A Transparent Correlation-Based Scheme for Energy Efficient Context Sensing and Fusion under Android Systems

Nicholas Capurso[1], Liran Ma[1], Tianyi Song[2], and Xiuzhen Cheng[2]

[1] Department of Computer Science, Texas Christian University, Fort Worth, TX, USA
[2] Department of Computer Science, The George Washington University, Washington, DC, USA

Abstract. A primary concern with modern smartphones is battery consumption. With so many different hardware components in modern smartphones, there are situations where certain components may be powered down or reduced in functionality without disrupting the user experience. We propose a transparent correlation-based scheme for energy efficient context sensing and fusion under Android systems. We experiment with the idea of disabling hardware functionality based on context. Our scheme focuses on inferring a user's location and subsequently disabling the GPS, which is considered to be one of the most energy-expensive components included in a smartphone. For example, when a user has connected to a Wi-Fi network with a known location, we disable GPS navigation and deliver the known location in its place. Based on our experiments, we conclude that this approach can significantly improve a device's battery life.

Keywords: Energy Efficiency, Android Systems, Context Sensing, GPS, Location, Wi-Fi.

1 Introduction

Modern smartphones are sophisticated enough that a GPS, a Wi-Fi and cellular radio, and a bank of sensors can all be packed onto the same device. As a result, one apparent issue that concerns all users is battery consumption. With so many different components, it is difficult for the average user to make the best decisions when it comes to saving power.

At many times, however, there are always certain components that can be safely powered down or reduced in functionality without disturbing the user's experience. For example, many users tend to leave Wi-Fi turned on, even when they are not using their phones. Thus, a simple solution would be to disable Wi-Fi while the phone is idle and the connection is not being utilized. It follows that similar logic can be applied to other smartphone components as well and overall, the battery life of a device would improve.

Typically, one of the most battery-intensive components on modern smartphones is the GPS. Today, applications that require location data pervade our society. Social networking applications often attach location data to status updates, weather applications use location data to fetch an accurate forecast, and, of course, there are mapping and navigational applications. Any application that requires prolonged use of the GPS results in the battery draining rather quickly. Many users rely on the functionality of navigational applications, but pay the price of a lower battery life throughout the day. In addition, there are times where the GPS may be unavailable or unreliable, such as in a building or underground. Thus, a smartphone's GPS is one of the most practical components to develop an energy-efficient solution for.

Ideally, one would like to minimize power consumption without sacrificing the accuracy or functionality of the GPS. An efficient way of providing location services to a user while using the GPS sparingly is by inferring a user's location based on context. For example, consider a user connected to their Wi-Fi network while at home. Because of the range of a typical user's Wi-Fi network, it can be assumed that the user's location will not change while they remain connected to their network. As a result, if the location of the user's home is known, the GPS does not need to be employed while the device remains connected to the user's home network. This specific case may also apply to a user's workplace or learning institution.

We propose such a scheme that attempts to infer a user's location based on context and reduce the use of a smartphone's GPS. Our solution is designed to disable GPS functionality while connected to a known Wi-Fi network with a location that has been pre-defined. It is also designed to be as transparent as possible such that already existing applications need not be modified and leaves the user unaware of whether their location is being inferred or obtained by the GPS.

This solution may be expanded to work with virtually any Wi-Fi networks, as locations can be looked up using online services. Further improvements may be made by incorporating logic and relationships to disable other smartphone components. For example, if Wi-Fi is not being utilized and the accelerometer is used to determine that a smartphone is idle, Wi-Fi can be safely turned off. In combination with our solution for disabling the GPS, we expect this approach would have significant battery savings. This approach mirrors those taken in [1], [2], and [3] to adjust or disable the GPS based on sensor readings, such as the accelerometer. We also perform a similar test using the accelerometer in order to verify its effectiveness.

Our main contributions are as follows: i) Propose a scheme to infer a user's location based on context and reduce the use of a smartphone's GPS; ii) Develop a solution that disables GPS functionality once a user's location can be inferred from Wi-Fi connection information; iii) Implement our proposed solution into the Android operating system transparently such that existing application code does not need to be rewritten; iv) Confirm that our proposed solution results in significant battery savings.

