OULUN YLIOPISTO
UNIVERSITY of OULU

DEGREE PROGRMME IN ELETRCIAL ENGINEERING

THESIS WORK

AUDIO-BASED DEVICE DISCOVERY AND PAIRING IN SMART URBAN SPACES

Author Pauli Marjakangas

Reviewer Hannu Kukka

Second reviewer Timo Rahkonen

(Technical supervisor Hannu Kukka)

April 2015

Marjakangas P. (2015) Audio Based Device Discovery and Pairing in Smart Urban Spaces. University of Oulu, Department of Electrical Engineering, Degree Programme in Electrical Engineering Master's Thesis, 80 p.

ABSTRACT

In this thesis we studied plausibility of using audio to discover and pair smart devices in smart urban spaces. We created SONDI, an audio-based device discovery and pairing system using pre-existing capabilities of smart devices like phones and tablets. Special emphasis was placed to make system as unobtrusive as possible to human hearing.

We designed and tested two different audio signature pairing methods and evaluated basic properties of audio signals in pairing process. We tested in the methods in variety of expected use environments and cases. Using this new found information we created a demonstration system for proximity passed secure pairing procedure with all necessary components.

System is based on smart phone application which detects and identifies unique audio signatures on near real-time without interfering with normal usage of the smart device. We also created all necessary server components to collect data from our demo setup. Based on our findings we also present several suggestions how to further develop the system.

Key words: Audio, demo, discovery, real time, pairing, smart device, secure.

Marjakangas P. (2015) Audio perusteinen laiteiden tunnistus ja yhdistäminen urbaanissa äly-ympäristössä. Oulun yliopisto, sähkötekniikan osasto, sähkötekniikan koulutusohjelma. Diplomityö, 80 s.

TIIVISTELMÄ

Tämän diplomityön tarkoitus on tutkia äänen käyttämistä erilaisten älylaiteiden löytämiseen ja yhdistämiseen käyttäen hyväkseen jo olemassa olevia älypuhelimien ominaisuuksia. Tarkoitus on saada älylaitteet automaattisesti ehdottamaan yhteyden muodostamista laiteiden ollessa näköetäisyydellä toisistaan häiritsemättä ihmisten normaalia toimintaa.

Suunnittelimme ja testasimme kaksi erilaista signaalintunnistusmetodia sekä tutkimme äänisignaalien perusominaisuuksia laitteiden löytämiseen ja yhdistämiseen. Testaus suoritettiin useissa odotettavissa olevissa käyttöympäristöissä ja käyttöskenaariossa. Kerätyn aineiston perusteella rakensimme demonstraatiosysteemin laiteiden läheisyyteen perustuvalle suojatulle yhdistämiselle.

Tämä demosysteemi on rakennettu älypuhelinsovelluksen ympärille ja se tunnistaa äänisignaalin lähes reaaliajassa häiritsemättä puhelimen normaalia käyttöä. Loimme myös kaikki tarvittavat palvelinkomponentit datan keräystä varten. Esittelemme myös testitulokset jotka pyrkivät selventämään luodun systeemin toimintaa ja todistavat systeemin toimivan myös käytännössä. Viimeiseksi esittelemme myös kehitysehdotuksia.

Avainsanat: Demo, löytäminen, reaali-aika, testaus, suojattu, yhdistäminen, älylaite, ääni.

TABLE OF CONTENTS

ABSTRACT
TIIVISTELMÄ
TABLE OF CONTENTS
ACKNOWLEDGEMENTS
ABBREAVIATIONS
1. INTRODUCTION 9
    1.1. Motivation 9
    1.2. Scope and objective 9
    1.3. Structure of the thesis 10
2. RELATED WORK 12
    2.1. Device Pairing 12
    2.2. Audio as a Medium of Communication 13
    2.3. Pairing Using Audio 13
    2.4. Location Aware Systems 14
    2.5. Audio Watermarking 15
3. CONCEPTUAL DESIGN 16
    3.1. Audio as Channel 16
        3.1.1. Audio versus Radio 16
        3.1.2. Human Hearing and Limitations of Equipment 17
        3.1.3. Audio Signature Parameters 18
        3.1.4. Doppler Effect 19
        3.1.5. Transmission equipment 20
    3.2. Audio Format 21
    3.3. Audio Signal Detection Methods 22
        3.3.1. Cross-correlation 22
        3.3.2. Frequency Transformation 22
        3.3.3. Binary Transmission and Error Correction 23
        3.3.4. Fourier Transformation 23
    3.4. Simple Majority Error Checking 24
4. SYSTEM DESIGN 26
    4.1. System Design 26
    4.2. Goals 27
    4.3. Signal Design 27
5. IMPLEMENTATION 31
    5.1. System Setup 31
    5.2. The Mobile Application: Audio Encoding 32
    5.3. Error Checking 33
    5.4. Loudspeaker System 34
    5.5. Messaging 34
    5.6. Signal Detection Methods 37

