GPS	Indoor	Updates every 30 min	4.2 points of 2909 reviews
Location alert [36]	Location-based alert	GPS	Indoor	Estimate location based on user's distance, speed and time	3.1 points of 171 reviews
Location aware lite [37]	Location-based alert	GPS	Indoor	N/A	2.3 points of 6 review
Location-aware wallpaper [38]	Wallpaper changing	GSM/WiFi and GPS	N/A	N/A	5 points of 3 users
Location aware pro [39]	Ringtones changing	GPS	N/A	N/A	5 points of 2 users
Lawn: location aware notes [40]	Location-based alert	GPS	N/A	N/A	No rates at all

we do in Foursquare [5] today. In an optimal situation concerning its efficiency of usability – "almost psychic" [42], an intelligent location-triggered application, can anticipate what the user may need it to do, in order to request location data and decide on which event to be triggered.

Although the location-triggered Apps are more convenient to use in relation to macro-location awareness, they still face challenges concerning the energy consumption. With the growing tendency of intelligent location-triggered application on smartphones, a user might run multiple location-triggered applications simultaneously. Another key finding is that none of the launched location triggered Apps care about sensing behaviors from another App. One possible reason is that the API to request GPS update is directly called by multiple Apps to meet their needs, but there is no layer between the GPS module and multiple applications that can coordinate all the power-intensive sensing requests. Moreover, different Apps usually adopt different business flows, which make it harder to coordinate these business-dependent calls. Along with the strategic lack we describe above, we believe that it is valuable to research on collaborative location update strategy especially involving GPS with other on-board sensors to achieve the energy-efficient goal in a unified cross-process framework.

3.3. Overview of location-triggered services

We give a microscopic view of automatic triggered services from the perspective of triggering factors and triggered elements on the smartphones in Fig. 2.

The location-triggered application is a special instance of all auto-triggered services in mobile applications. The target locations are defined as circular *geo-fenced areas*, which are associated with certain actions defined by the user in the application. Fig. 3 illustrates an example of geo-fenced area with coordinates at the center. The latitude and longitude of the target location are 59.8403 and 17.6487, while the radius of the geo-fenced area is 200 m. In real life, this location of interest may be a small local shop or restaurant. Mobile devices will compare the user's current location with any predefined geo-fenced areas dynamically, thus different

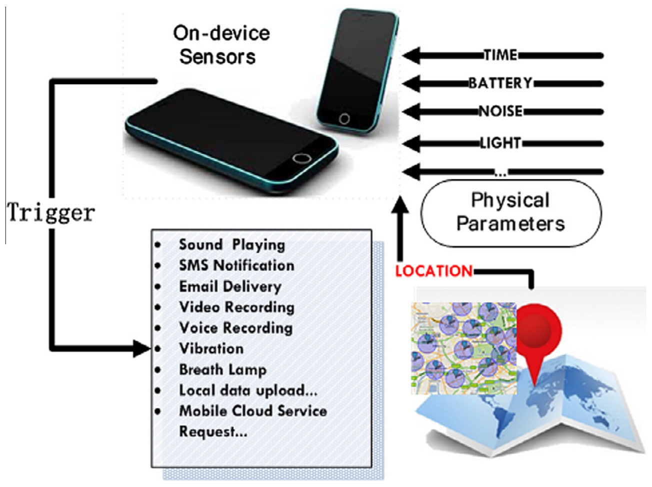

Fig. 2. Auto-triggered services in mobile applications.

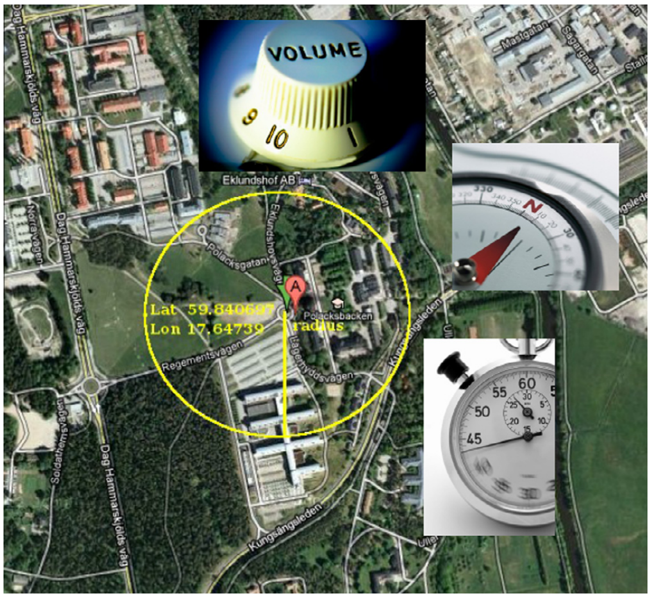

Fig. 3. An example showing an geo-fenced area in Uppsala, Sweden.

actions will be triggered according to the spatial relationship between the user and such geo-fenced areas.

The geo-fenced locations are stored in XML format, which provides interoperability to easily maintain the location database with simple operation interface. We define triggering policy that illustrates what actions to be triggered to initialize data collection when detecting the mobile device is inside the corresponding area as one policy tuple, $P = \langle \text{Location, Action} \rangle$. Each action has a unique identifier. For the possibility that multiple actions to be triggered in one area, we need to use action list to indicate the identifiers of the invoked actions. This allows us to compose one unique policy for every geo-fenced area regardless the number of actions. Similar to the locations, the policies are stored in XML format.

We have explained the reason why we chose GPS as location provider in pervious sections. Suppose we set T_{int} as one GPS fixed update interval. Fig. 4 shows the periodic location update workflow.

One problem of this strategy is that it risks draining the battery of a mobile device very quickly by setting a short interval for using GPS sensor too frequently, or missing the geo-fenced area the other way around.

4. Energy-efficient localization strategy

Location-triggered applications rely on GPS updates to detect the user's locations and activate the corresponding actions in the area of interests. The less sleeping time we set for the GPS module, the more accurate positions we will get. Thus the interval time T_{int} can decide how robust the system is on achieving relevant data. Our optimal goal is to avoid missing any targets using the least GPS updates on the smartphone. Therefore the time interval between GPS requests is extremely important as it decides the frequency of the power-intensive sensing operation. That is also the reason for us to investigate on estimating the dynamic time interval T_{int} between location updates in the location triggered applications.

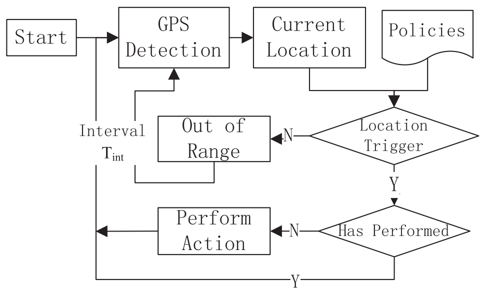

Fig. 4. Periodic location update workflow.

