PhoneEar: Interactions for Mobile Devices that Hear High-Frequency Sound-Encoded Data

Aditya Shekhar Nittala[1], Xing-Dong Yang[2], Scott Bateman[3], Ehud Sharlin[1], Saul Greenberg[1]
[1]Department of Computer Science, University Of Calgary, Calgary, AB, Canada
[2] Department of Computer Science, Dartmouth College, Hanover, NH, USA
[3] Faculty of Computer Science, University Of New Brunswick, Fredericton, NB, Canada
{anittala,saul}@ucalgary.ca, ehud@cpsc.ucalgary.ca,
xing-dong.yang@dartmouth.edu, scott.bateman@unb.ca

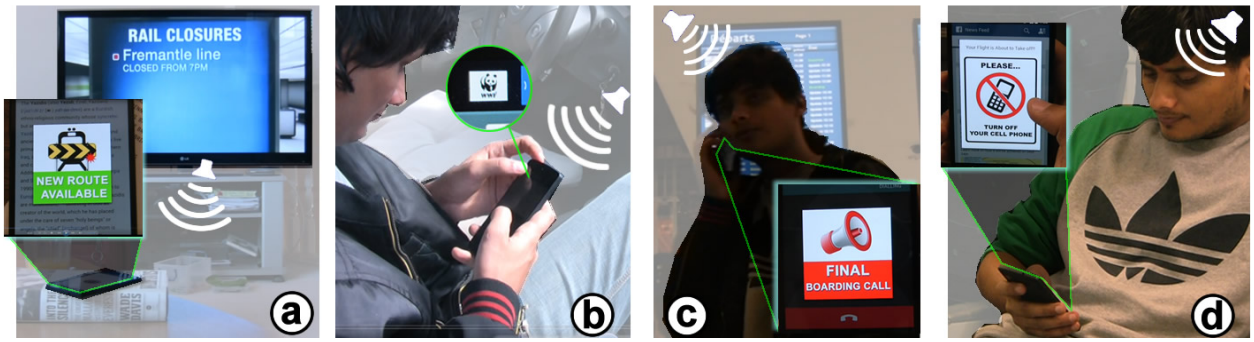

Figure 1: PhoneEar interaction techniques: (a) responding to traffic updates from a TV and suggesting a new route; (b) showing received charity advertisement on the status bar; (c) notifying a final boarding call; (d) suggesting powering off before take-off.

ABSTRACT
We present *PhoneEar*, a new approach that enables mobile devices to understand the broadcasted audio and sounds that we hear every day using existing infrastructure. PhoneEar audio streams are embedded with sound-encoded data using nearly inaudible high frequencies. Mobile devices then listen for messages in the sounds around us, taking actions to ensure we don't miss any important info. In this paper, we detail our implementation of PhoneEar, describe a study demonstrating that mobile devices can effectively receive sound-based data, and describe the results of a user study that shows that embedding data in sounds is not detrimental to sound quality. We also exemplify the space of new interactions, through four PhoneEar-enabled applications. Finally, we discuss the challenges to deploying apps that can hear and react to data in the sounds around us.

Author Keywords
Mobile interactions; Sound-encoded data.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

EICS'15, June 23 - 26, 2015, Duisburg, Germany
 ACM 978-1-4503-3646-8/15/06…$15.00
DOI: http://dx.doi.org/10.1145/2774225.2775082

INTRODUCTION
People frequently listen to the radio, TV or other digital media to receive important information that they can use in their daily lives. For example, a news broadcast might inform us of repairs to a rail line; suggesting that new travel plans are needed. In public places, sounds and audio messages are often used to make us aware of what is happening close by (e.g., that a flight has changed gates). While such information is meant for us to hear it when we need it, many times audio information can be missed for a variety of reasons: we are distracted, there is ambient noise in the environment, or we have moved away from the sound source; among other reasons. Because the information in sound streams is most often ephemeral (with no good way to hear or review information that has already been broadcasted), we have to be ready and able to hear what we need when it is presented. Further, audio information must be remembered or explicitly recorded to be useful (e.g., What gate has my flight been changed to? Which rail stations are closed?).

