Energy-Accuracy Aware Localization for Mobile Devices

Kaisen Lin
University of California San Diego
kaisenl@cs.ucsd.edu

Aman Kansal, Dimitrios Lymberopoulos, and Feng Zhao
Microsoft Research
{kansal,dlymper,zhao}@microsoft.com

ABSTRACT

Mobile applications often need location data, to update locally relevant information and adapt the device context. While most smart-phones do include a GPS receiver, its frequent use is restricted due to high battery drain. We design and prototype an adaptive location service for mobile devices, a-Loc, that helps reduce this battery drain. Our design is based on the observation that the required location accuracy varies with location, and hence lower energy and lower accuracy localization methods, such as those based on WiFi and cell-tower triangulation, can sometimes be used. Our method automatically determines the dynamic accuracy requirement for mobile search-based applications. As the user moves, both the accuracy requirements and the location sensor errors change. A-Loc continually tunes the energy expenditure to meet the changing accuracy requirements using the available sensors. A Bayesian estimation framework is used to model user location and sensor errors. Experiments are performed with Android G1 and AT&T Tilt phones, on paths that include outdoor and indoor locations, using war-driving data from Google and Microsoft. The experiments show that a-Loc not only provides significant energy savings, but also improves the accuracy achieved, because it uses multiple sensors.

1. INTRODUCTION

Mobile applications often need location information and a large number of methods for mobile device localization have been developed [22]. With GPS receivers becoming increasingly commonplace in mobile phones and the widespread availability of WiFi and cell-tower signature based location services from Google [9] and other providers, such location information is now becoming a reality. However, mobile applications still cannot assume continuous and ubiquitous location access in their design because of the high energy expense of using the location sensors such as GPS receivers [12]. The variability in accuracy provided by various location sensors and the limits on their coverage areas pose additional challenges for application developers. Using multiple location sensors simultaneously to make up for this variability in accuracy would further increase energy use.

Our goal is to develop location as a system service that automatically manages location sensor availability, accuracy, and energy. From an application developer perspective, this simplifies the use of the multiple existing, and potentially forthcoming, location technologies with varying characteristics. From a mobile user experience perspective, this allows the system to optimize battery life by intelligently managing the location energy and accuracy trade-offs based on available sensor capabilities. This is beneficial for mobile platforms that allow several third party applications to run on the platform, but at the same time must ensure long battery life for acceptable user experience.

To realize the above goal, we develop an approach based on two observations. First, location applications *do not always need the highest available accuracy*, such as that provided by GPS in open sky view locations. The accuracy needs vary as the user moves and we can exploit the slack in required accuracy to save energy. Second, a phone has multiple modalities to sense location aside from the GPS: WiFi triangulation [16, 3], cell-tower triangulation [22], Bluetooth vicinity, audio-visual sensing [2], among others [4]. The availability and accuracy of these modalities *vary as the user moves*, and appropriate modalities can be selected to efficiently meet the location needs at lower energy costs.

A typical scenario may involve the user starting a mobile search application such as Google Maps on their phone, and searching for a keyword, say "pizza." The search application wishes to determine the nearest pizza stores to display. If the user is in a densely populated area with multiple pizza stores, a high accuracy is needed to correctly determine the nearest entries. However, if the user is in a remote area with few pizza stores, knowing the location to the nearest mile may suffice to determine the correct entry. Clearly, in the latter case, a low energy location modality, simply based on cell tower association, could be used. Figure 1 illustrates the accuracy required at different locations in Portland, if the application was searching for the nearest five pizza stores. Variable accuracy requirements also apply to most search based scenarios including those where the user does not initiate a search but the application displays information proactively, such as show times for nearest movies. Another scenario may involve the mobile device adapting its role based on the user context where this context is resolved through

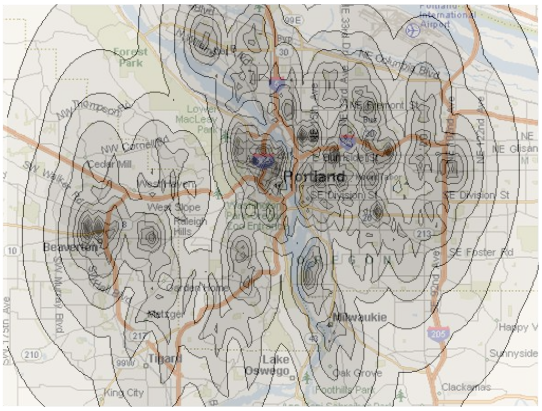


Figure 1: Contour plot of accuracy requirements for finding nearest five pizza stores in Portland region (darker shades represent higher accuracy requirement).

distinguishing among locations with distinct activities, such as home, office, shopping mall, beach etc. Again, the required location accuracy depends on the geographical separation among these places. Finally, variable accuracy needs also emerge for idle screen advertising and social networking applications [15].

Previous works have presented effective methods to reduce the energy overhead of GPS, but they do not fully leverage the energy-accuracy trade-off. One approach [12] to reduce energy use is to increase the time for which the GPS remains powered down by detecting when the user is stationary, using an accelerometer and predicting how far the user moved based on past speed. However, when a new location reading is required, the GPS is still used. Our goal is to further reduce the energy by using lower energy sensors when appropriate. Location based on WiFi was used in [23] to reduce reliance on GPS. However, a static model for availability and accuracy for both WiFi and GPS was used, that is not applicable in all scenarios. Our method uses dynamic models for both location accuracy requirements as well as the sensor characteristics, and continually tunes the location energy-accuracy trade-off to satisfy application needs. Specifically, we make the following contributions:

First, we develop a system service, named *a-Loc*, that automatically adapts location energy and accuracy based on dynamically varying sensor characteristics as well as application needs. Our method can be simply used as a system provided location service by multiple applications through a standard interface that accepts the accuracy requirement and returns the location. Internally, a-Loc minimizes the energy consumed for achieving the specified accuracy.

Second, we present experimentally measured data that characterizes some of the commonly available location sensors in terms of their accuracy and energy. We use this data to develop practical models used in our prototype implementation of a-Loc on a mobile phone. Bayesian estimation is used as the mathematical machinery behind our models. The a-Loc framework is extensible; additional localization techniques may be included by adding their sensor models. This allows battery performance to improve as new localization capabilities are added to mobile devices, without requiring changes to applications.

Third, we evaluate the effectiveness of the proposed methods through real-world experiments with an Android G1 phone using multiple built-in location modalities and the Android OS 1.6 interface to a Google location service for WiFi localization. Additional experiments are performed in emulation and presented using a Windows Mobile phone based on Microsoft's internal WiFi and cell-tower based war-drive data that allows us to generate several additional user trajectories, representing different realistic motion patterns. The performance of a-Loc is compared to existing techniques as well as a method to use multiple location modalities without considering the dynamic variations in sensor availability and accuracy.

2. RELATED WORK

Many location sensing modalities have been developed for mobile phones [22]. WiFi radios, available on many mobile devices, have been explored for localization [16, 24, 3] and WiFi war-driving data for many regions is available commercially. Encoding of location in WiFi SSID's has also been proposed [4]. Additional methods exist based on cell-tower signal strengths [16, 22] and FM radio station signal strengths [13]. More recently, the phone's camera and microphone have also been used for localization [2, 20, 18]. The focus of this paper is complementary to the above works. The goal of our system is to use the available location modalities, such as the above, and provide an energy efficient mechanism to obtain location to just the required accuracy.

Energy usage of some location modalities was studied before. In Microblog [6, 8], methods to reduce GPS use by predicting the user's path were presented. We add to such methods by explicitly trading off accuracy and energy. Our framework incorporates user movement models, sensor characteristics, energy models, shared real world sensor accuracy data, and methods to automatically determine evolving accuracy needs. Another method to reduce the energy use of GPS was presented in [12]. Accelerometer data was used to determine when the user is stationary and power down the GPS. Further the velocity of the user estimated by the GPS was used to infer time durations for which the user will stay within tolerable location error range, and GPS was shut down for those durations. Another method to reduce GPS energy was considered in [7], where GPS updates were assumed required only when the mobile device enters a specified region. Location prediction based on known mobile device velocity was used to infer time durations for which the device could not reach the specified region from its prior location, and GPS sampling was suppressed for those durations. In comparison, we use multiple location modalities,

account for their variable accuracies and availability, and allow using enhanced user location prediction based on a Hidden Markov Model. We also incorporate variable location accuracy requirements. Techniques to determine when location data is not needed [12, 7] can of course be used in addition to our proposed method.

The use of WiFi based location to reduce the reliance on GPS was also explored in [23]. The focus of the work however was on road traffic estimation. An initial method for reducing GPS use was included by reducing GPS sampling to once every k seconds, and using WiFi for the interim period of k seconds. Energy was optimized by choosing a value of k that allowed achieving the accuracy constraint. However, static models for GPS and WiFi error and energy were assumed, which do not apply in general. We use dynamic models for sensor accuracy that are acquired based on real world data, and also allow using multiple additional sensors.

