Daredevil: Indoor Location Using Sound

Ionut Constandache[†], Sharad Agarwal[‡], Ivan Tashev[‡], Romit Roy Choudhury[⋆]
[†]Duke University, [‡]Microsoft Research, [⋆]University of Illinois at Urbana-Champaign
[†]ionut@cs.duke.edu, [‡]{sagarwal,ivantash}@microsoft.com, [⋆]croy@illinois.edu

Abstract

A variety of techniques have been used by prior work on the problem of smartphone location. In this paper, we propose a novel approach using sound source localization (SSL) with microphone arrays to determine where in a room a smartphone is located. In our system called Daredevil, smartphones emit sound at particular times and frequencies, which are received by microphone arrays. Using SSL that we modified for our purposes, we can calculate the angle between the center of each microphone array and the phone, and thereby triangulate the phone's position. In this early work, we demonstrate the feasibility of our approach and present initial results. Daredevil can locate smartphones in a room with an average precision of 3.19 feet. We identify a number of challenges in realizing the system in large deployments, and we hope this work will benefit researchers who pursue such techniques.

I. Introduction

Making indoor location available ubiquitously is difficult. There are many challenges, including achieving accuracy with off-the-shelf phones, relying solely on existing infrastructure such as Wi-Fi access points, and scaling any human effort such as fingerprinting. However, there are specific situations where tailored indoor location solutions are valuable and one or more of these constraints do not apply. For instance, firefighters may be willing to carry custom equipment for indoor location but will require the solution to work in any burning building. A retail chain store may be willing to deploy custom equipment in their stores, but may want location to work with off-the-shelf phones that users carry.

In this work, we focus on the retail scenario as our motivation. A store owner may want to provide indoor location to shoppers for a variety of reasons. She may want shoppers to efficiently find items on their shopping list with minimal clerk assistance. She may want to entice shoppers with special discounts or product reviews depending on what product the shopper is looking at. In doing so, the store owner wants to minimize the burden on the shopper and not require anything custom on the end user devices beyond an app for the store. She may be willing to deploy custom equipment inside the store. While we focus on the retail scenario, our techniques are equally applicable to any scenario where user phones with a custom app need to be located in a large indoor room where the room can be augmented with additional equipment.

Prior work has considered a variety of ways to address such scenarios, including sensing ambient magnetic fields [4], fingerprinting Wi-Fi [1] and fingerprinting FM radio transmissions [3]. In this work, we focus on sound, either audible or ultrasound. The choice of sound comes with its own advantages and disadvantages. There are transducers on all phones (speakers and microphones) and sound is generally unaffected by changes in the store layout or human presence (unlike magnetic fields and Wi-Fi signal strength). However, depending on the frequency, amplitude and duration, it can be overheard by humans, and there is a lot of ambient noise that can interfere with the detection of the intended signal.

In this work, we attempt to accurately locate users indoors by detecting the angle of arrival of audio chirps emitted by smartphones. We examine how well off-the-shelf phones can emit such chirps, at a variety of different frequencies and amplitudes. We examine how we can encode small amounts of information in these chirps to distinguish different phones. To calculate the angle of arrival, we build microphone arrays that can be used in conjunction with SSL (sound source localization) algorithms [17] that we have customized for our use.

The novelty of our work is in applying SSL to locate smartphones. Shopkick is a company that has deployed sound-based location in stores. However, in their system the phone is detecting sound emitted by a custom device in the store, and only detects whether the phone is in the store, not where in the store the phone is. SSL techniques have been explored in depth in prior work, and are in use in products such as Microsoft Kinect, for distinguishing different humans speaking at the same time. In this work, we apply those techniques to locating phones, which has unique

challenges.

Our system, called Daredevil, operates at least two microphone arrays, and by calculating the angle from each array to the phone emitting a tone, can triangulate the user's location. In this paper, we present the design of the system and hardware and feasibility experiments in § III, evaluation results in § IV, and relevant prior work in § V. This early work demonstrates a working system that achieves low error – on average 3.8°, or approximately 3.2 feet on average. However, we also identify limitations of current hardware on smartphones that prevent us from deploying Daredevil in practical scenarios, and we point to future work in this space in § VI.

II. Motivation

We envision Daredevil to be deployed in retail stores where pairs of microphone arrays are mounted on walls or ceilings. Unmodified phones running an app can be identified and located using sound that they emit. There are several open questions that we need to answer when building such a system.

It is important to understand how well off-the-shelf smartphones can emit high frequency audio tones. This is partly a function of the speakers they have, audio processing hardware, and the software platform on the phones. A related question is how loud these tones are, or at what distance they can be heard using microphones. The longer the distance is, the fewer microphone arrays are needed, but higher is the potential for interference between multiple phones. The higher the frequency, the lower is the chance for interference from human voice.

The cost and robustness of our microphone array design are two more key factors. An inexpensive array can allow for more arrays to be deployed in the indoor environment. The array has to be robust enough to pick up audio tones from phones from a variety of angles and distances.

The ultimate question is how accurately the microphone arrays can detect the angle of incidence from phones, and subsequently the location of each phone by triangulation from a pair of arrays. Additional questions are how quickly tones can be generated and phones located, and how that impacts the scalability of the system in how many phones can be located and how frequently.

