Context-Aware Computing with Sound

Anil Madhavapeddy[1], David Scott[2], and Richard Sharp[3]

[1] Computer Laboratory, University of Cambridge
[2] Laboratory for Communication Engineering, University of Cambridge
[3] Intel Research, Cambridge
William Gates Building, 15 JJ Thomson Avenue, Cambridge, CB3 0FD, UK

avsm2@cl.cam.ac.uk, djs55@cl.cam.ac.uk, richard.sharp@intel.com

Abstract. We propose *audio networking*: using ubiquitously available sound hardware (i.e. speakers, sound-cards and microphones) for low-bandwidth, wireless networking. A variety of location- and context-aware applications that use audio networking are presented including a location system, a pick-and-drop interface and a framework for embedding digital attachments in voice notes or telephone conversations.
Audio networking has a number of interesting characteristics that differentiate it from existing wireless networking technologies: (*i*) it offers fine-grained control over the range of transmission (since audio APIs allow fine-grained volume adjustment); (*ii*) walls of buildings are typically designed specifically to attenuate sound waves so one can easily contain transmission to a single room; (*iii*) it allows existing devices that record or play audio to be "brought into the user interface"; and (*iv*) it offers the potential to unify device-to-device and device-to-human communication.

1 Introduction

Researchers have spent a great deal of time studying how electronic devices can be augmented with sensors that allow them to perceive their environment. This body of work, often referred to as *context-aware* or *sentient* computing [6], stems from the idea that in order to build large-scale ubiquitous systems, individual devices must be aware of their environment (e.g. where they are, what entities are nearby, what these entities are doing, where and when something is happening). Based on such information, applications and devices can make decisions *proactively*, only engaging in interactions with humans when absolutely necessary [23].

Over the last two decades, a wide variety of sentient computing systems have been designed, built and deployed. Such systems range from complex centralised location systems that provide a high degree of accuracy [11] through to much simpler distributed tagging techniques that enable devices to detect their proximity to each other [14]. Whilst these sensor frameworks have been implemented and tested on a research-lab scale, a factor that has prevented their wide-spread global deployment is that they all rely on custom hardware. Such hardware is not

usually available on the mass market and, furthermore, is sometimes expensive and difficult to install, configure and maintain.

In this paper we propose that *audio networking* can be used as the basis for developing context-aware applications. Audio networking allows standard devices fitted with speakers and microphones (e.g. PDAs, laptops, desktop PCs and mobile phones) to exchange data and infer information about their environment. One of the key advantages of audio networking is that it enables context-aware applications to be immediately deployed on a large scale without requiring users to purchase and install additional hardware.

We believe that the most important property of audio networking is *locality* since, in this respect, it has two main advantages over conventional wireless networking:

1. In the audio domain we have fine-grained control over the amplitude (volume) of the transmission and hence fine-grained control over the range of the transmission. For example, by instructing a PDA to transmit sufficiently quietly whilst holding it near a laptop one can ensure that only the laptop responds to the transmission (see Section 3.2). Conversely, one could broadcast a URL to an entire room simply by increasing the volume.
2. Walls of buildings are often *sound-proofed*; i.e. designed specifically to attenuate sound-waves in the audible spectrum. Thus, by transmitting data in the audible spectrum it is easy to infer room-grained location. Note that the same is not true for microwave radiation so, for example, it is not easy to infer location information using IEEE P802.11 [21].

We propose that the major use of audio networking is as a low-bandwidth, localised control channel. While the low-bandwidth constraint means that it is not suitable for transferring large objects, the technique is ideal for transmitting object identifiers (i.e. URLs, IP addresses etc.). Imagine a typical office environment where there are already machines connected using conventional wireless and wired networking technologies. We see audio networking providing distributed context-aware services on top of this existing infrastructure.

Putting our work into context, we observe that there is a long history of using audible signals to transmit information between electronic devices. For example, modems have been used for decades in digital communications and the telephone network uses a variety of audible signals to transfer data between handsets and exchanges [25]. More recently, researchers have revisited ideas originally exploited three decades ago in acoustically-coupled modems and devised coding-schemes that allow data to be transmitted and received through air using speakers and microphones [9]. The contributions of our research are (*i*) to extend these techniques by exploring new modulation schemes; (*ii*) to apply audio networking to location- and context-aware computing; and (*iii*) to implement applications and user interfaces that rely on audio networking.

The remainder of the paper is structured as follows: Section 2 describes the implementation and performance of various coding and modulation schemes that form the physical-layer of our audio networking implementation; Section 3, the main body of the paper, gives examples of a variety of applications that can

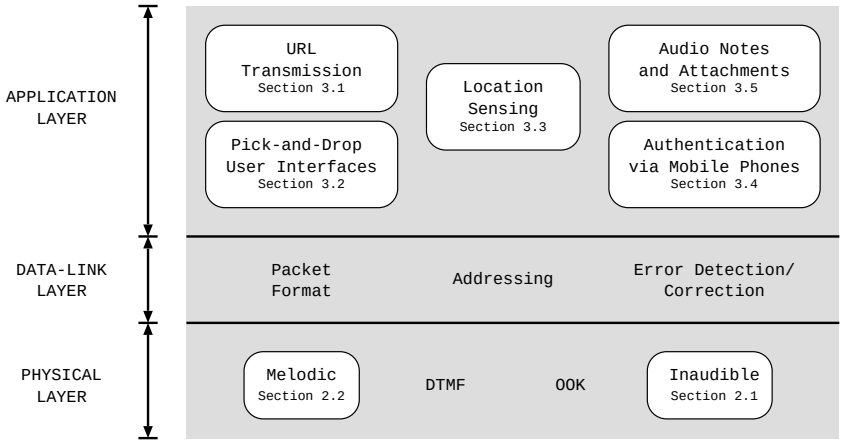

Fig. 1. A diagrammatic view of our audio networking research.

be implemented using audio networking; Section 4 discusses related work and Section 5 concludes and gives directions for future work. Figure 1 presents a diagrammatic view of this paper's structure in the wider context of our audio networking research.