4.1. GPS update interval estimation

We model the location and movement of users as follows. We consider the user at point a moving along the vector l at a speed of v (m/s). The circle centered at point b with radius r is the closest geo-fenced area to the user. The distance between a and b is d and $r < d$. The angle α denotes the bearing between vector l and the shortest path between a and b, $0 \leqslant \alpha \leqslant \pi$. Fig. 5 illustrates the above descriptions as follows:

Suppose that the user is moving through the geo-fenced area, which means the elongation line of vector l will pass through the circle at point D first, i.e. $\alpha \leqslant \arcsin(\frac{r}{d})$. Based on the Law of Cosines (4.1), we easily obtain the length of AD in the triangle ABD. Let x be the length of AD. Given $r^2 = x^2 + d^2 - 2 \cdot d \cdot x \cdot \cos\alpha$, we can obtain x by

$$x = d\cos\alpha \pm \sqrt{r^2 - d^2\sin\alpha^2}$$

We estimate the time interval for location update by calling the GPS considering the following three cases.

Case I. When the condition $r^2 - d^2 \sin\alpha^2 \geqslant 0$ is met, we can take the minimal x to estimate the time t that the system will trigger the action once the user enters the area of interest.

$$t = \frac{d\cos\alpha - \sqrt{r^2 - d^2\sin\alpha^2}}{v}, \text{ given that } 0 < \alpha \leqslant arc\ \sin\frac{r}{d} \text{ and } r < d.$$

Case II. When the conditions $r^2 - d^2\sin\alpha^2 < 0$ and $0 < \alpha < \pi/2$ are met, the user is approaching the predefined geo-fenced area. However, his actual trajectory will not go through the area of interest as shown in Fig. 6.

Case III. When the conditions $r^2 - d^2\sin\alpha^2 < 0$ and $\frac{\pi}{2} \leqslant \alpha < \pi$ are met, the user is not heading to the geo-fenced area as shown in Fig. 7.

Based on the above cases, the earliest time t for the user to reach the geo-fenced area occurs in case I, where $t = \frac{d\cos\alpha - \sqrt{r^2 - d^2\sin\alpha^2}}{v}$, and v is his moving speed. The normal speed of walking is 5 and 15 km/h for running or cycling. If the current moving speed of the user is v, then it is unlikely that his maximum moving speed will become three times faster than v in the coming time interval. We can then estimate the next time interval to request the GPS reading by

$$\Delta T = \frac{d\cos\alpha - \sqrt{r^2 - d^2\sin\alpha^2}}{v'},$$

where $v' = 3v$ is the maximum moving speed of the user.

4.2. Location estimation with sensors other than GPS

Location updates using GPS can consume a lot of energy in the mobile phone. In order to reduce energy consumption, we plan to use alternative on-board sensors to estimate the device's location.

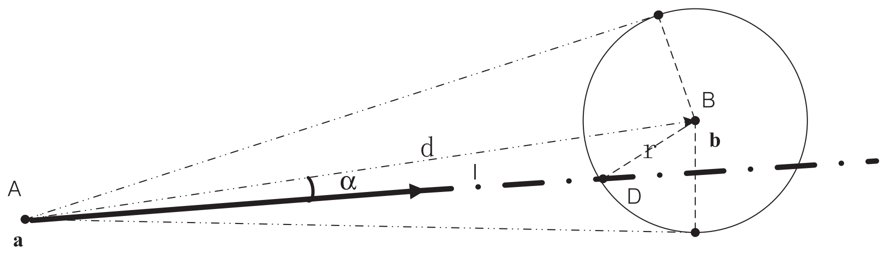

Fig. 5. The user is going to walking through the geo-fenced area.

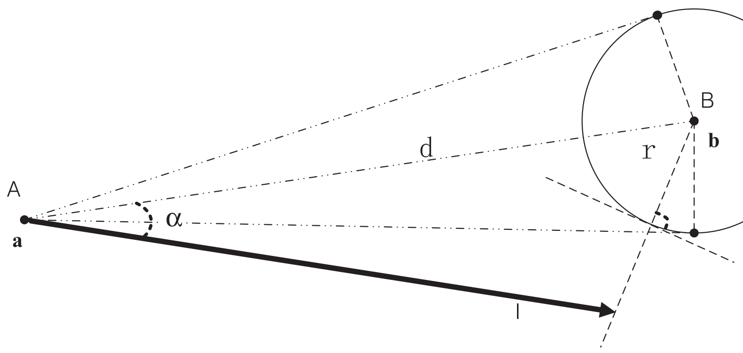

Fig. 6. The user is approaching but not passing through the geo-fenced area.

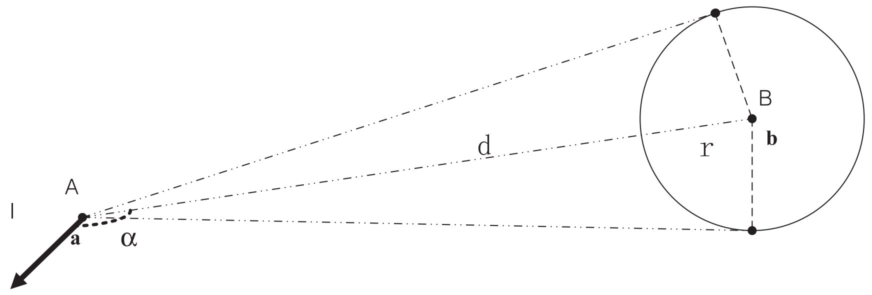

Fig. 7. The user is moving away from the geo-fenced area.

This includes utilizing the readings from the accelerometer and the magnetic field sensor on the smartphone in order to provide location estimations instead of performing GPS updates all the time.

We estimate the location of a user based on his moving direction and his latest GPS reading. We assume that a user keeps the same moving speed and direction during short period of time. The distance d that he moves during interval T_s will be $d = Ts \cdot v$. We can obtain the orientation φ of the user with the accelerometer and the magnetic field sensor on the smartphone, where $-\pi \leqslant \varphi \leqslant \pi$. For example, $\varphi = 0$ if the user is moving to the North. It is equal to $\pi/2$ and $-\pi/2$ for the East and the West.

Let (x,y) be the latitude and longitude from the latest GPS reading. We convert the reading to (μ,γ) in radian by $\mu = x \cdot \pi/180$ and $\gamma = y \cdot \pi/180$.