3. SYSTEM OVERVIEW

As a real-world example that provides a concrete testcase for a-Loc, we consider mobile search. Mobile search is an important application for two reasons. First, there is a much larger number of mobile devices than desktops, and these devices are rapidly becoming capable of obtaining information from the Internet, through either full fledged smart phone browsers such as Mobile Safari or limited capability browsers using WAP, iMode etc. This has made mobile search the fastest growing mode of search usage. Secondly, mobile search is especially important to search service providers such as Google, Yahoo, and Microsoft because a significant fraction of mobile searches involve searching for local services or products. This type of mobile search is easiest to monetize, not merely through advertisements accompanying search results but also through transactions initiated based on those results. This importance is evidenced by the rapid release of search-centric mobile applications such as Google Mobile [1], competing applications from other search providers, and voice-based search applications such as GOOG-411 for mobile devices without Internet capabilities.

We assume that location data is needed continually for the durations that the application is active, and the user has not been classified as stationary using techniques such as [12]. Search based applications may display nearest movie show times, nearest deals and coupons on products the user has expressed an interest in, locations of other mobile users from the user's social network, and contextually relevant idle screen advertisements. Mobile search is already known to be significantly slower than desktop search [11], and so it makes sense to not add the additional localization delay to every user initiated action in the search-based applications. Continuous access to location is also required for many other mobile applications, including those that act based on user's location, to control home thermostats [10], for instance. All these search-based applications naturally have a dynamically varying location accuracy requirement.

[1]http://m.google.com

At a high level, a-Loc considers multiple factors that affect location estimation, including a prediction of the user's location, the error performance of location sensors, energy costs, and application accuracy requirements. Figure 2 shows the key components of our proposed system, and they are discussed below.

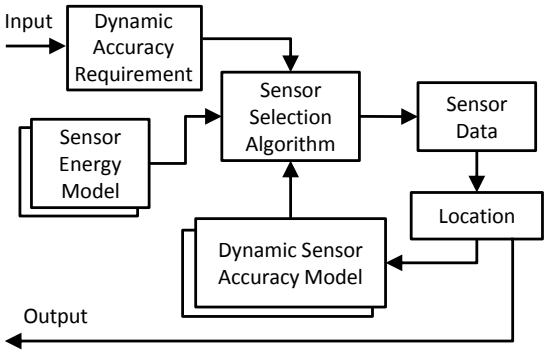


Figure 2: System block diagram.

Dynamic Accuracy Requirement: This block provides the location accuracy needed by the applications. For the mobile search-based applications, we provide a method to compute the accuracy requirement based on the entities searched (Section 5.2) but in other cases, the accuracy need may be directly specified by the application.

Sensor Energy Model: These models characterize the energy used by each available location sensor for obtaining location. We experimentally measure this for the modalities used and also compare the data to similar measurements on other phones in Section 4.2. In some cases, the energy spent depends on the location where the observation is made and we experimentally measure this effect.

Dynamic Sensor Accuracy Model: This model is developed for each sensor to characterize the quality of location information that it offers. A key challenge, not usually addressed in prior work, is that the availability and accuracy of location sensors varies with location. For example, the GPS may not work indoors or work poorly in areas with obstructed satellite view, WiFi triangulation may work better where the number of access points is high, and so on. The dynamic sensor model systematically characterizes such effects. The model is built using past sensor data for each location. In a real world deployment, this data, collected by multiple mobile devices for various regions, may be shared to build up a model with widespread geographical coverage, and appended to WiFi war-drive data sets. In our experiments, we learn the models on the mobile device itself. Section 4.1 describes these models in detail.

Sensor Selection Algorithm: The sensor selection algorithm determines the location sensor to be used at each time

step. The algorithm includes a method to model the user location trajectory and uses the sensor data as available to improve the location estimates. Maintaining a motion model and location estimate allows the algorithm to use the sensor accuracy and energy models in a location dependent manner. A Bayesian estimation framework is used to combine the sensor data and predicted location to provide a maximum likelihood estimate (Section 4.3).

After a location sensor has been selected, energy is spent to use that sensor, yielding *sensor data* that is used to generate a *location estimate*. The location estimate is output to the client application. Additionally, the location and sensor data may be used to enhance the sensor accuracy model.

Discretization: Before we describe the above models and methods in detail, it is worth noting that both space and time are discretized in these models. We use *discrete* probability distributions in the Bayesian framework. In general, the space discretization granularity may be set to be smaller than the minimum resolution provided by any sensor. A-Loc uses a 10m step size for space discretization. Time granularity depends on the frequency of location updates. A-Loc uses time granularity of 1 minute, which is the same order of magnitude as in [12]. These discretization steps are appropriate for the types of applications mentioned above, though they are unlikely to work for certain other applications such as street navigation which require high frequency and high accuracy location updates.

4. SYSTEM DESIGN

This section describes the key components of a-Loc. We use the following location modalities that were available on both the Android G1 and AT&T Tilt phones: GPS, WiFi, Bluetooth, and cell-tower.

4.1 Accuracy Models

The dynamic sensor sensor models characterize the accuracy and its variation with location, due to various factors that affect the performance of the sensor. For instance, the WiFi radio may be used to infer location by matching a list of visible WiFi access points, referred to as the scan fingerprint, to a database of known locations and fingerprints. This will yield varying accuracies depending on the density of access points and the spatial coverage of the database.

To facilitate the use of this model in the standard Bayesian framework used in the sensor selection algorithm, we represent the accuracy model for modality i using a probability distribution, $p\left(\mathbf{z}_i(t)|\mathbf{x}(t)\right)$. This distribution gives the likelihood that modality i yields observed location $\mathbf{z}_i(t)$ at time t, when the true (and unknown) location is $\mathbf{x}(t)$, where $i \in \mathcal{L}$ and $\mathcal{L}$ represents the set of location modalities available. This distribution depends on location $\mathbf{x}(t)$ and hence captures the variations in accuracy with changing location. The form of the distribution is assumed to be a two dimensional Gaussian distribution centered at $\mathbf{x}(t)$:

$$p(\mathbf{z}_i(t)|\mathbf{x}(\mathbf{t})) = \frac{1}{\sigma^2_{x(t)}\sqrt{2\pi}} \exp\left(-\frac{|\mathbf{z}_i(t) - \mathbf{x}(t)|^2}{\sigma^2_x(t)}\right)$$

Figure 13(b) visually illustrates such a distribution. The sensor errors in the two spatial dimensions are assumed to be independent and identically distributed (zero correlation and same variance in both spatial dimensions).

Here, the standard deviation $\sigma_{x(t)}$ depends on the error for the sensor at location $\mathbf{x}(t)$ and is learned from real world data, as described below.

GPS: A GPS receiver typically reports its estimate of error as horizontal dilution of precision (HDOP). Figure 3 shows the HDOP achieved with different number of visible satellites. An HDOP of 6 or less implies location error less than $12m$ [19]. For most outdoor locations in our experiments we observed 4 or more satellites, yielding an HDOP below 6 (acquisition times varied with location). We use $10m$ as the spatial discretization step and so the GPS sensor model uses $\sigma_{x(t)} = 1.2$, for locations where GPS is available, and infinity where unavailable.

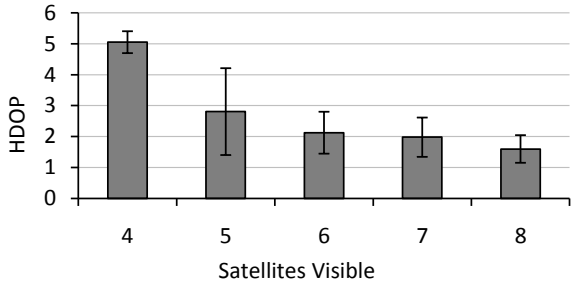


Figure 3: Experimentally measured GPS accuracy.

WiFi: Methods to convert a visible access point list to a location have been studied in [16, 3, 24]. The error, $\sigma_{x(t)}$, is expressed as a function of the number of access points, $n(t)$, visible at $\mathbf{x}(t)$. Our experiments use the conversion relation between $\sigma_{x(t)}$ and $n(t)$ found in one of the prior works [16]. The value of $n(t)$ can easily be determined at each location when WiFi is used and the conversion relationship provides the $\sigma_x(t)$ at that location.

As an alternative, an error estimate for WiFi localization is also provided by the Google location service used via the Android. This estimate, for all locations of interest to a mobile device can easily be cached on the phone. An example of WiFi location errors observed in our experiments for a sample user path of 0.55km is shown in Figure 4.