The rest of the paper is organized as follows. Section 2 lists related work. Section 3 discusses our development environment, information on the Android location framework, and relevant information about Wi-Fi. Section 4 gives the goals of our work and introduces our solution design. Section 5 discusses our testing environment and the results of our experiments. Finally, Section 6 concludes the paper.

2 Related Work

Reducing power consumption on mobile devices is an important and ongoing topic with many proposed approaches. Ideally, a solution should save energy while minimizing the harm done to the user experience. Proposed approaches include using middleware, trade-offs, minimizing GPS usage, or determining/inferring locations without the use of a GPS.

The development of middleware to reduce GPS power consumption is proposed by Zhuang, *et al.* in [4] and Nath in [5]. Zhuang, *et al.* proposes a middleware that uses context to determine whether it is more appropriate to invoke either a smartphone's GPS or a Wi-Fi/cellular location provider. Their solution also follows the idea of using low-power sensors to reduce unnecessary invocations of location services. In addition, they also dynamically adapt location sensing parameters according to a device's battery level and attempt to clump location requests together so as to increase the time interval between subsequent uses of the GPS. Nath proposes ACE, a middleware that defines context attributes (such as being at home, or jogging, etc) and relationships between them in order to derive their values efficiently. Energy efficiency is gained through attempting to use the least power-consuming components to derive an attribute. However, an issue that comes with middleware solutions, such as those proposed by Nath and Zhuang, *et al.*, is that existing applications must be modified to interface with the middleware.

Trading location accuracies for reduced power consumption is proposed by Lin, *et al.* in [6]. Their research recognizes that accuracy requirements may be relaxed based on context. Their work involves using various location sensors to determine locations based on energy cost and required accuracy.

Controlling a GPS to acquire or update a location when a device enters or leaves a region (crosses a boundary) is explored by Farrell, *et al.* in [7]. A similar concept is proposed by Paek, *et al.* in [2]. Their work focuses on controlling the GPS to only be turned on when accurate location readings may be obtained. This logic is based on the user's location, movement, and a user's history of visited locations and velocities obtained by a smartphone's sensors.

Attempting to minimize a device's power consumption based on a user's mobility is explored by You, *et al.* in [3] and by Kjærgaard, *et al.* in [1]. You, *et al.*'s research aims to adjust the rate at which the GPS is invoked to get a location fix (i.e., the sampling rate) based on a user's movements. Their research intends to reduce energy consumption of using location services and improve the location accuracy of mobile users. Kjærgaard, *et al.* proposes EnTracked, which also focuses on minimizing power consumption based on a user's mobility. EnTracked determines when to turn the GPS on or off and keeps track of whether the user is stationary or in motion.

The use of Wi-Fi to infer locations is researched by Youssef and Agrawala in [8]. Their work involves using Wi-Fi signal strength to track a user's location. Finally, VTrack, a system developed by Thiagarajan, *et al.* in [9] uses Wi-Fi, instead of the GPS, to estimate a vehicle's trajectory and travel time in the interest of reducing the amount of time that one spends in traffic.

Our work in this paper combines approaches to minimize GPS usage with inferring locations via Wi-Fi. As opposed to the middleware approach, where applications must be written to interface with the middleware, our solution is implemented transparently such that existing applications need not adapt to our modifications. We attempt to minimize GPS usage by disabling it when a user's location can be inferred from Wi-Fi connection information.

3 Background

First, we discuss our development environment and the capabilities of our prototype device. Then, we explain and illustrate the data flow in the Android location framework. Finally, we describe the features of Wi-Fi that are relevant to our work.

3.1 Development Tools

The Android Open Source Project (AOSP) provides for an environment where we can study the GPS operation and integrate our solution into the operating system itself. We develop our solution for a custom Android firmware called Cyanogenmod (CM) [10]. CM adds on various features that are not present in stock Android and thus, also provides us with greater flexibility when developing our solution. Our solution is written in Java and is developed with the Eclipse IDE and the Android SDK [11].

