SpyLoc: A Light Weight Localization System for Smartphones

Mostafa Uddin
Old Dominion University
muddin@cs.odu.edu

Tamer Nadeem
Old Dominion University
nadeem@cs.odu.edu

ABSTRACT

In this paper, we design, implement and evaluate the SpyLoc localization system. The design goal of SpyLoc is to develop a light weight, high accuracy, energy efficient and privacy preserving localization system for off-the-shelf smartphones. SpyLoc leverages both the acoustic interface (microphone/speaker) and the Wi-Fi interface at the kernel-level of the smartphones as well as the inertial sensors in the smartphones to achieve high localization accuracy. SpyLoc does not require any central controller unit nor any collaboration with nearby devices. Furthermore, in SpyLoc, each user's smartphone works autonomously to estimate it's location. We implement and evaluated the complete SpyLoc using commercial off-the-shelf smartphones. Our result shows that SpyLoc can achieve less then 1 meter accuracy for more than the 90% of the time for both indoor and outdoor environments.

1. INTRODUCTION

In many indoor/outdoor environments (e.g., airport terminal, railway station, shopping mall, office building, parking lot, etc.), knowing the user's location with high accuracy would enable several interesting applications and services. Several of the earlier localization works have been based on Radio Frequency (RF)-based techniques that leverages signal strength of RF signal from different nearby RF sources or infrastructures (e.g., WiFi access point, cellular tower).

Recently, in order to increase localization accuracy and reduce the overhead associated with the RF-based schemes, several works propose to utilize new sensing modalities and even to combine multiple modalities together to achieve high accuracy localization system [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]. For example, the localization schemes [1, 2, 3, 4] use an acoustic-based ranging scheme [11, 12, 13] combined with WiFi-based localization scheme. Other schemes focus on utilizing the inertia sensors in smartphones individually [14, 15] or in combination with other modality such as WiFi-based scheme [16]. Based on the literature review, we summarize the following problems with the existing schemes in achieving high accurate localization system for smartphones in practical environments:

- RF-based approaches, especially for indoor, could only achieve limited accuracy with error range of 6-8m even with sophisticated localization algorithms. Recent works show that existence of the same signature or fingerprint of RF signals at different distinct location prevents from achieving high accuracy localization system [1, 2, 7]. In addition, building an maintaining the RF fingerprint database for the environment is very tedious task and require significant overhead. Furthermore, the RF-bases schemes assume the user to be stationary during the collection of the RF fingerprint at his current spot, which make it not suitable for users with high mobility.
- High accuracy indoor localization systems that utilize the acoustic modality with the RF require either customized hardware or extra infrastructure hardware [3, 4, 17, 18], which make it not applicable to off-the-shelf smartphones. Moreover, recent schemes proposed for smartphones require some form of collaboration between several smartphones in the neighborhood, and/or exchanging certain information with a central controller. These requirements, in addition to the need of the RF fingerprints, hinder the usage of these localization systems in practical environments. In addition, these systems overlooked the impact of the user mobility on the system performance [1, 2].
- Other approaches, which utilize existing inertial sensor in off-the-shelf smartphones [15, 16], highly depends on the knowledge of the environment. For example, the proposed system in [15] highly depends on the layout of the building, which is not practical in large open public spaces (e.g. airport, metro station, shopping mall). The UnLoc system [16] requires to know enough landmarks of the environment in order to calibrate the estimated location using inertial sensors. However, besides the overhead of the initial training phase to build the database of landmarks, the UnLoc performance deteriorates when there is no enough distinguishable landmarks.

In this paper, we design and develop a light-weight high-accuracy indoor/outdoor localization system (*SpyLoc*) for off-the-shelf smartphones. SpyLoc leverages both the acoustic interface (microphone/speaker) and the Wi-Fi interface at the kernel-level of the smartphones as well as the inertial sensors in the smartphones to achieve high localization accuracy.

The proposed system uses a combination of both ranging-based and dead reckoning based localization approaches. In the ranging-based approach, SpyLoc utilizes the Time Difference of Arrival (TDoA) between the acoustic and the radio-frequency (RF) (i.e WiFi) signals by leveraging the slow propagation speed of the acoustic signal with respect to the RF signal for ranging estimation. In the dead reckoning-based approach, SpyLoc fuses the inertial sensors of the smartphone to estimate the direction and the distance of user's movements.

The basic idea of the SpyLoc is to leverage the benefits of both the dead-reckoning and the ranging scheme to build a practical localization system. Given the high errors of the inertial sensors especially indoor, SpyLoc uses a novel ranging scheme based on both the acoustic and WiFi interfaces to mitigate this error in order to improve the localization accuracy. Unlike the ranging-based or RF-based localization schemes that require multiple reference points (e.g., access points), using the dead reckoning in SpyLoc reduces the required number of reference points to only one reference to locate at track users accurately. This low dependency on ranging scheme and the no requirement of any initial training phases or calibrations make SpyLoc light-weight system and practically applicable to high mobile users. Moreover, this low dependency, in addition to no transmission requirement from the user's device, make SpyLoc energy efficient for the user's smartphone. Furthermore, SpyLoc does not require the interaction with any central controller or with any nearby device and each user's smartphone works autonomously to determine its location. Hence, the user's privacy is preserved in SpyLoc.

We summarize our contribution of this paper as follow:

- Implementation of SpyLoc; a light-weight high-accuracy localization system using off-the-shelf smartphones.
- Evaluation of SpyLoc under several real scenarios and different mobility conditions.
- Development of a step detection algorithm to efficiently detect the user's step using inertial sensors. Given different users have different steps and speeds, the proposed algorithm is adaptable to detect and estimate different step lengths corresponding to different user's speed and different users.
- Development of a robust direction detection algorithm that infer user's direction by fusing multiple sensors of the smartphone.
- Detailed study of the Line-of-Sight (LoS) and Non-Line-of-Sight (NLoS) acoustic signals and development of a classification method to differentiate between LoS and NLoS signals.
- Detailed study of the sound driver in Linux-based smartphones (e.g Android, Maemo etc.).

To the best of our knowledge, SpyLoc is the first localization system that utilizes the acoustic system of the smartphone at kernel-level, in addition to the inertial sensors, to improve the accuracy of the range estimation. In addition, we believe that the study of the sound signal propagation for both LoS and NLoS scenarios is a new contribution for localization systems.

The paper is structured as follows. We introduce SpyLoc with brief discussion about each component in the system in Section 2. In Sections 3-6, we describe in detail the different modules that construct client components of SpyLoc. We evaluate the performance of the system under different indoor/outdoor scenarios in Section 7. Then we discuss the related works in Section 8 and conclude with Section 9.

2. SPYLOC LOCALIZATION SYSTEM

Figure 1 shows the different modules of SpyLoc system. Similar to many localization systems, SpyLoc consists of two main components: infrastructure component that runs on infrastructure hardware, and client component that runs on the user's device (i.e. smartphone). In this paper, we refer to an infrastructure device running the SpyLoc infrastructure component by a *beacon device*. The

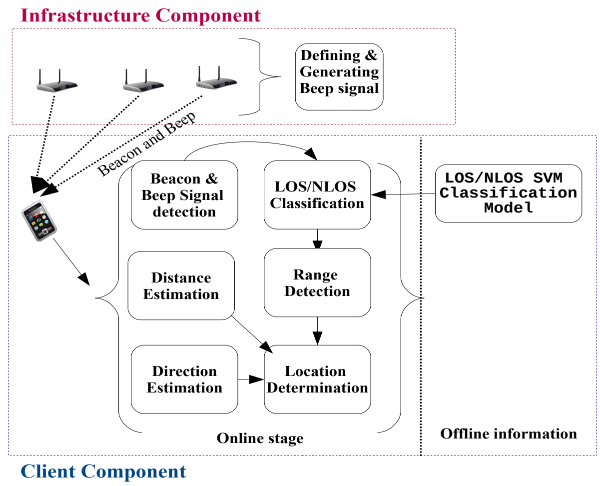


Figure 1: Architecture overview of the different components of SpyLoc localization system

beacon device periodically broadcasts a RF message (i.e. Wi-Fi beacon frame), which we refer to as a *beacon*. In addition, the beacon device also generates an acoustic signal, *beep* following each broadcasted beacon message. In a practical scenario, a beacon device could be an Access Point (AP) with additional acoustic interface (i.e. speaker, mic and sound driver). A user's smartphone running SpyLoc client component (e.g., SpyLoc application) will capture the beacon messages and the corresponding beep signals from the surrounding beacon devices. Using the captured beacon messages and the corresponding beep signals in addition to the inertial sensors in smartphone, SpyLoc application will infer the user's location.

