Spartacus: Spatially-Aware Interaction for Mobile Devices Through Energy-Efficient Audio Sensing

[1]Zheng Sun, [1]Aveek Purohit, [2]Raja Bose, and [1]Pei Zhang
[1]Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA
[2]Microsoft Silicon Valley, Sunnyvale, CA
{zhengs, apurohit, peizhang}@cmu.edu, {raja.bose}@microsoft.com

ABSTRACT

Recent developments in ubiquitous computing enable applications that leverage personal mobile devices, such as smartphones, as a means to interact with other devices in their close proximity. In this paper, we propose Spartacus, a mobile system that enables spatially-aware neighboring device interactions with zero prior configuration. Using built-in microphones and speakers on commodity mobile devices, Spartacus uses a novel acoustic technique based on the Doppler effect to enable users to accurately initiate an interaction with a neighboring device through a pointing gesture. To enable truly spontaneous interactions on energy-constrained mobile devices, Spartacus uses a continuous audio-based lower-power listening mechanism to trigger the gesture detection service. This eliminates the need for any manual action by the user.

Experimental results show that Spartacus achieves an average 90% device selection accuracy within 3m for most interaction scenarios. Our energy consumption evaluations show that, Spartacus achieves about 4X lower energy consumption than WiFi Direct and 5.5X lower than the latest Bluetooth 4.0 protocols.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems**]: Signal processing systems

Keywords

Interaction; audio sensing; device pairing; spatial interaction; gesture; mobile system; context aware

1. INTRODUCTION

Recent developments in ubiquitous computing enable applications that leverage personal mobile devices, such as smartphones, as a means to interact with other devices in their close proximity. Numerous application scenarios are either commonly observed or can be foreseen in the near future, ranging from *Person-to-Person* interactions – For example, conference attendees exchanging contact information, or friends playing multi-player pass-the-parcel mobile games, to *Person-to-Device* interactions – For example, a student printing out a document by interacting with a nearby printer, remote control of projectors, accessing product information in stores, controlling digital display screens, or changing thermostat settings.

Establishing such an interaction requires that the device initiating the interaction makes the target device aware of its intent and sets up a connection, without any prior configuration. While users themselves can intuitively know and identify the nearby target device based on its relative location, the existing methods require additional manual effort or prior configuration to translate this *spatial-awareness* to an identifier understandable by the device.

For example, device discovery features in Bluetooth or Wi-Fi can be used to initiate the interaction, where the user must scan for nearby devices and select the target from a list. Similarly, other methods have been proposed where users (the initiator and the target) must share information a priori [3] or perform synchronized actions [6].

Recent work, Point & Connect [15], has proposed a novel system that uses a pointing gesture to initiate interactions between a mobile device and its target. However, the system requires an initial channel of communication such as a local Wi-Fi or Bluetooth network to exist beforehand to enable a device to *point-and-connect* to the target device. In addition, due to the energy-constrained nature of mobile phones, the system service cannot be run continuously in the background and requires users to manually trigger it, on all nearby devices.

In this paper, we propose Spartacus, a mobile system that enables spatially-aware neighboring device interactions with zero prior configuration. First, Spartacus uses a novel acoustic technique based on the Doppler effect to enable users to accurately initiate an interaction with a particular target device in their proximity through a pointing gesture. Second, Spartacus uses a continuous audio-based low-power listening service that runs in the background and automatically triggers the relatively power hungry gesture detecting service. This enables Spartacus to run continuously in the background and removes the need for any manual user actions prior to the interaction. The system can be implemented as a software application on commodity mobile devices without any special hardware or software requirement.

As a proof of concept, we implemented the Spartacus

MobiSys'13, June 25−28, 2013, Taipei, Taiwan

system on the Android smartphones, and tested the system using various device models under different interaction scenarios. Our experimental results show that Spartacus support spontaneous device interactions with an average of 90% device selection accuracy within 3m for most interaction scenarios. Evaluation results of energy consumptions show that, Spartacus consumes about 150mW power. As a reference, we also compare energy consumption of Spartacus with the state-of-the-art peer-to-peer scanning techniques, and show that Spartacus achieves about 4X lower energy consumption than WiFi Direct and 5.5X lower than the latest Bluetooth 4.0 protocols, respectively.

The key contributions of this paper are as follows:

1. We proposed a novel acoustic technique based on the Doppler effect to enable spatially-aware interactions between devices that provides high accuracy and supports numerous natural human gestures.

2. We developed a novel undersampling audio signal processing pipeline that achieves better accuracy without increasing computational complexity, allowing the method to be used on commodity mobile devices.

3. We designed and implemented a low-power listening protocol using periodic audio sensing to trigger gesture detection that reduces energy consumption and enables the system to run continuously in the background. This enables the interaction to be truly spontaneous without any manual actions on part of the user.

4. We analyzed and experimentally validated the design tradeoffs in achieving low latency and power consumption given hardware and software limitations of commodity mobile devices.

The rest of the paper is organized as follows. Section 2 gives a system overview of Spartacus. We describe the design of algorithms in Section 3 and discuss implementation details in Section 4. Section 5 shows evaluation results. Related work is shown in Section 6. Finally, Section 8 concludes the paper.

2. SYSTEM OVERVIEW

Spartacus supports spatially-aware device selections in close proximity (i.e. within 5m), using intuitive pointing gestures. As illustrated in Figure 1, when a user wants to initiate an interaction with a nearby device, she makes a pointing gesture towards the device using her mobile phone. Then the interaction with the target device is automatically initiated.

To select the target device, Spartacus emits a continuous audio tone at a known frequency during the course of the pointing gesture. Other devices that are close to the user's phone will be able to capture the tone via their microphones. Due to the motion of the gesture, a Doppler frequency shift is observed in the received audio deviating from the original frequency, which is a monotonically increasing function of the gesture velocity [4]. Therefore, since the gesture is made in the direction of the target device, the target device can be isolated by finding the *peak frequency shift* among nearby devices.

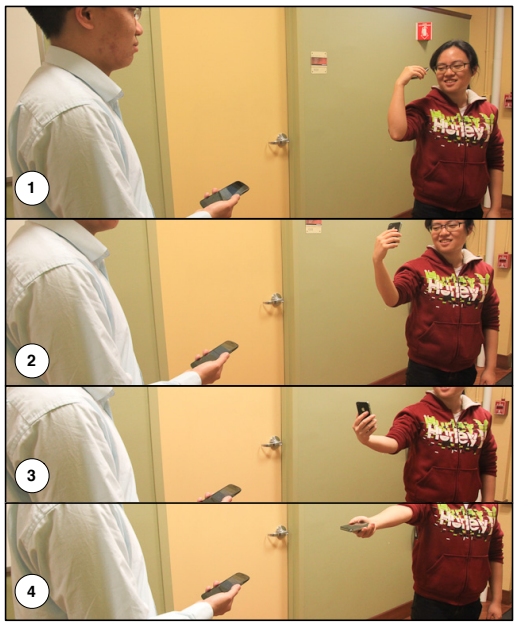


Figure 1: An interaction scenario of the Spartacus interaction system. A user selects a nearby device for interaction by quickly pointing her mobile phone towards the targeting device.

To eliminate any configurations from users in the entire interaction process, Spartacus needs to automatically trigger the gesture detection service on nearby devices before the gesture. Previous work adopts a continuous listening approach [15]. Given the limited energy budget on mobile devices, this approach becomes impractical to be active all the time and users are required to manually turn on the gesture detection service before the interaction. In contrast, Spartacus solves this problem by providing a zero-configuration interaction trigger mechanism, whereby mobile devices running Spartacus continuously perform a periodic low-power listening using their built-in microphones. Before a user issues the gesture, an audio beacon with a short duration (i.e. a couple of seconds) is emitted. Nearby devices that successfully capture this beacon trigger their gesture detection service and begin listening for the gesture tone from the initiator. After the gesture is over, the gesture detection service is turned off to conserve energy. Our experimental results show that, the low-power listening protocol is more energy-efficient than other state-of-the-art peer-to-peer scanning techniques, such as WiFi Direct and Bluetooth 4.0. (Detailed in Section 5.5)

We have implemented the Spartacus system on the Android mobile platform (Detailed in Section 4). The entire system is implemented in software, and runs as a mobile phone app, without any change to the existing mobile operating system and APIs. Since Spartacus leverages existing built-in microphones and speakers on commodity mobile devices, such as smartphones, it does not require any extra hardware.