2 Audio Data Coding

At an abstract level the audio networking concept is very simple: transmitters modulate their data in a suitable fashion and play the resulting audio data using the host machine's speakers. Receivers listen (via their microphone) and demodulate incoming signals to recover the transmitted information.

However, although the concept itself is simple, there are a plethora of audio data modulation schemes to choose from. Each of these has different characteristics in terms of bit rate, range of transmission and, equally importantly, what the data *sounds like* to humans. The latter characteristic is particularly thought provoking as it is not one that designers of traditional wireless networking technologies have to face.

In previous work researchers have explored various coding and modulation techniques that allow data to be transmitted as audio samples [9, 15]. As part of our research we have implemented four audio data transmission schemes:

1. Dual-Tone Multi-Frequency (DTMF) [25] (as used in touch-tone phones).
2. On-Off Keying (OOK) [9] modulated over a variety of audible carrier frequencies.
3. Inaudible data transmission (see Section 2.1).
4. Melodic data transmission (see Section 2.2).

Our transmission schemes divide data into fixed-size packets, each prefixed with a short preamble. A CRC field is used to detect transmission errors; invalid

packets are dropped. Our audio physical layers do not guarantee delivery: as in many networking stacks, we view this as the job of higher level protocols.

Using standard sound hardware (internal laptop speakers/microphones and a pair of $10 Harman Kardon HK 206 desktop speakers) we achieved a bit rate of 20 bits/s using DTMF across a 3 meter distance with a 0.006% symbol error rate (i.e. 0.006% of DTMF tones were decoded incorrectly). We found that our OOK coding scheme was better suited to short range transmission (less than 1 meter) since at low amplitudes one does not have to worry about pulse reflections. Using OOK modulated over a 10 kHz carrier we achieved a bit rate of 251 bit/s across a 30cm distance with a bit error rate of 4.4×10^{-5}. Although these bit rates are sufficient for the applications of Section 3 we are confident that with further implementation effort and cleverer DSP the throughput reported here could be improved significantly.

Since DTMF and OOK are standard techniques we do not consider them further in this paper. The remainder of this section describes the technical details of our inaudible data transmission scheme (Section 2.1) and our melodic transmission scheme (Section 2.2).

The interested reader may like to hear our audio coding schemes for themselves. For this purpose, we have placed a variety of sound samples on the web [1].

2.1 Inaudible Data Transmission

Whilst audible data transmission is acceptable in many situations and sometimes even beneficial (see Sections 3.3 and 3.5), it transpires that we can also use standard sound hardware to transmit data that is inaudible to humans.

Our current implementation of the inaudible data transmission scheme simply uses a variant of our OOK implementation, modulated over a 21.2 kHz carrier. The 21.2 kHz carrier frequency is conveniently chosen to be greater than the maximum frequency of human hearing and less than the Nyquist limit of standard sound-cards (which operate at 44.1 kHz). To ensure that users cannot hear audible clicks as the carrier switches on and off, we apply an amplitude envelope to each transmitted pulse. By increasing the attack and decay times of pulses we effectively band-limit the transmitted signal, confining it to the high-frequency (inaudible) region of the spectrum.

By analysing spectrograms of ambient noise in office environments we discovered that most noises in the 20 kHz–22 kHz region of the audio spectrum tend to occur in short bursts (e.g. doors closing, keys jangling etc.). In this environment, although our inaudible data transmission scheme is narrow-band, we can make it more robust against bursty high-frequency noise (at the cost of reducing the bit rate) by increasing transmitted pulse lengths. Our initial experiments involved broadcasting using a bit rate of 8 bits/s across a distance of 3.4 meters in an office with two occupants. The experimental setup consisted of a Dell desktop with on-board sound (Intel AC97 Audio Codec) transmitting data through Harman Kardon HK 206 speakers. The receiver was a Sony Vaio PCG-Z600LEK laptop. The occupants of the office generated ambient noise by continuing with their normal tasks (which included typing, bursts of conversation, walking around the

room, moving objects and papers, answering the telephone and drinking coffee). Under these conditions, out of the 172 16-bit identifiers transmitted, 95% of them were received correctly.

Whilst 8 bits/s is undoubtedly a low transmission rate, it is still adequate for broadcasting small identifiers. For example, a throughput of 8 bits/s and a packet receipt rate of 95% is sufficient for the room-grained location beaconing example described in Section 3.3.

We do not have enough information to state categorically that the high-frequency inaudible transmission scheme described here will work on all sound hardware. However, we have repeated the above experiment using a variety of different microphones, speakers and sound-cards. In each case we have been able to transmit data over the 21.2 kHz carrier.

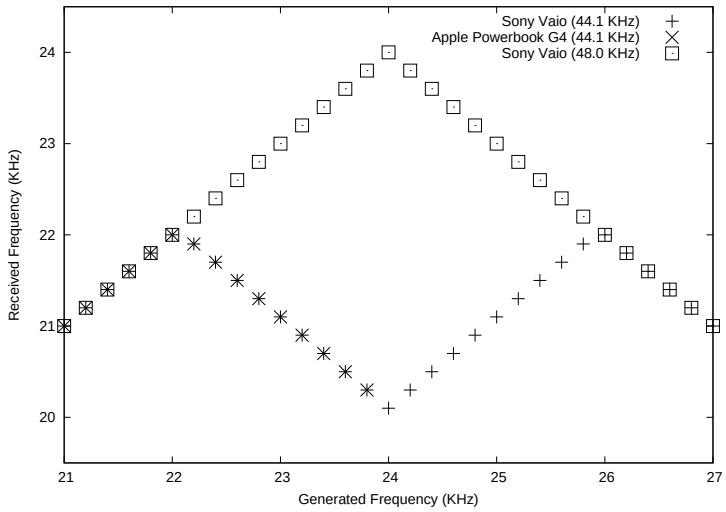

Fig. 2. A graph showing how standard sound-cards perceive frequencies above their Nyquist limit. The minima at a generated frequency of 24 kHz in the Vaio's 44.1 kHz sample-rate curve suggests, as expected, that the underlying codec is still sampling at 48 kHz.