Bluetooth: For Bluetooth, location is based on finding at least one static Bluetooth device in radio range (a computer mouse in an office, a Bluetooth advertisement device in a shopping mall [1], etc.).The error is taken to be the Bluetooth range, nominally set at $10m$ for the commonly used class 2 Bluetooth devices, implying $\sigma_{x(t)} = 1$ based on the spatial discretization step of 10m, where a static Bluetooth

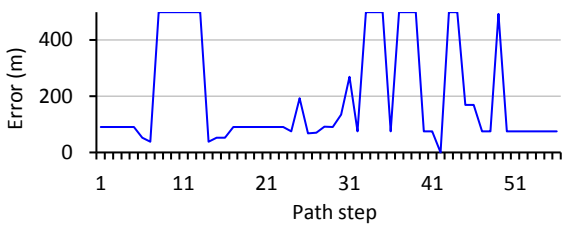


Figure 4: WiFi location error with Android G1. Errors above 500m are capped to 500m for plotting.

device is visible and infinity at other locations. Bluetooth localization can be extended to use multiple visible devices, if multiple static devices are indeed visible in a region, but refining Bluetooth localization methods is beyond the scope of this work.

Cell-Tower: Most current phones from cellular providers in the US only allow reading the currently connected cell-tower, even though the radio stack in the device maintains a larger list that includes other cell-towers within range. With only one tower's identity, the location error is essentially equal to the size of the cell within which that tower is likely to be the one with the strongest signal for a mobile device. We use the cell-size based on typical cell-tower density for dense urban areas since the experimentation region is a dense urban area. In the future or with certain cellular providers, if phones do allow reading the list of multiple visible cell-towers and their signal strengths, localization methods based on matching the visibility fingerprints or triangulation may be used [16, 22]. The error value could then be based on observed error performance of those methods.

4.2 Energy Models

We experimentally measured the energy usage for multiple location modalities on an AT&T Tilt (HTC TyTN II) mobile phone. This phone includes a Qualcomm gpsOne a-GPS, Bluetooth 1.2 and an 802.11 b/g WiFi radio. The phone's battery was removed and instead power was supplied from a Monsoon Solutions Power Monitor that allows logging the power supplied to the phone, at 200 microsecond intervals. Energy was measured using a location modality and external factors that may affect energy use were varied. All measurements are made based on application layer access to the underlying sensing modalities. Measurements include the energy in turning on, reading, and turning off the relevant sensing modality. The processing of sensor data and the sensor selection algorithm are not a part of the energy model. The energy model for modality i is denoted $E_i(t)$ and may depend on the location at time t.

4.2.1 WiFi Triangulation

Using WiFi entails powering up the WiFi radio, scanning[2] the access point identities (SSID's), and powering down the WiFi radio. Association with access points is not needed as localization only requires the SSID's of the access points.

Figure 5 shows the power drawn for scanning WiFi access points (obtained by subtracting the baseline power used by the phone when idle with all sensing modalities powered off from the measured total power drawn). The graphs shows that there is an initial energy spurt possibly for initialization, followed by a longer period of energy usage for the scan itself.

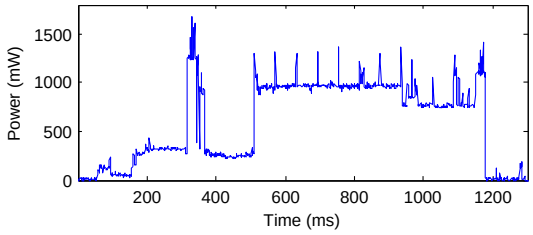


Figure 5: Measured power profile for WiFi.

One external factor suspected to affect the energy of the scan is the number of access points visible at a given location. The scan energy usage for WiFi measured at different locations and averaged for similar number of visible access points, is shown in Figure 6. The vertical bars represent standard deviation across measurements. Here, the energy

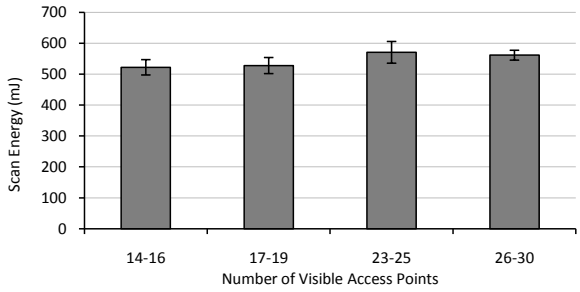


Figure 6: Energy usage for WiFi.

cost does not vary significantly with number of visible access points. Based on this data, we use the energy model $E_i(t) = 545.07mJ, \forall t, i =$ WiFi. Part of this cost is the energy to turn on WiFi, measured to be 115mJ and turn off WiFi, measured at 65mJ, averaged. If the WiFi radio is already on for other uses, the power up and down cost may be ignored. The latency of conducting the scan was $0.7s$ on an average. Given that the location update interval is a minute or larger in our scenarios of interest, this latency is not a concern.

We assume all data for matching the SSID's or triangulations is locally available on the phone. The data size required for such data for one of cities used in our tests was 20MB and this very reasonable for local storage given the multi-gigabyte flash storage capacities on most phones. On the other

[2]Active scan is used as it is more energy efficient than a passive scan due to reduced listening time.

hand, we also measured the energy used for communicating with a central server, using the 3G radio and its energy overhead (6000mJ-12000mJ depending on data size, Figure 21) was found to be higher than all the location modalities. Using server communication for localization would thus significantly increase energy overheads.

4.2.2 *Bluetooth Vicinity*

Localization using Bluetooth entails scanning the identities of Bluetooth devices in vicinity. Since Bluetooth has a short range of about 10m, visibility of a device suggests a distance of less than 10m from it. Most Bluetooth enabled devices are mobile and their visibility may not necessarily indicate a fixed location but if a device with a known static location is found, location can be determined.

The Bluetooth scan energy for different number of visible devices is shown in Figure 7. At first glance, comparing

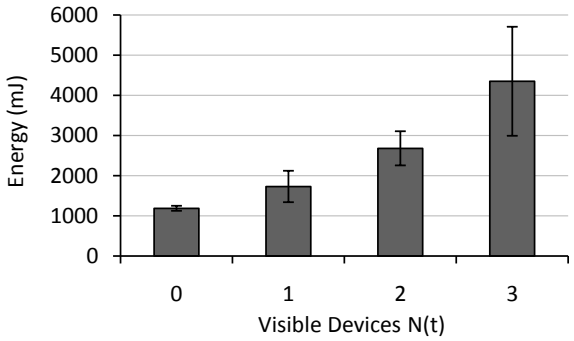


Figure 7: Bluetooth energy usage variation.

this energy with Figure 6, this data may appear surprising in that Bluetooth is using more energy than WiFi. However, the power measured for Bluetooth (Figure 8), was indeed much lower than the WiFi power draw. The energy use is

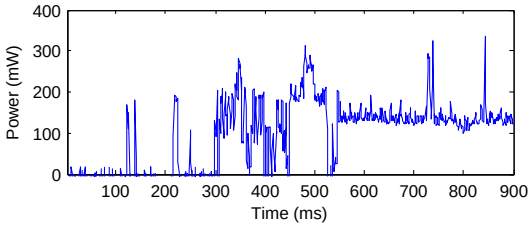


Figure 8: Bluetooth power usage during scan.

higher because Bluetooth takes much longer to perform a scan of visible devices. This is caused by the complexity of the Bluetooth scan protocol. The multiple steps and frequency hopping defined in the Bluetooth scan protocol [5] cause the scan phase to take considerably more time than a WiFi scan.

The energy does depend on the number of visible devices and would hence vary with location. Bluetooth scanning can be set to stop after a fixed number of devices are found, bounding the energy spent on a scan. The energy to power up and power down the Bluetooth radio was measured to be $160mJ$ and $35mJ$ respectively. Suppose $N(t)$ represents the number of visible Bluetooth devices at time t. Figure 9 shows linear and quadratic curves fit to the data. We take the linear model for simplicity, that gives $E_i(t) = 1299.6 * N(t) + 558mJ$ for $i =$ Bluetooth[3].

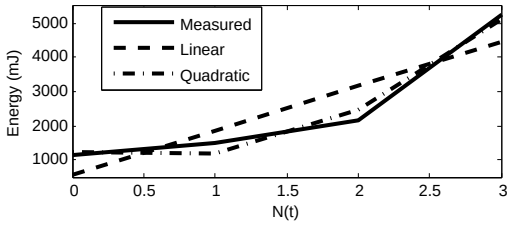


Figure 9: Linear and quadratic curves fit to measured Bluetooth energy.