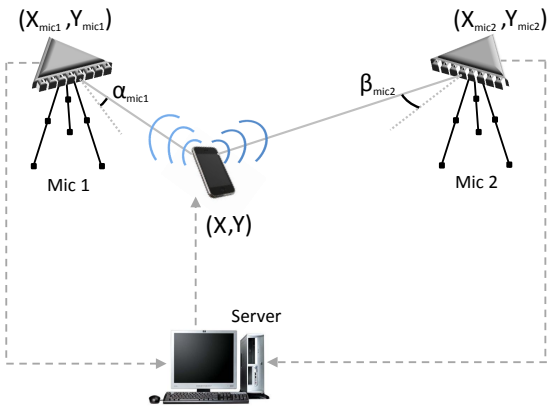

Figure 1: Daredevil overview

III. Design and Implementation

III.A. Overview

As shown in Figure 1, we assume that the indoor environment has been instrumented with at least two microphone arrays, with known positions. We expect the arrays to be mounted high on adjacent walls that are perpendicular to each other, or attached to the ceiling. The placement should be high enough such that there is good line of sight to most users in the environment, minimizing the impact of audio reflecting from indoor surfaces. The goal of each microphone array is to calculate the angle of incidence of received audio to the front surface of the array. With two angles (one from each microphone array), we can triangulate the position of the sound source.

Each phone has an app that generates and plays audio in a specific frequency band. The idea is to play audio at high enough frequencies to be inaudible by humans, yet be playable by smartphone speakers and be loud enough to be captured by microphone arrays.

In our application scenario, we expect multiple phones to be present that need to be located simultaneously. If multiple phones emit sound at the same time and frequency, it can be difficult to distinguish them and hence calculate an angle of incidence to each microphone array. We can use two basic approaches to address the problem of multiple phones – frequency division multiplexing (FDM) or time division multiplexing (TDM). As our experiments in § IV show, when attempting to emit sound at a particular frequency, we observe our signal at additional frequencies. As a result, FDM may be more challenging to achieve in practice. Hence, we focus on TDM in Daredevil.

Our smartphone app uses coarse-grained location information from the underlying mobile OS (typically based on Wi-Fi and cellular tower location) to esti-

mate which Daredevil-enabled store the user is in. The app will then receive a schedule from the Daredevil server (shown in Figure 1) that tells it when it can emit sound and in what frequency band. The app uses amplitude modulation to encode a unique phone ID in the sound it emits. Clock drift between phones and servers will limit how quickly multiple phones can be located.

Each phone registers with the Daredevil server and retrieves the common tone frequency, a unique phone ID, and the TDM schedule assigned to it. The phone constructs the audio in software and plays it at the assigned schedule. The microphone arrays stream audio to the server. SSL software running at the server identifies the tone, decodes the phone ID, and computes the tone angle of arrival at each of the two microphone arrays (angles α_{mic1} and β_{mic2} in Figure 1). The microphone array coordinates (X_{mic1}, Y_{mic1}) and (X_{mic2}, Y_{mic2}) are static values provided to the the server software as configuration values at deployment time. Using the microphone array positions and the tone angles of arrivals, the server computes the phone coordinates through *triangulation*. The phone coordinates (X, Y) are returned to the app, or processed further for higher-level services (such as providing directions on top of an indoor map, or sending a discount coupon).

The phone location is updated periodically, each time the app on the phone plays sound at the scheduled times. The schedule periodicity can be made adaptive based on the number of phones present in the environment. Additional factors, including the maximum user speed, and limits on user movement by walls and aisles, can be also used to dynamically adjust the schedule for different phones.

III.B. Sound Source Localization

Locating sounds using microphone arrays is a well established area in signal processing. One of the first applications was pointing a pan-tilt-zoom camera toward the current speaker in a conference room [21].

Direction estimation with a pair of microphones is done by computing the delay between the two received signals, and using the known speed of sound and the distance between the two microphones: $\theta = arcsin(\tau\nu/\delta)$. Here θ is the direction angle, τ is the estimated time delay, ν is the speed of sound, and δ is the distance between the two microphones. τ is computed using the Generalized Cross-Correlation function [10], typically with PHAT weighting.

Using more than two microphones improves precision, but increases the complexity of the estimation. The naive approach of combining angles from all possible pairs does not work well. Instead, Steered Response Power (SRP) algorithms are often used. Those are based on evaluating multiple angles and picking the one that maximizes certain criteria (power of the signal from the sound source [18], spatial probability [17], eigenvalues [14], etc.). In Daredevil, we use a modified and improved version of the algorithm described in [17]. On every audio processing frame we run a Voice Activity Detector (VAD), like the one described in [15] but modified for the type of audio we generate, and engage the sound source localizer only if there is a signal (real signal or interfering signal) present.

The audio frames with signal present are converted to frequency domain using short-term Fourier Transformation, and only the frequency bins containing the signal frequency band are processed. For each frequency bin k of audio frame n, the probability of it having a signal as a function of the direction $p_k^{(n)}(\theta)$ is estimated using the IDOA approach [17]. The probabilities from the frequency bins of interest are averaged to receive the probability of sound source presented as function of the direction $p^{(n)}(\theta)$. A confidence level is estimated as the proportion of the difference between the maximum and minimum values of the probability, divided by the probability average. If the confidence level is below a given threshold, the result from this audio frame is discarded, otherwise the direction angle is considered where the probability peaks. The time stamped angle, accompanied by the confidence level, is sent up the stack for further processing.

The angular precision based on a single audio frame with duration 10-30 ms is not high and hence we use post-processors. Their purpose is to remove outliers and reflections and to increase location precision by averaging and interpolating sound source trajectory over small time windows. A variety of methods can be used, including Kalman filtering [9], clustering [16] and particle filtering [22]. In Daredevil, we use the clustering algorithm described in chapter 6 of [16]. The output of the sound source localizer is a set of directions toward all sound sources tracked, each accompanied by a confidence level.