Our prototype device is the Samsung Galaxy S4. The device currently runs Cyanogenmod 10.2 with Android version 4.3.1 (Jelly Bean). The technical specifications of the device are as follows: a 1.9 GHz quad-core processor, 2 GB of RAM, and 16 GB of storage [12]. The Android location framework [13] allows a user's location to be derived from either the GPS or a network provider, such as Wi-Fi or the cellular network. However, our research focuses on the GPS as it typically consumes more power than the latter. The S4 also boasts a variety of hardware sensors, such as an accelerometer, which we may use in addition to Wi-Fi to make energy-saving decisions based on context. Finally, our prototype device lacks a SIM card, making cellular functions unavailable. As a result, power consumption is expected to be greater for a normal user's device that has cellular functions enabled.

3.2 Android Location Framework and Operation

To obtain information about the device's location, an Android application does the following. To declare its interest in receiving location information, an application registers a "listener" object with the operating system. When the device's location has been pinpointed by the GPS (i.e., a "fix" has been obtained), the Android operating system delivers this information to all registered listeners for applications to use.

Fig. 1 depicts a simplified view of the Android location framework. At the lowest layers, the Location Engine aids the GPS in obtaining a location fix. Upon obtaining a fix, the relevant information is passed up to the GpsLocationProvider (GLP) where it is all wrapped together into a single Location Object and passed to the LocationManagerService (LMS). The LMS is responsible for the delivery of Location Objects to interested listeners. The information contained in a Location Object includes a timestamp, a latitude, longitude, and an accuracy (a confidence radius, given in meters).

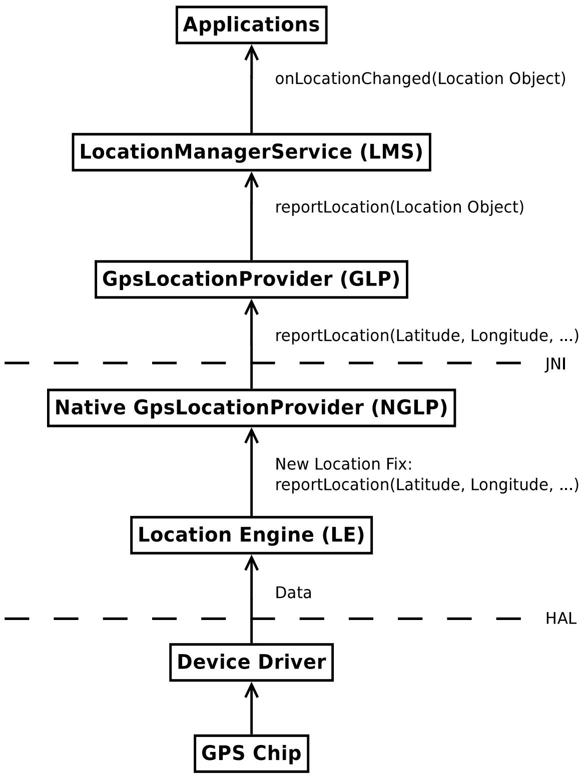
Applications
onLocationChanged(Location Object)
LocationManagerService (LMS)
reportLocation(Location Object)
GpsLocationProvider (GLP)
reportLocation(Latitude, Longitude, ...)
JNI
Native GpsLocationProvider (NGLP)
New Location Fix:
reportLocation(Latitude, Longitude, ...)
Location Engine (LE)
Data
HAL
Device Driver
GPS Chip

Fig. 1. Android Location Framework

The process of attempting to obtain a fix on a device's location is known as navigating (tracking) and contributes to the bulk of the power consumed when employing the GPS. This process runs continuously as long as there remains at least one registered listener for location information.

In the Settings application for Android smartphones, an option called "Location Services" (a.k.a. "Location Access") may be interacted with to manually enable or disable GPS navigating. Enabling or disabling Location Services causes the corresponding action to occur in the location framework - for example, disabling Location Services causes the LMS to disable the GLP which, in turn, stops the Location Engine from navigating. Whether or not the physical GPS chip is powered down is dependent upon implementation, however the energy cost to power the chip is negligible compared to the power consumed to determine a location.