2.1 Design Challenges of Spartacus

To support spatially-aware device selections and zero-configuration interaction triggers, the Spartacus system addresses a number of major technical challenges:

1. **High Resolution Doppler-Shift Detection** – Spartacus selects the target device through the analysis of peak Doppler frequency shifts in the pointing gestures. However, pointing gestures of average users are usually transient (i.e. generally shorter than 0.5s), and the gesture velocity is low, which leads to limited Doppler frequency shifts. Thus, accurately detecting the transient peak velocities requires digital signal processing techniques with a high frequency resolution, without sacrificing the time resolution. Spartacus addresses this challenge by utilizing an undersampling technique, and increases the frequency-domain resolution by 5X as compared to traditional FFT-based approaches. This translates to a 2X increase in angular resolution. (Detailed in Section 3.1)

2. **High-Accuracy Device Selection** – To successfully select the target device, Spartacus needs to accurately estimate the peak frequency shifts observed by each nearby device, and to select the one with the maximum peak shifts. To address this challenge, we design and implement a bandpass audio signal processing pipeline in Spartacus, which is robust against ambient and intermittent high frequency acoustic noises. (Detailed in Section 3.2)

3. **Energy-efficient Interaction Trigger** – To trigger interactions on nearby devices without user configurations, mobile devices need to be *always* ready to capture the pointing gestures, while keeping the energy consumption low. Spartacus addresses this challenge by designing a low-power audio listening protocol that periodically detects incoming interaction triggers. (Detailed in Section 3.3)

3. SYSTEM DESCRIPTION

Using Spartacus, a user initializes a device interaction process by pointing her mobile phone towards a target device, during which an audio tone with a known frequency is emitted. Spartacus utilizes audio signal processing techniques to detect the frequency shifts in the audio tone, such that the target device can be selected by searching for the maximum peak frequency shifts among multiple candidate devices. This section provides detailed system description of this process.

3.1 Detect Doppler Shift with High Resolution

To select the target device, Spartacus emits a continuous audio tone at a known frequency f_0 during the course of the pointing gesture. According to the Doppler effect, the frequency of the received tone can be calculated as $f = (\frac{c+v_R}{c-v_S})f_0$, where c, v_R, and v_S are the speed of sound, of the gesture receiver, and of the gesture sender, respectively [4]. In our case, assuming the receiver is stationary during the course of the gesture (i.e. $v_R = 0$), and the speed of sound c is constant, the observed frequency becomes a monotonically increasing function of v_S. Since the user made the gesture directionally towards the target device, the target device would be able to observe the maximum Doppler shift. Thus, by comparing the peak frequency shift among nearby devices, the target device can be selected.

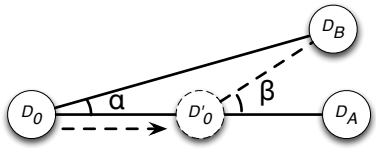


Figure 2: Geometry of tone transmission between mobile devices. When D_0 is moved towards the target device D_A during the pointing gesture, the phone displacement increases the directional difference from α to β.

3.1.1 Deriving Angular Resolution

Spartacus selects the target device D_A by measuring the time-varying Doppler frequency shifts. This is accomplished by running an audio signal analysis process over a series of overlapped audio frames. Applying the FFT transform on each audio frame, the quantity of the frequency shift in this audio frame of D_A can be calculated as $\Delta n_A = (f_A - f_0) \cdot \frac{N_{FFT}}{F_s}$, where f_A is the observed tone frequency of D_A, f_0 the frequency of the original tone, F_s the sampling rate, N_{FFT} the number of FFT points, and Δn_A the calculated frequency shift expressed in terms of FFT points. Assume the target device is stationary during the course of the gesture, i.e. $v_R = 0$, given the equation of the Doppler effect, Δn_A can also be expressed as $\Delta n_A = \frac{c}{c-v_S} \cdot \frac{f_0 \cdot N_{FFT}}{F_s}$, where c and v_S are the speed of sound and the gesture sender, respectively. As shown in Figure 2, suppose another device D_B is in close proximity to the target device D_A. Let the angle between the three devices be α, then the velocity observed at D_B will be $v_S \cdot \cos\alpha$. Therefore, correctly selecting D_A to be the target device requires Δn_A is at least one FFT point larger than Δn_B through the FFT analysis, i.e. $\Delta n_A - \Delta n_B > 1$. Given that c is constant and the maximum velocity of a particular gesture is fixed, this requirement can be expressed as $(\frac{1}{c-v_S} - \frac{1}{c-v_S \cdot \cos\alpha}) \cdot \frac{c \cdot f_0 \cdot N_{FFT}}{F_s} > 1$. Reorganizing this inequality, the minimum differentiable α_{res}, i.e. the angular resolution, can be expressed as

$$\cos\alpha_{res} < \left(c - \frac{1}{\frac{1}{c-v_S} - \frac{Q}{c}} \right) / v_S, \qquad (1)$$

where $Q = \frac{F_s}{f_0 \cdot N_{FFT}}$. Using Equation 1, we can compute the achievable angular resolution of Spartacus. For example, suppose a user used audio tones at 20KHz, and pointed her phone with an average 3.4m/s peak velocity. At the receiver device, the audio signal was sampled at 44100Hz, utilizing a 2048-point FFT, then the angular resolution would be 26.7°. Suppose a target device is 5m away, this translates to 2.3m spatial resolution, which is too large to conduct practical interactions! We will evaluate the velocity variance of the pointing gestures in Section 5.1.

3.1.2 Improving Resolution using Undersampling

To improve the angular resolution, Equation 1 indicates that α_{res} is a monotonically increasing function of Q, which implies three options to reduce α_{res}: 1) increasing the original tone frequency f_0, 2) increasing the number of FFT points N_{FFT}, or 3) decreasing the sampling rate F_s.

The first two options both have drawbacks. For the first option, audio tones with higher frequencies experience stronger energy degradation, which consequently reduces the

supported interaction range (We will evaluate the effect of energy degradation of sound in Section 4.2). For the second option, increasing the number of FFT points would involve a higher computational burden. Our experimental results indicate that, using 10ms audio frames and 2048-point FFT, processing one second audio takes 7 seconds on modern mobile devices, which is too slow for any interactions that require high responsiveness, such as mobile gaming.

To avoid these drawbacks, Spartacus exploits the last option, which is to decreasing the sampling rate F_s. This option seems to be impractical because Nyquist sampling theorem states that the sampling frequency has to be larger than twice the maximum signal frequency, to perfectly reconstructed the original signal, otherwise the sampled signal would be aliased [14]. However, if the bandwidth of a bandpass signal is significantly smaller than the central frequency of the signal, it is still possible to sample the signal at a much lower rate than the Nyquist sampling rate, without causing the alias.

This technique is called undersampling, which has been used in RF communication and image processing systems, for analyzing bandpass signals [5, 20]. In Spartacus, since the tones are transmitted at a very high frequency (i.e. above 18KHz), whereas the frequency shifts of tones are only a few hundred Herz, the entire received audio tone can be seen as a bandpass signal. Therefore, using the undersampling technique can significantly reduce the required sampling rate. The effect of the undersampling technique to the spectrums is shown in Figure 4.

Denote the lowest and the highest band limits of the received frequency-shifted tone as f_L and f_H, respectively, then the bandwidth of the signal is $B = f_H - f_L$. According to the undersampling theorem, the condition for an acceptable new sampling rate is that shifts of the bands from f_L to f_H and from -f_H to -f_L must not overlap when shifted by all integer multiples of the new sampling rate $F_s^\star$ [5]. This condition can be interpreted as the following constraint:

$$\frac{2 \cdot f_H}{n} \leq F_s^\star \leq \frac{2 \cdot f_L}{n-1}, \forall n \colon 1 \leq n \leq \lfloor \frac{f_H}{B} \rfloor, \qquad (2)$$

where $\lfloor \cdot \rfloor$ is the flooring operation, B the signal bandwidth, and $n = F_s/F_s^\star$ the undersampling factor. In our experiments, we observe that average users can generate pointing gestures with an average peak velocity of 3.4m/s, which equals to 200Hz frequency shift. Considering the edge effect of FFT transforms, we conservatively assume the bandwidth of the received tone signal is 2KHz, which is sufficient to avoid spectrum aliasing for human pointing gestures. Take this value of the bandwidth into Equation 2, we get a list of possible $F_s^\star$ and n combinations, as shown in Table 1.

3.1.3 Determining Undersampling Parameters

These parameter combinations provide rich options for us to design our system. However, when choosing undersampling parameters, there are a number of design considerations: 1) A higher n is generally favored, as it leads to a lower new sampling rate, which results in a better angular resolution. 2) Since the central frequency of the tones is $f_L + \frac{B}{2}$, a higher f_L increases the central frequency of the tones. During our experiments, as will be shown in Section 4.2, higher tone frequencies generally cause greater energy degradation of sound, which would significantly reduce the interaction range and the accuracy of tone detection. In practice, to support sufficient interaction range and achieve optimal device selection accuracy, we empirically avoided using f_L higher than 19KHz (i.e. the entire bandwidth of the audio tone resides between 19KHz and 21KHz). However, we would note that, thought this general design consideration holds, the frequency of the audio tones could also be varied according to the specific frequency response performance of the microphones and speakers. 3) Commodity mobile devices support a limited choices of audio sampling rates, which generally include 8KHz, 16KHz, 32KHz, 44.1KHz, and 48KHz, etc. This means the original audio sampling rate can not be arbitrarily set, which consequently limits the choices of $F_s^\star$ and n.