In addition to localizing sounds, signals from the microphones can be combined into one signal, which is equivalent of a highly directive microphone - an operation called beamforming. Commonly the beamforming happens in frequency domain, where we already converted the audio signal to perform sound source localization. In this domain the beamform-

ing output is defined as a weighted sum with direction dependent weight. The output signal contains less reverberation and noise, allowing us to process more easily the chirps from a phone. The beamforming operation requires knowledge of the direction to the desired sound source, which is obtained from the sound source localizer. In Daredevil, we use a time invariant steerable beamformer [16]. It consists of 21 pre-computed tables of weight coefficients for directions from -50° to +50° pointing at every 5°. Using the list of sound sources, obtained from the sound source localizer, for each of the tracked sound sources we snap the direction to the closest pre-computed angle and then compute the beamformer output. This means that we apply the beamforming procedure as many times as sound sources we track. The output of each beamformer contains the sound from the corresponding phone, while the other signals and noises are attenuated. It is used further for decoding the information coming from that particular phone.

Each microphone array works in the range of $\pm 50°$. In the ideal case we should be able to reject any sound source that is out of this angle and out of the line of sight. Otherwise we will get fake directions for sound sources. We mitigate this problem in the following ways:

- We use VAD to select only those audio frames where we have strong signals, presumably rejecting these that are out of the monitored zone or not in the direct sight of the array.
- We scan for sound sources only in the directions of the range above.
- After processing each frame we compute a confidence level ((max-min)/average) and reject all of the frames with confidence level below a threshold.
- We cluster the sound source localizations, and track only sound sources with presence in several frames. We then compute a confidence level for each sound source.
- We compute the beamformer toward a direction only when the confidence level after clustering is above a threshold.

Again, the key is to reject directions toward sporadic sound sources, reflections from walls and ceilings, and noise. This is part of the reason why we do not detect real sound sources at larger distances – with our thresholds we cannot distinguish those from soft sounds from nearby noise. Further tuning can be done to the thresholds to increase the detection range as much as possible while still keeping false positives at acceptable low levels.

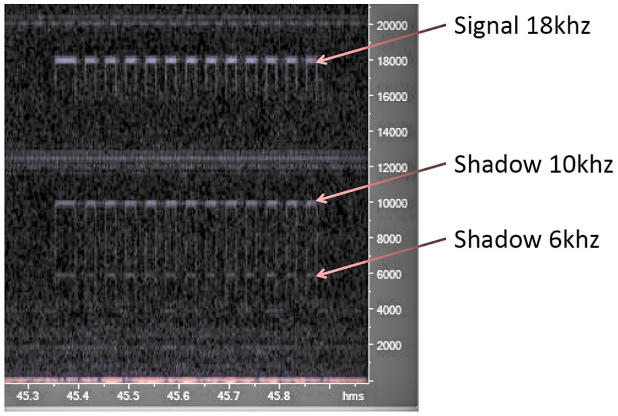

Figure 2: Spectrogram from Adobe Audition software of 18 kHz audio played on a HTC Surround phone and recorded on a microphone array

III.C. Audio Frequency Band

In our application scenario, the store owner is willing to deploy hardware in the store but cannot expect users to modify their phones beyond installing an app. Ideally, we want the audio emitted by phones to be well beyond human perceptible ranges. Medical literature indicates that human hearing can recognize sounds up to 20 kHz. Unfortunately, speakers on most smartphones are not designed to operate beyond human hearing ranges, and our experiments on a number of phone models (Apple iPhone 4, Samsung Focus, Asus E600, and HTC Surround) verified that expectation. As the emitted sound frequency approaches 21 kHz, the sound amplitude drops off quickly. Most phones are capable of achieving 18 kHz at loud volumes. It is well known that the upper range of audio frequencies detectable by the human ear varies with age. In a small experiment using 10 test users aged 20 and higher, playing audio on a PC with high end speakers, we observed no user perception of frequencies higher than 17 kHz. For this paper, we focus on 18 kHz, realizing that the short audio chirps that our app emits may be perceptible by children.

Phone loudspeakers at maximum volume produce substantial non-linear distortions. For 18 kHz sound, the second, third, and higher harmonics are above one half of the sampling rate and should be removed by the anti-aliasing filter integrated into the ADC. However, due to the high amplitude of these non-linear distortions, they still bleed over and are mirrored as signals with lower frequency. The spectrogram in Figure 2

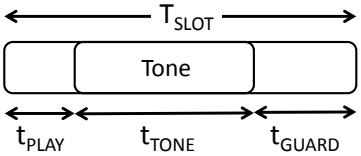

Figure 3: TDM time slot

shows these additional signals with frequencies of 6 and 10 kHz. While these two shadow frequencies are within the human audible range, their amplitudes are much lower than the primary signal which is clearly visible at 18 kHz. We have observed this behavior on multiple phones and we believe this is a limitation of the speakers built into phones. We observe similar spectrograms for frequency bands ranging from 18 kHz to 21 kHz.

Our findings are consistent with prior work. Recent work [5] investigated the feasibility of playing ultrasonic sounds on mobile phones. The authors tested four commercial phones (HTC G1, HTC Hero, Apple iPhone 3GS, and Nokia 6210 Navigator) playing tones at frequencies between 17 kHz and 22 kHz. They observe that all phones were capable of generating these high frequencies. While noise appeared at some other frequencies when the volume of the phone's speakers are set to maximum, there exists a combination of volume settings for each phone such that the noise is minimal.