5.6.1. Cross-correlation....38
5.6.2. Frequency Transformation....39
5.6.3. Doppler Effect Compensation....41
5.6.4. Signal Encoding....41
5.7. Demo Implementation....41
5.7.1. Communications and User Interface....42
5.7.2. Application UI....43
5.7.3. Display UI....44
5.8. Server Side....45
6. EVALUTION....46
6.1. Initial Stationary Testing....47
6.1.1. Cross-correlation Testing....48
6.1.2. Frequency Transformation Testing....49
6.2. Non Stationary Testing....49
6.2.1. Frequency Transformation Testing....49
6.2.2. Cross-correlation Testing....49
6.3. Obstructed Receiver....50
6.4. Demo Client Power Consumption Testing....50
7. RESULTS....52
7.1. Effects of Distance and Transmission Power....52
7.1.1. Cross-correlation....52
7.1.2. Frequency Transformation....53
7.2. Movement and Barriers....54
7.2.1. Cross-Correlation....54
7.2.2. Frequency Transformation....55
7.3. Error Correction Codes and Detection....57
7.4. Outdoors....58
7.5. Power Usage....59
8. DISCUSSION....60
8.1. Testing Process....60
8.1.1. Effects of Distance....60
8.1.2. Effects of Environment....61
8.2. Method Comparison....63
8.2.1. Cross-correlation problems....63
8.2.2. Frequency Transformation problems....64
8.3. Audio....65
8.4. Other Problems and Improvements....66
8.4.1. Location Awareness....66
8.4.2. Performance....66
8.5. Future Work....67
8.5.1. Platform....67
8.5.2. Security....68
8.5.3. Alternative Setups....69

9. CONCLUSIONS 70
10. REFERENCES 71
11. APPENDIX 75

ACKNOWLEDGEMENTS

This thesis has been made at MediaTeam research group, University of Oulu, Finland. I would like to thank MediaTeam for opportunity this and support during the long processes special thanks staff of the team. My special thanks go to all of those who provided input and advice during this whole process.

Oulu 7.4.2015

Pauli Marjakangas

ABBREAVIATIONS

AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
DAW	Digital Audio Watermarking
DFT	Discrete Fourier Transformation
DSP	Digital Signal Processing
FEC	Forward Error Correction
FFT	Fast Fourier Transformation
FIFO	First In First Out
GPS	Global Positioning System
ID	Identification
IFFT	Inverse Fast Fourier Transformation
IP	Internet Protocol
JSON	Java Script Object Notation
MP3	MPEG-1 or MPEG-2 Audio Layer III
MySQL	My Structured Query Language
NFC	Near Field Communication
PCM	Pulse-code Modulation
UI	User interface
WAV	Waveform Audio File Format
WLAN	Wireless Local Area Network

1. INTRODUCTION

1.1. Motivation

With regards to device discovery and pairing, there are two prevalent problems: how to make smart devices aware of their surroundings (in a non-invasive way) and, secondly, how to securely pair these devices together. This clearly illustrates the need for quick and proactive location-based pairing methods, specifically when the devices are in close proximity of each other. Further, the ever-increasing amount of mobile devices, coupled with the dwindling channel capacity in radio band has created the need for alternative communication methods.

Using sound waves as communications medium has raised interest especially with mobile platforms, as everything needed to utilize this channel already exists in modern mobile phones. Sound waves also have certain distinct advantages, such as directionality (when using a directional speaker), and easily limitable range. When combined with a directional speaker, we can determine when the user is in front of a transmitter, and hence establish line of sight.

In this work we wanted to establish and test a novel way of human machine interaction in device discovery and pairing by using inaudible sound. One requirement for the system is that it must run on existing technology, and as such be easily adaptable to off-the-shelf modern smart devices like phones and tablets. We are also interested how well such system would work in both indoors and outdoors.