Given the radius of earth $R = 6378.1$ km, we calculate the new location (x',y') of the user after time T_s by

$$x' = sin^{-1}\left(\sin\mu\cos\frac{d}{R} + \cos\varphi \cdot \cos\mu\sin\frac{d}{R}\right) \cdot 180/\pi,$$

$$y' = \left(\gamma + \tan^{-1}\frac{\sin\varphi\sin\frac{d}{R}\cos\mu}{\cos\frac{d}{R} - \sin\mu\sin x'}\right) \cdot 180/\pi.$$

The estimation of the new location of a user does not necessarily require the calling of the GPS. Based on the newly estimated location, we can calculate the new time interval T_{int} for GPS updates. The estimation process involves both the accelerometer and the magnetic sensors to increase the speed awareness of the application. The process is repeated until the estimated area is close to any area of interests. In this design, the application should be able to trade accuracy for energy consumption as long as it does not miss any area of interests. Fig. 8 shows the sequence diagram of GPS requests and location estimations. Our approach can determine the GPS interval time T_{int} adaptively according to the moving speed and orientation of the user.

5. Coordination for multiple location-based processes

5.1. Coordination of GPS updates for multiple applications

Due to the growing tendency of similar location-triggered applications in smartphones, it is common to run multiple location-triggered applications on the same smartphone. The asynchronous callings GPS among multiple applications can cause redundant location requests and waste energy. Let us take a look at the motivational example. In Fig. 9, location-triggered application

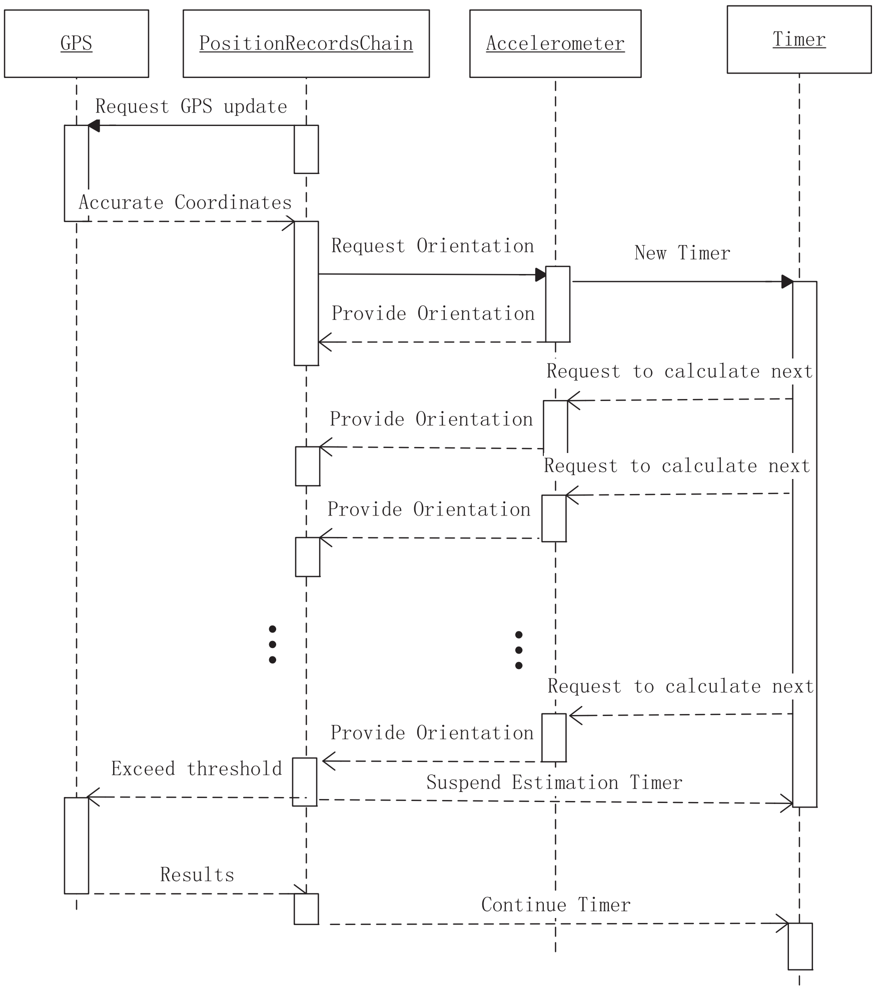

Fig. 8. Time sequence diagram to illustrate tasks among different sensing modules for location estimation.

AppA requests for update location at T_1. If the user launches another location-triggered application at T_2, the GPS request will be made again while the user is actually staying at the same place. The ideal update operations should occur only at T_1 and T_3. At time T_2 and T$_4$, the applications can use simply the most recent GPS readings. This can decrease expensive GPS update by 50% for energy saving.

We have programmed codes both on Android 2.1 and Android 4.1.4 to test how the system coordinates two GPS requests from different applications. The result further proves that the Android operating system coordinates only the GPS requests that occur simultaneously.

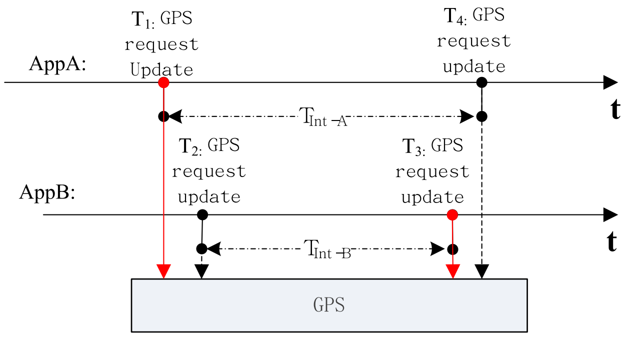

Fig. 9. Requesting GPS update from multiple location-triggered applications.

In order to avoid activating the GPS hardware unnecessarily, we intend to integrate the energy-efficient strategy in a unified middleware that can coordinate the GPS update operations and synchronize the access of location data of the smartphone for energy saving. Its core function is to get the requests from multiple applications in the upper layer in real time and decides the ideal time to request GPS update. The GPS readings will be shared among the applications to avoid making re Android only coordinates the GPS requests that occurs simultaneously redundant requests.

Fig. 10 shows the middleware design for coordinating multiple location-triggered applications on a smartphone. It includes an Intrinsic Data Generator (IDG) to estimate current location using the orientation and speed of users. The location estimation by IDG can be performed every time slot Δt. With the newly estimated location, each location-triggered application is responsible for fetching its own policies to estimate time interval T_{int} for making GPS requests. This will result in a series of T_{int} from different applications, i.e. $[T_{int1}, T_{int2}, \ldots]$.