After examining all these design considerations, we found that, only when n=5, 6, or 7 given F_s = 44.1KHz, or when n = 4 given F_s = 48KHz, the parameter combinations satisfy the central frequency requirement. Furthermore, among these parameter combinations, most of the cases lead to a low undersampling factor n, except the case where $n = 7$ given F_s = 44.1KHz. Thus, we finally choose this pair of parameters in Spartacus, which yields a new sampling rate $F_s^\star = 44.1/7 = 6.3$KHz. Given this lower sampling rate, based on Equation 1, the theoretical angular resolution is improved to 10°. As compared to the original angular resolution of 26.7° discussed in Section 3.1.1, this is more than 2.5X better than the original resolution.

3.2 Select Target Device with High Accuracy

3.2.1 Bandpass Signal Processing Pipeline

Using the undersampling technique, we set the central tone frequency at 20KHz and use a bandwidth of 2KHz, which satisfies the f_L requirement shown in Table 1. This design choice also avoids stronger energy degradations. Using the undersampling process, the spectrum of the original audio samples is essentially mapped from 19KHz-21KHz to a much lower spectrum from 0.58KHz-2.58KHz with a central frequency at 1.58KHz. However, since the new sampling rate is much lower than the Nyquist rate, aliasing arises in the original sampled audio signals. This makes the audio tone buried under ambient noises and makes frequency shift detection impossible.

Table 1: Candidate combinations of undersampling parameters. "⋆" is finally chosen.

F_s (KHz)	n	$F_s^\star = F_s/n$ (KHz)	Supported f_L (KHz)
44.1	5	8.8	(17.7, 20.0)
44.1	6	7.3	(18.4, 20.0)
44.1	7	6.3	(18.9, 20.0) ⋆
44.1	8	5.5	(19.3, 20.0)
48	4	12	(18.0, 22.0)
48	5	9.6	(19.2, 22.0)
48	6	8	(20.0, 22.0)
48	7	6.9	(20.6, 22.0)

To solve this problem, we designed and implemented an audio signal processing pipeline, as shown in Figure 3, to recover the frequency shifts. First, each mobile device receives and samples audio data at the default 44.1KHz rate. As will be discussed in Section 5.1, our experimental results show that the peak velocities of pointing gestures only lasts tens of milliseconds, we split consecutive audio samples into 10ms analysis windows, with a 75% overlapping ratio, achieving a high time-domain resolution. Then, a 10-order Butterworth

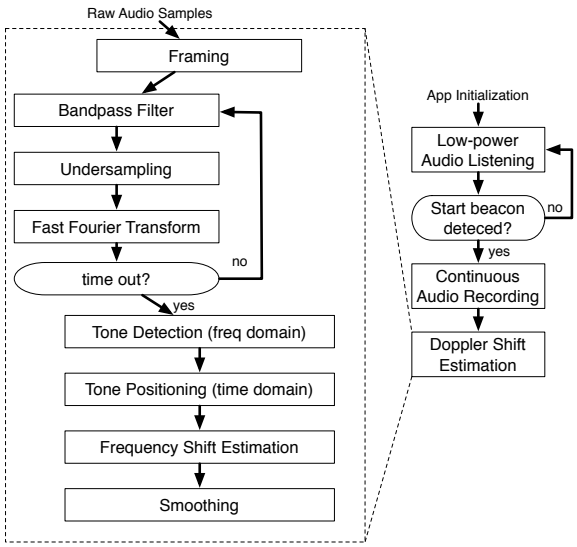


Figure 3: Signal processing pipeline.

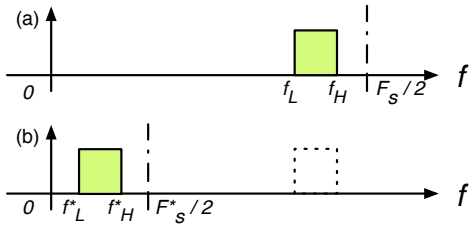


Figure 4: Illustration of the effect of undersampling to the spectrums. (a) The originally received audio tone is located from f_L to f_H. (b) After undersampling the audio samples, the tone is shifted to lower frequencies.

bandpass filter is used to attenuate out-of-band signals lower than 19KHz or above 21KHz. Then, the filtered audio samples go through the undersampling filter, which uses the new sampling rate $F_s^\star$. With $n = 7$, this filter essentially keeps every 7th sample and deletes other samples. Note that although this process reduces the number of samples being analyzed, it still preserves the necessary frequency-domain information due to undersampling. The entire bandpass-filtering and undersampling operations process audio samples as a streamline, with a total time complexity of only $O(N)$ time, which is efficient for responsive interactions.

To estimate frequency shifts, each undersampled audio frame is processed using a 2048-point FFT transform, and FFT transforms of multiple consecutive audio frames are processed altogether to find the frequency shift. Specifically, an energy-based tone detector first compares energy of the spectrum of each frame from 1.08KHz to 2.08KHz (i.e. corresponds to 19.5KHz to 20.5KHz of the original spectrum before undersampling) to that of the entire 0Hz to 3.15KHz (i.e. $F_s^*/2$) spectrum. If the former energy is M times greater than the latter energy, this audio frame is set to "1", otherwise "0". Second, a tone positioning operator determines the frequency bin with the highest energy for each frame that have been set to "1", and links these frequency bins through a moving-average smoothing operation. This operation eliminates intermittent high frequency spikes caused by acoustic noises, such as clangs of metals.

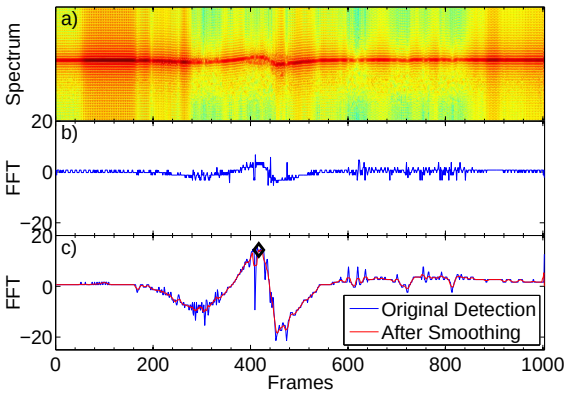


Figure 5: Effect of the undersampling analysis. (a) Spectrogram of the original audio. (b) Frequency shift detection using the traditional FFT approach. (c) Frequency shift detection using the proposed undersampling technique. The detected peak frequency shift is marked as "◇".

Finally, the peak frequency shift is determined by finding the maximum frequency shift among all the frequency bins. During our experiments, we found that $M = 1.5$ led to robust performance in various indoor environments. The effect of this process is shown on Figure 5. Note that, by using the undersampling technique, the amount of frequency shifts has been significantly increased as compared to the traditional FFT approach.

After each device detects the Doppler frequency shifts, all the devices that detect a positive shift value report their frequency shift to the sender device, along with the device's ID information. The sender device then compares all the received Doppler shifts and determines the target device.

We would like to note that, when Spartacus is used in a crowded indoor scenarios, such as an airport, where numerous devices might be interacting with each other, contentions may occur. However, due to the intrinsic rapid energy degradation of audio signals, as will also be shown in Section 4.2, different interaction sessions can be automatically separated if they are spatially far away from each other (i.e. more than 5m). If many interactions take place in close proximity, a coordination mechanism could be used to create a contention window that conditionally accepts the reports of the Doppler shifts, so that different interaction sessions can be separated in the time domain. Previous research provides various contention-control schemes, however, this is out the scope of the current work.

3.2.2 *Angular Gain through Pointing Gestures*

Spartacus uses the bandpass signal processing pipeline to estimate frequency shifts in audio tones. However, due to geometric constrains, the condition of whether a nearby device would be selected as the target device is determined by the angular resolution α_{res} of the system. As indicated in Section 3.1, when the number of FFT points is 2048, the smallest angular resolution is 10° when the undersampling factor n is equal to 7. However, during our tests in practice, we found that when candidate devices are close to the user (i.e. within 3m), the device selection accuracy is better than the analysis. This improvement is caused by

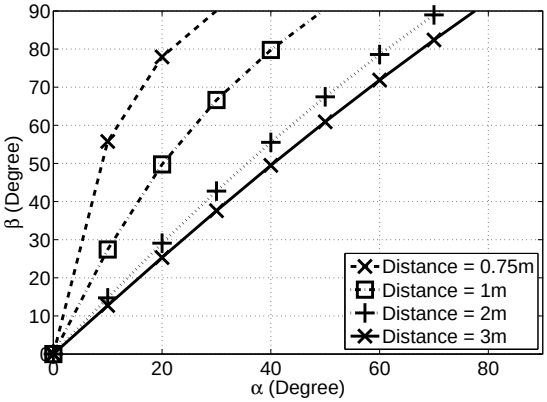


Figure 6: Angular gains caused by the stretch of user arms during the pointing gestures. These gains improve the device selection accuracy of Spartacus.

the increased directional difference, as illustrated in Figure 2. During a pointing gesture, since the user stretches her arm toward the target device D_A, the phone displacement makes the effective directional difference α increase. This angular change is significant when the candidate devices D_A and D_B are close to D_0. As shown in Figure 6, when D_A and D_B are 0.75m away and $\alpha = 10°$, the two devices are supposed to be barely differentiable. However, assuming the user's arm is 60cm, the effective angular difference β is increased to 55°, which makes the two devices much easier to be differentiated. Clearly, this angular gain diminishes when the distance between D_0 to the two devices increases, as well as when the user does not fully stretch her arm during the pointing gestures.