In a typical usage scenario, while the user runs the SpyLoc application in the smartphone, it will initially collect the beep signals from the surrounding beacon devices along their corresponding beacon messages. Each captured beep signal at user's smartphone will be classified either it is a Line-of-Sight (LoS) or a Non Line-of-Sight (NLoS) signal with respect to its corresponding beacon device. Only for the LoS signals, the application determines the relative ranges between the user's smartphone and the corresponding beacon devices. Then, the application combines three estimated ranges from three different beacon devices to estimate the user's initial location. After fixing the user's location, SpyLoc uses the inertial sensors to detect the distance and the direction to track the next user's locations. On a periodic bases, SpyLoc uses an estimated range to a single LoS beacon device to calibrate and calculate the accurate location of the user.

SpyLoc localization system has the following interesting features:

- SpyLoc exploits the multiple interfaces in the smartphones by utilizing both the RF interface and acoustic interface to design high-accurate localization scheme.
- SpyLoc does not require any central controller or backend server to coordinate between neighboring devices. In this system, user's smartphone calculate the location locally and no collaboration with neighboring devices is required. In addition, The user's application does not require to share any acoustic or WiFi information with any nearby device or access point. This enables to preserve and protect the user security and privacy.
- The SpyLoc application running by the users does not re-

quire any customized hardware. This enables any off-the-shelf smartphone to adopt and use the developed application.

- SpyLoc only needs one ranging estimation from the beacon device after knowing the user's initial position. In addition, such ranging requirement is only required to calibrate the estimated location by the dead reckoning approach. This enables SpyLoc to be light-weight and practically applicable during user's mobility.

- SpyLoc only requires one way transmission; the transmission of the beacon messages and the beep signals by the beacon devices. Furthermore, estimating one range from one beacon device eliminates the need to switch the Wi-Fi channel to receive multiple beacon messages from the multiple beacon devices. Therefor, the development of the SpyLoc is energy efficient application for user's smartphones.

In the following subsection, we provide a brief overview of the infrastructure component and the client component.

2.1 Infrastructure Component

The beacon device (e.g. Wi-Fi AP), which runs the infrastructure component of the SpyLoc system, periodically generates a RF beacon message followed by a beep signal. A single frequency sinusoidal acoustic signal defines the basic sound that we refer to as *tone* in the paper. A mixture of such tones (i.e., set of frequencies) defines the beep signal. Typical human hearing perception diminishes after 18kHz. Therefore, we utilize the 18kHz-21kHz audio frequency range, which is perceptible to the most of the off-the-shelf smartphones [19]. From experiments, we found that if the frequency space between two adjacent tone is 250Hz, then it is sufficient to avoid the interference and detect the tones on the client side. Therefore, we select 10 tones(i.e frequencies) $(f_1, f_2, ..., f_{10})$ from 18kHz-21kHz audio range, we can have up to 2^{10} unique beep signals. One of the major challenge in the infrastructure component is, *How to uniquely and autonomously define the tones of the beep signal for each beacon device?*

In selecting the tones we utilize the last ten bits of the MAC address from the Wi-Fi Interface card of the beacon device. In this last ten bit sequence, each bit position correspond to one tone among the 10 tones, $\{f_1, f_2, ..., f_{10}\}$ (i.e. the $0th$ bit map to f_1, $1th$ bit map to f_2, and so on). A value of 1 in one bit position indicates that corresponding tone will exists in the beep signal and vice versa for the value of '0'. For example, if the MAC address of a beacon device is $c4 : 2c : 03 : 3a : 2c : a1$, that has last ten bits as 0010100001, then the selected tones of the beep signal for that beacon device would be $\{f_1, f_6, f_8\}$. Selecting a tone is such way has almost 1% ($= (10 * 100)/2^{10}$) chance of having same tones for two different beacon device, if we consider 10 beacon devices in one place within each other proximity. Note that, at different public places we didn't found any cases where two Wi-Fi AP have same last 10 bit MAC address in one place.

In the implementation, we use the Wi-Fi beacon frame as our beacon message. Therefore, In SpyLoc system, each beacon device add their location information in the payload of the Wi-Fi beacon frame. Thus, SpyLoc client application knows the location of the beacon device by receiving the beacon message.

In section 3.1, we describe more details about the beep signal detection technique, and how to map the beep signal to its corresponding beacon frame. In the implementation, we use a smartphone (i.e. Nokia N900) with a speaker (i.e. Nokai MD-11) as a beacon device. In Section 5, we describe the implementation details of the beacon device using smartphones as well as the implementation details of the beep signal generation that follows each beacon message.

2.2 Client Component

In SpyLoc, client component consists of five main modules described in the following:

- **Distance Estimation module** Distance estimation module estimate the distance at each step. First, In order to estimate the distance, this module applies an adaptable step detection algorithm. Second, it utilizes a personal step model to infer the user's step length. The SpyLoc client application utilizes both the ranging scheme and the step detection algorithm to build the step model online.

- **Direction Estimation** Direction estimation module fuses multiple inertial sensors such as the geomagnetic sensor and the gyroscope sensor to infer the direct of the user. Note that, running the SpyLoc client application, for the first time, requires little effort from the user to calibrate the sensor reading in order to infer user's direction using this module. In section 4, we describe both distance and direction estimation modules in more details.

- **Non Line-of-Sight (NLoS)/ Line-of-Sight (LoS) Detection module:** The NLoS/LoS detection module consists of a binary classifier model and feature extraction components. We prebuilt this binary classification model through an offline training. The SpyLoc client component only utilizes the model to classify whether a received beep signal is LoS or NLoS signal. Once the SpyLoc application at user's smartphone receives a beep signal, it uses the feature extraction component to extracts the features from the beep signal to detect whether the beacon device corresponding to a received beacon is in the LoS or in the NLoS of the user's smartphone.

- **Range Detection module:** For a LoS beep signal, range detection module is responsible to estimate the user's relative distance to the corresponding beacon device. This module uses the Time Difference of Arrival (TDOA) of the beacon message and the corresponding beep signal to estimate the relative distance between the user's smartphone and the beacon device.

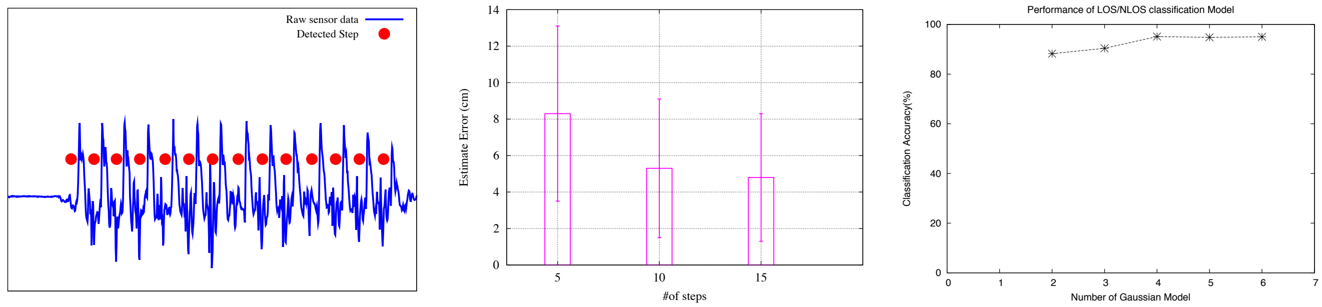
- **Location Determination module:** The Location determination module estimates the user location based on the outcomes of the range detection, distance estimation, and direction estimation modules. Knowing the current location of the user, this module estimate the next possible locations using the distance and direction estimation modules. However, at that time, If the range detection module provides the relative ranging of the user's smartphone for at least one beacon device (i.e., when the user is in LoS with at least 1 beacon device), then this module applies the outcome of the estimated range to calibrate the user's location. Under certain circumstances(e.g. locate user's starting point, and building step model) the range detection module utilize the relative ranging of the user's smartphone for at least three beacon devices (i.e., when the user is in LoS with at least 3 beacon devices), to apply triangulation technique to estimate user's location.