We implement an Interval Decision Maker (IDM) to decide on when to call the GPS. The IDM can pick the smallest T_{int} to call the GPS. Note that the GPS will be requested immediately if any T_{int} is equal to zero. We also set a valid time λ for each GPS reading. After a GPS request, no new GPS update will be requested within this elapsed time λ. Instead, the applications will take the latest GPS reading. This mechanism is designed to prevent close GPS

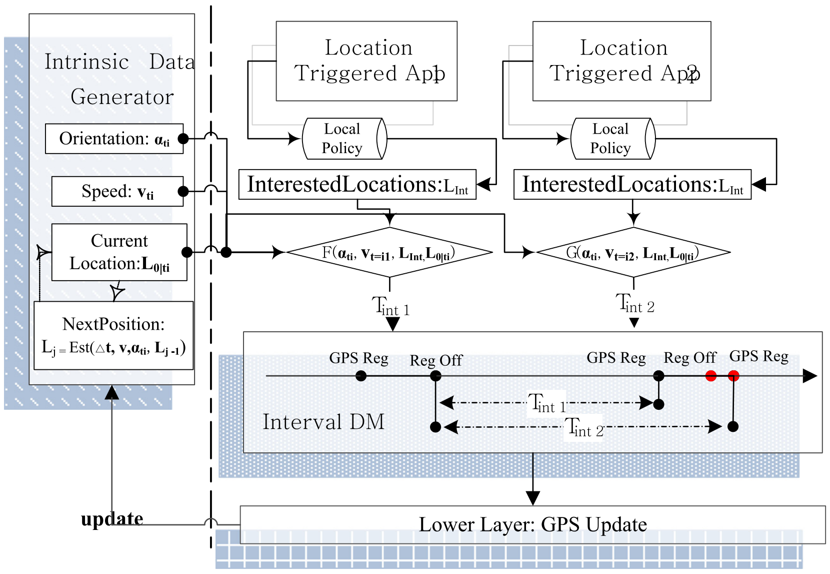
Intrinsic Data Generator
Orientation: α_{ti}
Speed: v_{ti}
Current Location: $L_{0|ti}$
NextPosition: $L_j = Est(\triangle t, v, \alpha_{ti}, L_{j-1})$
Location Triggered App1
Location Triggered App2
Local Policy
Local Policy
InterestedLocations: L_{Int}
InterestedLocations: L_{Int}
$F(\alpha_{ti}, v_{t=i1}, L_{Int}, L_{0|ti})$
$G(\alpha_{ti}, v_{t=i2}, L_{Int}, L_{0|ti})$
$T_{int\ 1}$
$T_{int\ 2}$
GPS Reg
Reg Off
GPS Reg
Reg Off
GPS Reg
Interval DM
update
Lower Layer: GPS Update

Fig. 10. Middleware design for coordinating multiple location-triggered applications on a smartphone.

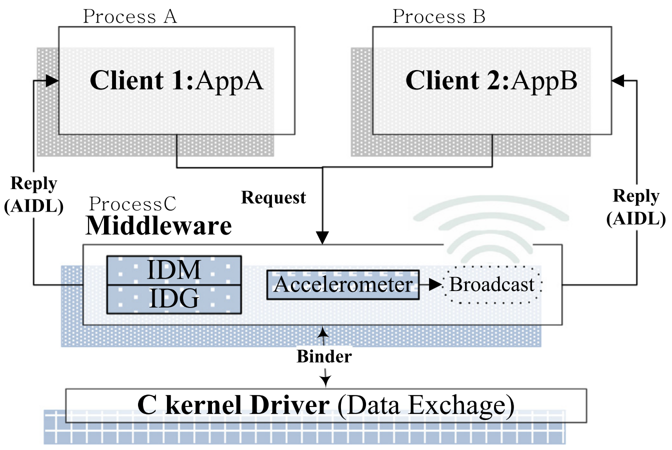

Fig. 11. Implementation of proposed framework in middleware.

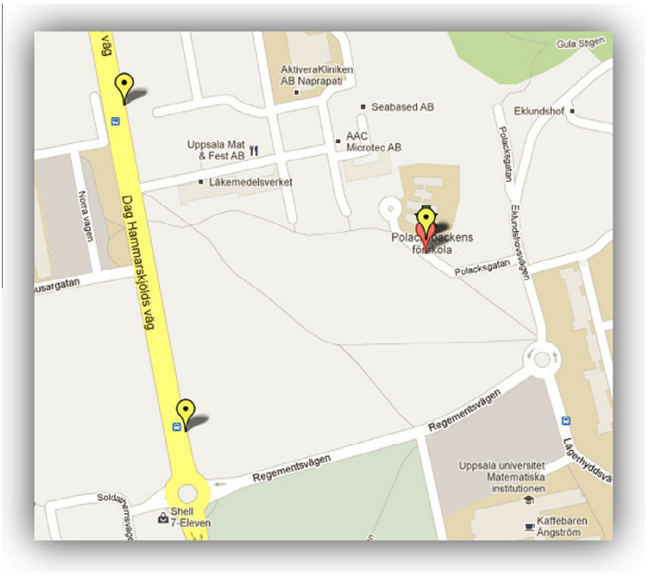

Fig. 12. App1 and PeriodicApp1 geo-fenced areas.

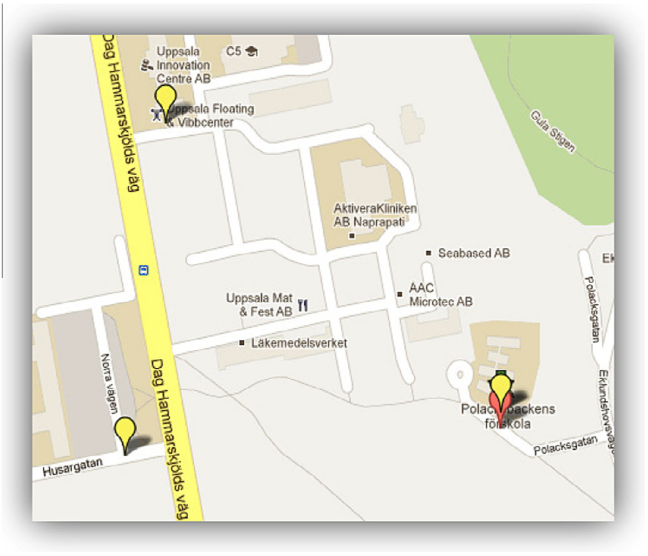

Fig. 13. App2 and PeriodicApp2 geo-fenced areas.

updates among multiple applications. In that case, multiple systems can benefit from each other without the actual energy-expensive operation.

5.2. Implementation of middleware in Android

The location-triggered applications apps are implemented as separated background services in Android. Instead of modifying the C++ libraries, we build the middleware above the Java Native Interface (JNI). It is easy to be imported and the users will not suffer from reinstalling firmware. The Binder Framework and intent broadcast are used for Inter-Process Communication (IPC). From Fig. 11, we can see how the data are transmitted between the middleware and multiple processes. Through the Binder Framework, the data can be exchanged in the C kernel driver.