III.D. Locating Multiple Phones

Using FDM to allocate distinct frequencies to different phones is possible, but complicated by the shadow frequency problem we observe. We instead use TDM. However, for a TDM approach to work, we need to ensure that the slots do not overlap. This is challenging because different phones and the Daredevil server may not have good time synchronization with each other. In addition, scheduling and processing overhead in smartphone OSes may introduce variable delay between our app issuing a command to play audio and it coming out of the phone's speaker.

Figure 3 represents a time slot in the Daredevil TDM schedule. We denote the length of the slot with T_{SLOT}. This represents the time interval allocated to a phone to play the tone of length t_{TONE} ($t_{TONE} < T_{SLOT}$). We mark with t_{PLAY} the delay between the request to play the tone and the phone actually playing it, and t_{GUARD} a guard time delay to account for clock drift. Parameter t_{GUARD} ensures that the next scheduled phone does not play the tone concurrently with the current phone due to poor clock synchronization.

To measure the time delay t_{PLAY} between issuing the play tone command and the tone coming out of the phone speaker, we ran a small experiment. We instrumented the Daredevil app on a phone to timestamp when tone play is requested by the OS. In parallel, the phone recorded sound through its microphone. In this way, we are able to estimate t_{PLAY} which is the difference in time between the app issuing the play command and the audio containing the expected frequency. After running this experiment 10 times on different phone models, we observed a maximum delay of 100ms. Hence we use a t_{PLAY} of 100ms.

We empirically evaluated a number of values for t_{TONE}, while measuring (i) tone detection and (ii) angle estimation accuracy. Both these values suffer when the tone length is short. If the tone length is too short, our SSL algorithm does not have enough samples to remove noise from angle estimates. In our experiments, 500ms was an adequate length, and hence we set t_{TONE} to 500ms.

To pick t_{GUARD} we ran experiments with two phones that were configured to synchronize their clocks with cellular towers. We attached both phones to a PC using USB cables and recorded on the PC the time reported by both phones at multiple points throughput the day. The two sets of reported times were very close to each other, and remained below 150ms, which is the value we pick for t_{GUARD}.

Putting together the values for t_{PLAY}, t_{TONE} and t_{GUARD}, we have 750ms for the value of T_{SLOT}. Hence, with a single frequency band, we can locate up to 40 phones every 30 seconds, in the coverage area of a pair of microphone arrays.

III.E. Hardware and Software Implementation

Figure 4 shows a photograph of one of the three microphone arrays we built for Daredevil. We used a laser cutter on a sheet of plastic to form the base and the front plate that holds the microphones. The geometry is a linear equidistant eight element microphone array. The distance between the microphones is one half wavelength for sound with frequency of 21 kHz (8.16 mm). We used cardioid electret microphones with a diameter of 6 mm, all of them pointing forward.

The microphones fit snugly into the holes on the front plate (bottom of picture), and wires connect them to simple circuit. The voltage bias for the microphones is provided by a 9 V battery. We also have variable resistors on each channel for individual gain calibration. Eight cables then run out to an A-D converter. For our experiments, we used the MOTU UltraLite-mk3, which supports 8 audio inputs and

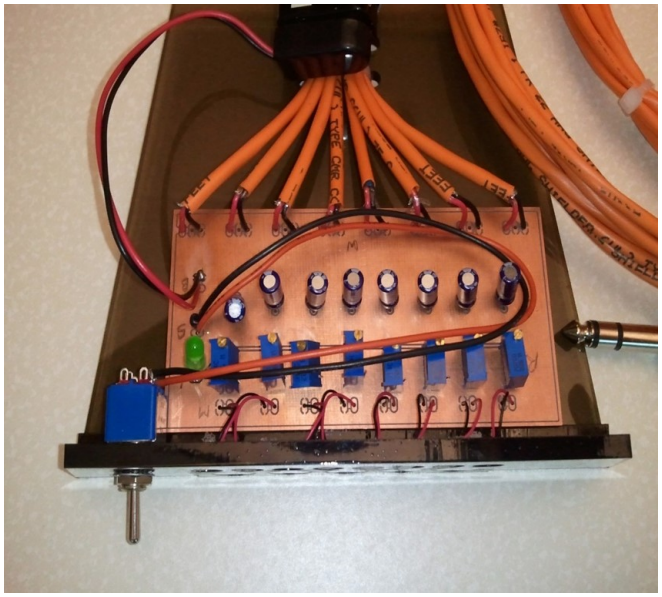

Figure 4: Top down view of one of our microphone arrays

connects to a PC via USB. For prototype and hardware debugging purposes, we made the microphone array larger than it needs to be. If productized the entire microphone array and the ADC can be made into a box that is the size of a deck of playing cards.

Each A-D converter outputs 8 channel audio through the audio driver stack in Microsoft Windows. We use a combination of C code and Matlab code to filter the audio, do VAD, do SSL, and triangulate the phone's position. Another piece of software maintains a list of active phones and allocates frequency bands, time slots, and unique IDs to each phone. This constitutes the Daredevil server.