As an interesting aside, we discovered that many sound-cards have poor or non-existent anti-aliasing filters. Experiments were performed which demonstrate that a number of sound-cards can receive sounds at significantly higher-frequencies than the Nyquist limit imposed by their sampling rate (see Figure 2). Although frequencies greater than the Nyquist limit occur as *aliases* in the received frequency spectrum, they can nevertheless be detected and analysed. This opens the possibility of having higher frequency transmitters (e.g. computers fitted with 96 kHz sound-cards) broadcasting data in the 20-30 kHz region, which can still be received and decoded by devices with standard (44.1/48 kHz) codecs. Although our 21.2 kHz carrier was inaudible to the adults we tested it on at

the amplitudes we used for transmission, increasing the separation between the threshold of human hearing and data transmission is still a desirable property. For example, young children have better high-frequency hearing than adults. By emitting data in the 24–27 kHz region and relying on the aliasing effects of standard hardware we could potentially increase transmission amplitude whilst ensuring that the signal remains inaudible to children. Investigating these ideas further is a topic of future work.

We observe that although signals modulated onto a 21.2 kHz carrier are inaudible to adult humans they are well within the hearing range of a dog. (A dog's threshold of hearing is an octave higher than that of a human: approx. 40 kHz). Further experiments are required to ascertain the effects that our inaudible data transmission scheme has on guide-dogs located in office environments.

2.2 Melodic Data Transmission

In this section we describe a framework that allows information to be transmitted as musical melodies. Although the generated melodies are unlikely to receive critical acclaim they are nevertheless "amusing little ditties" that sound surprisingly pleasant. We believe that the major application of this technique is to encode information in a form that can be played back using mobile phones (see Section 3.4).

Our melodic data transmission scheme assumes the existence of a set of 4 carrier frequencies. A two-bit value is signalled by playing a tone at one of these carrier frequencies for a pre-specified duration. Note that instead of activating two of these frequencies simultaneously as DTMF does, we only activate a *single* carrier frequency at a time. This enables us to encode data as *monophonic* ringtones: the base level supported by the majority of modern mobile phones.

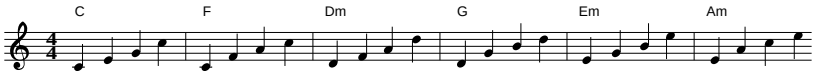

Fig. 3. Notes corresponding to carrier frequencies for first six bars of the "baroque" theme.

At any given time, we choose 4 notes from the C major (ionian) scale as our carrier frequencies. A melody is constructed by varying the carrier frequencies over time, according to a predefined sequence known both to the transmitter and the receiver. A musician is employed to choose carrier frequencies in such a way that, irrespective of the data being transmitted, the resulting melodies sound pleasing to the ear. We call a particular sequence of carrier frequencies a *theme*. Figure 3 shows an example of a theme that follows a chord sequence commonly found in baroque music. Each bar contains a group of four carrier frequencies. The nth carrier frequency (i.e. the nth crotchet) of each bar encodes the two-bit number n.

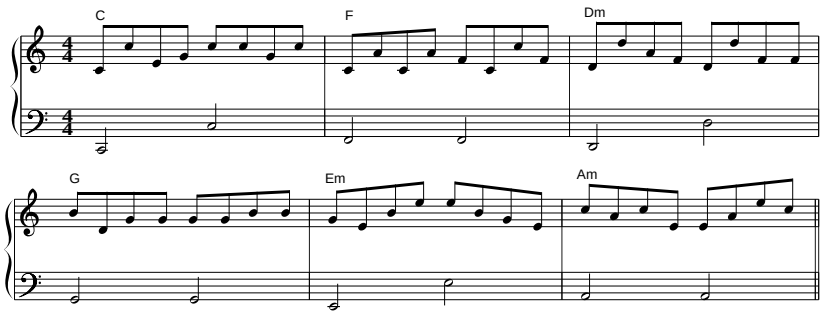

Fig. 4. An encoding of a 96-bit packet "`36FB 224D 3935 C55F 4BE4 981E`" using the *baroque* theme. (The top stave represents the data.)

Our current implementation changes carrier frequencies every 8 notes (i.e. every 16 bits). Under this scheme, the top stave of Figure 4 shows the melody corresponding to a 96-bit packet modulated according to the "baroque-themed" frequency hopping. Two notes of the melody encode a single hex digit.

Although we use monophonic melodies to encode our information, that does not stop devices capable of generating polyphonic music from incorporating other voices in their musical output. For example, the data-transmission example of Figure 4 is augmented with a bass-line. Although the receiver will ignore the bass-line[1] it will be heard by the human listener, making the data transmission sound more interesting.

Of course there are many themes other than the "baroque" one described above. As part of our melodic data transmission implementation, we have composed themes that represent an eclectic selection of musical styles ranging from "blues" through to "eighties pop"! We envisage a scenario where both transmitter and receiver have access to a pre-composed library of themes. The user configures their device to modulate data using their favourite theme (cf. personalisation of mobile phone ring-tones). The choice of theme is encoded as a brief preamble to the transmitted data broadcast using carrier frequencies known to both transmitter and receiver. In our current implementation we use two carrier frequencies (middle C and the C an octave above) to encode the preamble.