3.3 Energy-efficient Interaction Triggering

Spartacus seeks to enable spatially-aware interaction between an initiating device and receiving devices in its neighborhood. This requires that the receiving devices are *listening* to gesture broadcasted by the initiator. However, such continuous listening for gestures consumes a significant amount of energy and is a challenge to realize on energy-limited mobile devices. Therefore, in Spartacus we leverage a low-power audio listening protocol to save energy. This section describes the design details of this protocol.

3.3.1 Low-Power Audio Listening

The audio based low-power listening protocol has the following advantages:

1. Ubiquitous Hardware Support – Microphones and speakers are ubiquitous on most devices such as mobile phones, laptops, smart-TV sets, etc. No extra hardware modifications are need to implement this protocol. In addition, the cost of commodity audio hardware is very low.

2. Limited Range – The objective of this protocol is to initiate interaction strictly with devices in close proximity to the sender, typically line of sight and single space scenarios. Audio has a limited range and attenuates considerable when passing through walls. This makes the medium better suited for detecting neighboring devices within the same space, unlike radio based discovery such as WiFi and Bluetooth. This advantage also helps the Spartacus system to automatically separate concurrent interaction sessions that are spatially apart.

3. Energy Efficient – The continuous periodic listening and beaconing protocol using audio is orders of magnitude more energy efficient than discovery schemes using WiFi or Bluetooth, partly because these communication protocols are not designed for continuous discovery. Moreover, fine grained control or modification of WiFi and Bluetooth device discovery schemes is not possible on consumer hardware devices.

The audio based protocol has two major modes:

1. Periodic Listening – All devices periodically wake up (every T_{rx}) and record sound for a duration d_{rx}. The time d_{rx} is the amount of audio required to detect if a specific beacon tone frequency is being broadcasted by an initiating node.

2. Beaconing – Whenever a sender node wishes to broadcast a gesture, it first emits a beacon tone for a duration d_{tx}. This guarantees that all the receiving devices in range will receive the beacon and switch to continuous listening mode to record the gesture.

The theory of low-power listening states that, to guarantee every beacon will be successfully detected by nearby devices, the beacon duration d_{tx} must be at least as large as T_{rx} [16]. This relationship indicates that, the shorter the duration of the beacons, i.e. T_{rx}, the shorter the user needs to wait before starting the gesture, thus the more natural the gesture can be accomplished. However, a short beacon duration requires increasing the duty cycles of the interaction listening on the receivers, which consequently consumes more energy. We will evaluate the tradeoff between energy consumption vs. duty cycles in Section 5.5.

To identify the sender, Spartacus encodes the device ID using the Reed-Solomon coding, and sends the ID in a transmission immediately succeeding the beacon transmission. In the current implementation of Spartacus, the device ID is modulated using a 16 Frequency Shift-Keying (FSK) scheme with a central frequency at 19KHz. Keys are separated in the frequency domain using a 50Hz guard band. Note that, since the central frequency of the tone emitted during the pointing gesture is 20KHz, the transmission of the device ID is at least 200Hz lower than the gesture tone, thus will not cause ambiguities.

3.3.2 Dealing with Wakeup Jitter

The low-power listening protocol relies on the assumption that the duration of the beacon (d_{tx}) transmitted by the gesture transmitter is greater than the maximum interval (T_{rx}) between adjacent listening events on the receiving devices. However, since mobile platforms, such as Android, are not real-time operating systems, wakeup jitters can be observed between when an API starts recording sound and when the system actually begins recording. We empirically measure the latency of an API call on Galaxy Nexus Android phones. As shown in Figure 7, the average jitter in the wakeup timer is about 70ms, with a standard deviation of 15ms. This result indicates that, using the original beacon duration will cause the receiving devices to miss interaction triggers, which leads to a failure of capturing the upcoming gesture tone. To solve this problem, as shown in Figure 8, we include an additional guard band in the beacon length based on empirical measurements to account for the wakeup jitters.

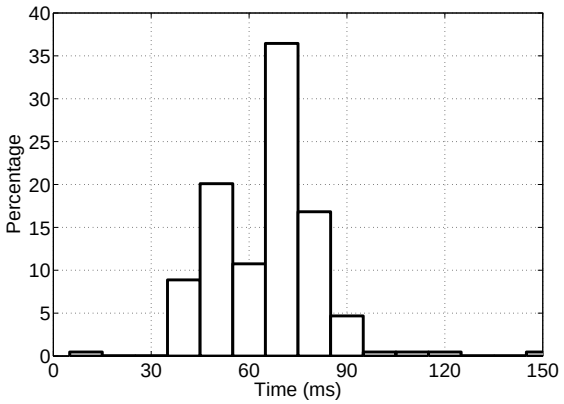


Figure 7: Distribution of wakeup jitters.

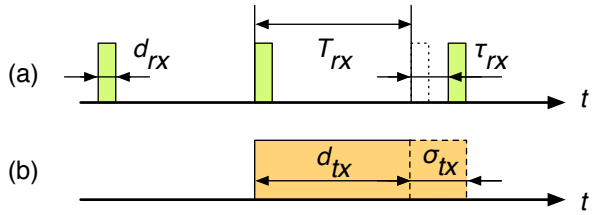


Figure 8: The graph shows the relationship between the wakeup period T_{rx} and the wakeup duration d_{rx} of the gesture receivers, and the beacon duration d_{tx} of the gesture transmitter. Note that, due to the existence of the wakeup jitter τ_{rx}, an additional guard band σ_{tx} is used in the beacons.

4. IMPLEMENTATION

To evaluate the interaction system, we implemented the Spartacus system on the Android platform, and tested it on various phones, including Galaxy Tab, Nexus 7, Galaxy Nexus, and HTC One S. This section discuss the implementation details of Spartacus.

4.1 Software Implementation

To evaluate the interaction system, we implemented the Spartacus system on the Android platform, and tested it on various phones, including Galaxy Tab, Nexus 7, Galaxy Nexus, and HTC One S. The current Spartacus has a client end and a server end. The client end runs as an Android app, which has 4 components: `GestureSensing`, `LowPowerListening`, `AudioModem`, and the GUI. The design of Spartacus strictly follows the Object-Oriented programming paradigm, such that each component is standalone, and can be easily incorporated into any existing apps as an add-on interaction support. For example, if an app needs to conduct the low-power audio listening, it simply calls the `LPL.start()` to start the listening thread, and a message handler is used to receive notifications in case an interaction request is detected. Similarly, for issuing gestures, an app could simply call `GestureSensing.makeGesture()`, then an audio tone at a predefined frequency will be generated during the gesture; on the receiver's end, a counterpart `GestureSensing.analyzeGesture()` will trigger a background thread that runs the signal processing pipeline described in Section 3.2 to compute the frequency shifts. The entire Spartacus project contains a total of 3500+ lines of Java code, including both the server and the client.

4.2 Hardware Limitations on Mobile Devices

To generate and receive audio tones emitted during the course of the gestures, Spartacus leverages the built-in speakers and microphones on mobile devices, and use software-implemented digital signal processing algorithms to detect the gestures. In our implementation process, we conducted experiments to understand the limitations of modern mobile devices in emitting high frequency audio tones.

Previous research has found that microphones on commodity mobile phones supports sampling rates up to 48KHz, which enables the use of tone frequencies as high as 24KHz [4, 7, 8]. In Spartacus, we use tone frequencies higher than 20KHz, which is inaudible [8]. However, a downside of using high frequency tones is the potential stronger energy degradation of sound. To investigate this effect on mobile devices, we model the transmission of audio tones as three consecutive processes, as shown in Figure 9. Similar to a RF communication system, the degradation of signal energy occurs inside of the speakers and the microphones, as well as during the transmission in the acoustic channel. We investigate each of these processes.

First, to quantize the energy degradation of sound, we benchmarked the frequency responses of speakers and microphones on various mobile devices. A Sennheiser MKE 2P microphone and a Yamaha NX-U10 speaker are used as references. As shown in Figure 10(a) and 10(b), we observe that, due to the characteristics of the hardware, a significant energy degradation exists for audio tones higher than 15KHz. This phenomenon is not surprising considering that the hardware of mobile phones are designed intentionally for human conversations and music, which feature dominant energies generally lower than 15KHz. We also found that, as tone frequencies go up, the degradation becomes even larger. To be specific, as the tone frequency increases every 1KHz, the degradation of sound energies increases 5dB on speakers, and 3.3dB on microphones, respectively. Second, we investigate the degradation of sound energy caused by the transmission through acoustic channels. As shown in Figure 10(c), we observe that there is an average 3.2dB/m energy decrease of sound from 1m to 6m, irrespective of the tone frequency.