In the following, we describe each of the client component modules in more details.

3. DISTANCE ESTIMATION MODULE

This module has two submodules i) *Step Detector*, and ii) *Personalized Step Model*

i) Step Detector: This submodule uses the commonly used accelerometer and the gyroscope sensors of the smartphone to detect



9(a) 9(b) 9(c)

Figure 3: (a) Step detection in raw sensing data. (b) Accuracy of the step model in estimation average step length. (c) Classification Performance of our LoS/NLoS Detection Technique

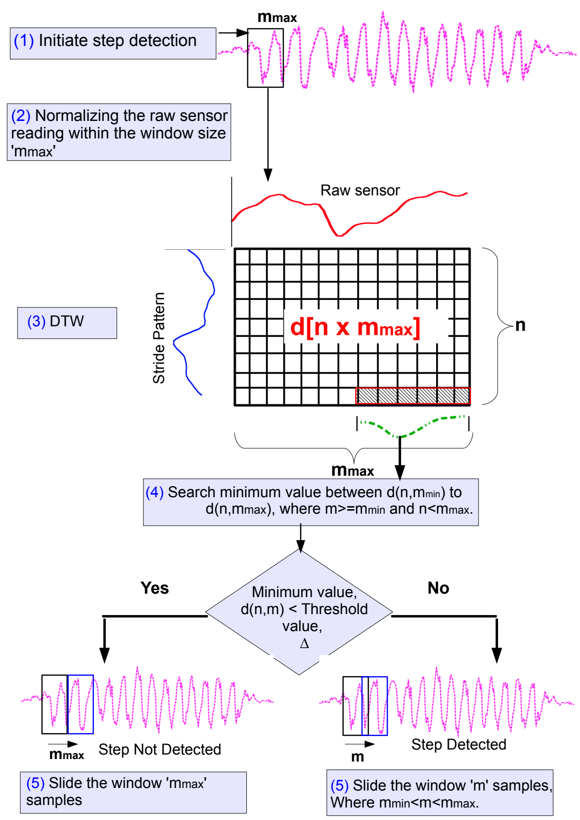


Figure 2: Adaptable Step Detection module in SpyLoc localization system.

and track user steps. Figure 2, shows the details of our adaptive step detection algorithm. Given the step detection should be activated when user moves, we use a change in gyroscope sensor reading above certain threshold as an indication of movement and initiates the step detection algorithm to start capturing the accelerometer data. In our implementation, we set this threshold value to 0.3. Unlike previous algorithms [14, 15, 16], we apply our step detection algorithm over a fixed size (m_{max}) batch of raw accelerometer samples to simplify and improve the computation efficiency of our algorithm. Given that step length is proportional to the walking speed [20], m_{max} represents the maximum length of a person step in terms of samples size and defined as follows,

$$m_{max} = \frac{s_{max}}{v_{max}} \times f_a, \tag{1}$$

where s_{max} is the maximum length of a person step, v_{max} is the maximum walking speed of person, and f_a is the collection frequency of accelerometer samples from the smartphone. Similarly, m_{min}, which is the minimum length of a person step in terms of samples size, is defined using s_{min} and v_{min} values. We used the $s_{max}, v_{max}, s_{min}$, and v_{min} values defined in [20] in our step detector algorithm. Since we used $f_a = 50$ samples/sec in our implementation, the corresponding m_{max} and m_{min} we used are 65 and 25 respectively.

After collecting m_{max} raw 3-axis accelerometer samples, we calculate the vector magnitude of each 3-axis accelerometer sample in order to make our algorithm independent of the user orientation. Similar to the other algorithms [14, 15, 16], we apply the IIR (Infinite Impulse Response) low pass filter to reduce the impact of the noise of the m_{max} samples. Then, we normalize the 'm_{max} samples before feeding it to the Dynamic Time Wrapping (DTW) algorithm [14]. The DTW algorithm compares the similarity between the predefined step pattern (with size n samples where $n \leq m_{max}$) and the captured m_{max} samples to detect whether a step exists within the m_{max} samples. Unlike correlation and threshold-based methods [15, 16], DTW adaptively detect user steps regardless of the different lengths corresponding to different walking speeds. In our implementation, we used the average of several walking samples from multiple users under different speeds to construct the predefined step pattern used in our system. From the experiments, we found that our predefined step pattern is of size $n = 45$ samples.

The DTW algorithm calculates $d[n \times m_{max}]$ matrix scores with positive values. The lower score of $d[i, j]$ indicates a better matching between predefined step pattern of size i and the captured samples of size j. Unlike common use of the DTW algorithm [14], we search for a cell $d[n, m]$ with the minimum value within $d[n, m_{min}]$ and $d[n, m_{max}]$ cells. (the minimum value at the green dotted curve In the Figure 2). If this minimum value is below a certain threshold Δ, then a step length of m samples is detected. Otherwise, there is no step detected within the captured m_{max} sample. By conducting several experiments, we set the threshold Δ to 0.4 in our implementation. If a step is detected, then we shift the searching window for detecting the next step by m samples. Otherwise we shift it by m_{max} samples. Figure 3(a) shows how accurately the steps detected by our scheme match the actual accelerometers samples corresponding to user steps in a walking experiment.

ii) Personalized Step Model: We use the commonly used following step length model [14, 15] as our personalized step model,

$$s = a \times f + b, \tag{2}$$

where s is the step length, f is the frequency of steps, and a, b are person-dependent constants. Unlike others, we build this model for each person with no or minimal efforts required from the user. In order to define the personalized step model, we have to calculate the constant parameters a, b for each user. SpyLoc utilizes the ranging scheme to track the user's consecutive locations using at least 3 ranges from three different beacon devices. Then the system matches those locations with the step counting to build the step model (2) of the user. Figure 3(b) shows the estimated step length error by building the step model from different number steps.

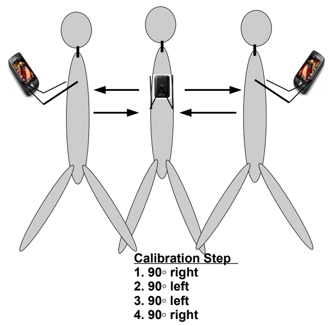


Figure 4: Calibration step for the direction detection module.

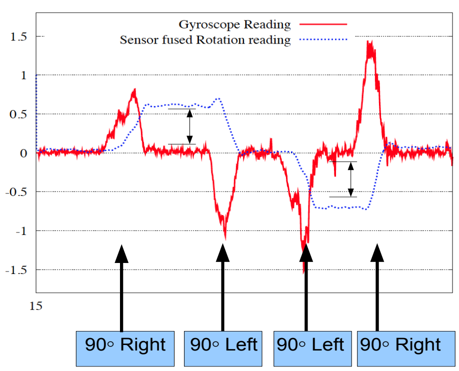


Figure 5: Sensor reading during calibration step.

4. DIRECTION ESTIMATION MODULE

The direction module utilizes the commonly used geomagnetic and gyroscope sensors to detect the direction of the user. In indoor environments, using the inertial sensor of the smartphone to detect the direction of the user's movement is extremely critical due to surrounding noises from the ferromagnetic devices. Therefore in the implementation, we use the function *getRotationMatri* provided by the android API to collect much cleaner ration sensor reading. We collect this reading whenever we detect a step from the step detector module. Note that, the rotation sensor reading provides the value of $\sin(\theta/2)$, where θ is the rotation made by the user. However, this rotation sensor is not totally error free. It shows some random nature in providing the angle, therefore we follow the steps in figure 4 to calibrate the rotation sensor reading in order to reduce the effect of that random noise.

Figure 5, shows the gyroscope sensor reading along with the rotation sensor reading while doing the calibration. During the calibration, we observe different value of θ from the rotation sensor reading for the same angle movement. For example, we receive two different values, 0.707 and 0.67 from the rotation sensor for same 90° movement. Thus we assume the following gaussian model for the rotation sensor reading θ,

$$\theta = \mathcal{N}(\theta_\mu, \theta_\sigma), \tag{3}$$

where θ_μ is the mean and θ_σ is the variation, We build the above model (3) by repeating the calibration steps several times(e.g. 4-5 times). In the implementation, we did the calibration process just five times before using the SpyLoc application to track the user positions. Later in location determination module (section 6), we use the model (3) to predict the possible locations of the user.