We have prototyped our melodic data transmission scheme using mobile phone ring-tones. Our mobile phones played the ring-tones at 7 notes per second leading to a transmission rate of 14 bits/s. We observed that although both receiver and phones are capable of faster transmission[2], increasing the tempo of the ring-tones beyond 8 or 9 notes per second makes them sound less pleasant to human listeners: the ear does not have enough time to pick out the melody. Since the melodic nature of the data encoding is a feature we wish to emphasise,

[1] It will be filtered out since it occurs outside the frequency range in which data is expected (and is thus treated as unwanted noise).

[2] We achieved 83 bits/s (the fastest that the mobile phone could play ring-tones) with an upper limit of 0.001% bit error rate; the phone was held 2cm from the microphone.

we see 14 bits/s as the upper limit of this encoding technique. This is more than adequate for transmitting short (e.g. 96-bit) IDs (see Section 3.4).

3 Audio Networking: Example Applications

In Section 1 we motivated audio networking as a "low-bandwidth localised control channel". Here we expand this idea further, giving concrete examples of applications based on audio networking technology.

The case studies presented here are designed to illustrate different properties of audio networking. Sections 3.1 and 3.2 use audio for local transmission of object identifiers (e.g. URLs); Section 3.3 demonstrates the room-grained locality property of audio networking; and Sections 3.4 and 3.5 show how existing devices that can record or play sounds can be "brought into the interface".

3.1 Local Transmission of URLs

To demonstrate the locality properties of audio networking we implemented a "shared collaborative environment" that allows devices in close proximity to each other to transmit and receive URLs in the spirit of HP's CoolTown project [13]. In our current implementation, received URLs are automatically added to a pre-specified bookmarks folder, allowing users to peruse them at their leisure.

Although transmitting short URLs is fine, we soon found that it can take an unacceptably long time to transmit long URLs over low bit-rate audio channels (e.g. consider a complex dynamic web application with a large number of HTTP-GET parameters). To alleviate this problem we introduce another layer of indirection via an *object server*, a networked machine that associates short *object IDs* (i.e. primary keys) with URLs. Recall that, in Section 1, we motivated audio networking as a control-channel for existing high bandwidth networks. By using an object server we can design protocols that transfer as much as possible using the existing wired/wireless networking infrastructure whilst still exploiting the locality properties of sound.

In this framework, our local shared collaborative environment works as follows: the transmitter (*i*) creates a new record on the object server that contains the URL to be transmitted (using the high-bandwidth network); and (*ii*) broadcasts the corresponding unique identifier over audio. Receivers decode the identifier and fetch the corresponding URL from the object server. Essentially object IDs are just *relative URLs* which, although not globally unique, are unique to a particular object server. Of course, we can make object IDs globally unique by prefixing them with the IP address of the object server on which they are stored. However, since our aim is to make object IDs as small as possible, we assume for now that both transmitter and receiver are configured to use the same object server. This setup is ideal for transmitting objects around an office-scale environment.

The object server implementation is straightforward, comprising a web-server, a SQL database and scripts that deal with insertion and retrieval of URLs (over

standard HTTP). A number of housekeeping tasks are performed automatically according to a configurable system-wide policy. For example, entries that time-out are automatically removed. This allows us to keep the pool of active objects small, reducing the lengths of object identifiers. In contrast to URLs, which are often designed to be persistent, object IDs frequently only exist for a few seconds: the time it takes devices to receive them over audio and dereference them over a high-bandwidth network.

3.2 A Pick-and-Drop User Interface

A *Pick-and-Drop user interface* [19] enables users to move virtual objects around in physical space by associating them with physical devices. We have implemented an interface that allows users to *pick up* virtual objects (e.g. files, documents) onto their PDAs, carry them around a building (using the physical PDA to represent the virtual object) and *put them down* on other devices. As a motivating example for this kind of user interface, consider a person who picks up a document from their desktop machine, walks across a building and puts it down on a printer (at which point the document is automatically printed)[3].

User interfaces such as this have been studied at great length. What makes our interface interesting is that it is implemented using audio networking. When an object is *picked up*, the transmitting device copies it to a local web-server, inserts the object's URL into the object server and transmits the corresponding object ID to the PDA over audio. Similarly, when an object is *put down*, the PDA broadcasts the object ID to the receiving machine over audio, which fetches the corresponding document and acts accordingly. We believe that there are two main advantages to this approach: (*i*) it is built out of existing, commonly deployed hardware; and (*ii*) giving the user fine-grained control over the transmission range (i.e. volume) gives the user an extra degree of flexibility. For example, transmitting the object ID over at an amplitude corresponding to a 4-meter range may copy the corresponding object to all computers in a room; conversely transmitting over a 1 cm range would only transmit the object ID to the single device held next to the PDA.

We have implemented a simple GUI for transmitting files using object IDs (see Figure 5). On the transmitter side, the user drags files into the *transmit window* (at which point the document is copied to the object server, an ID is assigned and broadcast over audio to a local PDA). On the receiver side the interface consists of a menu of recently received objects. By selecting an object, one can open it on the PDA (if the device supports the file-type) or retransmit the object ID as audio. A configuration dialog allows the interface to be customised to suit the user: e.g. whether objects should automatically be removed from the menu once they have been transmitted, or over what range the object IDs should be broadcast (see Section 3.2).

[3] Since printers don't have microphones we would probably have to place a device near the printer that receives object IDs and issues print commands. However, since in many cases there are print servers located next to printers, these devices could often be used for this purpose.