On the phone side, we have a simple app on Windows Phone 7, which communicates with the Daredevil server over the Internet, and plays audio when given a schedule. The app uses amplitude modulation to encode a 24 bit unique phone ID on top of the audio tone. Both the unique ID and the audio frequency are provided to the app by the Daredevil server. The unique phone ID is modulated at a baud rate of 50 symbols per second, and there are two additional guard bits at the start of the audio sequence. Each tone is 0.52 seconds long. We have also ported our app to the Apple iOS platform.

IV. Experimental Results

The primary question we want to answer in our evaluation is how accurately we can determine the location of a phone in Daredevil. We begin by evaluating how accurately Daredevil can determine the angle between a microphone array and where the phone was when it played a sound. We evaluate that error at different angles and distances to the microphone array. We also evaluate how that error changes when we change the frequency band in which audio is played. Finally, using the accuracy we achieve in determining the angle, we can use triangulation to calculate the error between a phone's actual and estimated positions.

In the experiments here, we set the volume of the phone's loudspeaker to 90% of the maximum, which does not have as much distortion as at 100%. The limited ability of phone loudspeakers to emit sounds at high amplitudes without distortion, and the noise abatement techniques in our sound source localization algorithm together limit the maximum distance at which we can locate phones.

IV.A. Angular accuracy at 18 kHz

To evaluate Daredevil, we deployed one of our microphone arrays in a conference room. We then played tones at 18 kHz on a Samsung Focus smartphone at specific locations in the room. Figure 5 shows the 12 locations in the room that we evaluated. The figure also shows for each location, the distance to the microphone and angle between the center of the microphone array and the shortest line between the phone's position and the microphone array. We calculated these distances and angles by using a laser range finder, a measuring tape, and trigonometry.

Figure 6 shows the results of estimating the angle at each of the 12 locations in Daredevil. As shown in the figure, many of the estimates are 5° or lower. However, our modified VAD is not able to detect the tone at three locations and in two locations, our SSL provides poor accuracy.

To understand why accuracy is poor at certain locations, we present Figure 7. The green baseline in the center represents the noise floor, while the spikes denote the tone being played by the phone. While the tones are distinguishable, the SNR is low – in many cases less than twice the noise floor. This low SNR results in missed tones and poor angle accuracy caused by tone amplitudes, erroneously correlated to the noise and not the actual tone signal. The SNR is poor at higher frequencies as well (we tested 19 kHz, 20 kHz, and 21 kHz).

IV.B. Angular accuracy at 10 kHz

The poor SNR at 18 kHz contributes to our poor accuracy, and we suspect the poor SNR is due to phone speakers not being able to produce sound at high am-

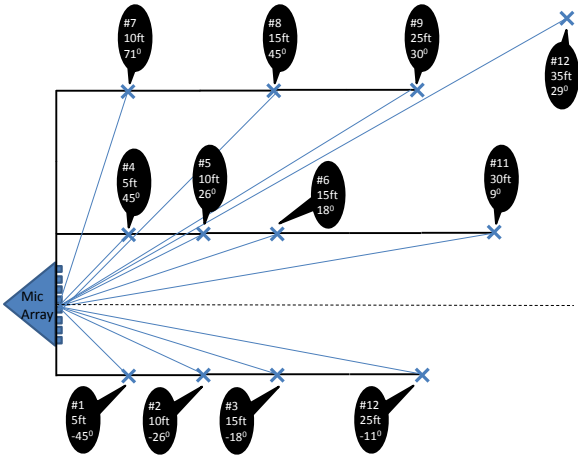

Figure 5: Evaluation locations when playing tones at 18 kHz. The "X" marks a spot where we played audio from a phone. The black bubble shows the position number, the distance and angle to the center of the microphone array. (not drawn to scale).

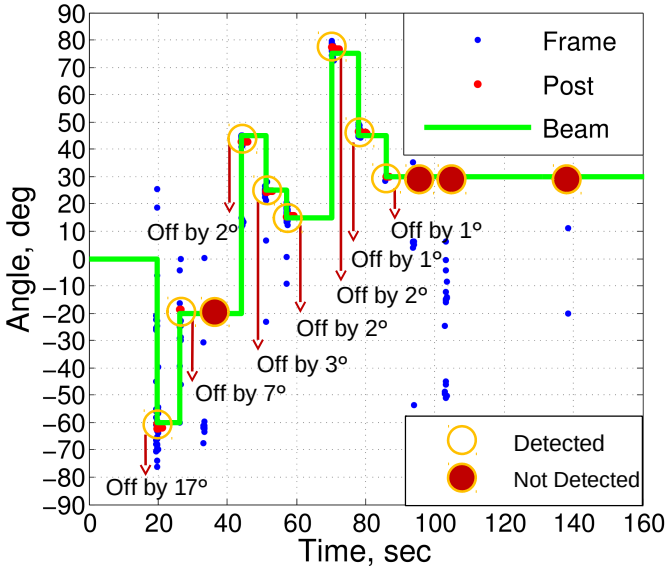

Figure 6: Results from calculating the angle at each of the 12 locations while playing tones at 18 kHz. Each orange circle indicates the time at which the tone was played on a phone, in order (location #1 is the left-most circle, location #12 is the right-most circle). Locations 3, 10, 11, and 12 were not detected by Daredevil. Blue dots denote the angle direction inferred per audio frame (1024 audio samples at 96 kHz). Green lines denote the beamforming directions. Small horizontal red lines represent the angles output by SSL after clustering and weighting the per-frame directions.