5. NLOS/LOS DETECTION MODULE

In this section, we start with describing the how SpyLoc client detect and capture the beep signal. Next, we analyze some common wavelet patterns of the beep signal under different LoS/NLoS conditions. In addition, we describe some anomalies that we have observed. Finally, we describe the binary classification model (LoS/NLoS) and the features we use to differentiate the beep signals from LoS or NLoS beacon device.

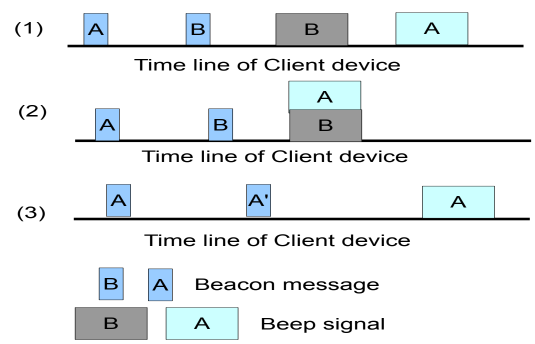


Figure 6: Time line of receive beacon message and beep signal at client device.

5.1 Beep Signal Detection

Precise detection of the beep signal at the receiving side is crucial in estimating the accurate range between the user and the beacon devices. We use the method described in RF-Beep[21] to detect the different tones in the beep signal. In addition, mapping the beep signal to its corresponding beacon frame from a particular beacon device is also challenging. This challenge has also been addressed in Cricket [3] system, where they fundamentally transmit a customized RF frame for a long duration to overlap audio tone transmission. However, such solution is not practically adaptable with the existing RF infrastructure (e.g. Enterprise Wi-Fi Network). In SpyLoc, we exploit the existing RF infrastructure except adding the audio interface. For example, In section 2.1, we have mentioned, how we are using the periodic Wi-Fi beacon frame from the Wi-Fi AP as our beacon message. Therefore, In SpyLoc we had to the address the mapping challenge in a different way without effecting the communication of the existing RF infrastructure.

Figure 6 shows three challenge scenarios of mapping the received beep signal to its corresponding beacon message. Considering the first scenario, where the client device first receives a beacon message from the beacon device 'A' followed by the beacon message from beacon device 'B'. Thereafter, the client device receives the beep signal from the device 'B' before receiving the beep signal from device 'A'. In such scenario, we need to define the beep signal in a unique way for each beacon device, therefore we can prevent the ambiguity of mapping the beep signal to the correct beacon message. In section 2.1, we describe a way to define the beep signal in an unique way. In second scenario, client device receives the beep signals from two device 'A' and 'B' at same time. In this scenario, it is not always possible to distinguishable two beep signals unless they both have some uncommon tones in their beep signal. However, if not, then only one beep signal will get detected which have the higher number of tones in the beep signal. In third scenario, the client device receives an another beacon message before receiving the beep signal of the previous beacon message. In the implementation, we use a timeout of 100ms which is the normal periodicity of the beacon message (i.e. beacon period). Considering the speed in the air, the timeout limits our range of detecting the beep signal is up to 33-35 meter, which is also the typical range of a Wi-Fi at indoor environment [7]. In section 6, we describe more detail about implementing the beep signal detection technique in off-the-shelf smartphones.

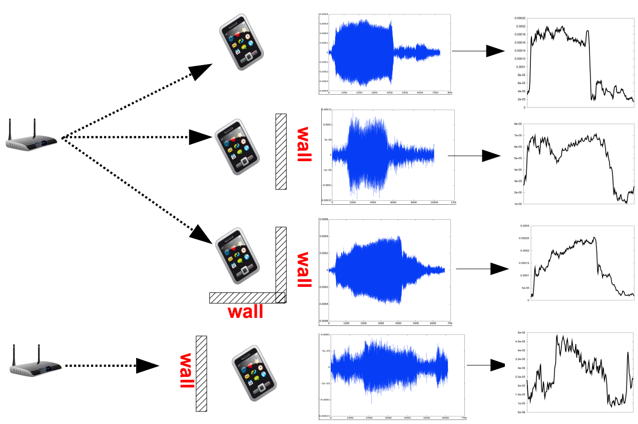


Figure 7: Detected beep signal under different LoS/NLoS conditions.

5.2 Beep Signal under LoS/NLoS Conditions

In this section, we describe the different observed patterns of the beep signal for different LoS and NLoS scenarios. Figure 7 shows typical arrangement configurations for both beacon device and the user's smartphone. In the first arrangement, the smartphone is in the LoS of the beacon device with no wall around the user's smartphone. In this configuration, the detected beep signal shows a flat with sharp start/ending pattern. For the second arrangement with a wall right behind the user's smartphone, the beep signal shows two spike; one from the direct path signal and the other from the reflected path signal of the wall. In the third case, where the user's smartphone is at the corner of two walls the beep signal shows a monotonous increment of the amplitude due to the aggregation of two reflected signal from the walls. All these patterns of the LoS beep signal have been verified by repeating the experiments over multiple times. Unlike LoS case, we observer that in NLoS case the beep signal has high variations pattern with multiple number of spikes. In addition, the amplitude of these multiple spikes also changes over time for NLoS cases. However, none of the patterns in the NLoS case matches with any of the LoS cases.

Figure 8 shows some anomaly patterns for LoS/NLoS cases. For example, the first pattern is corresponding to placing the user's smartphone in a long narrow corridor. In this scenario, the beep signal shows two spikes with short distance between them. One pike is corresponding to the direct path signal while the other one is corresponding to the reflected path signal from the surrounding walls. Another anomaly scenario is when the user body blocks the direct path signal from the beacon device. In this scenario, the pattern of the beep signal is not consistent. We have observed several different patterns under this scenario. Based on these observed patterns, we describe in the next subsection the classification model we built to detect whether the received beep signal is corresponding to NLoS or LoS scenarios.

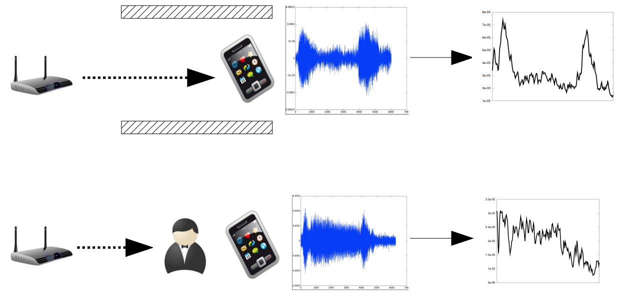


Figure 8: Detected beep signal under different anomaly conditions

5.3 Classification of LoS/NLoS Signals

This section describe both the training and testing phase of the classification model we developed to detect whether the beep signal is corresponding to a LoS or a NLoS scenario. In training the classification model and select the corresponding features, we apply the following steps:

- The beep signal is a time series data with both positive and negative values. Therefore, we start with taking the absolute values of the time series data of the beep signal. Since the amplitude of the beep signal reduces along the distance, we normalize the beep signal's amplitude to minimize the effect of the distance. Let $R = \{r_1, r_2, ..., r_n\}$ is the normalized samples of the beep signal of length n. Note that the length of the beep signal is not fixed and can vary from one signal to another due to the multi-path effect. In R, we assume r_i are independent and identical distributions.

- Next, we find the initial values of the Gaussian mixer model, $\theta_j = \{\alpha_j, \mu_j, \sigma_j\}$ using the expectation maximization algorithm [22] for $j = 1, ..., m$, where m is the number of the gaussian models, α_j is the weight vector, μ_j is the mean vector, and σ_j is the variance vector. We use the normalized time series beep signal (R) for estimating the likelihood parameters of the Gaussian models.

- After estimating the gaussian models, we compute the following $R(i, j)$ which is a $n \times m$ matrix,

$$R(i, j) = \frac{1}{\sigma_j \sqrt{2\pi}} exp(\frac{1}{2}(\frac{r_i - \mu_j}{\sigma_j})^2) \qquad (4)$$
$$i = 1, ..., n$$
$$j = 1, ..., m$$

where n is the length of the beep signal samples.