Fig. 11 illustrates the operation of the middleware for two location-triggered applications, AppA and AppB. The two applications are running separately in process A and process B. Once they

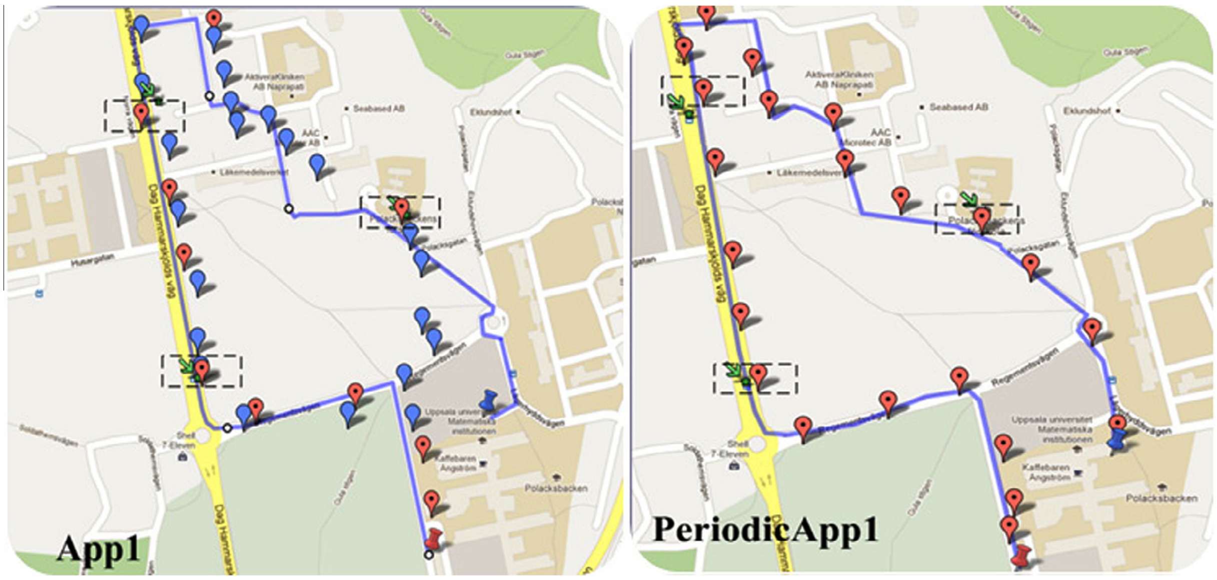

Fig. 14. GPS requests of location-triggered application App1 and PeriodicApp1 marked on the hiking track.

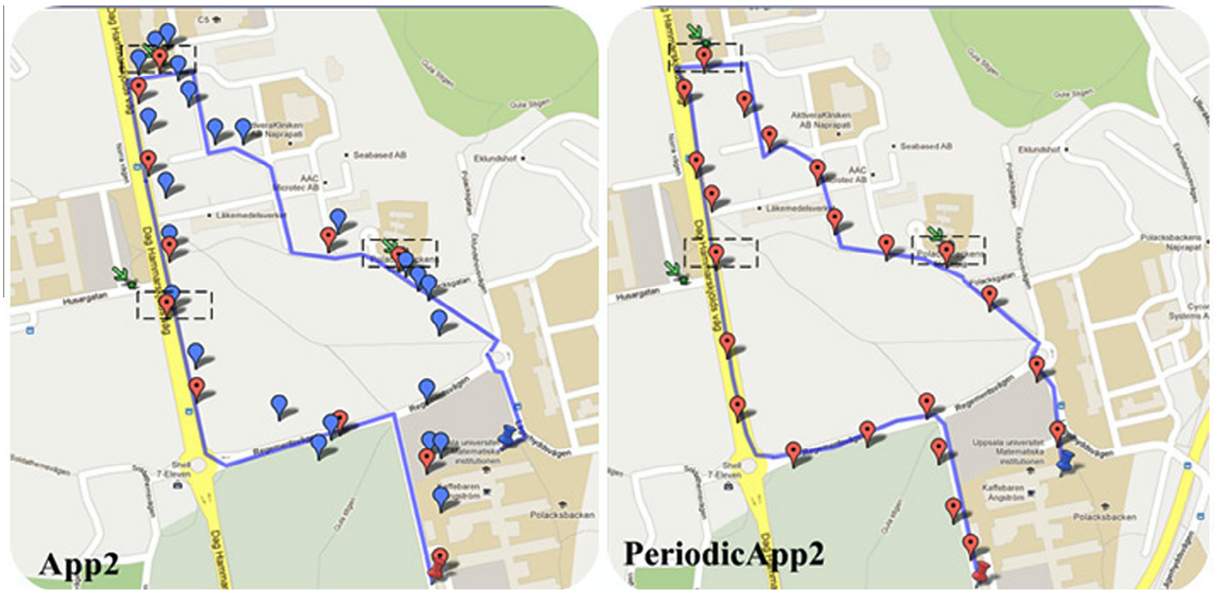

Fig. 15. GPS requests of location-triggered application App2 and PeriodicApp2 marked on the hiking track.

request for location updates, the messages will be sent to middleware. After location estimation, the reply from the middleware will be sent back through the Android Interface Definition Language (AIDL) interfaces. The middleware allows multiple location-triggered applications to synchronize the requests to the GPS and other on-device sensors.

6. Experiments

In the experiments we use Galaxy SIII i9300 that has 2100mAh battery as the test bed. In order to simulate daily scenarios and evaluate energy improvements we can gain by GPS updates, the user will hike in the area with mobile phones and trigger actions intentionally with the same trajectory. The model Galaxy SIII i9300 has been installed with the Android 4.1.4 code "Jelly Bean". Our goal is to evaluate the power intensive GPS update behaviors and actual energy consumption for both single and multiple location-triggered applications.

6.1. Case study: single application on Samsung Galaxy SIII

In this case study, we implement two location-triggered applications App1 (Bus Stop Discover) and App2 (UU Guider) in Android OS on two separate Samsung Galaxy SIII smartphone. The accelerometer cycle time is set to 60 s. App1 and App2 are in the Experiment Group.