In this paper, we present a range of new and working interactions that are made possible when mobile devices are able to *hear* the information in the broadcasted audio streams around us. When mobile devices (including smart watches, phones and tablets) have the ability to hear for us, it ensures that we don't miss important information. It also allows our mobile devices to augment audio information in new ways. For example, an app that understands an audio message could: proactively make recommendations and link to info related to a news story; vibrate and flash to alert

us of a critical message that we didn't hear; or, monitor our activities to present information at an appropriate time.

To demonstrate these novel mobile interactions, we developed PhoneEar, a system that augments audio streams with data, which is hidden in high-frequency sounds. High-frequency sounds are largely inaudible to people, but can be broadcast and captured effectively using conventional speakers and microphones. This means that our existing mobile devices could potentially be capturing data that describes and augments the audio messages that we listen to and that are played around us every day.

PhoneEar is a two-part solution. First, conventional audio streams are augmented with a high-frequency data channel. Second, apps are installed on mobile devices that run in the background and listen for information in the high-frequency data channel. Apps then take appropriate actions based on the content and type of data message received.

Our work makes three main contributions:

1. Four working example applications that demonstrate the new space of mobile interactions opened up when mobiles devices can understand data embedded in broadcasted sounds.
2. Two studies that show that high frequency, sound-encoded data is an effective method for transmitting information to mobile devices in realistic conditions.
3. A detailed outline of our approach, so that others can build mobile apps that understand sound-encoded data.

In this paper, we describe our implementation of PhoneEar, which allows any data message to be encoded and broadcasted with the typical sounds, audio messages and music broadcasted around us every day. PhoneEar-enabled apps allow messages to be captured by conventional mobile devices. Through a system evaluation and a user study, our work is the first to show that embedding data can be an effective and reliable way for mobile devices to capture audio information in realistic conditions. We also discuss the key challenges we overcame during the implementation of PhoneEar, and identify potential hurdles to its real world use that we will investigate in our continuing research.

RELATED WORK

Implicit Interactions

Many of today's mobile interactions are explicit; where users need to explicitly take a number of actions to indicate intentions or desires to a system (e.g., for the Shazam app [15] to be able to identify a song that is being played, a user must explicitly activate the system). In contrast, Buxton's concept of "background interactions" [1] has recently inspired a number of interaction techniques that work implicitly [2, 8, 17], without explicit user input. Our work was inspired by the concept of background interactions and systems like SurroundSee [17], which is a mobile phone that can see and react to the events happening in its surroundings. Extend this idea to explore what is possible when mobile devices can seemingly understand the audio information around them.

Sound-based Interactions

Audio signal processing technologies are increasingly being used for interactions; e.g., in voice recognition [5] or music identification [15]. Audio processing is also being used to allow mobile devices to understand a user's context. Lu et al. [10] created a mobile app that can distinguish between ambient music and human voice thus can be partially aware of users' activities. Madhavapeddy et al. [11] explored the use of ubiquitous microphones and speakers as a low-bandwidth wireless network for explicit interactions such as file transfer, indoor localization, and mobile authentication. Google's Tone is an experimental sound-based extension used to simplify URL transfers between nearby mobile devices, by encoding and broadcasting a URL as a short sequence of beeps. Nearby machines can hear and decode the beeps to allow a user to open the URL, see: http://www.webcitation.org/6YfmC3WJx. While previous work has considered the use of sound as input for mobile interactions, the new types of interactions that might be possible has not been fully explored.

"Understanding" Sound

Many successful sound-based interactions (like the ones described above) make use of machine learning [5,7,10,15] or template matching techniques [16]. This means that they are based on a-priori assumptions of what type of sound is being heard (e.g., music, voice, etc.) and a closed set of possible items that can be identified (e.g., pre-processed TV and songs, or certain pre-determined phrases). While these methods are technically robust within certain application domains, this means that mobile devices can only react to a limited set of potential inputs and conditions.