- Next, we calculate the probability density function for the i-th normalized sample of the beep signal given the above gaussian model

$$P(r_i|\Theta) = \sum_{i=1}^{m} \alpha_j R(i,j) \tag{5}$$
$$i = 1, ..., n$$
$$j = 1, ..., m$$

where Θ is the gaussian mixer model parameters.

- Finally, we reduce the variable length time series of the beep signal to a single size features using the following log-likelihood equation,

$$F = \sum_{i=1}^{n} \log P(r_i|\Theta) \tag{6}$$

The features values obtained from each input time series samples are used as input dataset for training and testing our SVM classifier model [23]. The SVM model basically performs binary classification to detect whether the received beep signal is corresponding to LoS or NLoS scenario. We experimented with different number of gaussian models to verify the performance of our SVM model as shown in Figure 3(c). We collected 200 LoS beep signals and 100 NLoS signals for our dataset. We used 10-fold cross validation over the dataset to calculate the overall performance (Figure 3(c)).

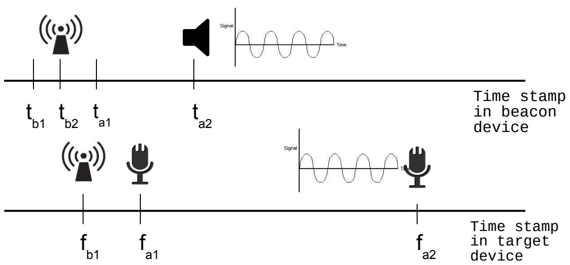


Figure 9: Overview of the Range Detection Scheme

6. RANGE DETECTION MODULE

The range detection module builds on [21] in estimating the relative ranging between smart devices. We start this section by describing an overview of the earlier proposed ranging scheme in [21]. Then, we describe details on how the the sound driver operates in Linux-based devices and the associated challenges. Finally, we describe how to enhance the accuracy of the ranging estimation by describing our novel temperature estimation for smartphones.

6.1 Range Detection Scheme

The work in [21], proposed a novel range detection scheme based on the Time-Difference-of-Arrival (TDoA) technique that utilizes the relative velocity of two different signals; RF and acoustic. Figure 9 shows an overview of this ranging scheme and the time-line of timestamps (summarized in Table 1) corresponding to different events at both the beacon device and the user's smartphone.

In this range detection scheme, the user's smartphone only receives both the Wi-Fi beacon and the beep signal in which it does not require to share any information with the beacon device. Given the high propagation speed of the Wi-Fi signal and the small length of the Wi-Fi beacon, we approximate both timestamps t_{b2} and f_{b1} to represent the same event on both timelines. Given the speed of the sound in air is s_a, the distance between the *beacon device* and the *target device* is D could be written as follow:

$$D = s_a.(t_{a2} - f_{a2})$$
$$D = s_a.(t_{a2} - f_{a2} + t_{b2} - t_{b2})$$
$$D = s_a.((t_{a2} - t_{b2}) - (f_{a2} - f_{b1}))$$
$$D = s_a.(\Delta t_{ab} - \Delta f_{ab}) \tag{7}$$

Both Δf_{ab} and Δt_{ab} values are measured locally at the user's smartphone and the beacon device respectively. Therefore, it is worthwhile to point out that there is no need for any type of synchronization between the two devices in order to calculate Δt_{ab} and Δf_{ab}.

The accuracy of the ranging method highly depends on the precision and the accuracy of measuring the arrival times of two different signals. In the next subsection, we describe the detail of the sound driver and corresponding challenges.

Table 1: Timestamps corresponding to beacon device and user's device events in the proposed range detection scheme.

Timestamp Parameters	
t_{b1}	Time when the beacon device puts the Wi-Fi beacon into the transmission buffer of the Wi-Fi driver.
t_{b2}	Time when the last bit of the Wi-Fi beacon is emitted from the beacon device's Wi-Fi interface.
t_{a1}	Time when the audio driver starts writing audio frames into the audio hardware buffer.
t_{a2}	Time when the speaker starts to generate the beep sound from the beacon device
f_{b1}	Time when the user's smartphone receives the last bit of the Wi-Fi beacon.
f_{a1}	Time when the microphone of the user's smartphone captures the audio samples in the audio driver buffer.
f_{a2}	Time when the user's smartphone detects the starting of the beep sound from the captured audio samples.

6.2 Operation of Sound Driver

In the range detection module, it is important to accurately timestamp the several events in both playing and recording sound. This requirement motivates us to understand the operation of the sound driver of the Linux-based smartphones (e.g. Android phones, N900, Maemo 5 phone etc.). In this section, we describe he general operation of the sound driver and how the sound samples are transferred between the user-space and the actual hardware thru the sound driver in both playing and recording sound operations.

In smartphones, the sound operation consists of two main components, i) the kernel-level sound driver and ii) the sound library. User applications can only access the sound library to communicate with sound driver for playing and recording sounds. The sound driver within the kernel has three layers: 1) the low-level layer that is responsible for accessing the hardware and is implemented as callback functions, 2) the middle-level layer that supports the common routines for different sound hardware components, and 3) the top-level layer that is the entry point for the sound library.

The sound library typically transfers the sound samples from the user-space application buffer to the Direct Memory Access (DMA) buffer within the kernel-space. This DMA buffer is also shared between the sound driver and the sound hardware. The DMA buffer implements a "ring buffer". The DMA "ring buffer" has two pointers, the application pointer that points to the current sample under processing by the application, and the hardware pointer that points to current sample under processing by the DMA. In major sound

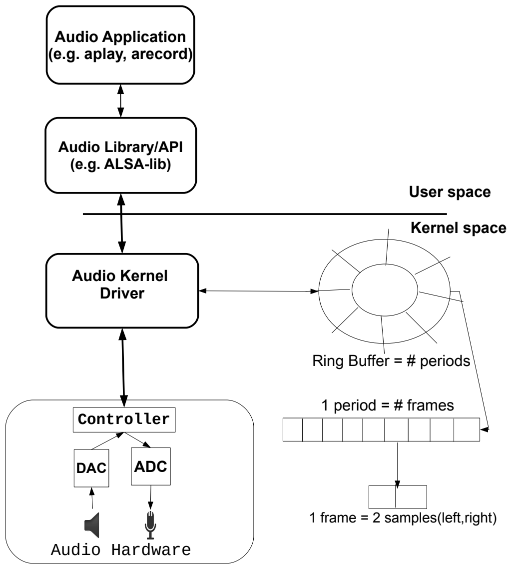


Figure 10: Operation Flow of Playing and Recording Sounds in Linux-based Smartphones.

hardwares, the DMA buffer is divided into several "epoch"s. The pointers (hardware and application) of the DMA buffer via, usually, the invocation of interrupts get updated at the end of each "epoch". The DMA buffer size is equal to the number of epochs multiplied by the "epoch size". The "epoch size" is the number of frames that is corresponding to one "epoch". The frame is the smallest unit consists of several sound samples.

Each of the playback stream (i.e. while playing sound) and the capture stream (i.e. while recording sound) has its own DMA buffer. In playing a sound, the application writes the sound samples to the DMA buffer within the kernel-space. After writing one epoch size of samples, sound driver initiates an interrupt operation to start the DMA data transfer from the DMA buffer to the hardware buffer. Following the completion of the interrupt, the sound driver updates the pointers of the DMA buffer thru the corresponding callback function. By recording the timestamp of this update operation, we capture the timestamp actual transmission of the sound signal event. For the sound recording operation, the hardware interrupts the sound driver to update the buffer pointers after each filling of the DMA buffer with one epoch size of sound samples. By recording the timestamp of this interrupt, we will be able to predict the actual time of receiving the sound signal.