We also implement a naïve fixed 60 s interval GPS update location-triggered application PeriodicApp1 as the control group compared with App1 and another 60 s periodic location-triggered application PeriodicApp2 as the control group compared with App2. PeriodicApp1 and PeriodicApp2 are taking location-triggered service pattern to fulfill the same business needs the same as App1 and App2 respectively. However, they do not use the sensor strategy and energy-efficient framework discussed in the paper

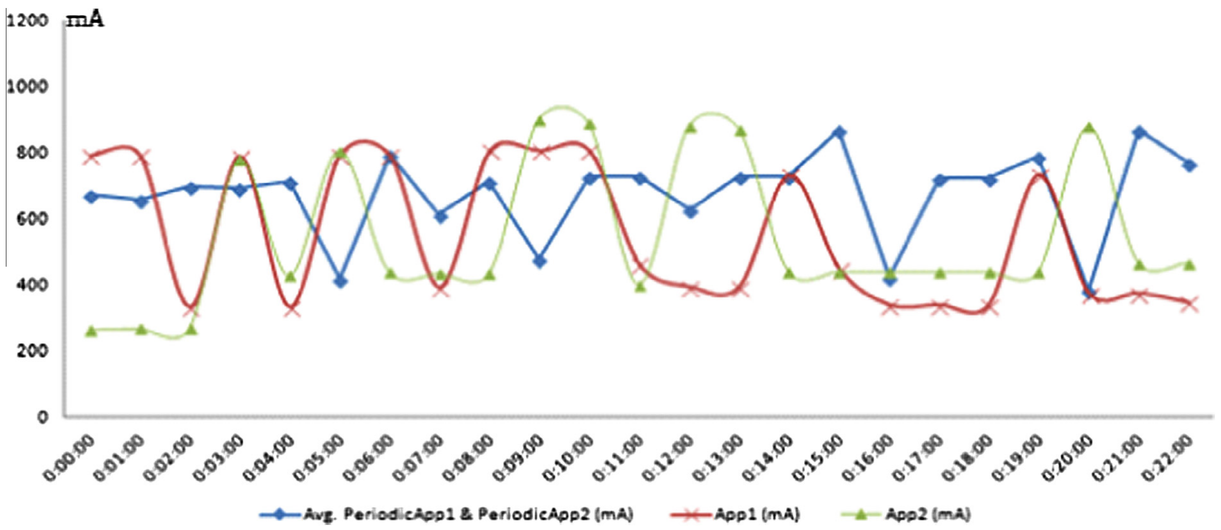

Fig. 16. Energy consumption of different applications.

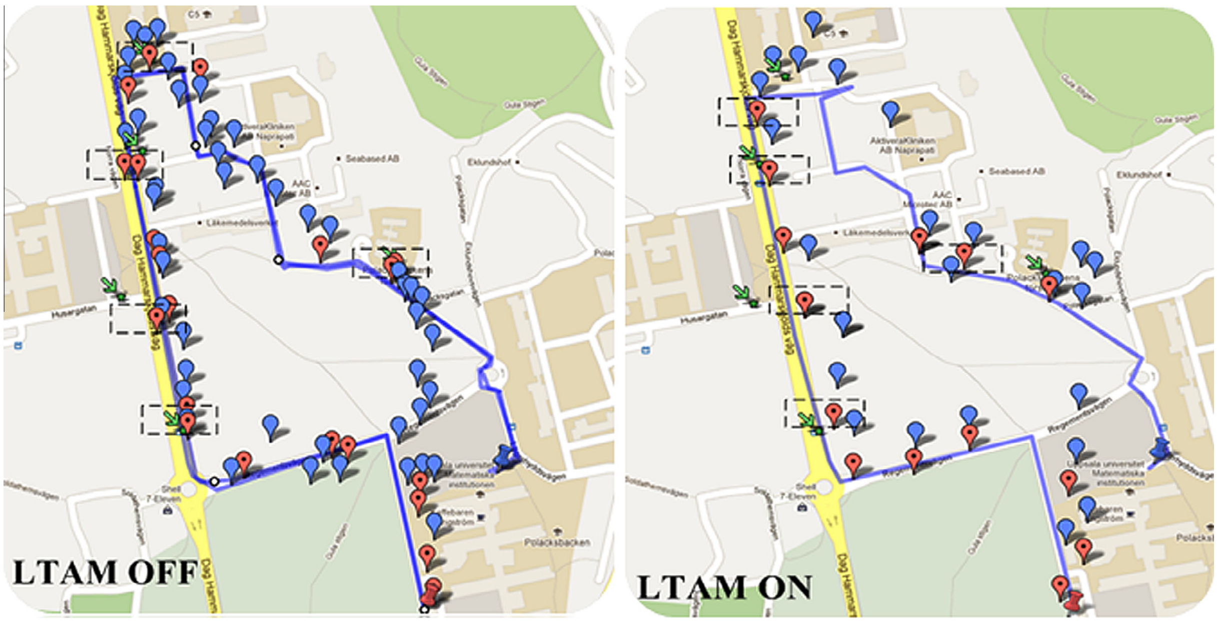

Fig. 17. GPS update places, estimated places and triggered events places without middleware and with middleware.

to obtain the adaptive interval. On the contrary, they only judge the user's location by periodical GPS update operations. The triggered area precision has been set to 50 m.

- App1 and PeriodicApp1 geo-fenced areas are set to:
  - (1) Grindstugan Bus Stop. Latitude: 59.85612, Longitude: 17.617285
  - (2) Uppsala Science Park Bus Stop. Latitude: 59.843327, Longitude: 17.638711

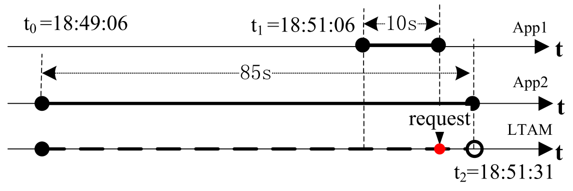

Fig. 18. GPS update places, estimated places and triggered events places without middleware and with middleware.

  - (3) Uppsala Polacksbacken Förskola, Latitude: 59.841942, Longitude: 17.644944. It is dot bubble on the right side in Fig. 12. Through anti-clockwise rotation based on (3), we can see the Position (2) and Position (1) are flagged with yellow dot bubbles.

- App2 and PeriodicApp2 geo-fenced areas are set to:
  - (1) BMC. Latitude: 59.841559, Longitude: 17.638206
  - (2) Uppsala BIO. Latitude: 59.844438, Longitude: 17.638936
  - (3) Uppsala Polacksbacken Förskola. Latitude: 59.841942, Longitude: 17.644944. It is the bubble without dot on the right side in Fig. 13. Through anti-clockwise rotation based on (3), we can see the Position (2) and Position (1) are flagged with yellow bubbles without dot.

In this scenario we have implemented App1 on one Galaxy SIII smartphone as the experiment group and PeriodicApp1 on the other Galaxy SIII smartphone as the control group, so is the similar deployment for App2 and PeriodicApp2. The mobile user carries the two mobile phones to move to make sure that the control phone experiences the same moving speed as the experiment phone. Our system has logged every GPS update location during the experiment hiking process so we will denote such places with red bubble with dots on the map. Besides, in the strategy of energy-efficient location-triggered applications, the system also automatically logged the estimated locations denoted by blue bubble without dots. We use the rectangle frame to denote the actual triggering action places and green arrow to denote the ideal triggering places.