In summary, we found that the energy degradation of sound is collaboratively caused by the frequency responses of the speakers, the microphones, and the transmission through acoustic channels. For interactions in close proximity, the degradation largely comes from the frequency responses of the hardware. These results indicate that, to reduce energy degradation and increase interaction range, audio tones with lower frequencies should be leveraged.

5. EVALUATION

In this section, we evaluate the performance of Spartacus given different interaction scenarios. Spartacus leverages the pointing gestures of users to accurately select the target device, without user configuration. To understand the boundary of system performance, we first conduct experiments to investigate trajectory and velocity variances of the pointing gestures of average users. Then we provide performance analysis of Spartacus, by discussing the device selection accuracy and the interaction range. Finally we discuss the

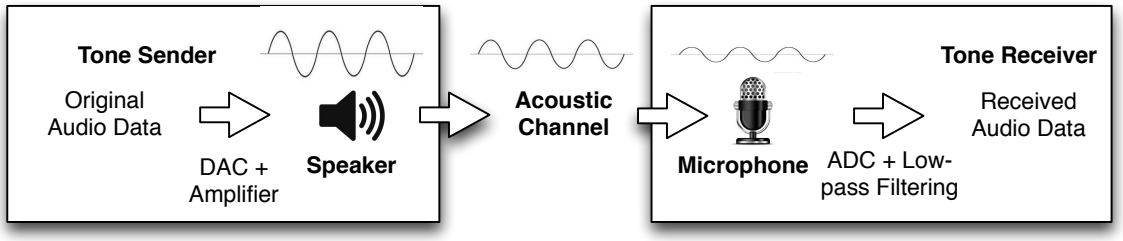
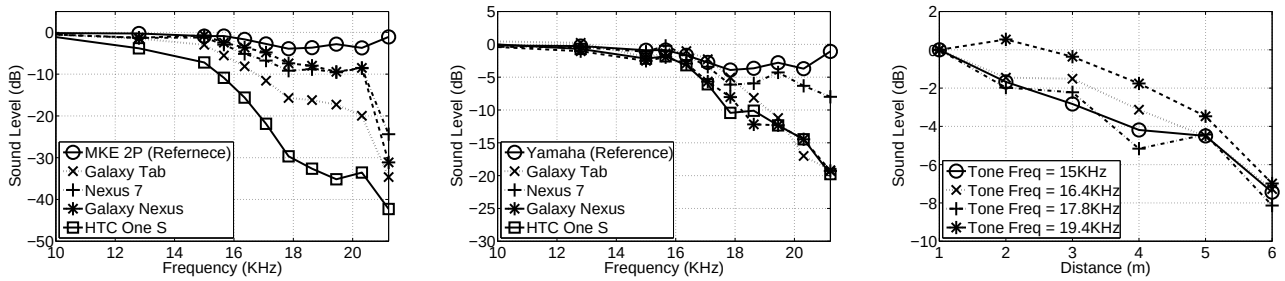


Figure 9: Audio tone transmission between mobile devices.



(a) (b) (c)

Figure 10: A graph showing frequency responses of various mobile devices (1/16 octave band-filtered from 15KHz). (a) Frequency responses of speakers as a function of the tone frequency. (b) Frequency responses of microphones as a function of the tone frequency. (c) Attenuation of sound energy vs. distance.

interaction latency, and present the power consumption of Spartacus as compared to other state-of-the-art techniques.

5.1 Evaluation of Pointing Gestures

To accurately select the target device, Spartacus makes an assumption that when a user points her phone towards the target device, the target device will always observe the largest Doppler frequency shift. However, in practice, user gestures can be significantly diverse in terms of directional precision, velocity, and trajectory. Such diversities make accurately estimating the peak frequency shift significantly challenging.

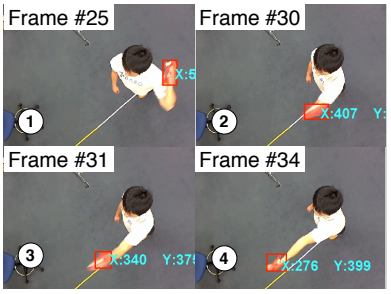
In order to investigate what factors may affect the performance of Spartacus, we conducted experiments to fully understand the characteristics of pointing gestures of average users. Some fundamental questions that we would want to investigate include:

1. *How diversely do users point their phones, and how fast can a user point?*
2. *If the user points fast enough, how often does the target device observe the highest frequency shift, thus the highest velocity, of the gesture?*
3. *If we want to estimate the frequency shifts, how much frequency- and time-domain resolution do we need to successfully capture the peak frequency shift inside of a gesture?*

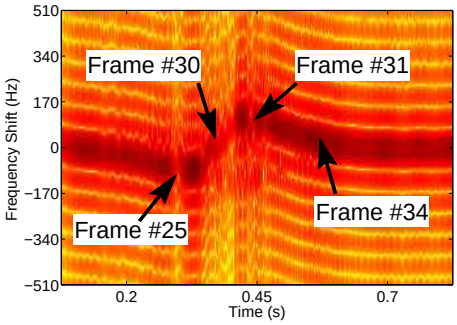
To answer these questions, we conducted experiments with 12 participants (6 females) and investigated their pointing gestures. To capture the gesture trajectories, we attached a video recorder on the ceiling right above the participants, and videotaped the entire experiment. The video we took recorded complete 2D trajectories of the gestures. Before doing the experiment, we briefed the participants on the idea of Spartacus, and let them to freely choose any *natural* gestures they wanted. During the experiment, each participant performed 10 gestures towards a target device 2m away from them, using a Galaxy Nexus phone. A red marker was attached to the participants' hands for motion-tracking. After the experiment, we detected hand trajectories of the participants using image processing techniques. Then, we estimated the velocities of the gestures given the frame rate of the video. We summarize our findings as below:

Finding 1: As shown in Table 2, three types of gesture trajectories were seen during the experiments. Among the three gesture types, a majority of the participants (10 out of 12) predominantly utilized a vertically downward pointing movement. To further analyze the Doppler frequency shifts, we plotted the spectrogram of each gesture and correlated the spectrogram with the recorded video. As shown in Figure 11(a) and 11(b), we found that the arm trajectory variance of the participants who performed this gesture was limited – in most cases, the velocity of users' arms increased during the process of stretching out the users' arms (i.e. Frame #30), and the peak velocities predominantly occurred close to the end of this process (i.e. Frame #31). Then the arm reached a full stretch, and the velocity diminished quickly. We found most of the participants fully stretched out their arms in the experiments, which translates to a 55cm to 75cm phone displacement and angular gains, as discussed in Section 3.2.2. We will show how this factor affects the performance of Spartacus in Section 5.3.1. Since gesture trajectories can be easily differentiated using built-in inertial sensors of the mobile devices, as a proof of concept, we focus on evaluating this vertically downward gesture trajectory in the current design of Spartacus.

Finding 2: To investigate how often the target device



(a)



(b)

Figure 11: The graphs show (a) a vertically downward pointing gesture performed by one student and (b) the corresponding Doppler frequency shifts of each frame. The yellow line indicates the direction of the target devices.

Table 2: Trajectory variations of gestures

Gesture Type	Total Count (Male/Female)
Vertically downward	10 (6/4)
Vertically upward	1 (0/1)
Horizontally outward	1 (0/1)

could successfully observe the maximum velocity of the gestures, we computed directions of the highest 70% velocities of the gestures, and calculated the angular deviation from the direction towards the target device. As shown in Figure 12(b), we found that in most cases, the highest velocities were predominantly facing towards the target device, with an average $\pm 7.5^\circ$ angular bias. When interacting with a device 3 meters away, this bias translates to 0.38m. This result indicates that participants are able to precisely point the phones towards the target device, and selecting the target device using the maximum velocity is appropriate.

Finding 3: The peak velocity of the gestures of all participants was 3.4m/s on average, as shown in Figure 12(a). This is similar to results reported in related work [4]. Using an audio tone at 20KHz and assuming the speed of sound is 343m/s, this translates to a maximum Doppler frequency shift of 200Hz. Moreover, we found that the peak velocity is transient. As shown in Figure 12(c), most of the gestures lasted less than one second, and the peak velocities appeared and diminished within 25ms. These findings indicate that to accurately estimate the peak velocity, Spartacus needs a high time-domain resolution to position the peak frequency shifts, as well as a high frequency-domain resolution to estimate the quantity of the shift.

After understanding the variances of the pointing gestures, we evaluate the performance of Spartacus under different indoor scenarios.