1.2. Scope and objective

This thesis presents a novel system for device discovery and pairing using so-called audio signatures. The goal of this work was to create a system capable of alerting people to the presence of fixed interactive devices in smart environments. The system should allow mobile clients equipped with a client capable of listening to "audio signatures" to serendipitously encounter fixed devices broadcasting identifying information using high-frequency audio, and proactively propose pairing without requiring pre-meditated action from the user, thus simplifying interaction and offering new kinds of services.

To meet these requirements, a system was designed to detect and pair various types of devices that are in close physical proximity. The position of the user (or line of sight) is important in cases such as public displays, which naturally are only usable if the user can see the screen. Such positioning is difficult to realize with undirected signals such as, for example, Bluetooth.

The basic premise of the system is presented in Figure 1. A fixed device broadcasts a unique audio signature to its immediate vicinity on a frequency that is inaudible to the human hearing using a directional loudspeaker. A mobile device listens to these signatures and, when a signature is detected, can determine that the user is close enough to a particular target device for pairing to make sense; and that the user is in front of the device.

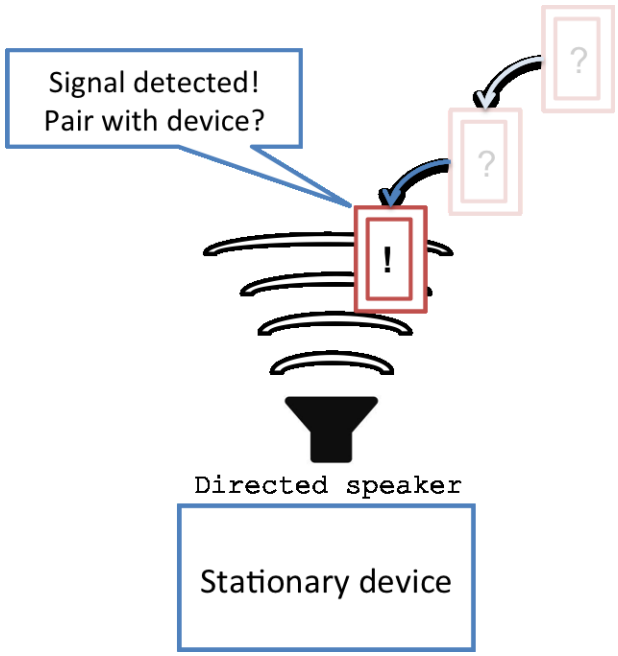


Figure 1. SONDI concept.

After the mobile client recognizes a signature, the user is notified via tactile or audio feedback that a device supporting pairing is close by, and the user can then decide whether or not he or she wants to go ahead with the pairing. Since secure pairing is one of the major current problems in the communications field [53], we also wanted to create setup that would demonstrate secure pairing using this implementation.

Modern smartphones were selected as the implementation platform, as they fulfill all of the requirements presented above. The resulting system is called SONDI. The main goal of this thesis is to establish whether the audio signatures broadcasted by a SONDI-enabled device could be used in human machine interaction in noisy environments such as common urban settings both indoors and outdoors. Further, as mobile devices are typically carried inside a piece of clothing, *e.g.* in a pocket or purse, we will look at how such common barriers to sound affect the signature detection and recognition.

1.3. Structure of the thesis

First chapter is introduction where we shortly go through scope and objective of this thesis. In chapter two, we look at related work which also forms basis to future chapters. Chapter three is conceptual design where we take closer look to our problem and how to solve it in general level and difficulties associated with it. Chapter four presents our design on basic level and delves deeper into design of our work. Chapter five is the implementation of designs presented in previous chapter and demonstration setup done as proof of concept. This is followed by chapter six where we go over our evaluation methods before going into results at chapter seven which presents our testing data. Then at chapter eight we discuss briefly about test results effects revealed in testing. We present our conclusions in chapter nine where we also discuss shortly about possible future work and what we would have done

differently or what could be improved and how to best to take advantage of this work. We also lists several suggestions which could be used further develop the system.