One significant challenge is the high variation in the time delay between issuing the transmission command of the acoustic signal by the application and the actual emission of the acoustic signal by the speaker. In [21], we discussed our finding that the delay in the acoustic signal transmission consists of three main components that correspond to the following three actions: i) Powering up the playback stream, ii) Data transfer from the application to the sound driver, and iii) DMA data transfer from the sound driver to the actual sound hardware. Thru experiments using Nokia N900 smartphones, we showed that this delay varies significantly between 2 and 6 milliseconds (solid line in Figure 11), which lead to few meters of error in measuring the relative range module. However, we showed that we could limit this delay to less than 1 millisecond (dotted line in Figure 11) by implementing the range estimation scheme at the driver-level instead of at the application-level as in previous work.

We also showed that the delay in receiving the beep sound at the user's smartphone correspond to two actions: i) Powering up the capture stream, and ii) Detecting the starting event of the captured Beep sound. Similarly, we showed how inout proposed ranging scheme to limit the impact of this delay on the ranging accuracy.

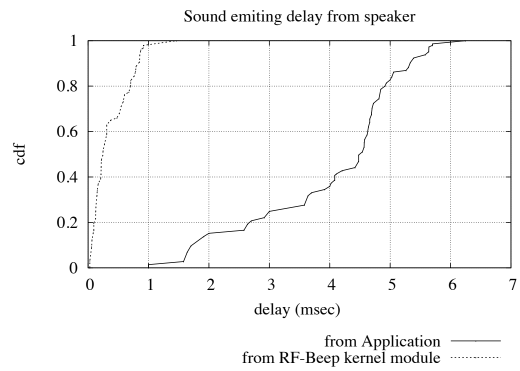


Figure 11: CDFs of the time delay between issuing the transmission command of the acoustic signal and the actual emission of the acoustic signal when the transmission command is issued at the application-level (solid line), and at the kernel-level as in *SpyLoc* (dotted line). CDFs are calculated based on 1000 sample runs in which each consists of transmitting 4400 samples of sound data at 44100Hz sampling rate

6.3 Temperature Estimation

In the range detection module, we use the sound speed to calculate the relative range. The actual sound speed depends highly on the environmental factors such as temperature, relative humidity and atmospheric pressure. Among these environmental factors, relative humidity and atmospheric pressure has the least impact on sound speed in the air [3]. In contrast, the sound speed is very sensitive to the changes in the temperature. For example, a change of one Celsius degree in the temperature could cause an approximately of 18% change in the sound speed. This sound speed sensitivity forces us to measure the air temperature prior to ranging estimation . In Beacon-Beep system user's smartphone estimates the temperature of the atmospheric air before calculating its location.

Figure 12: Position of the speaker and microphone

Now-a-days, smartphones come with internal temperature sensor to protect the internal circuit for getting over heated. Such sensor are not suitable to measure the air temperature of the surrounding environment. To over come this problem, we use the acoustic hardware of the smartphone to measure the temperature of the surrounding air. In doing this, we identify the smartphone's speaker and microphone with the farthest distance in between. For example, Figure 12 shows Nokia N900 smartphone that has two speakers; one at the top(right) and the other one at the bottom(left). The microphone is also located near the bottom of the phone. As shown in the figure, the top speaker and the bottom mic has the farthest distance, which is 10cm. We turn on both the speaker and the microphone of the smartphone. Later we generate a sinusoidal tone from the top(right) speaker and capture that sound by the microphone. We utilize the sound driver to control the tone generation from the top(right) speaker. We record both the timestamps of the actual transmission of the tone from the speaker, and the reception of that tone at the microphone. In the experiments, we use the following model [24] to calculate the air temperature:

$$\theta = \frac{\frac{d}{t_b - t_a} - 331.3}{0.6} \tag{8}$$

where θ is the air temperature in Celsius, d is the distance between the speaker and the mic (i.e 10cm from Figure 12), t_a is the timestamp (in microsecond) of tone transmission from speaker, and t_b is the timestamp (in microsecond) of the tone reception by the microphones. Based on the above equation and with t_a, t_b, d measurements, we estimate the temperature at different indoor/outdoor environments. For each environment, we have collected 20 samples to measure the temperature. Figure 13 shows the statistics of our estimated temperature measurement using Equation (8). From the figure, the estimated temperature usually is within 4-5° Celsius of the actual temprature, which has minor effects on the sound propagation speed in the air.

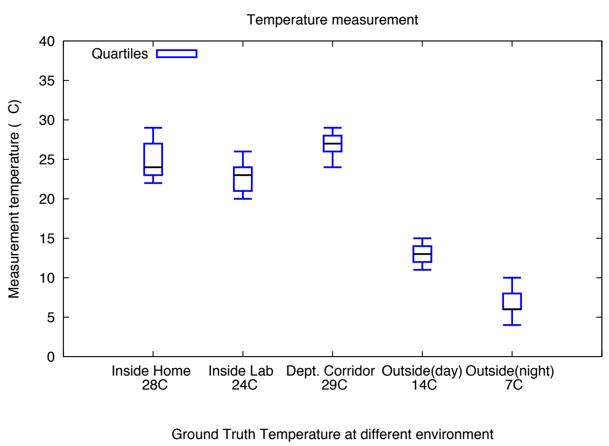


Figure 13: Estimated Temperature over different indoor/outdoor environment

7. LOCATION DETERMINATION MODULE

The location determination module is responsible to estimate the user's final location based on both the range estimation and the dead reckoning approach using inertial sensors. If the user is in the LoS of at least three beacon devices, location determination module applies the triangulation technique to estimate the user's actual location. However, in scenarios where the users is moving, (e.g. walking or running) such triangulation technique is not always practically applicable due to time limitation of computations. In addition, existence of three beacon device (e.g WiFi AP) in LoS is not always a practical scenario. Therefore, our location determination module utilizes the inertial sensors to predict the user's location. However, such technique is not highly reliable in predicting the location. Since localization error in dead reckoning scheme get cumulated over time. Thus, instead of using three ranges from three beacon devices, our location determination module uses one range estimation to calibrate the estimated location from the inertial sensors. In situation where no LoS beacon device exists, location determination module relies on the inertial sensors to predicts the location.

This module utilize the direction/distance estimation module to predict the user's possible locations from the known initial location. The variation in direction estimation, due to the noise in calculating angle of rotation (eq. (3), provides possible set of locations,

$$\begin{aligned} L &= \{l_1, l_2, ..., l_i, ..., l_N\} \\ l_i &\equiv (x_i, y_i)\ i = 1, 2, .., N \end{aligned} \tag{9}$$

For the scenarios where there is no LoS beacon device, location determination module selects the location l, which is calculated from the rotation angle θ_μ from equation (3). For scenarios with a single LoS, we use the following calculations to infer the user's final location by selecting the location points (i.e. equation 9) closest range r to the beacon device, where r is the range estimated from the beacon device.

$$d_{min} = \min_{l_i} |r - \sqrt{(a - x_i)^2 + (b - y_i)^2}| \tag{10}$$

where a, b are the center coordinate of the LoS beacon device.