4.2.3 *GPS*

The GPS chip on the AT&T Tilt is a Qualcomm gpsOne, an assisted GPS solution, implying that satellite almanac and ephemeris data is obtained through the cellular data connection rather than from satellites. This allows for faster fix times, and consequently, lower energy usage. Once a fix is acquired, GPS uses power at a steady rate, with intermittent higher usage, presumably due to almanac and ephemeris acquisition. Figure 10 shows the power profile for turning on and obtaining location using GPS. For the Android, the GPS

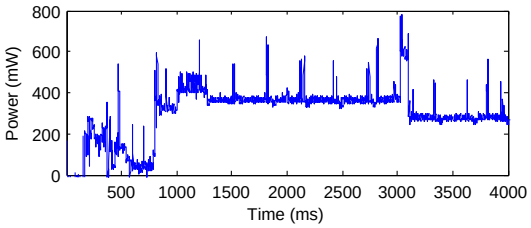


Figure 10: Measured GPS power profile.

power draw has been measured at 230mW [21]. These numbers are comparable to Nokia N95 [12], that drew 324mW. GPS energy for the iPhone was measured in [23] but because this device requires the application to run in the foreground, it includes the energy use of the entire system with the LCD screen powered on, making it hard to directly compare the energy numbers. The current draw estimated from the measured battery life and known battery capacity is 455mW, including the idle system energy.

Our experiments indicated that GPS energy does depend on location, as was also noted in [12]. Figure 11 presents GPS energy for a location fix at three different locations:

[3]Quadratic fit is $E_i(t) = 684N^2(t) - 752N(t) + 1242.6mJ$.

(1) a road intersection, (2) a large park with open sky view, and (3) in front of an office building. The measurements are based on getting a fix with a horizontal dilution of precision (HDOP) better than 6. Very long fix times as high as 114s (not shown), were sometimes observed in locations with poor sky view. The measurements in the figure are from a cold start of the GPS, assuming no previous almanac or ephemeris data. With a warm start, the satellite acquisition time dropped to as low as 5s at some of the locations, with a corresponding reduction in energy usage.

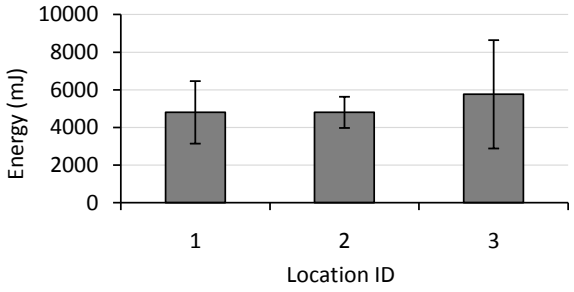


Figure 11: GPS energy usage (cold start).

A closed form equation does not exist as usage depends on location and time in a complex manner (due to satellite coverage quality). Instead we use an average energy value for quantifying GPS energy use, using two separate averaged values for warm and cold starts: $E_i(t) = 1425mJ$ for $i =$ Warm GPS, and $E_i(t) = 5700mJ$ for $i =$ Cold GPS. We use the warm start energy model for each time step, when GPS was also selected at the previous time step, implying a lower cost for GPS when used at consecutive steps.

4.2.4 Cell-Tower Association

A cellphone maintains a list of cell-towers that are visible to its radio receiver. Based on this, the phone may determine its location [16]. The energy to use this location modality is thus negligible as it only consists of reading data available on the local device. This energy was measured to be under 20mJ, averaged over multiple readings.

Figure 12 collects the energy spent on various modalities for a relative view; the maximum and minimum energy measured for each are plotted. While we use only the above four location modalities, other options have also been proposed, such as using phone's camera [2]. The figure includes the energy to capture an image and save it as a Jpeg file on internal flash. The processing energy for image based localization, unlike other modalities, may not be negligible and should be accounted in the energy model. The energy usage varies by orders of magnitude and hence selecting lower energy modalities when feasible is likely to yield significant savings.

4.3 Selection Algorithm

The goal of the selection algorithm is to determine the most energy efficient sensor to be used, such that the required location accuracy can be achieved. In addition to the sensor accuracy and energy models, this algorithm also maintains an estimate of the user's location that is based on a prediction of user movements. The prediction helps select the appropriate location for the sensor energy and accuracy models and may even help avoid sensing when predicted location has a high confidence.

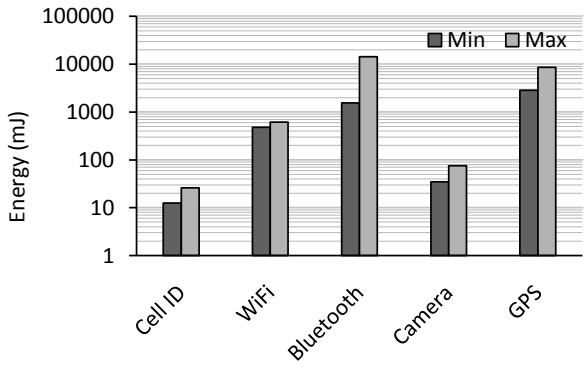
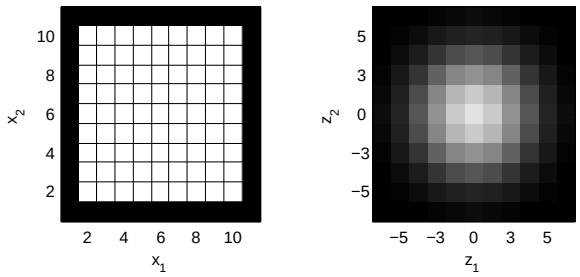


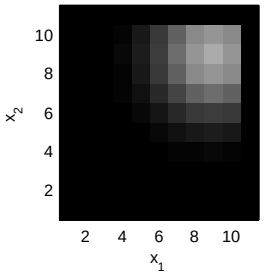
Figure 12: Relative energy costs of location modalities.

Location at discretized time t is denoted using a random variable $\mathbf{x}(t)$, that takes values in a two dimensional space. Suppose the location observation from sensing modality i at time t is denoted $\mathbf{z}_i(t)$ as before. Suppose $\overline{\mathbf{z}(t)}$ represents all observations made up to the time instant t, ie $\overline{\mathbf{z}(t)} = \{\mathbf{z}(t), \mathbf{z}(t-1), ..., \mathbf{z}(0)\}$ for any i. Then, the probability distribution of location at current time t given all previously made observations and prior models is given by $p(\mathbf{x}|\overline{\mathbf{z}(t-1)})$. As an illustration, consider the probability distribution shown in Figure 13(a), where the example distribution is uniform over a discretized two dimensional square region, such as initialized at $t = 0$.

The prior distribution captures our knowledge about the user location up to the time at which the last observation was made. We use this to predict the location at the current time step. Prediction of user location has been studied extensively in literature and we use one of the commonly used approaches, based on a Hidden Markov Model (HMM). Specifically, we use a second order Hidden Markov Model (HMM) that uses the past two observed locations to yield a distribution of predicted location, $p(\mathbf{x}(t)|\mathbf{x}(t-1), \mathbf{x}(t-2))$, providing a probability distribution of location before spending energy on sensing at the current time step. A second order model takes the direction of motion into account, significantly improving prediction performance over a first order HMM, but higher orders yield diminishing returns. The transition probabilities between locations are learned from past user motion and are updated as new observations are made. Such an approach allows learning an arbitrary probability distribution of location transitions. Also, only a small portion of the learned probabilities corresponding to the region



(a) Prior Distribution (b) Sensor Model.



(c) Posterior Distribution.

Figure 13: Illustration of the (discretized) stochastic models: (a)prior distribution of $\mathbf{x}(t)$, (b) sensor model $p(\mathbf{z}_i(t)|\mathbf{x}(t))$ at $\mathbf{x}(t) = [0, 0]$ with standard deviation = 2, and (c) posterior distribution $p(\mathbf{x}(t)|\mathbf{z}_i(t))$ for an observation $\mathbf{z}_i(t) = [9, 9]$. Lighter shades represent higher probabilities.

around the user's location may be loaded into the memory at a given time, making the approach highly scalable. If the user is at a new location with no prior history data, methods such as linear extrapolation on the past few locations may be applied. Likely user locations learned from other mobile user locations or even land use [14] can be employed.

Other location prediction models can of course be used. For instance, a Kalman filter may be employed to predict the next location based on past observed locations. However, when past observations over several days are to be used, the size of the filter and matrices involved becomes very large and the scalability of the filter may become a concern. Another possibility is a method based on Conditional Random Fields that learns the significant places visited and transportation modes used [17]. This information is then used to predict future user location. The method relies on detailed parameter learning.