In order to see how much energy we can save at least once adopting the adaptive location-triggered schema, we set the periodic interval to 60 s in the control group. This is the least GPS request frequency that we can maintain without missing any location of interests. As PeriodicApp1 and PeriodicApp2 have the exactly the same implementation based on the naïve fixed interval GPS update strategy, they both make 22 GPS updates. On the other hand, only 10 GPS updates are made for AppA (54.5% less) and 11 for AppB (50% less) in this 22minute test. Since the energy efficiency strategy for PeriodicApp1 and PeriodicApp2 is the same and use the same work flow but only different in geo-fenced areas, we take the average power consumption of ε (PeriodicApp1) and ε (PeriodicApp2) in battery monitor for their high similarity.

We can see the blue line, which stands for periodic update strategy are always fluctuating at a high level while the red and green lines, which stand for energy-efficient update strategy have dropped to lower level in the later half the experiment. Especially on the way to Uppsala Polacksbacken, the system in the optimized framework has done a good job to control the GPS update. This avoids using power intensive modules when it is not necessary.

Certainly, we have trade accuracy for energy saving. As we discover from the recorded map in Figs. 14 and 15, the green arrows that have been used to identify the ideal triggering places are not located in the exact same points as the actual triggering places. In this way we are comparing "the optimized reality" vs. "the perfect ideal". In the experiments, it turns out that the triggered events are either a little ahead of time or with a short delay; but still, they are all in the range of these geo-fenced areas. The key point is to guarantee that the traded accuracy loss will not jeopardize the triggering actions when the user is entering the area. The short delay of such location-triggered events is tolerable as long as we do not miss any targets.

The standard capacity specification for Galaxy SIII battery is 2100 mA h. We calculate the battery capacity consumed in the unit of milliampere-hour for periodic and adaptive GPS strategy based on the following procedure: First we record the average current per minute for the 22 min experiment time in Fig. 16, considering other unrelated background and battery monitor service consume 220–280 mA on the Galaxy SIII phones. Then we get the mean current by summing it up the average monitored current in these 22 min and later dividing by 22. That is 9800/22 mA. Therefore the total capacity consumed is approximately 9800/22 mA * 22/60 h, which is 163 mA h. This means, in the past 22 min, we have already run out one "battery" that can originally release stable current of 163 mA for 1 h theoretically. Considering the original battery capacity specification of Galaxy III 2100 mA h, we have already released 163/2100 of the total battery, which is about 7.7% of the battery amount. In physics, the unit for the battery capacity consumed is actually Coulomb(C) because 1mAh equals to 3.6C. So the consumed batter capacity is also equal to 3.6 * 163 mA h, which is 588C.

For App1 and App2, the calculation is the repeating the same process. We get the mean current 6700/22 mA. So the total capacity consumed is 6700/22 mA * 22/60 h, which is 112 mAh. The consumption is 5.3% of the total battery capacity. We calculate the ratio of the conservation amount of the battery by using (7.7–5.3)/7.7%. Hence, we save about 30% battery power in this experiment scenario.

We also evaluate the power consumption from the accelerometers and magnetic field sensors. We record the average current, being drawn by these sensors is merely around 30–70 mA. On the contrary, GPS updates draw much current, which is at least 400 mA in Fig. 16. Since we set the periodic time interval for accelerometer to 60 s, the benefit it creates far exceeds the cost it makes. Accelerometers and magnetic field sensors consume much less energy than GPS. Our results show that the actual energy consumption of the application actually goes down. Therefore, we show that optimizing the performance by using other low power consumption sensors is very important in our future work.

6.2. Case study: multiple application on Samsung Galaxy SIII

In this experiment, we evaluate the efficiency that has brought by the implementation of middleware to support multiple location-triggered applications on one smartphone.

For the management convenience of the middleware, we have developed it to Location-Triggered Application Manager (LTAM) so it has usable user interface for us to set up in this case study. App1 and App2 in the last case study have asynchronous requests and totally different business purposes. We turn on both App1 and App2 on the same phone with LTAM on so that the middle can coordinates their sensing behaviors to further save energy. We compare the performance with another phone with the deployment but LTAM off in the figure stated above (see fig. 17).

We find that GPS coordination works very well for multiple location-triggered apps. It successfully removes asynchronous orientation detection and reduces the total number of GPS updates from 21 to 14, which is 33% less than running both apps without coordination. To better illustrate how the GPS update operation is further optimized with the coordination from the middleware, we also design the real time logging system within such framework to get their event time stamp. One precise example would be the here in Fig. 18 below.

At time t_0 which is 18:49:06, App2 sends a GPS update interval to the low layer to intend to update location at t_2 which is 18:51:31. However, App1 calculates the dynamic update interval at t_1 which is 18:51:06. It intends to update the location in 10 s (the moment where the red spot stands). So LTAM adjusts the update moment for App2 to the red spot, 15 s earlier. It obviously avoids redundant GPS update behavior when the new GPS listener seems to be needed while the user is still at the place.

With LTAM the multiple business-independent location-triggered applications have drew the capacity of 138 mA h, just 20–24% higher than the energy consumption by single location-triggered application. Considering about 223 mA h capacity drew by running App1 and App2 separately without any coordination in the lower layer, the integrated application framework helps these two independent location-triggered applications App1 and App2 to save up to 38% energy.

7. Conclusions and future work

This paper presented a unified framework for energy-efficient location-triggered mobile applications. In our system, mobile clients maintain locations of interest with associated actions for triggering data collection activities. We proposed a scheduling algorithm to reduce energy consumption for detecting location of the mobile phone. With the assistance of other on-device sensors, our scheduling algorithm minimizes GPS updates through predicting the future locations of the mobile phone and its distance to the closest location of interest. We further extended the system to support GPS coordination among multiple location-triggered applications running on the same smartphone. We implemented the proposed framework in a middleware on Android and conducted extensive experiments to evaluate its energy consumption compare with the performance of periodic location update strategy and multiple running LBS situation. The experimental results showed that our algorithm could significantly reduce the number of GPS and reduce energy consumption on smartphones. In addition, the localization module in our framework can be easily extended to utilize WiFi and cellular tower to assist indoor localization. In the future work we will extend the location service to include localization using WiFi and cellular towers, etc. We also plan to further optimize the performance of location estimations by coordinating sensing activities of accelerometer and geomagnetic field sensor based on the real-time mobility and the historic moving trajectory of the users.

References

[1] JiePang android application, <https://play.google.com/store/apps/details?id=com.jiepang.android&hl=en>.

[2] Facebook android application, <https://play.google.com/store/apps/details?id=com.facebook.katana>.