8. PERFORMANCE EVALUATION

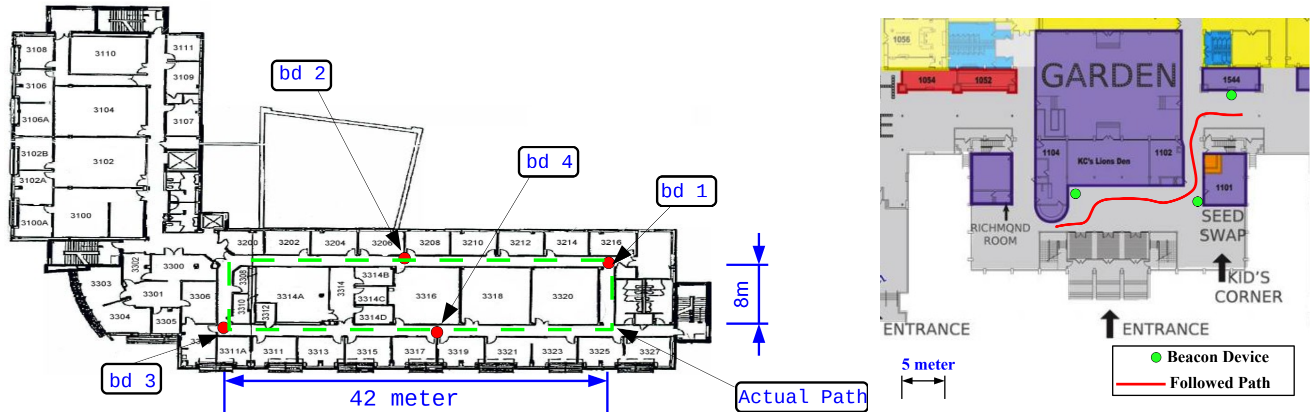
8.1 Experiment Equipment

In the experiments we use the Nokia N900 smartphone as a beacon device (i.e. Wi-Fi AP). In addition, we add a mobile speaker Nokia MD-11 with the N900 smartphone to generate the *beep* signal. In the experiments, we were able to detect the beep signal up to 35m (on average) from the beacon device for both indoor and outdoor environments, which is typical the Wi-Fi ranges in the indoor environments [7]. Since the Wi-Fi will have larger range in outdoor, we can also extended the range of the beep signal by using more powerful speaker. In our experiments, we use the same speaker Nokia MD-11 with the same transmission power of 3000mW for both indoor and outdoor environments. We use the same smartphone model for implementing the SpyLoc client application. N900 runs Maemo 5 linux based OS powered by TI OMAP3 processor which can support up to 1 GHz [25]. The phone has 256MB of high performance RAM with 1GB of VM support, two speakers laid out at the top and the bottom surface of the phone, and a single microphone located at the bottom of the front surface (figure 12). The audio features of the device is provided by the TI TWL4030 chip supported by ALSA SoC driver. The Wi-Fi chipset of the N900 phone is TI WL1251 supported by wl12xx driver. SpyLoc application requires some modifications of the wl12xx and the ALSA SoC drivers which are both part of the linux kernel open source code.

8.2 Experiment Scenarios

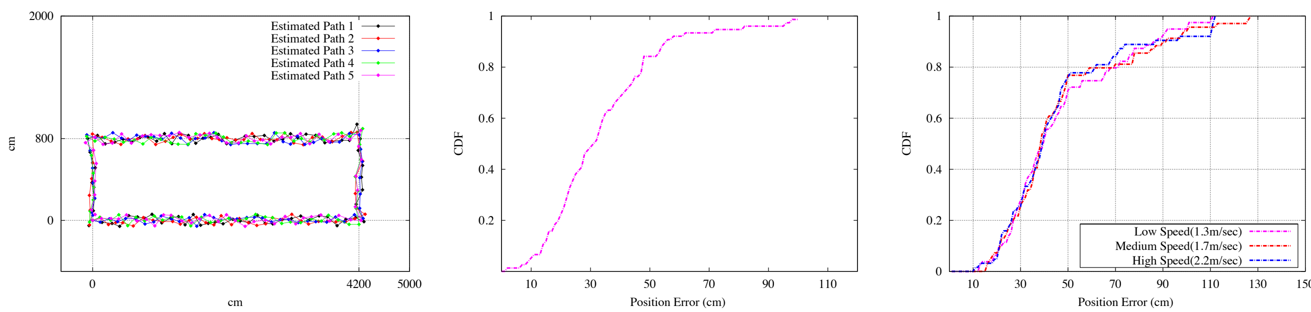
We evaluate the SpyLoc localization system under the following different scenarios:

- **Scenario-I**: In this scenario, we conducted the experiment in a defined route at indoor environment. Figure 14(a) shows the location of the beacon devices and the actual followed route. Note that, during this experiment we use one range



9(a) 9(b)

Figure 14: (a) Indoor experiment setup at department building, following the green line as a route. (b) Indoor experiment at Web Center following the red line route.



9(a) 9(b) 9(c)

Figure 15: (a)Estimated path of following the route in 14(a) for five times (b) CDF of the overall estimation error for following the route in 14(a) for five times (c) CDF of the estimation error for following the route in 14(a) at different speed.

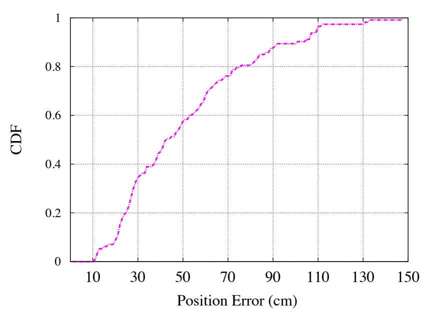


Figure 16: CDF of the estimation error for the route in Web Center 14(b)

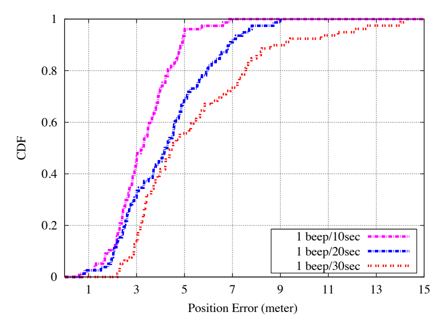


Figure 17: CDF of the estimation error for the route in Web Center 14(b) at different beep rate.

estimation and the inertial sensors to calibrate user's final location. In this experiment setup, we also evaluate the SpyLoc system under different walking speed of the user.

- **Scenario-II**: This experiment was conducted in a public space (i.e. Web Center) where we follow a casual route 14(b). The Figure 14(b) also shows the position of the three beacon devices and the route. During this experiment we use one range estimation and the inertial sensors to calibrate the user final location. In this experiment setup, we evaluate the SpyLoc system for different rate of beep (e.g. beep/1sec, beep/10sec, beep/20sec and beep/30sec.) in order to mimic the circumstances, where SpyLoc client application don't receive any LoS beep signals for a certain duration of time.

- **Scenario-III**: In this scenario, we conducted our experiment inside the WebCenter building. Figure 18(c) shows the location of the three *beacon devices* and the positions where we estimate the location. In this experiment, we only use the range estimation scheme to locate the user's position.

- **Scenario-IV**: This experiment is taken place in a open space parking lot. Figure 18(a) shows the location of the three *beacon devices* and the positions where we estimate the location.

In this experiment, we evaluate the range estimation scheme for outdoor scenarios.

For the scenario III and IV, we have repeated the location estimation 500 times for 50 distinguish locations.

8.3 Experiment Results

Scenario-I: Figure 15(a) shows the estimated path by the SpyLoc Client system while we follow the route in 14(a) for five times. The plot 15(b) shows the CDF of the overall estimation error over for five time experiment. During this experiment, we found estimation error less then 50 cm for 90% of the time. The plot 15(c) shows the CDF of the estimation while we follow the route in 14(a) at different speed. In this experiment, we found estimation error less then 90cm for 90% of the times for different speed. These results proofs the feasibility and the efficiency of our SpyLoc system under different mobility condition of the user.

Scenario-III: The plot 16 shows the CDF of the estimation error while we follow the route in 14(b) at Web Center. This results proofs that SpyLoc shows good accuracy even for normal casual walking pattern of our daily life. In some scenarios, SpyLoc client application might not be able to receive any LoS beep signal from the beacon devices. In such cases, SpyLoc would need to rely on the inertial sensors to predict the locations until it receives a LoS beep signal. In order to mimic this situation, we change the rate of beep generations from the beacon devices. The plot 17, shows the CDF of the estimation error for different beep rate. The result indicates that relying just on the inertial sensor for long time can lead to a high position error. In that case, users need to calibrate their location by using ranges from three LoS beacon devices. The following experiment, shows the evaluation of the range estimation based location determination for both indoor and outdoor environment.

Scenario-III: Consider the scenarios, when the user just lunch the application to find out the initial position of the user, or the application was not receiving any LoS beep signal for a long time, in those circumstances SpyLoc needs to use three ranges from three LoS beacon device to initiate or calibrate user's position. In order to evaluate such situation, in this experiment we just use the range estimation technique to find out the positions. Figure 18(a) shows the approximate locations of the collected samples (i.e., blue circles). This experiment was done in a public place with a lot of surrounding students. In addition, the place has several furnitures such as tables, chairs and desktop PCs. The distance ranges between the beacon devices in the range of 20-22m. Plot 18(d) shows the error distribution of the estimated location with an error less then 1.5 meter for more than 90% of the time.

Scenario-IV: This experiment was conducted at outdoor to proof the feasibility of using our SpyLoc application in such environment. In Figure 18(a), the blue circles shows the locations of the collected samples. Plot 18(b) shows the distribution of the distance error in cm for this experiment setup. The plot shows that the error in location estimation is less then 1meter for more than 90% of the samples collected in this outdoor experiment.