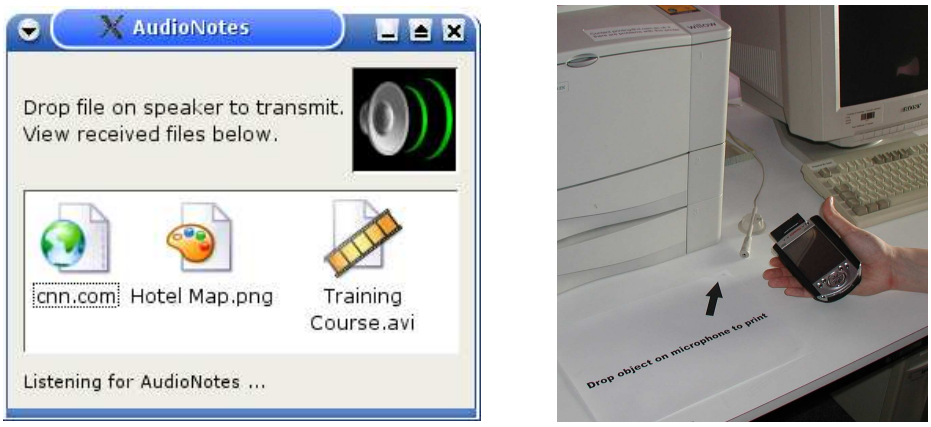

Fig. 5. *Left:* The GUI interface for receiving/transmitting object IDs; *Right:* Transmitting a document ID for printing using audio networking

Although machines that access the objects themselves must be connected to high-bandwidth networks, one can still move object IDs around using devices without networking capability. However, there is one subtlety which arises if the PDA shown in Figure 5 is not networked: in order for users to browse the object IDs currently held on the PDA it is essential that human-readable names (e.g. filenames) are associated with object IDs. When a device receiving an object ID has access to conventional higher bandwidth networking then they can use this medium to query the object server, obtaining the object name from the ID. In this case only the object ID itself has to be broadcast over audio. However, when a device receiving an object ID does not have access to higher bandwidth networking then the object name must be transmitted over audio along with the corresponding ID.

Short range transmission: Although room-scale multicast is a useful property of audio networking, there are a number of scenarios where very short range transmission (i.e. 0–2 cm) is appropriate (see above). Short range audio transmission has a number of useful properties. In particular:

1. Transmission amplitude can be reduced so as not to cause distraction to others in the same environment. We have demonstrated that by holding the device within approximately 2 cm of a receiver's microphone we can transmit data at comparable volume to playing quiet music through headphones or listening to a voice message through a telephone handset.
2. In a room that contains several devices, short range transmission provides a simple and intuitive mechanism for a user to select a single device to communicate with. Devices only communicate if they are less than a few centimeters apart.

3.3 Room Grained Location

In Section 1 we observed that, since buildings are specifically designed with sound-proofing in mind, sound tends to be attenuated by walls. As a result, audio signals provide a convenient mechanism to infer room-grained location. Figure 6 shows experimental evidence that justifies this claim. We measured the received amplitude of a 21.2 kHz pulse (transmitted from a pair of desktop speakers) on both sides of an office wall, using the built-in microphone and sound-card on a Sony Vaio laptop. (Recall that 21.2 kHz is the carrier frequency used for our inaudible transmission scheme.) Although inside the room the tone is loud, outside the room it is not detectable.

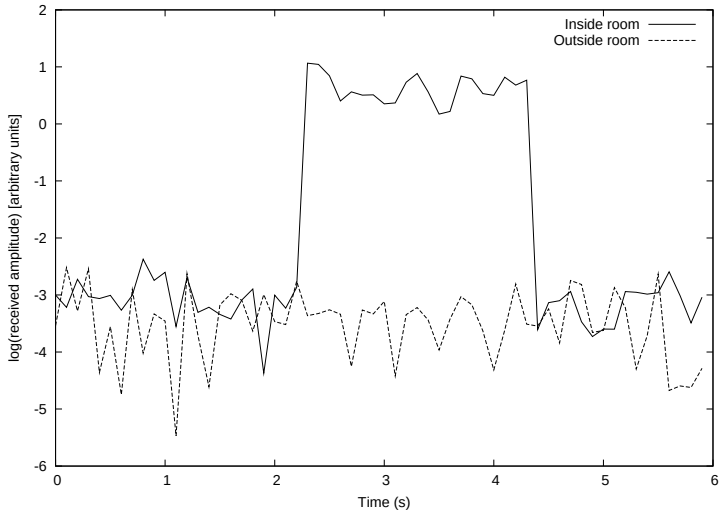

Fig. 6. A graph showing the received amplitude of a 2 second 21.2 kHz pulse on the both sides of an office wall 3 meters away from the speaker. Outside the room the tone is barely detectable: attenuation of > 20 dB.

The simplest audio location technique involves placing a *beacon* in each room that repeatedly broadcasts a unique *room identifier*. Devices placed in the same room as the beacon decode the room identifier and use this information to infer their location. We do not advocate the use of extra hardware for room beacons. Instead, we note that in many office environments there is at least one device in each room that seldom moves (e.g. a desktop computer). A simple *beaconing service* (which repeatedly plays a sound sample containing the room id) is installed on this machine.

The major concern with this method is that the constant beaconing may annoy users. To alleviate this problem we employ our inaudible transmission scheme (see Section 2.1) in the beaconing service. Our prototype implementation transmits a room identifier every 15 seconds; room identifiers take 3.5 seconds

to be transmitted. To demonstrate the utility of room-grained location we have implemented an application that automatically configures a laptop to use its nearest printer as it is moved around a building.