3.3 Android Wi-Fi Operation

The actual operation of the Wi-Fi subsystem is not pertinent to our research. However, it is important for us to receive connection information such as the state of the connection, a network's SSID, and an access point (AP)'s MAC address. After a connection has been made to a Wi-Fi network, the Android operating system fires a broadcast containing connection information throughout the system. Thus, it suffices for our solution to listen for these specific broadcasts and make decisions based on the connection information.

One should also take into account that the average user may leave Wi-Fi on at all times, though it may or may not be connected to an AP. It follows that the total power cost combines the cost of using Wi-Fi with that of the GPS.

4 Design

The main goal of our solution is to reduce a device's energy consumption by analyzing the device's context and turning off unnecessary components, such as GPS, Wi-Fi, or sensors. Our solution must also be implemented in a seamless manner such that applications need not be modified to accommodate any changes.

To meet our power consumption goal, we first experiment with the idea of disabling a device's GPS when the device has connected to a Wi-Fi network with a location that has been previously user-defined or may be looked up using online services such as WiGLE [14], or another wardriving database. If this location is known ahead of time, the GPS may be turned off while the device remains connected to the network. The known location can just be pushed regularly to interested applications while the GPS remains unused.

Because control over the physical GPS chip is left to the device driver, directly powering off the GPS is not straightforward and an approach that works on one device may not work on another device with GPS hardware from a different manufacturer. Thus, an alternative approach to tackling the power consumption problem is to turn off GPS navigation, as discussed in Section 3.

A few considerations surround this approach. First, applications must not be notified that the GPS is unavailable as per the standard protocol when the user manually disables the Location Services option. Many applications are programmed to adapt to the user enabling and disabling Location Services, but in our case, we wish to disable GPS functionality and push a predetermined location in its stead. Second, a suitable location within the location framework must be chosen to implement our solution. To achieve a seamless integration with existing applications, we cannot require that application code be adapted to our changes to the location framework.

GPS navigation may be disabled as high as the GpsLocationProvider (GLP) level, shown in Fig. 1. By disabling the GLP, it calls downward to the Location Engine to stop navigating. It is favorable to make our modifications to higher levels in the location framework so as to maximize compatibility. The typical way to disable the GLP is to interact with an Android class called SettingsProvider, which manages changes to a user's settings, such as enabling or disabling Location Services. However, going through the SettingsProvider class to disable a given location provider causes applications to be notified that the provider is unavailable. Thus, to circumvent this, we create a class that directly interacts with the GLP to enable or disable it when needed. Because it is not the responsibility for a location provider to notify applications that it is disabled, applications still believe it is enabled even when we disable it. Our modification to the location framework is shown in Fig. 2.

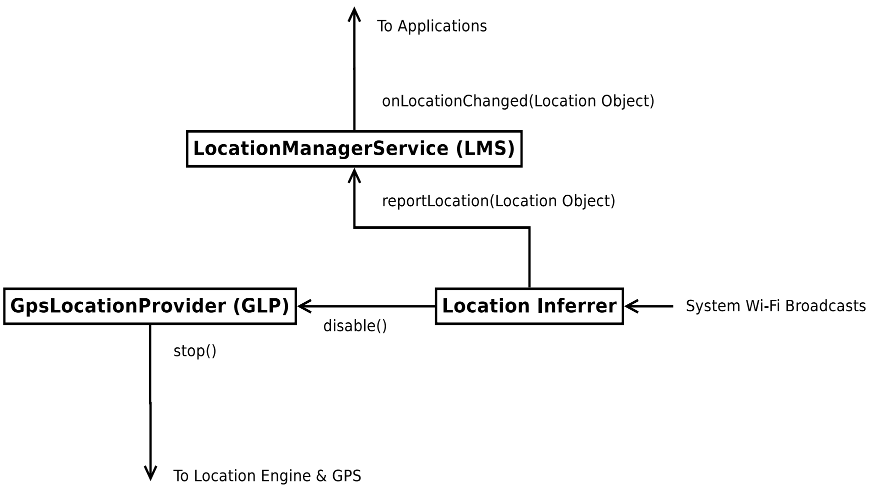
To Applications
onLocationChanged(Location Object)
LocationManagerService (LMS)
reportLocation(Location Object)
GpsLocationProvider (GLP)
disable()
Location Inferrer
System Wi-Fi Broadcasts
stop()
To Location Engine & GPS