2. RELATED WORK

2.1. Device Pairing

Device pairing is a process used in computer networking that helps to set up an initial link between two devices, with the purpose of establishing a communication channel between them. Device pairing can be divided into two categories: *secure* and *non-secure* [43]. In non-secure pairing, two devices are connected when a willing candidate is found. However, the lack of prior security context and common trust infrastructure opens the door for Man-in-the-Middle (also known as Evil Twin) attacks [28]. Thus, secure device pairing is the process of bootstrapping a secure channel between two previously unassociated devices over a (usually wireless) human-imperceptible communication channel [52.

This exchange of contact information requires some sort of initiator. For exchanging contact information visual approaches have been quite popular, including 1D barcodes [30], 2D barcodes [34], and so-called fiducial markers [25], [45]. Proposed non-visual methods have included RFID tags [24] and Bluetooth [26]. These other solutions include using NFC and touching the phones together to initiate pairing, such as found in the Samsung Galaxy S3 device among others, or using external apps such as the now-discontinued Bump application to initiate pairing through physical proximity.

However, devices equipped with RFID/NFC readers are still somewhat rare. Further, pairing devices with audio requires fewer steps than e.g. scanning a visual code, and audio does not suffer from real-world constraints such as poor lighting or glare, although it has its own set of problems: for example, very loud noise may naturally mask the audio signal. The common denominator with these methods is their limited range: For example, NFC range is limited to few centimeters (this however is intentional since it helps to prevent eavesdropping and man-in-the-middle-attacks [21]).

Device pairing methods can further be divided into *proactive* and *reactive* methods based on user initiative. In the former, the pairing procedure is initiated by an entity other than the user's personal mobile device. As an example, Bluetooth hotspots can be used to continuously scan the environment for other Bluetooth enabled devices, and push information to mobile clients when they are in range. This approach has been used for e.g. Bluetooth-based advertisement (see for example [1], [46]), or topical information delivery [27]. Reactive pairing approaches, on the other hand, require the user the initiate pairing and subsequent data exchange by e.g. scanning a visual code (as used in for example [2]), or performing some other physical activity such as "scratching" a printed barcode, as described in [19].

A difficulty with reactive approaches, such as visual codes, is that they require quite many steps from the user. The user has to first become aware that he or she is in the vicinity of a device that supports pairing; become motivated enough to approach the device and initiate the procedure; often times launch a mobile application such as a barcode reader; and scan the presented visual code, after which pairing can be initiated. Previous research has also found that people may get confused about how to interact with a visual code, and for example try to click on the code instead of scanning it with a mobile device [37]. Proactive approaches, such as Bluetooth, on the other hand can suffer from long latency in device discovery, and

users may also get confused as they have no way of knowing when they are within range of the Bluetooth transmitter [27].

Prior research also yielded a number of other interesting methods utilizing various auxiliary human-perceptible channels, e.g., visual, acoustic or tactile. These methods engage the user in authenticating information exchanged over human-imperceptible channels, thus mitigating Man in the middle attacks and forming the basis for secure pairing. Various methods of secure device pairing were studied in "Caveat eptor: A comparative study of secure device pairing methods" [28] which is excellent baseline for performance comparison.

2.2. Audio as a Medium of Communication

Besides the obvious example of speech as a means of communication, there are several other historical and technical examples of using audio as a medium. The simplest form of non-verbal use of audio for communication is the use of different kinds of signal sounds like bells, drums and whistles. Morse code can also be regarded as example of using audio as medium. Electrically transmitted Morse code was typically decoded by human listening audio cues. In the field of human-to-machine and machine-to-human communication, audio has been a “hot topic” for years [10], as spoken language is a very natural form of human communication [23].

However, for machine-to-machine communication, audio has not been very popular. One example is the PhotoCircle app[32], which is very similar to our work but focuses on short range communication. PhotoCircle app bootstraps connection between two smartphones by exchanging information in form of token over audio channel in audible to humans. Another example of this is Google Chromecast guest mode [15] where 4-digit PIN is transmitted using short, inaudible audio tone.

One field where audio has been used extensively as communication medium is underwater communication [48] where sound waves are used in long range communication between submerged objects and/or ground stations. Unfortunately air and water are very different mediums [50], this means we cannot directly use same techniques. We can learn great deal about air as medium like spatial variation in the sound field from biological studies to animal sounds [57].