One approach that does not have any limitations in terms of the number of inputs is encoding data into sound by modulating sound wave properties— such as frequency, amplitude, or phase [3,4,11,13]. Both audible and nearly inaudible frequencies have been shown to be effective channels for data transmission using this technique [3,6,9]. This means that no pre-existing knowledge of a sound is required, and any arbitrary data can be encoded into a sound. While previous work focused on near-field data transmission in quiet, controlled environments, we assessed performance and feasibility in a more realistic setting by embedding sound-encoded data into existing sound tracks.

THE PHONEEAR APPROACH

In this section, we describe the two general steps required for PhoneEar-based interaction techniques: (1) acquiring information embedded in sounds that are in the proximity of a mobile device; and (2) triggering personalized actions based on information separated from other sounds.

Step 1 – Acquiring Ambient Information from Digital Sound Sources

PhoneEar listens to the audio being transmitted from TV, radio programs, public announcements, online videos or

any other audio source, and looks for high-frequency sound that might contain data. If sounds exist within a high-frequency range, PhoneEar attempts to decode data from the sound. If data is successfully decoded, it is parsed for metadata that describes its purpose, and the content of the data message is routed to an appropriate app to trigger an action. Unlike existing methods that require databases of potential sound classification samples (e.g., [16]), our approach is simple and flexible. For example, content data might include the TV channel name, a program title, a song title, the artist, a weather forecast, etc. And, it would be up to a particular PhoneEar app to make use of the information to inform or to interact with user. Additionally, content can include supplemental information (e.g., a web link).

Step 2 – Triggering Actions: The Design Space of Interactions Enabled by PhoneEar

Once data has been captured, a PhoneEar-based system can proactively take actions on behalf of a user, present shortcuts for likely or potentially relevant actions, or simply notify the user about the captured content data. Through our work with PhoneEar—brainstorming potential interactions, and developing example interactions—we have identified five types of interaction techniques that PhoneEar enables: Context-Based Recommendation, Enhanced Directions, Enhanced Notifications, the Display of Complementary Information and Learning User Preferences.

THE SPACE OF PHONEEAR INTERACTIONS

In this section, we describe and illustrate each type of interaction through a prototype system we have successfully built using PhoneEar.

Context-Based Recommendation. People use TV and radio news about public events, traffic conditions or the weather to make plans. For example, people may change their travel plans when a radio broadcast informs them of traffic congestion. PhoneEar can capture these events and provide ad-hoc recommendations for users. In our implementation, PhoneEar allows a smartphone to respond to traffic updates from TV and radio audio streams, and suggests alternative routes based on a user's current travel plan: a notification on the screen of the phone provides a recommendation, clicking it provides alternative routes (Figure 1a).

Enhanced Notifications. In public spaces, audio announcements are commonly used to broadcast important messages, which can be missed by people whose attention may be focused elsewhere. For example, at an airport, a user may miss their final boarding call because they are talking on the phone. With PhoneEar, their phone could capture the message sent with the boarding announcement and interrupt the user's call to notify them that their flight is about to depart (Figure 1c). This example highlights flexibility over other technologies. For example, using communication channels such as Wi-Fi or Bluetooth could require new infrastructure (wireless routers or Bluetooth beacons) prior to using the service. But, PhoneEar is meant to operate in locations where existing sound broadcasting infrastructure (i.e., speakers) are already available. In such locations, PhoneEar would always be listening, so the chance of missing important information is reduced.

Enhanced Directions. In public spaces, audio announcements are also commonly used to request important actions from people. For example, before planes takeoff and touchdown, an audio announcement requests that electronic devices be turned off or switched to flight mode. However, it is difficult for airlines to enforce, and difficult for passengers to know exactly when they must comply. With PhoneEar, as the passengers receive an action request from a recording, so do their phones. In our implementation, the phone displays a message explaining that the phone should be set in 'airplane' mode or turned off (Figure 1d). Enhanced directions might optionally trigger the required action on the device directly (i.e., the phone is automatically turned off).