Fig. 2. Proposed Location Framework Modifications

This class, which we refer to as the Location Inferrer (Inferrer, for short), is instantiated by the LocationManagerService (LMS) after the GLP has been instantiated. It receives Wi-Fi-related broadcasts and, if connected to a Wi-Fi network with a known location, will disable the GLP which, in turn, stops GPS navigating. While remaining connected to such a network, the Location Inferrer regularly sends the known location to the LMS which will deliver the location information to interested applications. Once disconnected from the network, the GLP will be enabled again and will enable GPS navigation, thus taking over from the Inferrer. By choosing to implement our solution this way and at this level, minimal changes are made to the location framework itself and our solution achieves transparency. The LMS still is responsible for delivering Location Objects to listeners and the Location Engine/driver are still responsible for obtaining a location fix when necessary. The operation of the Location Inferrer is shown in Fig. 3.

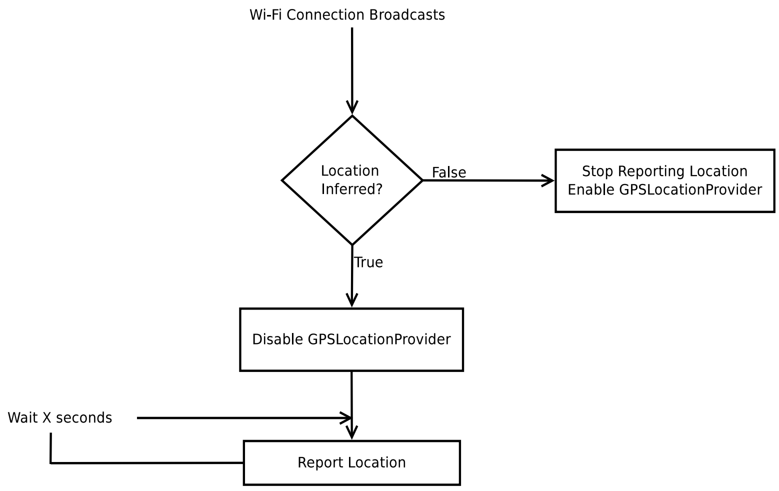
Wi-Fi Connection Broadcasts
Location Inferred?
False
Stop Reporting Location
Enable GPSLocationProvider
True
Disable GPSLocationProvider
Wait X seconds
Report Location

Fig. 3. Location Inferrer Flow Diagram

We expect this approach may result in power consumption to be close to that of Wi-Fi while the Location Inferrer is in use. It also should be noted that the interval on which the Location Inferrer will report a location can be adjusted to meet application requirements.

Aside from the GPS, we hypothesize other components may also be powered off or unused based on context. As mentioned in Section 2, a user's movement can be used as a trigger to disable certain components, such as the GPS. If a user is not moving and is not utilizing their Wi-Fi connection, then Wi-Fi may be shut off to further energy savings. Thus, to measure the battery impact of this idea, we add an additional test to our evaluation in which we use the accelerometer to turn off Wi-Fi while the device is idle.

5 Evaluation

First, we give the settings and conditions under which we measure power consumption of our implemented Location Inferrer. Then, we discuss the results and remarks of our tests.

5.1 Settings

We evaluate our solution by measuring the change in our prototype device's battery level over time in different situations. This approach is taken as opposed to looking at battery usage statistics as those statistics typically do not list hardware components, such as the GPS, by themselves. These tests are done indoors with the phone stationary so that movement is not a factor in determining the device's location from the GPS. The location where testing is done is predefined such that it is known to the Location Inferrer. Our first set of tests are performed with the device set in airplane mode, with screen brightness at the lowest level, and using an application that continually needs location data, but does little with it. Tests are performed in this fashion so as to keep other battery draining factors as minimal as possible. Two tests are performed using the Maps application from Google to evaluate performance under a realistic setting. Finally, we conduct one additional test with the minimal application in which the accelerometer is used to disable Wi-Fi while the phone is idle, but still allows the Location Inferrer to work.