2.3. Pairing Using Audio

There are two options for using sound waves in device pairing: one is to use audio as a means of communication (*e.g. previously mentioned underwater communication* [48]); the other one is to use audio as an initiator (*e.g. bee-bee method* [28]). The typical *modus operandi* is to utilize the audio channel to exchange public keys, similarly to normal secure exchange. An early example of using audio as a method of pairing is phone trunk lines, where frequency tones were used to control a telephone switch ([56] and [6]). This is known as in-band signaling, which is sending metadata and control information within same band or channel as voice. Problems with security (blueboxing) forced phone companies to adopt out-of-band communication methods for phone lines.

Goodrich et al. [14] used the audio channel to securely exchange public keys between devices by either having the devices present human readable or human-understandable sentences in either written or spoken form. They also built another

implementation with no human intervention where one device broadcasts its public key encoded as audio, and the other decodes and interprets the audio to locate the key. Audio has also been used as an interface. For example, Harrison et al. [19] created various patterned objects that could be "scratched" with other objects such as mobile phones, pens, or fingernails and decoded the resulting sounds into control signals for computer systems. A user could, for instance, control an iPod Touch by scratching these "acoustic barcodes" with it.

Another system, called Skinput [18], used an acoustic-sensing armband to detect and localize finger taps on the skin. Similarly, TapSense [17] and Sonically Enhanced Touch [35] used microphones attached to interactive surfaces to detect and decode taps made with different objects or parts of the hand. Dearman and Truong [12] presented a system called BlueTone, which used dual-tone multi-frequency sounds transmitted by pressing the keys on a mobile phone carried over a Bluetooth connection to interact with an interactive public display. BlueTone enabled simple interaction tasks such as text entry cursor manipulation or menu selection without requiring additional software to be installed on the mobile device.

2.4. Location Aware Systems

Audio has been previously used for several prototypes and concepts in both mobile and ubiquitous computing research. There are several existing technologies that can be used to realize a location based proactive pairing. Audio signals have been used to detect location based on ambient noise [9], or based on multiple reference sources like in DOLPHIN [13]. However, such systems can be difficult to set up and maintain. One design goal of SONDI is to offer a simple solution for determining user orientation in relation to display by utilizing a directional speaker.

Pairing with a stationary device could be accomplished by simply making the mobile device location aware, and matching devices based on proximity. The most typical way to create systems that are location aware is GPS, but utilizing a satellite-based positioning system has its own limitations. The standard GPS accuracy is approximately 15m, but can be enhanced to 5m [38]. However, in indoors and urban areas the accuracy is typically less than 15m, and especially in areas with tall buildings, GPS suffers from dead spots [38].

Instead of using a global positioning system (GPS), location aware systems can be built in local scale. WLAN is a typical choice, as it has a longer range than Bluetooth and most smart devices automatically search for available WLAN networks. We can approximate the distance from a WLAN-device through signal strength but, as with Bluetooth, we cannot derive orientation information (*e.g.* whether the user is behind or in front of the device). As such, a WLAN-based system cannot guarantee line of sight. WLAN networks can also get overloaded during peak hours.

Smith et al. [47] used ultrasonic pulses coupled with wireless RF signals for indoor positioning of moving mobile devices. Similarly, the Active Bat location system presented by Harter et al. [20] utilized ultrasonic "chirps" and small sensor tags carried by the users to track their location in indoor spaces. The WHISPER system presented by Vallidis [54] used a spread-spectrum audio approach to obtain precise distance measurements by encoding information on an audio stream and using time-of-arrival information to obtain distance estimates, and Peng et al. [41] used acoustic "beeps" for measuring the distance between two mobile devices.

2.5. Audio Watermarking

Another relevant usage for audio channel is data transfer. Relevant example of this is audio watermarking [51]. Digital audio watermarking (DAW) has been used to add meta-information, such as copyright, to audio recordings, and similar methods can be used to add information into audio played from speakers. DAW systems are similar to the SONDI concept because in both, a digital signal is hidden and subsequently retrieved using smart device. One example of where smart phone has been used to decode information with (DAW) is in [36]. Difference is, in DAW the signal is hidden in another piece of audio, whereas SONDI signatures do not have to hide the signal except to make it inaudible to the human ear, while still keeping it distinguishable from the ambient background noise.