Whilst the beaconing technique allows devices to infer their own location, it does not provide a mechanism to query the location of other devices. To address this limitation we implemented another system in which fixed desktop machines run *listener services* that detect short (0.3 s) 800 Hz tones and report back to a centralised *location server*. We assume that the devices whose location we want to query already have conventional (wired or wireless) networking capability. In this framework a location query can be performed as follows: (*i*) the device to be located is signalled over the network; (*ii*) the device modulates a short code over a 800 Hz tone; and (*iii*) any listener services that hear the tone report back to the location server which aggregates the information in order to infer which room the device is in.

Instead of the location server continually polling devices, the system is intended to query the location of a device only in response to a user's request. In this context, we deliberately choose an audible signal (800 Hz) for the device's "I'm here" message, since it performs the useful function of notifying nearby humans that the location of a device has been queried. It is interesting to note that in ORL's Active Badge System [26] a speaker on the badge was specifically employed to notify the wearer when their location had been queried. Stajano argues that this *principle of reciprocity* makes the location-system more palatable to users worried about loss of privacy [22]. A nice feature of our implementation is that we are able to unify the signalling of the listening service and the notification of the human observer in a single audible broadcast.

There are a number of other audio location architectures which one can envisage. For example, whereas the location system proposed above signals the device to be located over an existing (wired or wireless) network, one can imagine a system that signals the queried device using an audio broadcast played simultaneously in all rooms. This avoids the requirement that locatable devices must be connected to an existing network. Another variant of this idea is to modify the listening service to detect mobile phone ring-tones. In this context, mobile phones (and therefore, with high probability, their owners) can be located. In our experiments with this technique we configured our mobile phones to play a single-note ring-tone when called from a known phone used to initiate location queries. In this way one can easily disambiguate location queries from genuine phone calls. (The *caller groups* feature, which allows specific ring-tones to be associated with specific callers, is a common feature on modern mobile handsets.)

3.4 Authentication using Mobile Phones

Section 2.2 shows how data can be encoded as melodies. Using this encoding scheme, unique identifiers can be sent to users' mobile phones as ring-tones. An interesting application of this technology is to use ring-tones as capabilities: by playing back a received ring-tone users can authenticate themselves to a device.

As a real-world application of this technique, consider an automatic ticket-collection machine (of the type commonly found in cinemas) that prints tickets that have been pre-booked over the Internet or telephone. Systems of this type typically use credit card numbers to authenticate users: when a credit card is inserted into the collection machine, the appropriate tickets are generated. However, since children tend not to own credit cards (and may not be trusted to borrow their parents' cards) they are precluded from using the ticket-collection device.

Consider the scenario where a parent purchases cinema tickets for a group of 14 year-old children[4]. As part of the booking system the parent enters the mobile phone number of one of the children. An authentication ring-tone is sent to this phone via SMS, allowing the children to collect the tickets without having to borrow the adult's credit card.

Using ring-tones to transmit data has different properties to transmitting data using Bluetooth [2], which is increasingly becoming integrated into mobile phones. In particular, whereas Bluetooth offers significantly higher bandwidth, audio supports stricter locality properties. These properties complement each other well: consider a scenario where a mobile phone uses an audible exchange to authenticate itself against a local Bluetooth device. Once audio has been used to introduce the devices, higher-bandwidth communication can be initiated automatically over Bluetooth. This removes the current requirement for the user to enter an authentication PIN into the Bluetooth device they wish to communicate with.

We believe that the mobile-phone audio authentication technique described here will become more flexible as the Multimedia Message Service (MMS) continues to gain acceptance over SMS [7]. Using MMS one can transmit large audio samples to mobile phones.

3.5 Audio Notes and Attachments

Audio networking allows devices that do not currently inter-operate with computers, to be "brought into the interface". In Section 3.2 we have seen how object IDs can be recorded on PDAs to create a user interface where users can physically move objects around a building. However, we do not have to rely on PDAs to record object IDs: any device that can record audio is sufficient. In this section we present two applications that use existing recording devices and transmit object IDs. In both cases, the transmitter uses our drag-and-drop interface to copy object URLs to the object-server and transmit unique audio IDs (see Section 3.1).

By recording object IDs using standard voice recorders (i.e. cellphones, analog dictaphones, voice mail systems etc.) we can *attach* digital documents to voice notes. For example consider the memo: "Dave sent me revision 2 of this document

[4] We conveniently choose an age at which children are (*i*) old enough to go to the cinema by themselves; (*ii*) not old enough to own credit cards; and (*iii*) likely to own a mobile phone!

today ⟨insert object ID_1⟩; must forward it to Anil ⟨insert object ID_2⟩" where ID_1 corresponds to a Word Document and ID_2 corresponds to Anil's email address (i.e. a `mailto:` URL). When the object IDs are played back near a networked computer it can receive the IDs, fetch the associated URLs and act accordingly (e.g. open the Word Document and open a blank email window with the `To:` field containing Anil's email address).

Another similar example involves embedding objects IDs in telephone conversations. As part of a conversation the transmitter tells the receiver that they would like to transmit an object. At this point, the transmitter holds the handset to their computer's speaker and drags the document to the transmit window (resulting in an audible object ID); the receiver holds the handset to their computer's microphone to receive the object ID[5]. As long as both computers are configured to use the same object server, the file simply appears on the receiver's computer. Both users then pick up their handsets again and continue their conversation. Whilst the idea of using acoustic-coupled software-modems as an out-of-band channel to control higher bandwidth networking infrastructure is technically straightforward, we have found the resulting applications to be useful and intuitive. Just as documents can already be attached to emails, audio networking allows files to be attached to telephone conversations and voice notes.