Measuring the change in the device's battery level is done as follows. First, the phone is charged fully (when the battery indicator reads "Charged" rather than "Charging, 100%"). Second, our testing application is set up to receive battery-related broadcasts. Such broadcasts are fired every time the battery level changes (for example, from 100% to 99%). Thus, the application keeps track of the amount of time it takes for the battery level to decrease by 1%. Each test was measured in the amount of time it took for the battery level to decrease from 100% to 99%.

A total of 10 types of tests were conducted, 8 of which used the minimal application, 2 of which used the Maps application. First, a baseline test was conducted in which the device was purely in airplane mode, with Wi-Fi and Location Services disabled. Then, a test was conducted with Wi-Fi enabled and idle, but without the Location Inferrer implemented. A test was also conducted where GPS and Wi-Fi were both on, but the Inferrer was not implemented and Wi-Fi was not connected - just left idle. The GPS acted normally to find a location fix. This test was intended to simulate a user who leaves Wi-Fi on constantly. Following those, were three tests with the standard GPS to see if GPS's energy consumption was different under different situations. The first of the three was to let the GPS act as normal and find a location fix naturally. The second was taken in a basement to test the energy consumption of the GPS in a situation where it would be unable to obtain a location fix. The third test was performed after the GPS had already obtained a fix and was constantly updating its location.

Two tests types were conducted with the Location Inferrer. The first involved letting the Inferrer operate normally after the smartphone had connected to a Wi-Fi network with a predefined location. The second test incorporated the use of the accelerometer to disable Wi-Fi while the smartphone was idle. In this case, the Location Inferrer was still allowed to run while Wi-Fi was turned off.

The final 2 of the 10 total test types were performed with the Maps application to measure the differences in using a commercial application - one test with the Location Inferrer and one test with the standard GPS.

5.2 Results

Table 1 displays the results of each of the 10 tests performed. The results are the averages of multiple tests of the same type. As can be expected with minimal functionality, the baseline test took 13.58 minutes to decrease the battery level by 1%. Adding on Wi-Fi functionality, although idle, decreased the time by about two minutes, or 12.8%.

Table 1. Test Results - Battery Life from 100% to 99%

Test	**Time (seconds)**	**Time (minutes)**
Baseline	814.83	13.58
Wi-Fi Idle	710.44	11.84
GPS & Wi-Fi Idle	541.96	9.03
GPS - Normal	546.89	9.11
GPS - No fix Obtained	572.97	9.55
GPS - Obtained Fix	558.18	9.30
Location Inferrer - with Wi-Fi	687.22	11.45
Location Inferrer - with Accelerometer	781.64	13.03
Maps - Location Inferrer	544.83	9.08
Maps - GPS	396.67	6.61

Implementing the Location Inferrer yielded an average time of 11.45 minutes, which is only about 4.7% less than the Wi-Fi Only test and 15.7% less than the baseline. It is expected to be slightly less as the Inferrer has to report a location at a regular interval and also has to manage Wi-Fi-related broadcasts. All three of the GPS tests yielded a time between 9 and 10 minutes which, at best, is 30% worse than the baseline. However, it does show that the GPS consumes about the same amount of energy regardless of the progress it has made in obtaining a location fix. Having Wi-Fi idle while using the GPS seemed not to make too much of difference compared to the other GPS tests, however it had the most effect on the battery life of all the initial tests.

The main feature to note is that the Location Inferrer, on average, performs two minutes better than the standard GPS operation. Compared to the GPS - Normal test, using the Inferrer is a 25.7% increase in battery life and is a 19.9% increase compared to the GPS - No Fix Obtained test. This two minute relationship is also held when performing the tests with the Maps application. In this case, there is a 37.4% increase in battery life by using the Inferrer. As expected with the Maps tests, the battery drains faster than with the minimal application - the Maps - GPS test is shown to consume about twice as much energy as the baseline. Finally, using the accelerometer to disable Wi-Fi while the smartphone is idle and still allow the Location Inferrer to run yielded an average time that was only 4.1% less than the baseline.