[3] Dianping android application, <https://play.google.com/store/apps/details?id=com.dianping.v1&hl=en>.

[4] Google places, <http://www.google.com/places/>.

[5] Foursquare, <https://foursquare.com/about/new>.

[6] J. Ryder, B. Longstaff, S. Reddy, D. Estrin. Ambulation: a tool for monitoring mobility patterns over time using mobile phones, in: International Conference on Computational Science and Engineering, 2009, pp. 927–931.

[7] I. Constandache, S. Gaonkar, M. Sayler, R.R. Choudhury, L.P. Cox, EnLoc: energy-efficient localization for mobile phones, in: Proceedings of INFOCOM, 2009, pp. 2716–2720.

[8] Llama, <https://play.google.com/store/apps/details?id=com.kebab.Llama>. Access date: 2013-11-23.

[9] F. Abdesslem, A. Phillips, T. Henderson, Less is more: energy-efficient mobile sensing with senseless, in: Proceedings of ACM MobiHeld '09, Barcelona, Spain, 2009.

[10] J. Paek, J. Kim, R. Govindan, Energy-efficient rate-adaptive GPS-based positioning for smartphones, in: Proceedings of the 8th International Conference on Mobile systems, Applications, and Services, San Francisco, CA, USA, 2010.

[11] A. Küpper, Location-based Services – Fundamentals and Operation, John Wiley & Sons, 2005.

[12] Axel Küpper, G. Treu, Efficient proximity and separation detection among mobile targets for supporting location-based community services, Mobile Comput. Commun. Rev. 10 (3) (2006) 1–12.

[13] A. Leonhardi, K. Rothermel, Protocols for updating highly accurate location information, in: A. Behcet (Ed.), Geographic Location in the Internet, Kluwer Academic Publishers, 2002, pp. 111–141.

[14] K. Lin, A. Kansal, D. Lymberopoulos, F. Zhao, Energy-accuracy trade-off for continuous mobile device location, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, San Francisco, CA, USA, 2010.

[15] L. Wang, Y. Cui, I. Stojmenovic, X. Ma, J. Song, Energy efficiency on location based applications in mobile cloud computing: a survey, Springer Comput. (2013) 1–17.

[16] K. Dhondge, et al., Energy-Efficient cooperative opportunistic positioning for heterogeneous mobile devices, in: Proceedings of 21st International Conference on Computer Communications and Networks (Icccn), 2012.

[17] D.H. Kim, Y. Kim, D. Estrin, M.B. Srivastava, SensLoc: sensing everyday places and paths using less energy, in: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, Zürich, Switzerland, 2010.

[18] U. Bareth, Privacy-aware and energy-efficient geofencing through reverse cellular positioning, in: Proceedings of 8th International Wireless Communications and Mobile Computing Conference (Iwcmc), 2012, pp. 153–158.

[19] Gimbal, <https://gimbal.com>. Access date: 2014-03-13.

[20] iBeacon, <http://daveaddey.com/?p=1252>. Access date: 2014-03-13.

[21] IDC report, <http://www.idc.com/getdoc.jsp?containerId=prUS24442013>. Access date: 2014-03-13.

[22] Y. Chon, E. Talipov, H. Shin, H. Cha, Mobility prediction-based smartphone energy optimization for everyday location monitoring, in: Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems, Seattle, Washington, 2011.

[23] M.Y. Mun, Y.W. Seo, Everyday mobility context classification using radio beacons, in: Proceedings of IEEE Consumer Communications and Networking Conference (Ccnc), 2013, pp. 112–117.

[24] M.B. Kjærgaard, J. Langdal, T. Godsk, T. Toftkj, EnTracked: energy-efficient robust position tracking for mobile devices, in: Proceedings of the 7th International Conference on Mobile Systems, Applications, And Services, Kraków, Poland, 2009.

[25] T. Brezmes, J.L. Gorricho, J. Cotrina, Activity recognition from accelerometer data on a mobile phone, in: Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, Salamanca, Spain, 2009.

[26] H. Lu et al., The jigsaw continuous sensing engine for mobile phone applications, in: Proceedings of 8th ACM Conference on Embedded Networks Sensors Systems, SenSys'10, ACM, 2010, pp. 71–84.

[27] Z. Zhuang, K.-H. Kim, J.P. Singh, Improving energy efficiency of location sensing on smartphones, in: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, San Francisco, CA, USA, 2010.

[28] M.B. Kjærgaard, S. Bhattacharya, H. Blunck, P. Nurmi, Energy-efficient trajectory tracking for mobile devices, in: Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, Bethesda, MD, USA, 2011.

[29] R. Jurdak, P. Corke, D. Dharman, G. Salagnac, Adaptive GPS duty cycling and radio ranging for energy-efficient localization, in: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems, Zürich, Switzerland, 2010.

[30] R. Jurdak et al., Energy-efficient localization: GPS duty cycling with radio ranging, ACM Trans. Sensor Networks 9 (2) (2013) 23:1–23:33.

[31] Y. Cai, T. Xu, Design, analysis, and implementation of a large-scale real-time location-based information sharing system, in: Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services, 2008.

[32] Andlroid 4.4, <http://developer.android.com/about/versions/kitkat.html#44-hce>. Access date: 2014-03-13.

[33] Location Based Services: From Promise to Reality, <http://www.comsocscv.org/docs/Talk_091008_LBS.pdf>. Access date: 2014-03-13.

[34] Incoming GPS Tracker, <https://play.google.com/store/apps/details?id=iwuana.imcomingstd>. Access date: 2013-11-23.

[35] Call, GPS, SMS Tracker <https://play.google.com/store/apps/details?id=com.call_gps_sms_tracker>. Access date: 2013-11-23.

[36] Location alert, <https://play.google.com/store/apps/details?id=com.mofirst.locationalert>. Access date: 2013-11-23.

[37] Location aware lite, <https://play.google.com/store/apps/details?id=com.coldvapor.locationawarelite>. Access date: 2013-11-23.

[38] Location-Aware Wallpaper , <https://play.google.com/store/apps/details?id=com.ppypsoftware.locationwallpaper>. Access date: 2013-11-23.

[39] Location aware pro, <https://play.google.com/store/apps/details?id=com.locaware>. Access date: 2013-11-23.

[40] Lawn: Location Aware Notes, <https://play.google.com/store/apps/details?id=com.tglcoding.lawn>. Access date: 2013-11-23.

[41] L.M.S. Morino et al., Outdoor exit detection using combined techniques to increase GPS efficiency, Expert Syst. Appl. 39 (15) (2012) 12260–12267.

[42] B. Sherwood-Jones, Quality in use scoring scale. http://www.processforusability.co.uk/QIUSS/QIUSS.pdf. Access date: 2013-11-23, 2008.