DTMF is a suitable transmission mechanism for these applications since it uses the voice frequency-band and therefore performs well over devices such as telephones and dictaphones. Our current implementation of these applications uses fairly low-speed DTMF (4 tones or 16 bits/s) to transmit object IDs. The main reason for slow transmission in these examples is that object IDs are recorded and replayed a number of times in the analog domain. We have to make sure that transmitted data is robust against the signal degradation that occurs as a result. Of course, 16 bits/s is perfectly adequate for transmitting short object IDs.

4 Related Work

Many devices exist that offer wireless data transmission capabilities. Amongst the most popular, IEEE 802.11b [8] and Bluetooth [2] use relatively low power radio transmissions in the 2.4 GHz band to provide local area networking. On a smaller scale, motes [12] are low-power sensor devices that have been successfully deployed in an environmental monitoring project [16]. Motes use a short range radio link to communicate sensor readings with each other and with a base station. The PEN [5] project (previously named "piconet") focused on using very infrequent, low power radio transmissions to create a low-bandwidth localised control channel.

[5] Of course one can envisage a more developed framework where computers are given a direct audio interface to the user's telephone. However, since this kind of interface is not considered standard hardware it is not the focus of this paper.

There are a number of factors that differentiate our work from these technologies. Firstly, our "audio network interface cards" are ubiquitous – after all, most computers and PDAs, as well as other common devices such as mobile phones and dictaphones, come equipped with the ability to record and play back sound. Secondly, many existing microphones are designed to be unidirectional whilst, in contrast, radio antennae are typically omni-directional. As a result audio devices must be roughly pointing at each other in order for communication to take place. We see this as an advantage: it makes it more difficult for devices to communicate by accident. (However, note that audio is not as directional as IrDA). Thirdly, we directly benefit from the low-level nature of the sound interface. Whereas a network card typically offers a fixed set of APIs, the sound interface is just a DAC. Since all signal processing must be done in software we are able to directly manipulate the representation of packets at the sample level, gaining the same kind of flexibility as software radios [24]. One major advantage is that we can easily implement and test new application-specific features. For example we added a *send packet quietly* API to transmit a packet over a shorter range by reducing the amplitude of the generated samples. A further benefit is that we can infer more information from each transmission than just a packet's payload. For example, the received amplitude of the signal is an indication of distance between sender and receiver.

There are a wide variety of location systems described in the literature. Notable examples include active bats [27], active badges [26] and crickets [18]. Unlike our audio-based system, these projects all require custom hardware. The RaDaR [3] project, which attempts to infer location from the signal strength of an IEEE P802.11 WaveLAN interface, is closer to our approach. However, it suffers from two critical problems: (*i*) it is limited to using only the APIs that come with the WaveLAN card (i.e. the software does not have access to the raw RF signal); and (*ii*) WaveLAN is designed to permeate buildings in order to provide network coverage. Since, in contrast, buildings are explicitly designed to *block* audio transmissions we argue that audio is a more appropriate medium for inferring room-level location.

Although a lot of work has gone into the design of human-to-computer audio interfaces [20] (and conversely computer-to-human audio interfaces [17]), the idea of using audio for computers to communicate with each other seems to have been largely overlooked in mainstream computing. We believe that this may be, at least partly, due to a concern that audio transmission is too intrusive to humans. We hope that the applications presented here (in particular, see Sections 3.3 and 3.2) demonstrate that this is not the case.

Garasimov and Bender [9] have investigated a number of techniques for transmitting information in the audible spectrum. They concentrated mostly on increasing bandwidth e.g. fast On-Off Keying (OOK) and Code Division Multiple Access (CDMA). The bandwidths they report are high (3.4 Kbits/s for CDMA) although no error rates are presented. Their paper states that it is difficult to do inaudible data transmission using standard hardware, claiming that standard 44.1 kHz DACs do not work well enough to transmit data over frequencies

greater than 18.4 kHz (a frequency that is audible to most people). In contrast, we found that the DACs on cheap sound-cards worked well at much higher frequencies (even above the Nyquist limit due to lack of anti-aliasing filters, see Section 2.1). The bit-rates we achieved whilst modulating data over a a 21.2 kHz carrier, although low, are sufficient for a variety of context- and location-aware applications (see Section 3.3).

Lopes and Aguiar have also studied mechanisms for encoding data as melodies [15]. The main contribution of our work in this area is the application of melodic data encoding to mobile phone ring-tones. Also, our scheme is the first to allow data to be *themed* in the musical style preferred by a user. The existence of a multi-million-dollar industry in themed ring-tones suggests that this is an important consideration!

The use of audio communication has interesting security properties. Balfanz et al [4] considers the use of audio as a "privileged side channel" to bootstrap trust between strangers. His proposed system relies on the property that although the side channel can probably be snooped (e.g. case using a very sensitive microphone) it is not possible for attackers to subvert the protocol without drawing attention to themselves (e.g. by making a loud noise). The implementation and deployment of such a system using audio networking technology is an interesting area of future work.

5 Conclusions and Further Work

In this paper we have motivated audio networking as a low-bandwidth localised control channel and described the implementation and performance of a variety of audio data transmission schemes. A number of applications based on audio networking have been described demonstrating that location- and context-aware applications can be based on existing deployed hardware using audio networking techniques.

In future work there are a number of other audio transmission mechanisms that we are keen to explore. For example, by mixing our high-frequency narrow-band transmission scheme (see Section 2.1) with band-limited samples in the audible spectrum we are able to combine inaudible data transmission with audible noises. A possible use for this technique is to accompany inaudible transmitted data with an audible sound, the sole purpose of which is to provide information to human listeners. For example, an object ID (see Section 3.1) broadcast in the 21 kHz band could be augmented with a sample that speaks the name of the object (e.g. via voice synthesis) in the 0–10 kHz band.