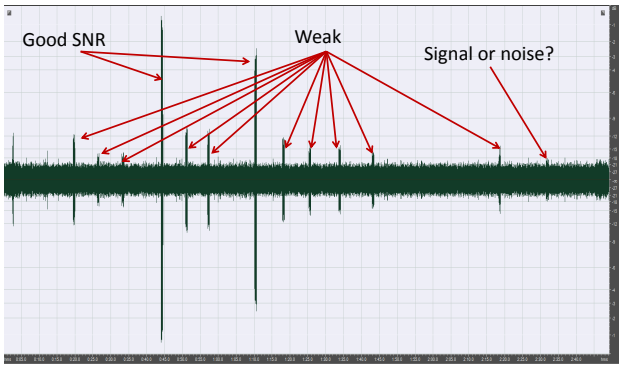

Figure 7: Amplitudes of the 18 kHz audio tones received at one of the 8 microphones, after amplifying the signal by 40 dB, applying a band pass filter around 18 kHz, and then amplifying again by 40 dB. The vertical axis is amplitude, and the horizontal axis is time.

plitudes at 18 kHz. To test this theory, we repeated our experiments at 10 kHz. Figure 8 shows the locations we tested in the same room at this frequency.

Figure 9 shows the angular accuracy at 10 kHz. We observe that all the 7 locations are detected, and in most cases with relatively low error. Daredevil can detect the phone's audio as far away as 35 feet. The average error is 3.8°.

Figure 10 shows the amplitude of the received signal at one microphone after processing. As expected, we see a very strong set of 7 signals (the vertical bars). Unfortunately, since we are operating at audible frequencies, we also catch human voice, as shown by the other peaks. These sometimes appear in Figure 9 as calculated angles (short red horizontal lines without an orange circle), but since the phone's ID cannot be decoded off that signal, no phone is detected at those angles.

Unfortunately, the lower frequency of 10 kHz poses problems for Daredevil. While each tone played by a phone is very short, it can become annoying for humans. While human voice can also appear in this frequency range, it can be filtered out. However, it can interfere with the phone's audio signal when humans speak at the same time. While 10 kHz is not a feasible option in a real deployment, it demonstrates the effectiveness of Daredevil if this limitation of phone speakers were to be mitigated in the future.

IV.C. Location accuracy

To evaluate location accuracy, we simulate a range of scenarios with different angular accuracies from our results above. Rather than stand at many locations inside a room and run the application on the phone, we

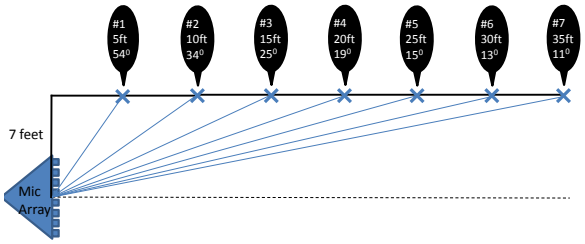

Figure 8: Evaluation locations when playing tones at 10 kHz. The "X" marks a spot where we played audio from a phone. The black bubble shows the position number, the distance and angle to the center of the microphone array. (not drawn to scale).

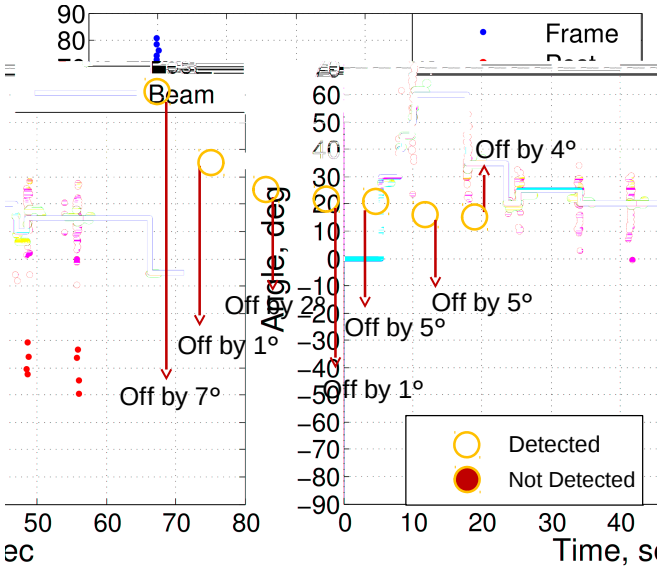

Figure 9: Results from calculating the angle at each of the 7 locations while playing tones at 10 kHz. Each orange circle indicates the time at which the tone was played on a phone, in order (location #1 is the left-most circle, location #7 is the right-most circle). All locations were detected by Daredevil. Blue dots denote the angle direction inferred per audio frame (1024 audio samples at 96 kHz). Green lines denote the beamforming directions. Small horizontal red lines represent the angles output by SSL after clustering and weighting the per-frame directions.

use our prior measured angular accuracies in a simulation for convenience. We define the location error as the distance in feet between the Daredevil computed location and the actual location of the user. We randomly select 50 user locations within 25 feet of each of two microphone arrays. At each location, we use two angular errors that are randomly chosen from Figure 9 (for each angle error we considered both the positive and the negative value). Figure 11 shows the CDF of the Daredevil location error in feet. The min-

Figure 10: Amplitudes of the 10 kHz audio tones received at one of the 8 microphones, after amplifying the signal by 40 dB, applying a band pass filter around 10 kHz, and then amplifying again by 40 dB. The vertical axis is amplitude, and the horizontal axis is time.

imum location error is 0.52 feet, the maximum error is 17.59 feet while the average error across all 50 locations is 3.19 feet (less than 1 meter). As baseline values we include the CDFs of two other localization schemes: Centroid (estimate the user location as the mid-point between the two microphone arrays) and Best Mic (estimate the user location as the location of the microphone array that is closest to the user). The CDF graphs show that Daredevil offers promising location accuracy, and can meet the requirements of the scenarios we outlined in § I.