Display of Complementary Information and Learning User Preferences. Shazam [15] is a phone app that allows complimentary information to be accessed based on a song that is being played. Once explicitly enabled, Shazam provides information about the song, artist, album, etc. PhoneEar doesn't require explicit interaction for a similar interaction, because it listens to the songs user listens to in the background. Additionally, PhoneEar does not rely on a pre-existing database of songs. This allows it to recognize live sports or breakout news that are not possible with Shazam. We implemented a simple app where the status bar of a phone automatically displays the content of the data messages it receives (Figure 1b). Users can then selectively tap the notification to get more details. Such functionality would also allow PhoneEar to create models of user preferences that could be used in personalizing services. For example, only provide me information about artists with whom I am not already familiar.

SYSTEM IMPLEMENTATION

In developing PhoneEar, we had three major requirements. First, the quality of the original sound should be largely preserved. Second, PhoneEar should work on existing hardware infrastructure (e.g., the off-the-shelf speakers and microphones in mobile devices). Third, PhoneEar should

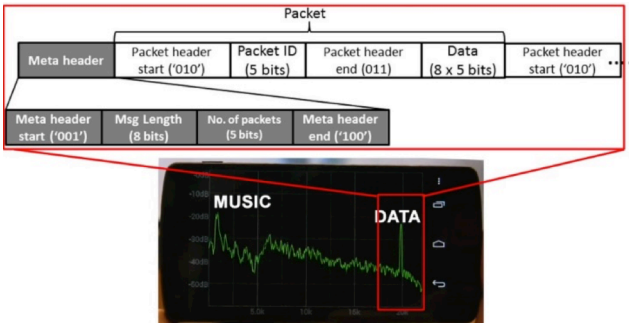

Figure 2: PhoneEar message template (the Hamming code is not shown in the diagram).

work in realistic conditions; noisy environments with various operating distances and device placements.

To preserve sound quality and to support existing hardware, we implemented PhoneEar using unused and nearly inaudible frequencies from 17 kHz to 20 kHz. This range of frequencies can isolate data from major noises introduced by music and human speech. To achieve sufficient operating distance, we ensure adequate wave energy and signal-to-noise ratio (SNR). The high frequencies we chose are hardly audible by adults [12], especially when the data is mixed with music or other noise. As we will show later, these frequencies are within the capabilities of smartphone microphones and off-the-shelf speakers. Below we describe our implementation in more detail.

Our PhoneEar prototype consists of an encoder and a decoder. The encoder can be integrated in the infrastructure emitting the PhoneEar messages, where messages can be composed on the fly (e.g., sounds representing data can automatically be played by a PA system when an attendant announces a flight gate change). Alternatively, the encoder can be part of the offline process of preparing messages ahead of the time (e.g., a sound engineer or radio DJ could incorporate data into a song). The decoder, an app that runs on a mobile device, listens to the sounds around it and extracts data messages (if any).

Encoding Mechanism: For encoding, we used a simple version of frequency-shift keying (FSK) modulation to encode information by varying the instantaneous frequency of a sound wave [2]. FSK allows us to encode data into the higher end of the frequency spectrum (17–20 kHz; Figure 2) that is less sensitive to human ears and more robust to the interference caused by music and human speech.

Message Encoding: Any ASCII message is translated into a binary stream. The binary stream is encapsulated into packets of 8 ITA2 characters (5 bits/character). A four-bit Hamming code is used to provide forward error correction. If the information is longer than 8 characters, it is separated into multiple packets, each of which has control bits that specify its ID and length (Figure 2).

Transferring Message: Messages can be sent in binary or other forms, e.g. decimal. Increasing the base of the number system allows for more compact message. In our implementation, we used decimal values and mapped them to discrete frequencies ranging from 18–19.8 kHz in 200 kHz intervals. Dedicated frequencies of 17.8 and 20 kHz are added to the message in order to robustly indicate its start and end, respectively.