9. RELATED WORK

Most of the localization research works have been based on Radio Frequency (RF)-based techniques that leverages signal strength of RF signal from different nearby RF sources or infrastructures (e.g., WiFi Access point, Cellular Tower) [17, 4, 26, 27]. Recently, researchers are combining multiple modalities such as sound with the WiFi to achieve high accuracy localization system [1, 2, 28, 3]. For example, the localization schemes [1, 2, 12, 11], utilize the acoustic based ranging [13] scheme and combine it with the RF at the application layer. In addition, some localization system uses multiple modalities of the smartphone to determine location of user at different level of accuracy[9, 16, 14, 15]. The following table compares between the recent proposed schemes and SpyLoc.

	[1, 2]	[16]	[14, 15]	[3]	SpyLoc
RF or Acoustic Fingerprint dependent	Yes	Yes	No	No	No
Smartphone Usability	Yes	Yes	Yes	No	Yes
Usability at motion	No	Yes	Yes	No	Yes
Accuracy	1-5m	1.7m	1-5m	<1m	<1m
Depend on floor layout	No	Yes	Yes	No	No
Backend system or centralized system required	Yes	Yes	No	No	No

10. CONCLUSION

In this paper, we propose a practical location determination system that leverages the balance between the high accuracy ranging scheme and the light weight dead-reckoning approach. In this paper, we show how the sound system and the RF interface in the off-the-shelf smartphone can be utilized to develop a high accuracy localization system. In addition, we also develop an efficient and light way to utilize the inertial sensors to determine the distance and direction of the user movement. In this paper, we leverages the unique characteristics of the acoustic signal to detect LoS and NLoS acoustic signals. Finally, we evaluate our system under different practical world scenarios where we considered different walking speed of the user. In ideal situation, SpyLoc is able to achieve less then 1meter level of accuracy under different speed.

11. REFERENCES

[1] R. Nandakumar, K. K. Chintalapudi, and V. N. Padmanabhan, "Centaur: locating devices in an office environment." in *Mobicom '12: Proceedings of the 18th annual international conference on Mobile computing and networking*, new York, NY, USA: ACM Press, 2012, pp. 281-292.

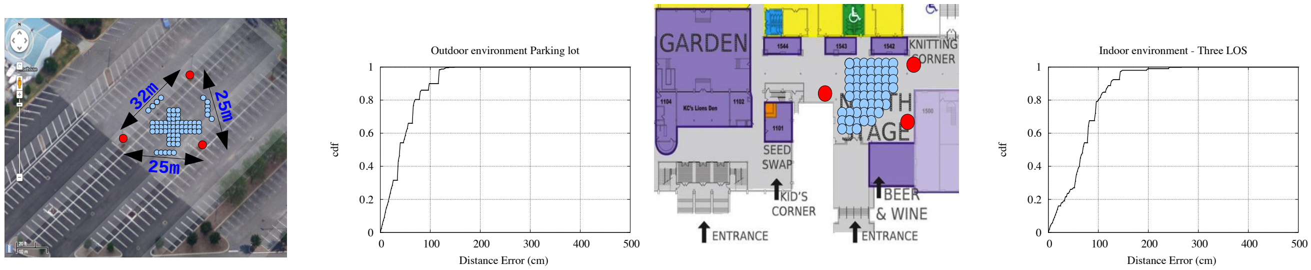
[2] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen, and F. Ye, "Push the limit of wifi based localization for smartphones." in *MobiCom 2012: Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, istanbul, Turkey, August 2012.

[3] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system." in *MobiCom Õ00: Proceedings of the 6th annual international conference on Mobile computing and networking*, new York, NY, USA, 2000.

[4] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp, "Walrus: wireless acoustic location with room-level resolution using ultrasound," in *MobiSys Õ05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, new York, NY, USA: ACM Press, 2005, pp. 191Ð203.

[5] A. Matic, A. Popleteev, V. Osmani, and O. Mayora-Ibarra, "Fm radio for indoor localization with spontaneous recalibration." *Pervasive Mob. Comput.*, vol. 6, December 2010.

[6] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka., "Spot localization using phy layer information." in *In MobiSys Õ12, 2012.*



9(a) 9(b) 9(c) 9(d)

Figure 18: (a) Outdoor experiment at parking lot with three LoS beacon device. (b) CDF of the outdoor experiment localization error for Three LoS. (c) Indoor experiment at Web Center with three LoS beacon device. (d) CDF of the indoor experiment localization error for Three LoS.

[7] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha, “Fm-based indoor localization.” in *In Proceedings of The 10th International Conference on Mobile Systems, Applications and Services, (MobiSys Õ12)*, lake District, UK, June 25-29, 2012.

[8] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai, and M. W. I, “Indoor location sensing using geo-magnetism.” in *In MobiSys Õ11, 2011.*

[9] M. Azizyan, I. Constandache, and R. R. Choudhury, “Surroundsense: mobile phone localization via ambience fingerprinting.” in *MOBICOM 2009*, pages : 261-272.

[10] S. P. Tarzia, P. A. Dinda, R. P. Dick, and G. Memik, “Indoor localization without infrastructure using the acoustic background spectrum.” in *MobiSys 2011*, pages : 155-168.

[11] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda., “Swordfight: Enabling a new class of phone-to-phone action games on commodity phones,” in *MobiSys’12: Proceedings of ACM MobiSys*, low Wood Bay, Lake District, UK, 2012.

[12] J. Qiu, D. Chu, X. Meng, and T. Moscibroda, “On the feasibility of real-time phone-to-phone 3d localization.” in *SenSys Õ11: Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, new York, NY, USA: ACM, 2011, pp. 190Ð203.

[13] C. Peng, G. Shen, Y. Zhang, Y. Li, and K. Tan, “Beepbeep: A high accuracy acoustic ranging system using cots mobile devices.” sydney, Australia, ACM SenSys 2007, Nov 2007.

[14] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, “A reliable and accurate indoor localization method using phone inertial sensors,” in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, ser. UbiComp ’12. New York, NY, USA: ACM, 2012, pp. 421–430.

[15] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, “Zee: zero-effort crowdsourcing for indoor localization,” in *Proceedings of the 18th annual international conference on Mobile computing and networking*, ser. Mobicom ’12. New York, NY, USA: ACM, 2012, pp. 293–304.

[16] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, “No need to war-drive: unsupervised indoor localization.” in *MobiSys 2012: 197-210.*

[17] P. Bahl and V. N. Padmanabhan, “Radar: An in-building rf-based user location and tracking system,” in *INFOCOM Õ00: the 19th Annual IEEE Conference on. Computer Communications*, tel-Aviv, Israel: IEEE Infocom, March 2000.

[18] M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala, “Pinpoint: An asynchronous time-based location determination system.” in *MobiSys 06: Proceedings of the 4th international conference on Mobile systems, applications and services*, new York, NY, USA, 2006.

[19] J. Yang, S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser, and R. P. Martin, “Detecting driver phone use leveraging car speakers.” in *International conference on Mobile computing and networking, 2011.*

[20] S. H. Collins and A. D. kuo, “Two independent contributions to step variability during over-ground human walking.” *Journal of Applied Physiology*, 2011.

[21] Removed for Blind Review.

[22] Y. Athavale, S. Krishnan, and A. Guergachi, “Pattern classification of signals using fisher kernels,” *Hindawi Publishing Corporation Mathematical Problems in Engineering*, 2012.

[23] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.

[24] Wiki, “Speed of sound,” http://en.wikipedia.org/wiki/Speedofsound.

[25] ——, “Nokia n900,” http://wiki.maemo.org/Nokia_N900.

[26] M. Youssef and A. Agrawala, “The horus wlan location determination system.” in *MobiSys Õ05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, new York, NY, USA: ACM Press, 2005, pp. 205Ð218.

[27] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal, “Ariadne: a dynamic indoor signal map construction and localization system,” in *MobiSysÕ06: Proceedings of the 4th international conference on Mobile systems, applications and services*, new York, NY, USA, 2006.

[28] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, “The anatomy of a context-aware application.” in *MobiCom Õ99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, new York, NY, USA, 1999.