Since the sensor is to be selected before actually using it, we need an estimate of sensor accuracy of each sensing modality i at the predicted user location. Formally, the location estimate after using modality i is characterized by the posterior probability distribution $p(\mathbf{x}(t)|\mathbf{z}_i(t))$. For each sensor modality i, we can use the spread of the distribution of $\mathbf{x}(t)$, given a reading from that modality $\mathbf{z}_i(t)$, as a measure of the error in the estimated location. The trace of the covariance matrix is used to characterize this spread for the two dimensional distribution, much like variance is used for one dimensional random variables. The error for modality i given an observation, denoted $e_i(t)|_{\mathbf{z}_i(t)}$, becomes:

$$e_i(t)|_{\mathbf{z}_i(t)} = tr\left\{Cov(\mathbf{x}(t)|\mathbf{z}_i(t))\right\}$$

The computation of the covariance matrix requires the posterior distribution, which can be computed using the sensor accuracy model and the prior location distribution, via Bayes rule:

$$p(\mathbf{x}(t)|\mathbf{z}_i(t)) \propto p\left(\mathbf{z}_i(t)|\mathbf{x}(t)\right) p(\mathbf{x}(t)|\overline{\mathbf{z}(t-1)}) \quad (1)$$

As an illustration, with the prior distribution and sensor accuracy model shown in Figures 13(a) and 13(b) respectively, the posterior distribution for a potential observation $\mathbf{z}_i(t) = [9, 9]$, is as shown in Figure 13(c). The observation has caused the prior distribution to get concentrated in a smaller region, as expected.

However, since we must compute the error that would result from using modality i before spending the energy to obtain $\mathbf{z}_i(t)$, we compute multiple posteriors $p(\mathbf{x}(t)|\mathbf{z}_i(t))$ for different possible $\mathbf{z}_i(t)$ that may be observed, resulting in a different error estimate for each of the multiple possible observations, $\mathbf{z}_i(t)$. We then take a weighted average of these error estimates, where the weights are the probabilities of getting different observations $\mathbf{z}_i(t)$ for modality i. The probability of getting an observation $\mathbf{z}_i(t)$ depends on the current location, and since we do not have the current location, we use an estimate for the probability of getting observation $\mathbf{z}_i(t)$. The probability distribution of the observations is obtained using the distribution of predicted location as follows:

$$\hat{p}(\mathbf{z}_i(t)) = \int_{\mathcal{X}} p(\mathbf{z_i}(t)|\mathbf{x}(t))p(\mathbf{x}(t))d\mathbf{x}(t) \quad (2)$$

where $p(\mathbf{z}_i(t)|\mathbf{x}(t))$ comes from the sensor accuracy model, and $p(\mathbf{x}(t))$ comes from the location prediction. The weighted average of $e_i(t)|_{\mathbf{z}_i(t)}$ for all the possible observations $\mathbf{z}_i(t)$ becomes:

$$\hat{e}_i(t) = \int_{\mathcal{Z}|\mathbf{x}(t)} \hat{p}(\mathbf{z_i}(t))tr\left\{Cov(\mathbf{x}(t)|\mathbf{z}_i(t))\right\} d\mathbf{z_i}(t) \quad (3)$$

where $(\mathcal{Z}|\mathbf{x}(t))$ represents the support of $\hat{p}(\mathbf{z}_i(t))$. This $\hat{e}_i(t)$ quantitatively characterizes the expected error for modality i at the current time step.

Having computed the estimated accuracy $\hat{e}_i(t)$ of sensor i, the sensor selection problem can be expressed as:

$$\hat{i} = \arg\min_{i \in \mathcal{L}} E_i(t) \quad \text{Subject} \quad \text{to}: \quad \hat{e}_i(t) < e_r^2(t)$$

where e_r represents the desired location accuracy, and $E_i(t)$ represents the energy used by modality i. Square of the desired accuracy is used since variances and trace of the covariance matrix characterize error as a square of the variable estimated.

This energy minimization problem may be solved using Algorithm 1, shown below.

Algorithm 1. $SelectSensor(\mathcal{L}, \mathcal{X})$

1. Initialize $p(\mathbf{x}(0)|\overline{\mathbf{z}(0)})$, $p(\mathbf{z}_i(0)|\mathbf{x}(0))$, $t = 0$, and $\Lambda = \phi$
2. Obtain $e_r(t)$. For all $i \in \mathcal{L}$:
   (a) Compute $\hat{e}_i(t)$
   (b) If $\hat{e}_i(t) \leq e_r^2(t)$ then $\Lambda = \Lambda \cup i$.
3. Among all $i \in \Lambda$, select i that has minimum $E_i(t)$.
4. Obtain $\mathbf{z}_i(t)$.
5. Compute $p(\mathbf{x}(t)|\mathbf{z}_i(t))$ and use it to compute $\hat{x}(t) = \mathbb{E}(\mathbf{x}(t)|\mathbf{z}_i(t))$. Return $\hat{x}(t)$ as the current location.
6. Set $t = t + 1$ for next time step.
7. Update $p(\mathbf{x}(t))$ using location prediction method.
8. Go to Step 2 at next time step.

The initial prior distribution $p(\mathbf{x}(0)|\overline{\mathbf{z}(0)})$ in Step 1 could be set to a uniform distribution, or initialized based on land use data or observed mobile phone user locations from shared databases. The $p(\mathbf{z}_i(0)|\mathbf{x}(0))$ is initialized based on experimentally measured sensor accuracy models as described in Section 4.1. Λ represents the set of modalities expected to satisfy the accuracy constraint. The distributions are maintained only over a small region of $k \times k$ grid cells ($k = 100$ in our implementation) surrounding the user location as the probability values are negligible outside this area. This means that the memory overhead of this distribution is only 10kB, acceptable on mobile devices.

Briefly, the algorithm is performing the following operations. For each location modality, the $\hat{e}_i(t)$ is computed using equation (3). Eligible modalities are added to Λ. Among the modalities in Λ, the one with the lowest energy is selected [4].

Energy is spent on the selected modality, obtaining $\mathbf{z}_i(t)$. This is used to compute $p(\mathbf{x}(t)|\mathbf{z}_i(t))$, using (1), and allows computing the estimated location using the expectation:

$$\hat{\mathbf{x}}(t) = \int \mathbf{x}(t) p(\mathbf{x}(t)|\mathbf{z}_i(t)) d\mathbf{x}(t)$$

This is the output provided to the application requesting location.

Moving on to the next time step, the location model is now used to obtain the prior distribution $p(\mathbf{x}(t)|\overline{\mathbf{x}(t-1)})$, where t represents the subsequent time step.

[4]If $\Lambda = \phi$ cell tower based localization may be used as it is almost free.

The above algorithm uses $e_r(t)$ at Step 2b, where $e_r(t)$ is the application specified accuracy requirement, obtained for mobile search based applications as described in Section 5.2.The computation of $e_r(t)$ requires the location estimate $\hat{\mathbf{x}}(t)$, but since it is performed before using any sensor the prior estimate is used.

Computational Overhead: The computational overheads for the location prediction step (a second order HMM), update of the location and sensor models (incrementing a small set of values), and comparison of energies are not significant. The dominant overhead is the computation of the covariance matrix trace, that involves computing the posterior distribution. The sensor accuracy model has non-negligible values only in a small region (up to $k \times k$ grids) around the predicted location and the region of predicted locations is only a small number of grids, n, say. Then, the computation of one posterior (such as shown in Fig. 13(c)) involves nk^2 scalar multiplications, followed by a computation of the two variances $\sigma_{x_1}^2$ and $\sigma_{x_2}^2$ required for the trace of the covariance matrix. The number of posteriors computed depend on the size of the support of $\hat{p}(\mathbf{z}_i(t))$ in (2), and is a small multiple of n, say αn. The dominant computation thus has an overhead $O(\alpha n^2 k^2)$ which for $k = 100$ as before, $n = O(10)$, and $\alpha = O(10)$, requires only a small fraction of a second on a 100MHz or better processor found on mobile phones. The overhead of this computation performed once every location update interval of one minute or more is negligible.

The total compiled code overhead on the mobile device for our implementation was 32kB and the execution time was extremely small.

5. EXPERIMENTS AND EVALUATION

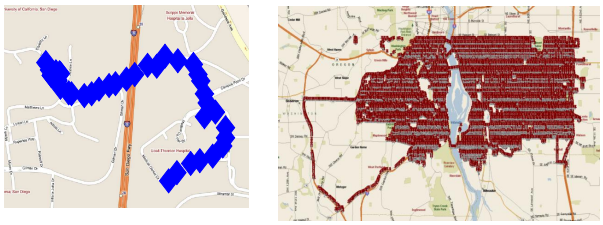
We now evaluate the performance of the proposed method in terms of both the energy savings and the application specified accuracy constraint satisfaction.

5.1 Prototype Implementation

We implemented a-Loc on an Android G1 phone and tested its operation on a real world path while the phone was carried by a mobile user in a region in San Diego (Figure 14(a)). A location service provided by Google, that is accessible through the Android's LocationManager API [9] was used. The responses from this service for various locations along the path were stored locally, using Android's Sqlite data structure, to realize the equivalent functionality of locally stored WiFi war-driving data for this region. The energy use of the network communication was not included since a-Loc assumes the availability of the war-driving data locally as discussed in Section 4.2.1.