Digital watermarking research has explored a number of ways in which inaudible data can be hidden in audible sounds. A technique that we find particularly interesting is hiding data in music [10]. We are keen to ascertain whether we could use this technique to hide data in background music played in department stores, restaurants etc. Note also that hiding data in other audio-streams would allow us to use audio networking techniques in environments where people are already using their PC's speakers for other purposes. This would be extremely

useful in office environments, for example, where people often use their desktop speakers to play music.

One of the advantages of audio networking is that the flexible APIs allow one to infer more information from a received packet than just its payload data (see Section 4). In future work we intend to explore this idea further. For example, how accurately can one measure distances between devices using an audible "ping"?

Audio networking has a variety of interesting properties that differentiate it from conventional wireless networking technologies:

1. the software has fine-grained control over the audio physical layer;
2. building sound-proofing constrains transmission to a single room; and
3. device-to-device and device-to-human communication can be unified.

Furthermore, since any device that can play or record sound can take part in audio networking, the technique relies entirely on ubiquitously available hardware. We hope that the research reported in this paper paves the way for location- and context-sensitive applications that can be deployed immediately using existing infrastructure.

Acknowledgements

This work was supported by Network Appliance, Inc., Intel Research, and the Schiff Foundation. The authors wish to thank Derek Mcauley, Alistair Beresford, Andrew Moore, Dorothy Moore, Madeleine Glick, Charles Berlin, Mark Hagaard, Jon Crowcroft, Alan Mycroft, James Scott, Tim Harris, Frank Hoffmann and the anonymous reviewers for their valuable comments and suggestions.

References

1. Audio Networking Example Sounds: `http://www.recoil.org/audio/`.
2. The Bluetooth Specification version 1.1. `http://www.bluetooth.com/`.
3. P. Bahl. User location and tracking in an in-building radio network. Technical Report MSR-TR-99-12, Microsoft Research, February 1999.
4. D. Balfanz, D. Smetters, P. Stewart, and H. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Network and Distributed Systems Security Symposium*. Internet Society, 2002.
5. F. Bennett, D. Clarke, J. Evans, A. Hopper, A. Jones, and D. Leask. Piconet: Embedded Mobile Networking. *IEEE Personal Communications*, 4(5):8–15, 1997.
6. Diego Lopez de Ipina and Sai Lai Lo. Sentient computing for everyone. In *Distributed Applications and Interoperable Systems*, pages 41–54, 2001.
7. ETSI. Universal Mobile Telecommunications System (UMTS); Multimedia Messaging Service (MMS); stage 1. TS 122 140 V5.4.0 (2002-12) - (3GPP TS 22.140 version 5.4.0 Release 5).
8. IEEE Standards for Information Technology. *Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Network - Part 11: Wireless LAN Medium Access Control and Physical Layer Specifications*. 1999.

9. V. Gerasimov and W. Bender. Things that talk: Using sound for device-to-device and device-to-human communication. *IBM Systems Journal*, 39(3 and 4), 2000.
10. Daniel Gruhl, Anthony Lu, and Walter Bender. Echo hiding. In *Information Hiding*, pages 293–315, 1996.
11. Andy Harter, Andy Hopper, Pete Steggles, Andy Ward, and Paul Webster. The anatomy of a context-aware application. In *Mobile Computing and Networking*, pages 59–68, 1999.
12. J. Hill and D. Culler. A wireless embedded sensor architecture for system-level optimization. Submitted for publication, 2002.
13. Tim Kindberg and John Barton. A Web-based nomadic computing system. *Computer Networks (Amsterdam, Netherlands: 1999)*, 35(4):443–456, 2001.
14. D. Kirsch and T. Starner. The Locust Swarm: An environmentally-powered, networkless location and messaging system. In *1st International Sumposium on Wearable Computers*, pages 169–, 1997.
15. Cristina Videira Lopes and Pedo M.Q. Aguiar. Aerial acoustic communications. In *International Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, October 2001.
16. Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, and John Anderson. Wireless sensor networks for habitat monitoring. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, GA, September 2002.
17. Elizabeth D. Mynatt, Maribeth Back, Roy Want, Michael Baer, and Jason B. Ellis. Designing Audio Aura. In *Conference on Human-Computer Interaction*, pages 566–573. ACM, 1998.
18. Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The Cricket location-support system. In *Mobile Computing and Networking*, pages 32–43, 2000.
19. Jun Rekimoto. Pick-and-drop: A direct manipulation technique for multiple computer environments. In *Proceedings of User Interface Software Tools (UIST)*, pages 31–39, 1997.
20. Alexander I. Rudnicky, Alexander G. Hauptmann, and Kai-Fu Lee. Survey of current speech technology. *Communications of the ACM*, 37(3):52–57, 1994.
21. Siddhartha Saha, Kamalika Chaudhuri, Dheeraj Sanghi, and Pravin Bhagwat. Location determination of a mobile device using IEEE 802.11 access point signals. Master's thesis, Indian Institute of Technology Kanpur, 2002.
22. Frank Stajano. *Security for Ubiquitous Computing*. John Wiley and Sons, February 2002.
23. David Tennenhouse. Proactive computing. *Communications of the ACM*, 43(5):43–50, 2000.
24. David L. Tennenhouse and Vanu G. Bose. Spectrumware: A software-oriented approach to wireless signal processing. In *Proceedings of The First International Conference on Mobile Computing and Networking*, pages 37–47. ACM, 1995.
25. International Telecommunication Union, editor. *CCITT Blue Book Recommendation Q.23: Technical Features of Push Button Telephone Sets*. Geneva, 1989.
26. Roy Want, Andy Hopper, Veronica Falcao, and Jon Gibbons. The active badge location system. Technical Report 92.1, ORL, 24a Trumpington Street, Cambridge CB2 1QA, 1992.
27. A. Ward. *Sensor-driven Computing*. PhD thesis, University of Cambridge, august 1998.