Message Decoder: The decoder was implemented as an Android application. The app runs in the background of a device and checks for the presence of frequencies assigned to the data scheme (above). A tone is identified for decoding if it appears longer than a certain amount of time (500ms) and its amplitude is greater than a threshold value (30% above the mean amplitude of the frequencies in the

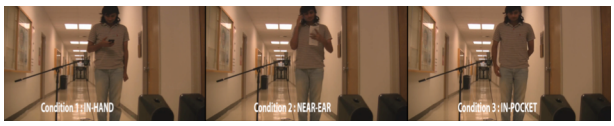

Figure 3: Conditions used in the evaluation (from left to right): IN-HAND – Phone held in front of the user with mic pointed towards the chest. NEAR EAR – Phone held near the ear with mic pointing towards the audio source. IN POCKET – Phone in the user's front trouser pocket.

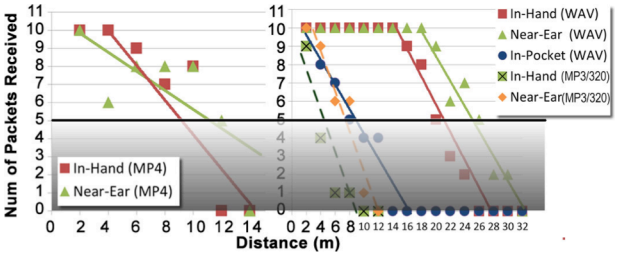

Figure 4: Number of received packets using MP4 video (left), and WAV and MP3 (320kbps) audio formats (right). The black line represents half of the sent packets.

range from 15.8–17 kHz). We choose to use a 500ms pulse to ensure that data can be received from a good distance. Due to the fact that the pulse duration is proportional to wave energy, higher pulse duration introduces higher SNR and higher power (the received power varies inversely with the square of the distance), allowing data to be reliably transferred farther (our results provide a higher operating distance when compared to previous work). Our current implementation allows a speed of 8 bits per second, which is sufficient for sending 1 byte/sec. up to 18 m. Speed can be improved for apps requiring shorter operating distances. Our current implementation is meant to demonstrate the upper bound of PhoneEar's working distance and the corresponding lower bound of data transformation speed.

Once a packet is received, it is converted into binary form. Single-bit errors are corrected using the Hamming code. If more errors occur, the packet is dropped. Since PhoneEar is a simplex system (the encoder only transmits and the decoder only listens), the transmitter continuously plays the message, broadcasting it until the end of a predetermined time. The encoder has no knowledge of whether it was received. The decoder, on the other hand, is aware of a packet being lost, and will then attempt to receive it in the next loop. Lost packets are identified by IDs.

Adding PhoneEar to Other Sounds: Sound streams can be augmented either when the original audio stream is being created, or on the fly. A PhoneEar message can be incorporated into any existing video/audio file using free editing software (e.g., Audacity or Windows Live Movie Maker). It should be noted that mixing data with an existing file might affect the quality of the original audio stream. However, based on our testing, conflicting harmonies (which impact the quality of the original sound) can be avoided by playing the sound data on a single audio channel

or by adjusting the amplitude ratio between the data track and the original sound. This step may not be necessary, depending on the quality of the original sound. Alternatively, sound data can be played as a standalone audio stream.

DATA TRANSFER EVALUATION

To validate the feasibility of our approach for real world use, we conducted an evaluation, where we measured the performance of our implementation in three common mobile phone usage conditions. In the first condition, the phone was held in front of the experimenter's chest in a texting position (IN HAND). In the second condition, the experimenter read a paragraph of text with the phone in a talking position against their ear (NEAR EAR). Finally, in the third condition, the phone was located in the experimenter's trousers pocket (IN POCKET). For each condition, we tested successful data transfer rates from various distances by repeating a data message 10 times. The message we used for the study contained a single character sent 10 times—equivalent to sending a 10-character message once—allowing us to gauge raw performance.

Apparatus and stimulus

The data was embedded into a song file (Kelly Clarkson's "Stronger") using Audacity, and a music video file of the same song using Windows Live Movie Maker. Three audio files (a WAV and 2 MP3 files), and a video file (MP4) were generated for testing. The MP3 and MP4 formats (lossy compressed formats) and WAV (a lossless format) were used to test the performance of the system under different sound quality conditions. The two MP3 files were generated using a compression rate of 128 and 320 kbps respectively. To remove harmonic effects, the data was only played on the left channel for the three audio files, and played on Logitech Gigabyte T40 speakers. Volume ratio between data and sound was adjusted to 140% for all the tested files to ensure sufficient data volume. Finally, the volume of the speakers and the tested files were calibrated in such a way that the audio levels were from 65-72 dB SPL, 2 meters from the speakers (measured by a Nexus 5 phone). The dB level we used in the evaluation represents what is commonly experienced in people's daily lives.