In addition to the real world experiments, we also performed an emulation. The emulation was based on a war-driving dataset for the entire city of Portland, Oregon, obtained through Microsoft. The region covered is shown in Figure 14(b) where the darker shades represent regions for which data was available. The entire dataset is only 20MB

in size, justifying the assumption that it can be saved locally on the mobile device. This data allows us to generate arbitrary user paths simulating various types of user movement patterns for a more controlled investigation. In particular, we emulate (i) a commuter traveling repeatedly between home and work with some side trips and (ii) a tourist in the city, not repeating any of the routes.



(a) San Diego (b) Portland

Figure 14: Mobile device path regions for evaluations.

In both the real world path and the emulated paths, GPS is not available when the user is indoors. This causes both the accuracy and energy to vary between different location sensor selection methods. Hence, to perform a comparison where even the naïve method of using only the GPS can achieve high accuracy, we also include a completely outdoor path in our comparisons.

Both the Android and Tilt phones have built-in GPS, WiFi, Bluetooth, and cellular radios, which were used for the experiments.

5.2 Application Accuracy Requirement

As mentioned before, mobile search is used as the application example for deriving the accuracy constraints. For the experiments, we assume that the application wishes to determine the nearest five pizza or coffee businesses. The location coordinates for such businesses in the experimental regions were obtained from available yellow pages datasets.

The dynamic location accuracy requirement for the example scenario is determined as follows. Suppose the application wants to search for a list of k entities. Intuitively, the accuracy required depends on the spatial density of the entities being searched around the user's location. If the density is high, finer location granularity is required. We wish to determine the constraint on location accuracy such that if the location error is within that constraint, the set of nearest entities can be correctly determined.

Suppose the true user location is represented by a two dimensional vector $\mathbf{x}(t)$. Suppose the nearest k entities are located within a circle of radius r around $\mathbf{x}(t)$. We *define* the accuracy requirement to be the maximum tolerable error in the location estimate such that the list of k entities nearest to $\mathbf{x}(t)$ is produced correctly, regardless of the order within the list[5].

[5]An extension to preserve the order is straightforward and omitted for brevity.

If the true location is known, determining the tolerable error is straightforward. However, this accuracy requirement must be *determined using only the estimated location* $\hat{\mathbf{x}}(t)$, without knowing the true $\mathbf{x}(t)$. The following theorem provides this accuracy constraint:

THEOREM 5.1. *Given estimated location $\hat{\mathbf{x}}(t)$, the maximum location error, $e_r(t)$, that may be tolerated while preserving the correctness of searched entity list, with respect to true location $\mathbf{x}(t)$, is given by:*

$$e_r(t) \leq \max\left\{\frac{r'_{k+1} - r'_k}{2}, \Delta(t)\right\} \quad (4)$$

where r'_n represents the radius of the smallest circle centered at $\hat{\mathbf{x}}(t)$ enclosing the nearest n entities, and $\Delta(t)$ is a threshold on the smallest error that may ever be requested. [6]

The proof is provided in Appendix A.

Thus, if after obtaining the location estimate, the estimation error is known to be within $e_r(t)$, then the list produced is same as that produced by a mobile device which knows the true location. Theorem 5.1 could be extended for the scenario where the locations of the entities being searched are not known accurately (eg., in a social networking scenario displaying a list of nearest k buddies, where each phone in the group of buddies wishes to save energy). Here $\Delta(t)$ acts as a lower bound on the best error that may be requested. It may be based on the highest accuracy available from any location sensor, or based on the user preference, such as not caring about differences in distance less than a certain threshold.

Figure 1 shows the accuracy required across Portland, Oregon for an application interested in displaying the nearest five pizza stores during meal times on a phone. The locations of pizza restaurants, obtained from a mobile yellow pages database, included 155 pizza places in and around Portland. Darker shades represent higher accuracy needed. Here, $\Delta(t)$ was set to 10% of the distance to the nearest entity (for instance, if the nearest entity is 10 miles away, the user may treat two entities at distances 10 miles and 11 miles equally suitable).

5.3 System Performance

We now evaluate the system to check if the added benefits justify the increase in complexity, compared to simply using the GPS.

The sensor accuracy models are assumed to be learned before the performance of the system is measured. In our implementation, the learning is performed simply using an extra traversal of the user path. In a real system, the models could be learned for different regions by different users who happen to reach that region first and a shared database of the models would be built up.

[6]Since the distances r'_n are measured from the estimated location, the true location is not required to be known.

The HMM for the user motion prediction is unique to the user and we do not expect the user to train their mobile device before using any location based application. Hence the HMM parameters are learned on the fly as the user moves. A uniform distribution is used to initialize the HMM and for regions where the model has not been learned.

As reasonable points of comparison for the a-Loc system, we use the following alternative strategies:

Static: This method assumes static values for error in location measured by various modalities, as has been assumed in prior work. The parameters used are the typical accuracies expected from different sensors: 10m for Bluetooth, 50m for WiFi, 150m for cell-tower, and 12m for GPS. These do not vary with location of the mobile device. Unlike previous work, we do allow this method to also use a dynamic accuracy requirement and provide it the same energy model for the sensors as used in a-Loc, giving it a fair opportunity for saving energy. This method selects the minimum energy cost sensor that is expected to satisfy the location accuracy constraint based on the static accuracy models.

Periodic: This method simply uses a single location sensor periodically, similar to the periodic use of GPS as a base case in [12, 23]. In addition to periodic GPS, we also compare to periodic WiFi, that is a likely candidate for low energy localization.

Perfect Models: The accuracy models used in our experiments have been learned by a single mobile device using one path traversal. As the system is used by more and more users, more data may be collected to refine the accuracy models. Hence we also compare against a hypothetical case where the system has perfect accuracy models at all locations for each sensor. This approach is same as a-Loc in other respects.

The above cases are represented as Static, GPS, WiFi, and Perfect respectively, in the results below.

Consider first the real mobile user experiments with an Android G1 in San Diego. After acquiring the accuracy models, the user moved on a similar path three times. The HMM was learned on the fly. The location accuracy requirement for this 0.5km path, for a search application that wishes to display coupons for the nearest five coffee shops, is shown in Figure 15. Significant slack in accuracy exists allowing sensors other than GPS to be used.

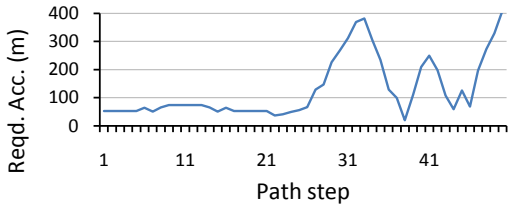


Figure 15: Accuracy requirement on experimental path.

The fraction of the path for which the required location accuracy constraint is satisfied, denoted accuracy, is plotted in Figure 16 for this path, labeled Full Path in the figure. The energy consumed is shown in Figure 17.

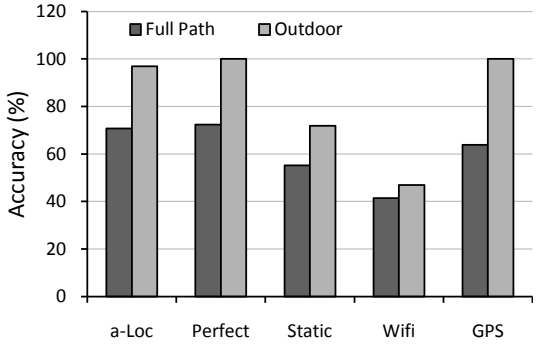


Figure 16: Fraction of the path for which the accuracy requirement is satisfied, in the San Diego experiment.

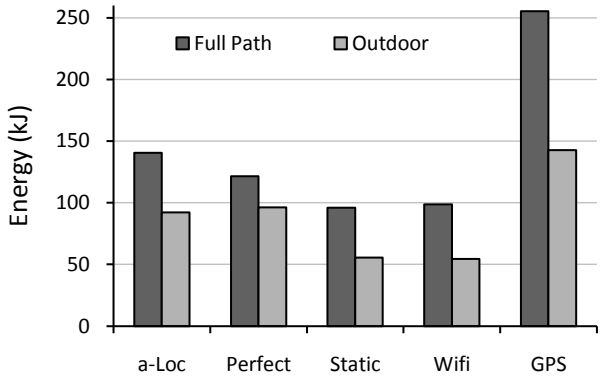


Figure 17: Energy consumption in San Diego experiments.

Clearly, if we compare the accuracy and energy use for periodic GPS and a-Loc, we see that a-Loc achieves higher accuracy with 45% lower energy use. The accuracy achieved by a-Loc is very close to that achieved when using perfect models. The energy use is also only slightly higher than the system with perfect models. This shows that not only can a-Loc reduce energy usage by exploiting the slack in accuracy requirement but also improve the accuracy by exploiting other location sensors. To make a quantitative comparison with GPS in terms of energy saved alone, we also consider the outdoor portion of the experimental path separately. On this portion, both a-Loc and periodic GPS have near 100% accuracy but a-Loc uses 35% lower energy.