5.2 Experimental Setup

The first set of experiments was done in a student lounge in our university, in which multiple ambient noise conditions were tested. The other two sets of experiments were done in a student cubicle area and along a hallway. During each experiment, a target device was placed in front of a student with distance d; another device was placed at α° apart from the direction of the target device, with the same distance. For each test, a student pointed a Galaxy Nexus phone 25 times towards the target device, with a peak velocity of about 3m/s. To guarantee each individual gesture was not made too fast or too slow, after the experiments, we manually investigated the duration of each gesture, and selected 20 from the 25 gestures that achieved the closest peak velocity for analysis. To compute the frequency shift, audio tones were captured at the two candidate devices at 44.1KHz, undersampled 7 times to 6.3KHz, and then processed using the signal processing pipeline discussed in Section 3.2. A 2048-point FFT was applied for each 10ms analysis window, with a 75% overlapping ratio. These parameters were used throughout the entire evaluation process.

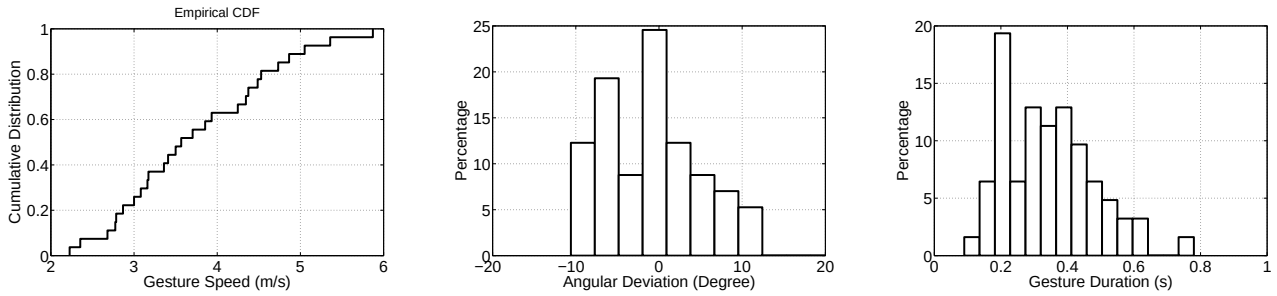
5.3 Device Selection Accuracy

5.3.1 *Performance with Distances and Angles*

Spartacus selects the target device by comparing the detected peak velocity of user gestures among nearby devices. As shown in Figure 15, as the distances between devices increase, the device selection accuracy drops gradually. This is consistent with our analysis of sound energy degradation. As described in Section 3.2, Since Spartacus leverages an energy comparison method for tone detections, the energy difference between tones and other frequency bands decreases as the distances increase. Moreover, due to the performance gain caused by the stretch of arms, for all tested device directions, Spartacus achieved 90% accuracy within 4 meters, and 80% within 5 meters.

To evaluate how directional changes affect the performance, we keep the devices at fixed distances, and change α. As shown in Figure 14, as α decreases, the accuracy of device selection drops. For interactions within 3 meters, the device selection accuracies are above 90% when devices are at least 30° apart. When the devices are at least 45° apart, the accuracies are above 90% within 5 meters.

It is worthing noting that, although the theoretical angular resolution of Spartacus is 10°, we still achieved high device selection accuracy (i.e. above 90%) when the directional difference between devices are lower than 20° and the interaction range is within 1 meters. This is because when the interaction range is short, the angular gain due to the stretch of arms is significant. These evaluation results show that the performance of Spartacus is robust against directional and distance changes for interactions in close proximity. However, as distances keep increasing (i.e. above 5 meters), the performance drops gradually due to the decrease of sound energy and the resultant difficulty in detecting the audio tones. This observation justifies the close-range interaction scenarios of Spartacus, where various ubiquitous applications take place.



(a) (b) (c)

Figure 12: Characteristics of pointing gestures. (a) Speed distribution. (b) Directional precision. (c) Duration variance.

Figure 13: The three different indoor scenarios where we conducted experiments.

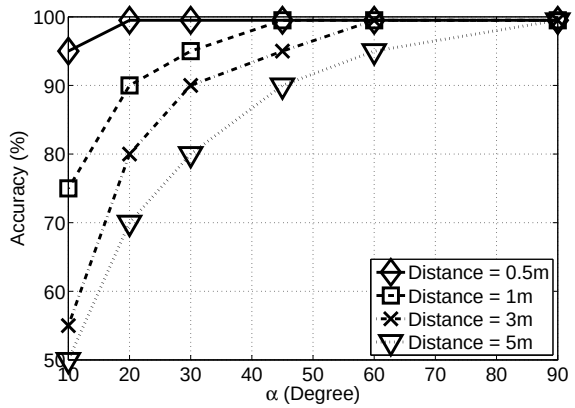


Figure 14: Performance of device selection when device directions change.

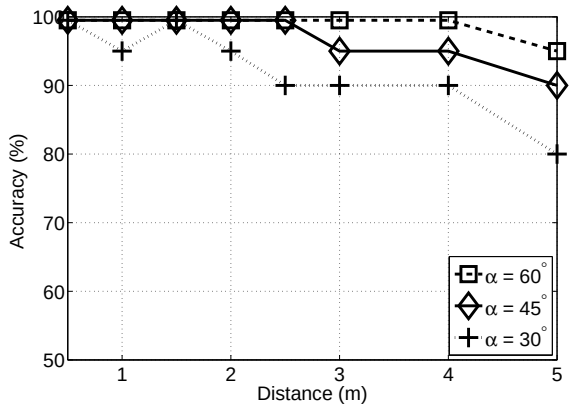


Figure 15: Performance of device selection when device distances change.

5.3.2 *Performance Under Noisy Conditions*

To evaluate the performance of Spartacus against different noise backgrounds, we conducted experiments in the student lounge at different times. The first experiment was done when a couple of students were having a group discussion in the lounge, resulting human conversations in the background. Previous research has shown that the sound of clanging metals would generate high frequency noise [8]. To evaluate this effect, we purposely played a piece of rock music (i.e. "Burn It Down" of Linkin Park) in the background, which features rich clangs of metal instruments. In both experiments, we changed the distance from 1m to 2m. The experimental results of the lounge are used as a comparison. As shown in Figure 17, the experimental results show that under 1m distance, the performance for all three cases for all tested directions are constant, with higher than 95% accuracy. When the distance is increased to 2m, except the 30° case under human conversations, all the three cases achieved above 90% accuracy. Audio spectrum indicates that, as shown in Figure 16, metal clangs can hardly reach frequencies above 18KHz, which has limited effect to Spartacus. These experimental results indicate that the performance of Spartacus is robust against common indoor noises.

5.3.3 *Performance with Different Scenario*

To evaluate the performance under different indoor scenarios, we conducted experiments in cubicle areas and a hallway, as shown in Figure 13. Due to limited space in these scenarios, we only tested performance up to 1.5m with 30°. Figure 18 shows the results. For the distance of 0.5m, all three cases had 100% device selection accuracies. As the distance increases, the performance of the cubicle area and the hallway is seen to have slight decreases. This is primarily

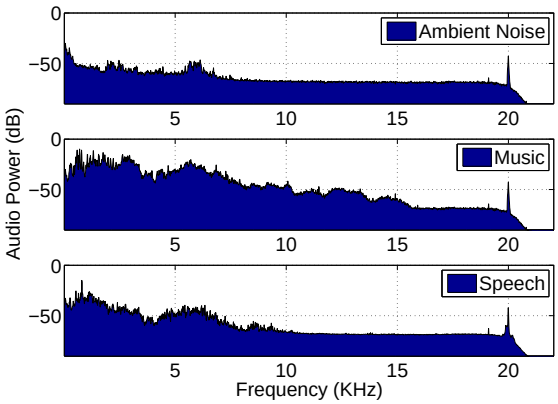


Figure 16: Spectrums of the three noise cases measured at 1m. Note that, in all cases, the audio tone at 20KHz (i.e. the spikes on the spectrums) for the pointing gestures is easily detectable.

due to the stronger multi-path effects in the two scenarios as compared to the student lounge. However, in all three cases, Spartacus has achieved higher than 85% accuracy.

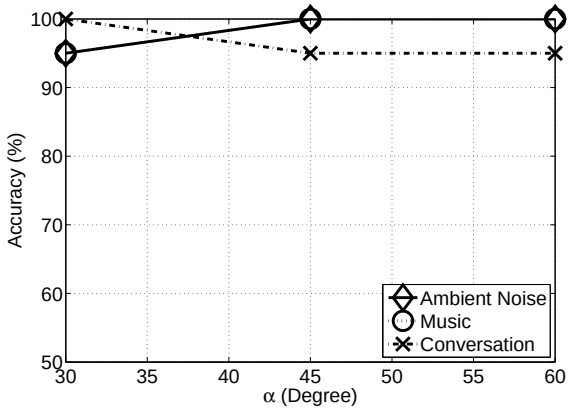
5.4 Interaction Latency

Using the undersampling technique, Spartacus reduces the number of audio samples that need to process, without decreasing the time-domain resolution. As discussed in Section 3.1, if undersampling technique was not used, Spartacus would have to increase the number of FFT points to achieve the equivalent angular resolution, which as a consequence, involves longer processing time. To evaluate this performance, we tested the processing latency of Spartacus using different FFT points. As shown in Figure 19, Spartacus leverages a 2014-point FFT processing, which takes 1.5s to process a 1-second gesture audio. To achieve the same angular resolution, a traditional FFT approach would have to use at least an 8192-point FFT processing, which takes 8.7s! Such a processing latency is impractical for any responsive interactions, such as mobile gaming. These results justify the use of undersampling in Spartacus.