Results

We recorded the number of received packets starting from 2m away from the speakers. Using the lossless WAV file, our system successfully received all packets up to 14m away, and up to 18m with the phone being held in front of the chest (IN HAND) and against the ear (NEAR EAR) respectively. The talking position led to longer distance because in this position, the phone's microphone was pointed towards the sound source. Surprisingly, even when placed in the pocket (IN POCKET), the phone could still receive $\geq$ 70% of the packets from up to 6m from the speakers. The results suggest that the PhoneEar is practically feasible given the interaction techniques we proposed (above). However, audio compression did impact the performance of the system and as expected, data transfers completely failed with the 128 kbps MP3. Using the 320kbps MP3, the phone could receive $\geq$ 60% of the packets when held in front of the chest and against the ear, from up to 2m and 8m respectively. Although Figure 4 (right) shows that data could still be received from 32m, the system suffered significant packet loss. We considered it a failure if more than half of the packets were lost.

Compressing data using MP4 also impacted data transformation. The system completely failed when placing the phone in the pocket. When held in front of the chest or against the ear, the system received $\geq$ 70% of the packets up to 10m away from the speaker. The result can be considered good as 10m is farther than the distance at which people typically watch a video.

SOUND QUALITY EVALUATION

We conducted a second evaluation to determine whether embedding data into sound files would have a noticeable impact on sound quality. In particular, we were interested in knowing whether people could notice a difference in quality when data was embedded in sound.

Apparatus and stimulus

We used the same speakers and song as in the first evaluation. We used a song as our stimulus because unlike human speech, music largely relies on higher frequency sounds, so there is more potential for sound-encoded data to interfere with music.

Participants and experimental design

We recruited 32 participants, including an 11-year old boy. All except the boy were adults with ages between 22 and 50 (mean = 26.625 and SD = 6.866). All participants reported normal hearing ability.

The study was conducted in a quiet room, where each participant was asked to listen to two versions of "Stronger"; one embedded with data and without. Participants were positioned 2 meters away from the speakers, were asked to compare the quality of the two clips, and asked to indicate which one they preferred. The presentation of the two versions was balanced (half had the data version first). This evaluation is similar to social acceptance evaluations in previous HCI research [14].

Results

Overall, the adult participants could not find any consistent difference between the two audio clips. Nineteen of the 31 adults (61.29%) could not find any difference between the two audio clips. The remaining 12 adult participants had inconsistent views. Six liked the version without data, while the other six liked the version with data. As expected, the 11-year old boy could hear the nearly inaudible clip embedded in the original audio clip, and commented, "I can feel the ringing in my ears". This indicates that PhoneEar needs to be studied further to identify which frequencies are audible to which user demographics. Overall, adults with normal hearing cannot perceive sound-encoded data

embedded in a song. We discuss the potential challenges highlighted by this study below.

DISCUSSION AND CHALLENGES

The PhoneEar system, four example applications and results from two studies highlight the strong potential of the approach to serve as a new and simple data channel that provides many new interaction possibilities for mobile devices. While our current work has shown that PhoneEar can work well, it has also raised several challenges that we discuss below and will address in future work.

Speed: The speed of our current PhoneEar implementation is 8 bits/second. As previously mentioned, our current implementation is meant to be a demonstration of the upper bound of PhoneEar's working distance and the corresponding lower bound of data transformation speed. Future work will focus on refining our data transfer technique to increase speed, while maintaining or increasing operating distance. We will also investigate the use of mark-up schemes to improve the semantic richness of messages but to maintain minimal data requirements.