The behavior of other alternatives is also shown in the above figures. While the energy use of the static algorithm is lower, its accuracy is also significantly worse than a-Loc. The same hold for periodically using only the WiFi, indicating that the dynamic models and Bayesian prediction components of a-Loc do indeed provide a benefit. The fraction of the time steps at which various localization modalities were

used are shown in Figure 18. The periodic WiFi and GPS based method only used the single respective sensor. The static approach clearly wastes energy on WiFi even when that modality is not providing sufficient accuracy because it lacks the dynamic models to determine when WiFi is a useful alternative. This region did not have a significant number of known static Bluetooth devices and we disabled that modality to avoid wastage of energy in the static method, that would have caused it to perform even worse.

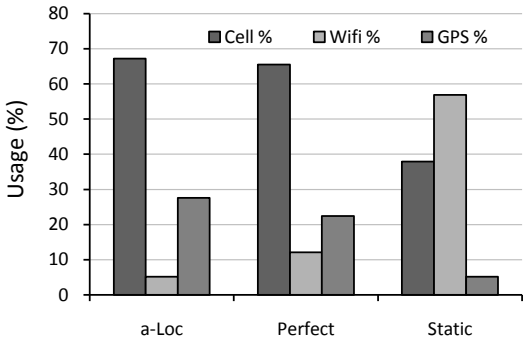


Figure 18: Modalities used by a-Loc and other approaches.

Next, we consider multiple user motion patterns, using the city-wide war-drive data for Portland, to simulate common user behaviors such as a commuter and a couple of tourists. We also include the outdoor-only portion of the Tourist-2 path separately, for a comparison with GPS where it has high accuracy. The application accuracy requirement was as shown in Figure 1 for the five nearest pizza stores. The satisfaction percentage for accuracy and corresponding energies used are shown in Figures 19 and 20 respectively. Again, a-Loc demonstrates a significant energy advantage compared to periodic GPS and even the system with perfect models, though with a small reduction in accuracy. The other alternatives perform significantly worse on accuracy, and in some cases even use more energy. Using WiFi periodically for instance, is worse for both energy and accuracy for some of the paths, compared to a-Loc.

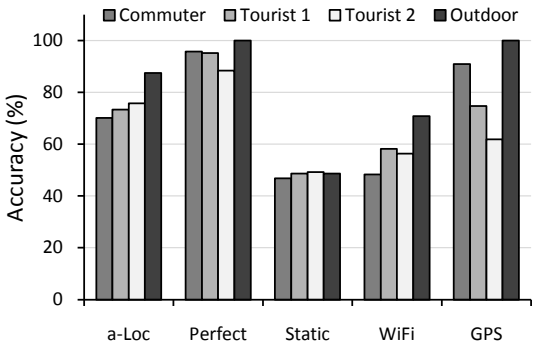


Figure 19: Accuracy achieved in the Portland.

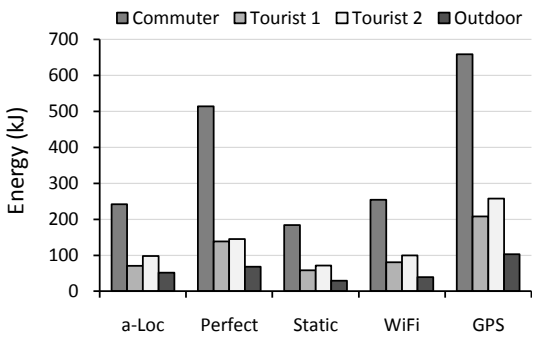


Figure 20: Energy consumption in Portland.

The experiments and simulations show that not only does a-Loc help save energy but there is significant potential to trade-off accuracy and energy based on application needs.

5.4 Discussion

The design of a-Loc and experiments presented above reveal several interesting challenges that are discussed below.

Communication Energy: Our design assumed that all location modalities used locally stored data. Since flash storage is readily available, both sensor availability model data as well as databases of interesting entities, such as mobile yellow pages, can be stored on the phone itself. Model updates can be uploaded and new shared data downloaded when the phone is plugged in, reducing the impact on battery life.

However, for certain applications, communication with the network may be unavoidable. For instance, the social networking scenario [15] requires the mobile devices to share their location data over a communication network. For certain location modalities, such as image matching [20] communication to a central server may be needed. In these cases, the communication energy cost should be considered in the energy model. For instance, if WiFi is anyway being used to send location updates, then running a scan of visible access points may be considered significantly cheaper than when not using WiFi for any other task.

The amount of data required to be communicated by a location modality may also be a concern since if only a few bytes are to be sent, short message service (SMS) may be used at a much lower energy cost than using a TCP connection over a 3G cellular data network. The energy costs for cellular data communication (TCP over 3G), experimentally measured on the AT&T Tilt phone, are presented in Figure 21. As a comparison, SMS messages used only 1200mJ on average.

These numbers were measured at 100% signal strength. At locations with weaker signal (as low as 74% could be found around our campus), the energy usage increased by up to 50%. The energy usage also increased by up to 30% when measured at different times of the day, reflecting the effect of varying network congestion. Both TCP over 3G and SMS energies showed similar trends in variation with

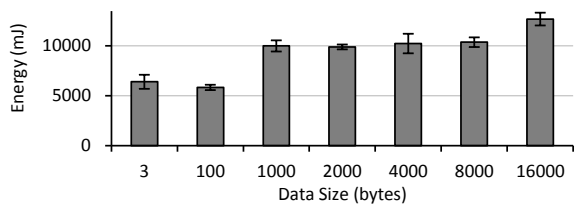


Figure 21: Communication energy measured for the 3G connection on AT&T Tilt.

signal strength and congestion.

Joint Optimization of Accuracy and Energy: In the problem formulation considered in the current work, the accuracy requirement was treated as a constraint and the problem was to minimize the energy usage. However, significant energy savings may be feasible with controlled violations of the accuracy constraints for limited time durations. A variation of the problem may then be to maximize accuracy and minimize energy simultaneously. One technique to solve such joint optimization problems is to consider a combined objective function, such as a linearly weighted sum of energy and error, and optimizing the combined objective: $C(t) = E(t)+\lambda e(t)$, where $C(t)$ is the objective function to be minimized, $E(t)$ represents energy used and $e(t)$ represents the location error achieved. Here, λ is a scalar weight that reflects the relative importance of energy and accuracy for a given scenario.

Multi-step Optimization: The sensor selection strategy presented above selects a location sensor for a single time step at a time. The problem may be extended to optimize energy over multiple time steps, potentially yielding greater energy savings. For instance, using a higher energy cost sensing modality at one time step may help improve the location prediction for multiple future time steps and avoid sensing energy expenditure in the future. Such an optimization could be modeled as a Markov Decision Problem (MDP) where the energy cost over multiple time steps is represented as an optimization objective.

6. CONCLUSIONS

We presented the a-Loc system that can automatically tune the location energy and accuracy trade-off by continually adapting to the dynamic location sensor characteristics and application needs. The end result is a system service that can free applications of the burden of location error and energy management. The structured approach systematically models multiple factors that influence location estimation using a probabilistic framework. The system also incorporates a prediction mechanism for the user's location and a method to extract the dynamic accuracy constraints for mobile search based applications. The proposed approach provides significant energy savings that go beyond existing techniques. We also showed that the a-Loc system is practically implementable on a real mobile phone with low memory and storage overheads. The design and prototype effort also revealed several additional research challenges that may lead to interesting future work.