5.3 Remarks

As expected, using our Location Inferrer causes energy consumption to be closer to that of Wi-Fi, rather than the GPS. Many users leave Wi-Fi and Location Services on at all times and users of Wi-Fi often have their devices set to automatically connect to a familiar network once it is in range. It follows that a user with the implemented Location Inferrer gets the most energy savings while at home, the office, or on a college campus. The use of Wi-Fi is pervasive enough in today's society that it is often available in many locations. Moreover, while indoors or in underground locations, the accuracy of the GPS drops significantly and a location fix may not even be possible. However, because locations may be user-defined or looked up online, our solution is still able to deliver accurate location data when the GPS is not reliable.

Android 4.4 (KitKat) introduces a battery saving mode for the GPS. Although our prototype device is not running KitKat, this battery saving mode uses fewer reference points to find a location fix. As a result, the location fix is less accurate. In this respect, the Inferrer may still be able to deliver more accurate location data while still only using Wi-Fi.

6 Conclusion

In this paper, we devise a solution to disable GPS functionality based on context in order to conserve battery life. Specifically, once a location can be inferred from Wi-Fi connection information, our solution disables GPS navigation. Once GPS navigation has been disabled, our solution reports the inferred location in its stead. Our solution is implemented transparently into the Android location framework such that application code needs not be adapted to our modifications. Our results show that battery usage with our solution implemented is significantly improved from that of the natural GPS operation. In addition, the battery usage is similar to that of using Wi-Fi rather than using Wi-Fi plus the GPS. Finally, our results show that implementing a solution to disable Wi-Fi while the smartphone is idle causes battery usage to be close to our baseline. We plan to expand upon this solution by introducing additional modules to disable other smartphone components, such as Wi-Fi or sensors, based on device context.

Acknowledgement. This work was supported in part by the National Science Foundation of the US (CNS-1352726 and CNS-1265311).

References

[1] Kjærgaard, M.B., Langdal, J., Godsk, T., Toftkjær, T.: Entracked: Energy-efficient robust position tracking for mobile devices. In: Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services. MobiSys 2009, pp. 221–234. ACM, New York (2009)

[2] Paek, K.G.: Energy-efficient rate-adaptive gps-based positioning for smartphones. In: MobiSys 2010 Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, pp. 299–314. ACM (2010)

[3] You, C.W., Huang, P., Chu, H.H., Chen, Y.C., Chiang, J.R., Lau, S.Y.: Impact of sensor-enhanced mobility prediction on the design of energy-efficient localization. Ad Hoc Netw. 6(8), 1221–1237 (2008)

[4] Zhuang, K.S.: Improving energy efficiency of location sensing on smartphones. In: MobiSys 2010 Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, pp. 315–330. ACM (2010)

[5] Nath: Ace: Exploiting correlation for energy-efficient and continuous context sensing. In: MobiSys 2012 Proceedings of the 10th International Conference on Mobile Systems, Applications and Services, pp. 29–42. ACM (2012)

[6] Lin, K., Kansal, A., Lymberopoulos, D., Zhao, F.: Energy-accuracy trade-off for continuous mobile device location. In: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys 2010, pp. 285–298. ACM, New York (2010)

[7] Farrell, C.R.: Energy-efficient monitoring of mobile objects with uncertainty-aware tolerances. In: IDEAS 2007 Proceedings of the 11th International Database Engineering and Applications Symposium, pp. 129–140. IEEE (2007)

[8] Youssef, A.: The horus wlan location determination system. In: MobiSys 2005 Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services, pp. 205–218. ACM (2005)

[9] Thiagarajan, A., Ravindranath, L., LaCurts, K., Madden, S., Balakrishnan, H., Toledo, S., Eriksson, J.: Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones. In: Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems, SenSys 2009, pp. 85–98. ACM, New York (2009)

[10] Inc, C.: Cyanogenmod Official Website, `http://www.cyanogenmod.org/`

[11] Android: Android SDK, `http://developer.android.com/sdk/index.html/`

[12] Samsung: The Galaxy S4 smartphone specifications, `http://www.samsung.com/global/microsite/galaxys4/index.html/`

[13] Android: Android Location Strategies, `http://developer.android.com/guide/topics/location/strategies.html/`

[14] WiGLE: WiGLE Official Website, `https://wigle.net/`