Security risk: With our current scheme, it is possible for mischievous or erroneous data to be broadcasted, which could be misleading to a user. This raises significant security risks that warrant careful consideration. One solution could use data signing of messages, similar to what is done on the Web with HTTPS, to certify trusted data.

Occlusion And Environmental Noise: Although, our study showed that it is possible for data to be received when a smartphone is in a pocket, there are other conditions, in which the microphone might be covered or where environmental sound might interfere. Further investigation is required to find out how such challenges can be overcome, including more testing in real world scenarios. However, one simple scheme to decrease packet loss might be to broadcast PhoneEar at different frequencies in its play loop rather than repeatedly playing it at the same frequency.

Audible by kids and pets: Although our implementation uses audio frequencies that can be barely heard by adults, they can still generate noticeable sounds for children and, likely, pets and other animals. Whether this causes important problems for the technique is an open question. A potential alternative solution is to use much higher frequencies, e.g., ultrasonic waves. However, this will require microphones that can detect higher frequencies, which are not currently available in today's smartphones.

CONCLUSION

PhoneEar enables mobile devices to understand messages from everyday audio sources, ensuring that people do not miss important details and providing new interaction possibilities. Our approach enables digital audio content creators to embed simple text-based messages into audio streams, allowing mobile devices to 'hear' data, and people to interact with sound information in new ways. Through two studies we have shown that mobile devices can capture sound-encoded data with enough reliability to be used in real world situations, and the approach does not detract from the original sound quality. We built several novel example apps that illustrate a new design space of mobile interactions made possible by PhoneEar.

REFERENCES

1. Buxton, W. Integrating the Periphery and Context: A New Model of Telematics. *Proc. of GI*, 1995, 239-246.
2. Rodden, Tom, Keith Cheverst, K. Davies, and Alan Dix. "Exploiting context in HCI design for mobile systems." In *Workshop on human computer interaction with mobile devices*, pp. 21-22. 1998.
3. Deshotels, L. Inaudible sound as a covert channel in mobile devices. *Proc. USENIX conference on Offensive Technologies*. 2014. pp 16-16.
4. Gerasimov, V. & Bender, W. Things that talk: using sound for device-to-device and device-to-human communication. *IBM Syst. J.,* 39 (3-4), 530-546, 2000.
5. Google voice search. http://www.google.com/mobile/voicesearch/.
6. Hanspach, Mi., & Goetz, M. On covert acoustical mesh networks in air. 2014. *arXiv preprint arXiv:1406.1213*
7. Harrison, C. & Hudson, S. E. Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces. *UIST'08*, pp 205-208. ACM, 2008.
8. Ju, Wendy, Brian A. Lee, and Scott R. Klemmer. "Range: exploring implicit interaction through electronic whiteboard design." In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pp. 17-26. ACM, 2008.
9. Lopes, C.V. and Aguiar, P.M.Q. Acoustic modems for ubiquitous computing. *IEEE Pervasive Computing, Mobile and Ubiquitous Systems*,2(3),62-71. 2003.
10. Lu, Hong, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew 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*, pp. 165-178. ACM, 2009.
11. Madhavapeddy, A., Scott, D. and Sharp, R. Context-Aware Computing with Sound. *UbiComp'03*. 315-332.
12. Moore, B. An Introduction to the Psychology of Hearing. Academic Press. 1997.
13. Nandakumar, Rajalakshmi, Krishna Kant Chintalapudi, Venkat Padmanabhan, and Ramarathnam Venkatesan. "Dhwani: secure peer-to-peer acoustic NFC." In*ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 63-74. ACM, 2013.
14. Rico, J. and Brewster, S. Usable gestures for mobile interfaces: evaluating social acceptability. *Proc.CH'10*.pp 887-896. ACM, 2010.
15. Shazam: http://www.shazam.com/.
16. Wang, A. An Industrial-Strength Audio Search Algorithm. *Proc. ISMIR*. 2003. 7-13.
17. Yang, Xing-Dong, Khalad Hasan, Neil Bruce, and Pourang Irani. "Surround-see: enabling peripheral vision on smartphones during active use." In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pp. 291-300. ACM, 2013.