V. Related Work

Indoor location is an active area of research, with a large number of published works. We cannot even begin to comprehensively cite all prior work in this space, but instead focus on location systems that use sound.

ActiveBat [6] relies on small devices, called Bats, to transmit ultrasonic pulses when instructed over RF. Bats are carried by users or attached to objects. Known-location receivers listen for the Bat ultrasonic pulse, and compute the distance to it based on the time difference between the RF request and the device ultrasonic pulse. The Bat location is computed centrally by the system using distance estimates to at least three receivers. ActiveBat avoids collisions by scheduling the Bat transmissions: it broadcasts over RF which Bat devices should transmit the ultrasonic pulse. The system determines the Bat location with accuracy of a few cm.

In Cricket [13], mobile devices correlate RF signals and ultrasonic pulses transmitted by hardware beacons installed in the environment. The mobile device

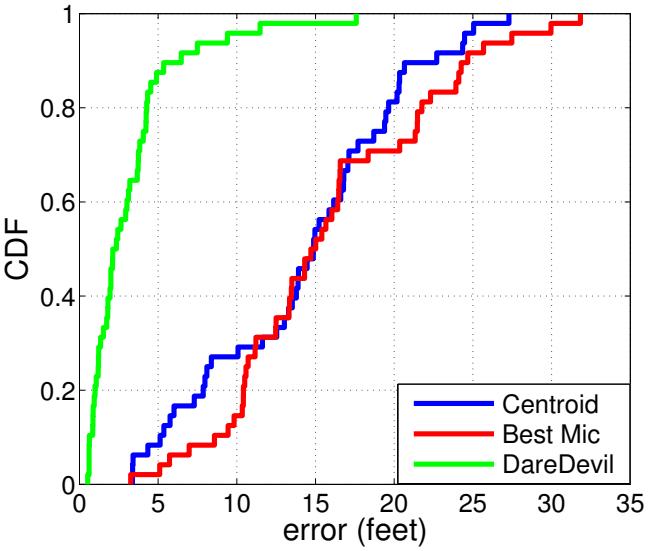

Figure 11: CDF of location error in feet for Daredevil, Centroid and Best Mic. Centroid and Best Mic represent baseline localization schemes. Centroid always estimates the user location at the mid-location between the two microphone arrays. Best Mic places the user at the closest microphone array location from the user ground truth location.

is located at the closest emitting beacon. The beacons are carefully placed in the environment to maximize location coverage and minimize interference. To avoid collisions and make correct RF-ultrasonic pulse correlations, the beacon transmissions times are randomized. The system achieves location granularities of 4x4 square feet.

We believe that Daredevil provides an interesting solution in a different part of the design space, where fewer devices are installed (two microphone arrays per room or area) and a different technique (SSL) is used to calculate location. In situations where a large number of devices can be installed in the environment, such as in ActiveBat and Cricket, perhaps finer location accuracy can be achieved.

WALRUS [2] provides room-level mobile device location. Each room is instrumented with a desktop computer which broadcasts simultaneously Wi-Fi and ultrasound beacons. Ultrasound beacons are confined by room walls and, as a result, these beacons are overheard only by devices co-located with the desktop computer. The mobile device infers room-location by correlating received Wi-Fi packets (possibly from multiple rooms) with the overheard ultrasound beacon(from the same room). To make correct correlations, WALRUS uses a maximum time difference between the Wi-Fi and ultrasound beacon and periodic variations in desktops broadcast times. The authors show that even in the case of 6 neighboring rooms, room-identification is correct 85% of the time. In contrast, Daredevil is locating phones within a room at much finer granularity using microphone arrays.

WebBeep [11] proposes a scheme based on time of flight to locate mobile devices. These devices inform over RF when they are about to play an audible tone (at 4kHz). Microphones detect the tone, and then compute a distance estimate to the device based on the speed of sound and the difference between the advertised time of play and the time the tone was heard. These distance estimates are affected by a constant, unknown time-delay between when the sound was advertised to play and when it actually played. By factoring in this unknown delay and relying on distance estimates to three microphones, the location of the device can be inferred. In a room 20x9m^2 covered with 6 microphones, the authors show an accuracy of 70cm, 90% of the time in a quiet environment.

Authors in [7, 12] built or used off the shelf UWB transmitters/receivers for location in an indoor environment. The advantage of wideband ultrasound over narrow band ultrasound (as used in the above systems) is to allow multiple, simultaneous access to the medium. A wide frequency spectrum can support code division multiple access (CDMA) techniques. In CDMA, special codes are assigned to transmitters to ensure their signals can be separated at the receiver, even when transmissions occur in parallel.

In contrast to these prior systems, we use a unique approach of using microphone arrays at carefully spaced intervals to use SSL to calculate the angle of incidence of sound. In § III.B, we already described the underlying prior work in SSL. However, the prior work in that space has focused on locating human voice, not locating phones, which comes with associated challenges of how phones can emit appropriate sound and how to schedule those transmissions and what accuracy can be achieved. SSL has been used in robotics, typically to locate human voice so that a robot can follow humans. For example in [20], the authors designed a sound location system to point the robot camera toward the direction of the sound source. When the source was placed at $3m$ from the robot, the mean direction error was 3°. Authors in [8] allow for simultaneous robot and multiple fixed sound source location. The sound sources are assumed fixed, while the robot can move freely through the space of a room. The bearing to sound sources is computed with accuracies between 3° and 9°. Similar work [19], tracked multiple people (2-4) while speaking. The system

used a combination of beamforming and particle filtering to distinguish between multiple sound sources in the environment. The authors showed direction accuracies of 10° when the sound source is up to $7m$ away from the robot.