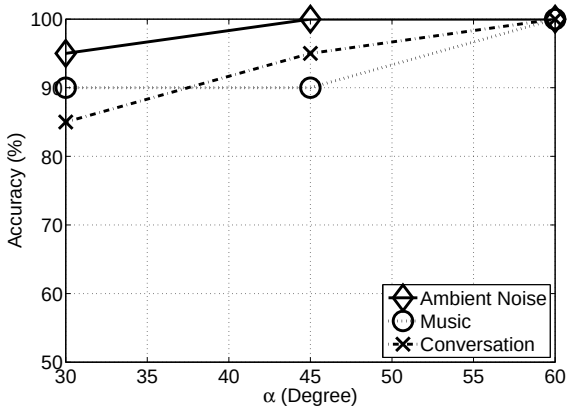
5.5 Power Consumption

Spartacus leverages a low-power audio listening technique to automatically sense the potential interaction requests from nearby devices, so as to eliminate any user involvement. In this section, we evaluate the performance of Spartacus in terms of power consumption in low-power listening. To compare the performance under different duty cycles, we fixed the duration of each listening session to 200ms, and changed the periods. To eliminate hardware variations, all the experiments were done on the Galaxy Nexus mobile phones. In the experiment, the screens and the CPUs of the mobile phones were completed shut off, and the power consumption was measured using an oscilloscope. For each test, we measured the power consumption by running Spartacus's low-power listening task for 5min, and we compute the mean and the standard deviation of the collected data.

The experimental results are shown in Figure 20. As a baseline, we first measure the power consumption of the devices being idle, i.e. turning off the screen and shutting down all the application processes. This yields the base-



(a) Distance = 1m



(b) Distance = 2m

Figure 17: Performance of device selection with various background noise conditions.

line power consumption at 75mW. Then we evaluate the performance of Spartacus. When the duty cycle is 5% (i.e. the period equals 4s), Spartacus consumes 120mW power. As the duty cycle increases, the power consumption goes up gradually. At the maximum 25% duty cycle, Spartacus consumes 250mW. As a comparison, we tested the state-of-the-art peer-to-peer scanning techniques, and found that WiFi Direct consumes 460mW of power, and Bluetooth 4.0 consumes 670mW. These results indicate that on average, Spartacus achieves about 4X lower energy consumption than WiFi Direct and 5.5X lower than the latest Bluetooth 4.0 protocols, respectively.

6. RELATED WORK

Spartacus leverage built-in microphones and speakers on commodity mobile devices for both active device interaction and passive listening for potential interaction requests. In this section, we compare Spartacus with previous work that leverage audio signals in two main categories: mobile sensing and device interaction.

6.1 Audio Processing in Mobile Sensing

Microphones have been widely used in mobile sensing applications to capture audio data. For example, Miluzzo et al. has used human conversation snippets for analyzing

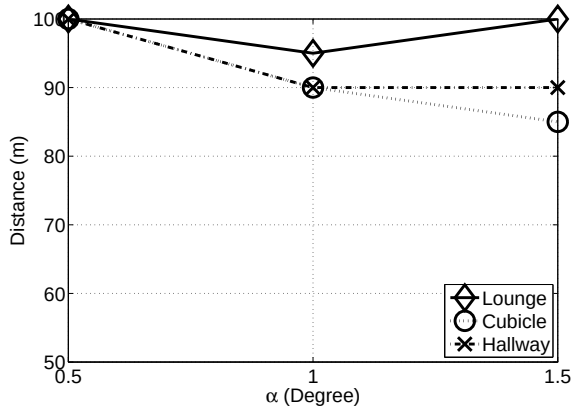


Figure 18: Performance of device selection in different indoor scenarios. α is fixed at 30°.

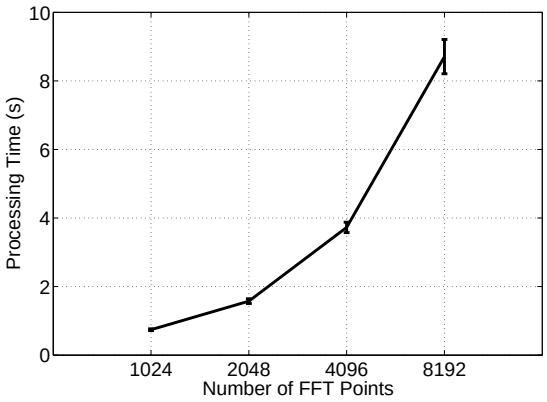


Figure 19: The processing time of a 1-second audio recording using different number of FFT points.

social activities [12]. SurroundSense also utilized audio data, combined with data from other sensing modalities, such as accelerometers, cameras, and magnetometers to detect locations of users for social context inferences [1]. Similarly, Lu et al. has provided a tailored signal processing pipeline for audio sensing and learning, such that unknown social events can be automatically identified and easily labeled [9]. These early projects have largely focused on the knowledge discovery tasks of using audio data, while assuming a fixed or default audio sensing mechanism.

For energy-efficient continuous audio sensing, some more recent work has been published. JigSaw and Darwin Phones focused on enabling energy-efficient continuous sensing and collaborative learning techniques [10, 11]. MoVi presented an approach of using collaboratively sensed audio data from multiple participants to create integrated social event records [2]. SwordFight provide a continuous and accurate distance ranging technique using time difference of sound arrivals [22].

All the previous work mentioned above leveraged microphones in a continuous manner. Spartacus differs from them by proposing a low-power audio listening technique for passive interaction sensing. Since interaction requests from nearby devices can happen spontaneously, it requires Spartacus to coordinate the tradeoff between detection latency and

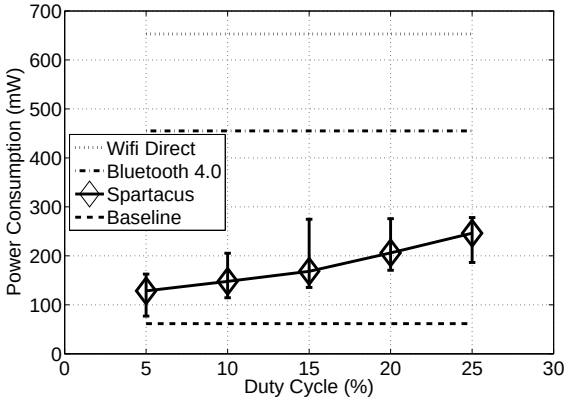


Figure 20: Power consumption of Spartacus under different duty cycle settings. As references, power consumptions of scanning of Bluetooth 4.0 and WiFi Direct are also shown.

energy efficiency. Other papers have also presented systems using audio tones [7] for indoor counting or localization [8]. Differing from them, Spartacus leverages tones at high frequency to generate Doppler effect for device interaction.

6.2 Spatially-Aware Device Interactions

Interactions through mobile devices generally require two pieces of functionalities, i.e. device selection and interaction detection. For selecting devices, previous work has shown that touching, scanning, and pointing are the most common interactions when performing user-mediated object selection and indirect remote controls [17]. Point & Connect (P&C) proposed an interaction technique based on time difference of sound arrivals. Both Spartacus and P&C can be used for device interactions in close proximity. However, the two systems differ significantly in several aspects. First, P&C requires users to manually set up a dedicated broadcast WiFi channel or start discoverable mode of their Bluetooth. Enabling P&C may prevent the users from using their default WiFi networks. In contrast, Spartacus runs in the mobile devices as a system service, and the periodic interaction detection mechanism automatically detects potential interactions. Second, P&C was focused on providing the device selection solution, while making an assumption that the surrounding devices have already launched the related service and continuously waiting for interaction requests. In practice, this continuous audio listening would consume significant energy. Therefore, Spartacus uses the duty-cycled audio listening mechanism, and our evaluation results have shown this mechanism is energy-efficient, as compared to the traditional peer-to-peer communication techniques, such as WiFi Direct and Bluetooth 4.0.

SoundWave leverages the Doppler effect to sense user gestures for close-range (i.e. within 1m) single-device interactions [4]. As the Doppler effect is made by moving user arms in front of a laptop, and the laptop is both the transmitter and the receiver, the generated frequency shift is doubled. This significantly increases the detection accuracy of the frequency shifts and makes detection easier. In contrast, Spartacus selects the target device by comparing the peak frequency shifts between multiple devices, which requires higher frequency- and time-domain resolution. We solve

these problems by proposing a bandpass signal processing pipeline to increase Doppler shift estimation accuracy.