7. REFERENCES

[1] L. Aalto, N. Göthlin, J. Korhonen, and T. Ojala. Bluetooth and WAP push based location-aware mobile advertising system. In *MobiSys*, pages 49–58, 2004.
[2] M. Azizyan, I. Constandache, and R. Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *MobiCom*, pages 261–272, 2009.
[3] P. Bahl and V. N. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *INFOCOM*, volume 2, pages 775–784, 2000.
[4] J. Barrus. Geofi: Global positioning for wifi-enabled devices. In *Ignite Where and Lauch Pad at Where 2.0*, May 2008.
[5] Bluetooth Specification. http://www.bluetooth.com/.
[6] I. Constandanche, M. Sayler, S. Gaonkar, R. R. Choudhury, and L. Cox. Energy-aware localization using mobile phones. In *Poster, ACM MobiSys*, June 2008.
[7] T. Farrell, R. Cheng, and K. Rothermel. Energy-efficient monitoring of mobile objects with uncertainty-aware tolerances. In *IDEAS '07: Proceedings of the 11th International Database Engineering and Applications Symposium*, pages 129–140, 2007.
[8] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt. Micro-blog: Sharing and querying content through mobile phones and social participation. In *ACM MobiSys*, June 2008.
[9] Google Inc., Android developers reference: Locationmanager. http://developer.android.com/.
[10] M. Gupta, S. S. Intille, and K. Larson. Adding gps-control to traditional thermostats: An exploration of potential energy savings and design challenges. In *Pervasive '09: Proceedings of the 7th International Conference on Pervasive Computing*, pages 95–114, Berlin, Heidelberg, 2009. Springer-Verlag.
[11] M. Kamvar and S. Baluja. Deciphering trends in mobile search. *Computer*, 40(8):58–62, 2007.
[12] M. B. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær. Entracked: energy-efficient robust position tracking for mobile devices. In *MobiSys*, pages 221–234, 2009.
[13] J. Krumm, G. Cermak, and E. Horvitz. Rightspot: A novel sense of location for a smart personal object. In *Fifth International Conference on Ubiquitous Computing (UbiComp)*, pages 36–43, October 2003.
[14] J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. In *UbiComp 2006: The Eighth International Conference on Ubiquitous Computing*, Orange County, CA, USA, September.
[15] A. Küpper and G. Treu. Efficient proximity and separation detection among mobile targets for supporting location-based community services. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10(3):1–12, 2006.
[16] A. Lamarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. In *Proceedings of the Third International Conference on Pervasive Computing*, May 2005.
[17] L. Liao, D. Fox, and H. Kautz. Extracting places and activities from GPS traces using hierarchical conditional random fields. *Int. J. Rob. Res.*, 26(1):119–134, 2007.
[18] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan. BeepBeep: a high accuracy acoustic ranging system using COTS mobile devices. In *ACM SenSys*, pages 1–14, 2007.
[19] J. Person. Writing your own gps applications: Part 2.

http://www.developerfusion.com, January 2005.
[20] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *CVPR '07: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.
[21] J. Sharkey. Coding for life – battery life, that is. In *Google IO Developer Conference*, May 2009.
[22] G. Sun, J. Chen, W. Guo, and K. Liu. Signal processing techniques in network-aided positioning: a survey of state-of-the-art positioning designs. *Signal Processing Magazine, IEEE*, 22(4):12–23, July 2005.
[23] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson. Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. In *ACM SenSys*, pages 85–98, 2009.
[24] M. Youssef and A. Agrawala. The Horus WLAN location determination system. In *MobiSys*, pages 205–218, 2005.

APPENDIX

A. SEARCH ACCURACY REQUIREMENT

Theorem 5.1 Proof: The time index is dropped as a single time instant is involved during this calculation. Locations are denoted using boldface variables, that represent two dimensional vectors. $\Delta(t)$ is ignored in the proof as the other term represents the correctness constraint with zero tolerance and combining with $\Delta(t)$ only allows tolerating a larger error in location.

We prove the theorem by contradiction. Suppose there exists a true location $\mathbf{x}$ such that the error in estimated location $\hat{\mathbf{x}}$ is less than e_r but there exists some entity closer to $\mathbf{x}$ than the entities in the list produced at $\hat{\mathbf{x}}$.

Suppose the entities being searched over are located at locations $\mathbf{y}_j$, where $j = \{1, ..., k, ...\}$, arranged in ascending order of their distances from $\hat{\mathbf{x}}$. The list of nearest k entities produced at $\hat{\mathbf{x}}$ is $\{\mathbf{y}_1, ..., \mathbf{y}_k\}$. Suppose some entity not in this list, say the one at $\mathbf{y}_{k+m}$, should have been in this list, and hence some entity within the list, say the one at $\mathbf{y}_{k-n}$ should not be in the list. Here n is an integer such that $0 < n < k$. Then the distances from the true location $\mathbf{x}$ satisfy:

$$r_{k+m} < r_{k-n} \tag{5}$$

Expressing distance as magnitudes of appropriate vector differences, and taking squares since distances are positive:

$$|\mathbf{x} - \mathbf{y}_{k+m}|^2 < |\mathbf{x} - \mathbf{y}_{k-n}|^2 \tag{6}$$

Considering the left hand side first:

$$\begin{aligned} |\mathbf{x} - \mathbf{y}_{k+m}|^2 &= |\hat{\mathbf{x}} - \mathbf{y}_{k+m} - \hat{\mathbf{x}} + \mathbf{x}|^2 \quad (7)\\ &= |(\hat{\mathbf{x}} - \mathbf{y}_{k+m}) - (\hat{\mathbf{x}} - \mathbf{x})|^2 \\ &= |\mathbf{r}'_{k+m} - \mathbf{e}_r|^2 \end{aligned}$$

where (7) follows by adding and subtracting $\hat{\mathbf{x}}$, $\mathbf{r}'_{k+m}$ is used to denote the vector difference $\hat{\mathbf{x}} - \mathbf{y}_{k+m}$, and we define $\mathbf{e}_r = (\hat{\mathbf{x}} - \mathbf{x})$. Given that the square of the magnitude of a vector is equal to its dot product with itself, we get:

$$\begin{aligned} |\mathbf{x} - \mathbf{y}_{k+m}|^2 &= (\mathbf{r}'_{k+m} - \mathbf{e}_r) \cdot (\mathbf{r}'_{k+m} - \mathbf{e}_r) \\ &= |\mathbf{r}'_{k+m}|^2 + |\mathbf{e}_r|^2 - 2|\mathbf{r}'_{k+m}||\mathbf{e}_r| cos\theta_1 \\ &\geq |\mathbf{r}'_{k+m}|^2 + |\mathbf{e}_r|^2 - 2|\mathbf{r}'_{k+m}||\mathbf{e}_r| \quad (8)\\ &= (|\mathbf{r}'_{k+m}| - |\mathbf{e}_r|)^2 \quad (9) \end{aligned}$$

In the dot product expansion θ_1 represents the angle between vectors $\mathbf{r}'_{k+m}$ and $\mathbf{e}_r$. Also, (8) holds because magnitudes are positive and hence the smallest value is obtained at $cos\theta_1 = 1$.

Similarly, the right hand side of (6) may be expressed as:

$$\begin{aligned} |\mathbf{x} - \mathbf{y}_{k-n}|^2 &= |\hat{\mathbf{x}} - \mathbf{y}_{k-n} - \hat{\mathbf{x}} + \mathbf{x}|^2 \\ &\leq (|\mathbf{r}'_{k-n}| + |\mathbf{e}_r|)^2 \quad (10) \end{aligned}$$

where $\mathbf{r}'_{k-n}$ denotes the vector difference $\hat{\mathbf{x}} - \mathbf{y}_{k-n}$.

Combining (6), (9), and (10) we obtain:

$$(|\mathbf{r}'_{k+m}| - |\mathbf{e}_r|)^2 < (|\mathbf{r}'_{k-n}| + |\mathbf{e}_r|)^2 \tag{11}$$

Now, take the square root. Since magnitudes are always positive, the quantity on the right has only one square root. The quantity on the left has two square roots:

Case 1: $|\mathbf{r}'_{k+m}| \geq |\mathbf{e}_r|$. Here, (11) yields:

$$\begin{aligned} |\mathbf{r}'_{k+m}| - |\mathbf{e}_r| &< |\mathbf{r}'_{k-n}| + |\mathbf{e}_r| \\ \Rightarrow e_r &> (r'_{k+m} - r'_{k-n})/2 \quad (12)\\ \Rightarrow e_r &> (r'_{k+1} + \delta_1 - (r'_k - \delta_2))/2 \\ \Rightarrow e_r &> (r'_{k+1} - r'_k)/2 + (\delta_1 + \delta_2)/2 \quad (13)\\ \Rightarrow e_r &> \frac{r'_{k+1} - r'_k}{2} \quad (14) \end{aligned}$$

where (12) holds because the magnitude of $\mathbf{e}_r$ is same as error e_r, and using scalar variables to represent magnitudes. Also, from the estimated location, since r'_{k+m} is farther off than r'_{k+1} for any $m > 1$, we can express $r'_{k+m} = r'_{k+1} + \delta_1$ using a positive quantity δ_1. Similarly, since $r'_k > r'_{k-n}$, the quantity δ_2 is also positive, and hence (14) follows from (13). But (14) is a contradiction, and hence in this case the theorem holds.

Case 2: $|\mathbf{r}'_{k+m}| < |\mathbf{e}_r|$. Then, (11) yields:

$$\begin{aligned} |\mathbf{e}_r| &> |\mathbf{r}'_{k+m}| \\ \Rightarrow e_r &> (r'_{k+1} - r'_k)/2 \quad (15) \end{aligned}$$

But (15) is a contradiction, and hence in this case also, the theorem holds.

The accuracy specified in the theorem is thus sufficient to ensure correctness. The accuracy shown is also necessary as otherwise there may exist a true location $\mathbf{x}$ for which $r_{k+m} < r_{k-n}$ leading to an incorrect result, as is easy to prove.□