VI. Conclusions

Daredevil is a system for locating phones in an indoor environment that has been instrumented with small microphone arrays. Each array can be as small as a pack of playing cards. Our system uses sound source localization to calculate the angle between a phone and each microphone array, and then triangulation to calculate the position. We use time-division multiplexing to schedule multiple phones that want to be located. Our implementation and evaluation demonstrate that low error can be achieved: 3.8°, or approximately 3.2 feet on average.

Our current implementation is limited by the quality of audio speakers on modern smartphones. At frequencies higher than 18 kHz, the amplitude of sound that these speakers generate is too low compared to the ambient noise floor. At human audible ranges, sufficient amplitude can be achieved. Our hope is that better speakers will be available on smartphones in the future. Nonetheless, at frequencies above 18 kHz, it is unknown what the impact will be on animals, especially service animals for the blind. One possibility of using audible frequencies such as 10 kHz is to embed our unique audio signal into music or sounds that the UI of an app may play while the user is interacting with the app.

There are two directions for future work in this space that we believe are promising. In this paper, we did not address the problem of segmenting the floor plan of a large room (such as a large department store) into squares, where each square is served by a pair of microphone arrays. In such a configuration, the schedule of audio transmissions by phones in one square would need to be coordinated with those in adjacent squares – a traditional coloring problem that would need to be solved. A second direction that we did not explore in this work is reversing the direction of audio. If we had wall mounted speakers that are transmitting at inaudible, high frequencies, can a microphone array on a phone determine its location? In such a scheme, no scheduling is needed since the same signal would be useful by all phones in the vicinity. Some modern smartphones have multiple microphones, primarily for noise cancellation during audio calls. If each audio stream from each microphone were exposed to the software stack on the phone, potentially SSL could be applied.

VII. Acknowledgments

We thank Mike Sinclair (MSR) for letting us use his hardware lab and helping us with the laser cutter, and we thank Bruce Cleary (MSR) for help with circuit boards.

References

[1] P. Bahl and V. Padmanabhan. RADAR: An In-Building RF-based User Location and Tracking System. In *IEEE INFOCOM*, 2000.

[2] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp. WALRUS: wireless acoustic location with room-level resolution using ultrasound. In *ACM MobiSys*, 2005.

[3] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha. FM-based Indoor Localization. In *ACM MobiSys*, 2012.

[4] I. Constandache, R. R. Choudhury, and I. Rhee. CompAcc: Using Mobile Phone Compasses and Accelerometers for Localization. In *IEEE INFOCOM*, 2010.

[5] V. Filonenko, C. Cullen, and J. Carswell. In *IPIN*, Sept. 2010.

[6] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *ACM MobiCom*, 1999.

[7] M. Hazas and A. Hopper. Broadband ultrasonic location systems for improved indoor positioning. *IEEE Transactions on Mobile Computing*, 5:536–547, 2006.

[8] J.-S. Hu, C.-Y. Chan, C.-K. Wang, and C.-C. Wang. Simultaneous localization of mobile robot and multiple sound sources using microphone array. In *IEEE ICRA*, 2009.

[9] S. Julier and J. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 1997.

[10] C. Knapp and G. Carter. The generalized correlation method for estimation of time delay. In *IEEE TASSP*, Aug. 1976.

[11] C. V. Lopes, A. Haghighat, A. Mandal, T. Givargis, and P. Baldi. Localization of off-the-shelf mobile devices using audible sound: architectures, protocols and performance assessment. In *ACM MC2R*, April 2006.

[12] K. Muthukrishnan and M. Hazas. Position Estimation from UWB Pseudorange and Angle-of-Arrival: A Comparison of Non-linear Regression and Kalman Filtering. In *LoCA*, pages 222–239, 2009.

[13] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *ACM MobiCom*, 2000.

[14] R. Schmidt. Multiple emitter location and signal parameter estimation. In *IEEE Transactions on Antennas and Propagation*, 1986.

[15] J. Sohn, N. Kim, and W. Sung. A statistical model based voice activity detector. In *IEEE Signal Processing Letters*, Jan. 1999.

[16] I. Tashev. Sound capture and processing: Practical approaches. In *John Wiley and Sons*, 2009.

[17] I. Tashev and A. Acero. Microphone array post-processor using instantaneous direction of arrival. In *IWAENC*, 2006.

[18] H. V. Trees. Optimum Array Processing. Part IV of Detection, Estimation and Modulation Theory. In *John Wiley and Sons*, 2002.

[19] J.-M. Valin, F. Michaud, and J. Rouat. Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. *Robotics and Autonomous Systems*, 55(3):216–228, 2007.

[20] J.-M. Valin, F. Michaud, J. Rouat, and D. Letourneau. Robust sound source localization using a microphone array on a mobile robot. In *IEEE IROS*, 2003.

[21] H. Wang and P. Chu. Voice source localization for automatic camera pointing in videoconferencing. In *ICASSP*, 1997.

[22] D. Ward and R. Williamson. Particle filter beamforming for acoustic source localization in reverberant environment. In *ICASSP*, 2002.