To detect nearby devices and interaction requests, various techniques can be used. However, existing techniques either required extra infrastructure (i.e. laser tags and pointers [13]), or did not provide spatial accuracy sufficient for supporting device interactions (i.e. WiFi-based indoor localization [21]). Other work involved extra effort from users to initiate interactions (e.g. the simultaneous shaking required by the Bump app), or is not energy efficient (e.g. WiFi or Bluetooth techniques). PANDAA provided an automatic device locationing service using ambient sound generated in indoor environments, without requirements of extra infrastructures or extra manual effort, and has achieved centimeter-level locationing accuracy [18]. However, PANDAA only supports devices in stationary placements, thus if moving users are involved in interactions, other mechanisms would have to be used. More recent work in Polaris provided an indoor orientation determination technique that could support spatially-aware indoor device interactions [19]. However, since Polaris dealt with only absolute directional relationships of devices (i.e. relative to the earth's magnetic field), nearby devices can hardly be differentiated if their relative orientations change, such as in the interaction scenarios that involve moving users. In this paper, we describe the low-power audio listening technique used in Spartacus to detect nearby interaction requests, which periodically wakes up the CPU and microphones on mobile devices, and detects audio beacons emitted by nearby devices. We also present experimental results that show the low-power audio listening technique is much more energy-efficient than other state-of-the-art techniques, such as WiFi Direct or Bluetooth 4.0.

7. DISCUSSION

In the current stage, the Spartacus system focuses mainly on the signal processing techniques and prove the novel interaction concept through the use of the Doppler effect. This section addresses a number of related system issues and discusses their possible solutions.

7.1 Energy-Efficient Interaction Triggers

To support automatic interaction triggers without user involvement, Spartacus leverages energy-efficient audio sensing to automatically detect interaction intents from nearby devices. This makes the system more appropriate to be used on mobile devices with limited energy budgets. While the current system adopts the continuous low-power audio listening technique, we would note that, if the energy constraint is not a major concern in particular applications, the system could also be enabled on demand. Moreover, in addition to the proposed energy-efficient audio sensing technique, the system could also be triggered by other traditional communication schemes, such as Bluetooth or WiFi Direct.

When choosing between different interaction triggering mechanisms, there are a few design tradeoffs. On one hand, as we have pointed out in Section 5.5, the proposed low-power audio listening technique is more energy-efficient than the Bluetooth 4.0 and the WiFi Direct techniques. On the other hand, due to the intrinsic slow data rate of audio signals, the user has to wait for a couple of seconds for a "warmup beacon" before doing the gesture. As a comparison, this issue can be solved, given that the relatively faster wireless communication schemes could be used, such as the Bluetooth and the WiFi techniques.

7.2 Security Issues

Spartacus leverages the Doppler effect to support spatially-aware device interactions. This interaction may be vulnerable to security attacks. For example, when a user generates a pointing gesture, a malicious device standing close by could pretend to have detected higher Doppler shifts than other devices, so that it deceives the sender into thinking it was the receiver; similarly, the same trick could also be used to prevent other receivers from being connected with the sender. To avoid these issues, a secured connection mechanism could be used, so that only trusted and authenticated devices are allowed to report their Doppler shifts. Alternatively, knowledge from the users could also be used to reduce the possibility of malicious recipients. For example, after the user's device determines the potential receiver who has reported the maximal Doppler shifts, the name and identity of receiver's owner would be shown on the user's device. Then the user would be able to determine the correctness of the interaction recipient.

7.3 Contentions Among Interaction Sessions

When the Spartacus system is used in a crowded scenario, e.g. an airport, where many users would use the system concurrently, contentions could be an issue for device pairing techniques. However, since Spartacus leverages an audio sensing mechanism for close-range interactions, concurrent interactions can be automatically separated if they are far away from each other (i.e. beyond 5m), due to the fast degradation of sound energy. If the interactions do occur with close proximity, a contention coordination mechanism (e.g. a contention window that receives the Doppler shift measures) could also be leveraged to solve the problem.

8. CONCLUSION

This paper presents Spartacus, a spatially-aware interaction system that allows users of mobile devices to establish spontaneous interactions with high accuracy, low latency, and low energy consumption. Spartacus does not require extra hardware. Leveraging sensors ubiquitous in most commodity smartphones, Spartacus enables users to use intuitive pointing gestures to select target devices with zero prior configuration. We provide a comprehensive evaluation of the Spartacus system in various use conditions. Our experimental evaluations show that Spartacus performs significantly better than existing device interaction systems in terms of intuitiveness, accuracy, latency, and energy consumption. This new paradigm of mobile interactions will enable natural, fast, and seamless interactions in numerous emerging applications.

9. ACKNOWLEDGEMENT

This work was partially the results of the generous support from Intel Inc., and National Science Foundation grant number CNS-1135874 and 1149611. In addition, the authors would like to thank our shepherd Prof. Shyam Gollakota for providing valuable guidance during the process of finalizing the camera-ready version of the paper. Moreover, the authors would also like to thank Prof. Ian Lane of CMU for clarifying audio-related concepts and for lending us the reference microphones and speakers for doing the experiments.

10. REFERENCES

[1] M. Azizyan, I. Constandache, and R. Roy Choudhury. SurroundSense: mobile phone localization via ambience fingerprinting. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, pages 261–272, New York, NY, USA, 2009. ACM.

[2] X. Bao and R. Roy Choudhury. MoVi: mobile phone based video highlights via collaborative sensing. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, pages 357–370, New York, NY, USA, 2010. ACM.

[3] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*, page 10, 2006.

[4] S. Gupta, D. Morris, S. Patel, and D. Tan. SoundWave: using the doppler effect to sense gestures. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 1911–1914, New York, NY, USA, 2012. ACM.

[5] H. Harada and P. Ramjee. *Simulation and Software Radio for Mobile Communications*. Boston: Artech House, 2002.

[6] K. Hinckley. Synchronous gestures for multiple persons and computers. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, UIST '03, pages 149–158, New York, NY, USA, 2003. ACM.

[7] P. G. Kannan, S. Padmanabha, M. C. Chan, A. Ananda, and L.-S. Peh. Low Cost Crowd Counting using Audio Tones. In *The 10th ACM Conference on Embedded Networked Sensor Systems (SenSys 2012)*, pages 155–168, 2012.

[8] P. Lazik and A. Rowe. Indoor Pseudo-ranging of Mobile Devices using Ultrasonic Chirps. In *The 10th ACM Conference on Embedded Networked Sensor Systems (SenSys 2012)*, pages 99–112, 2012.

[9] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell. SoundSense: scalable sound sensing for people-centric applications on mobile phones. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, MobiSys '09, pages 165–178, New York, NY, USA, 2009. ACM.

[10] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell. The Jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 71–84, New York, NY, USA, 2010. ACM.

[11] E. Miluzzo, C. T. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu, and A. T. Campbell. Darwin phones: the evolution of sensing and inference on mobile phones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, pages 5–20, New York, NY, USA, 2010. ACM.

[12] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the CenceMe application. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, SenSys '08, pages 337–350, New York, NY, USA, 2008. ACM.

[13] B. A. Myers, R. Bhatnagar, J. Nichols, C. H. Peck, D. Kong, R. Miller, and A. C. Long. Interacting at a distance: measuring the performance of laser pointers and other devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 33–40, New York, NY, USA, 2002. ACM.

[14] A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing*. Prentice Hall.

[15] C. Peng, G. Shen, Y. Zhang, and S. Lu. Point & Connect: intention-based device pairing for mobile phone users. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, MobiSys '09, pages 137–150, New York, NY, USA, 2009. ACM.

[16] A. Purohit, B. Priyantha, and J. Liu. WiFlock: Collaborative group discovery and maintenance in mobile sensor networks. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 37–48, Apr. 2011.

[17] E. Rukzio, K. Leichtenstern, V. Callaghan, P. Holleis, A. Schmidt, and J. Chin. An experimental comparison of physical mobile interaction techniques: touching, pointing and scanning. In *Proceedings of the 8th international conference on Ubiquitous Computing*, UbiComp'06, pages 87–104, Berlin, Heidelberg, 2006. Springer-Verlag.

[18] Z. Sun, A. Purohit, K. Chen, S. Pan, T. Pering, and P. Zhang. PANDAA: physical arrangement detection of networked devices through ambient-sound awareness. In *Proceedings of the 13th international conference on Ubiquitous computing*, UbiComp '11, pages 425–434, New York, NY, USA, 2011. ACM.

[19] Z. Sun, A. Purohit, S. Pan, F. Mokaya, R. Bose, and P. Zhang. Polaris: getting accurate indoor orientations for mobile devices using ubiquitous visual patterns on ceilings. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, HotMobile '12, pages 14:1—-14:6, New York, NY, USA, 2012. ACM.

[20] P. Vandewalle, L. Sbaiz, J. Vandewalle, and M. Vetterli. How to take advantage of aliasing in bandlimited signals. *IEEE Conference on Acous- tics, Speech and Signal Processing*, 3:948–951.

[21] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 197–210, New York, NY, USA, 2012. ACM.

[22] Z. Zhang, D. Chu, X. Chen, and T. Moscibroda. SwordFight: enabling a new class of phone-to-phone action games on commodity phones. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, MobiSys '12, pages 1–14, New York, NY, USA, 2012. ACM.