The basic premise of signal generation and retrieval process is similar in both cases, and we can use or at least learn from several methods that have been proposed previously in DAW literature. For example Boney et al [4] presented a spread-spectrum approach for audio watermarking using a pseudo-random sequence that is filtered in in several stages in order to exploit the long and short-term masking effects of the human auditory system in making the watermark inaudible. Similarly, Bassia and Pitas [3] applied a straightforward time-domain spread-spectrum watermarking technique to audio signals, and Lee and Ho [33] utilized inversed Fourier transformations to hide a narrow-band watermark into the ceptsral components of the audio signal using a technique analogous to spread-spectrum communications.

Löytynoja et al. [36] presented a method for embedding a digital watermark into radio broadcast music. The user could record a segment of the music on his/her mobile phone, and the extracted watermark could contain value-added information such as metadata about the song or a link to a site where the song could be bought. While robust against various attacks, their method managed to reach an 83% recognition rate on the mobile phone, while recording from a few centimeters away from the audio source.

3. CONCEPTUAL DESIGN

In this chapter we will look at audio based device discovery and pairing in conceptual level and present basic facts about audio as medium related to our work. We will also take a short look at the requirements at the general level and present few possible setups for physical devices. Last we present theory behind two possible different signal detection methods and short run down on involved techniques.

Physically the device ensemble can consist of all mobile clients (i.e. two or more mobile phones), or one or more mobile client(s) and a stationary end-point such as an interactive public display. Third option would be all stationary devices but we are interested pairing mobile systems. There are several ways which we can establish this device pairing. The most simplistic form of pairing for mobile clients is using phone numbers, for example sending a picture to another person as a multimedia message. Other typical used technique is to use Bluetooth to form a connection (Bluetooth keyboards).

With Smartphone system requirements can be divided into following sub problems: Communication, power consumption, pairing and signal detection. Communication is meant to be context free as possible. We also aimed for proactive pairing since one of the big unsolved problems with systems like this is how to get user motivated enough to use the services. Phones also have extensive array of sensors in addition to audio in them [29] which we can also use for support.

As previous studies show one of big problems with systems like this is power consumption. Generally with smartphones there are three big power consumers: GPS, WLAN, and Bluetooth [8]. Thus we should at least try to minimize power consumption of these components by only using them, when really needed. Also as supplement to this scheme and to add additional features

We designed the system to use combination of radio and audio signals to enhance location awareness. Idea is have to overlapping methods to support each other to increase reliability and security. Also using combination techniques, we can active application near hotspot and just briefly check presence of audio barcode thus saving battery.

3.1. Audio as Channel

Line of sight has two meanings. Signal line of sight meaning that signal has unobstructed path between transmitter and receiver. Another visual one is line of sight between user and target object with audio these two are closely related this means we can take advantage this effect. We can limit how and when signal received and if we then combine transmitter and target device can create system which can deduce when user has line of sight to device.

3.1.1. Audio versus Radio

Audio as channel is similar to the radio channel in that many of the same techniques can be used on both channels. The two main differences are the usable frequency range, and signal speed. Speed of sound at 20 Celsius at sea level is about 340 m/s, while the speed of light is 299 792 458 m/s. This has some notable effect to our system, the most notable one being the Doppler compensation. Producing low or

high frequency sound waves is more difficult than creating similar radio waves, which by necessity dictates a more limited frequency range.

Unlike most of radio band, audio band is unlicensed. What this means is, that anything and anyone can utilize it without permission. There are some restrictions and exceptions that must be considered, most importantly health and safety and noise regulations. Foremost, we want to avoid any damaging effects, but in addition to those sound waves can possess irritating qualities [49]. These must be avoided if the system is to be used outside of a laboratory environment, which mandates the used frequency range to be outside of the normal human hearing range.

3.1.2. Human Hearing and Limitations of Equipment

Effects of high frequency sounds on humans has been researched surprisingly little. One study, titled "Damage to human hearing by airborne sound of very high frequency or ultrasonic frequency" [31] recommended keeping sound levels below 85 dB for high frequencies to avoid side effects. Further, many animals have different hearing range to humans and may be affected by high frequency noises.

Since we are dealing with audio signals it important to make sure that we keep signals levels in reasonable level as even though humans cannot hear the signal it still can damage hearing. Usually in similar cases signal levels have not been problem since communication has communication has been at touch distance (pairing) or very short in duration (distance measurement) but as we want to use SONDI system continuously and in distances up to 5m. With this specifications signal levels can grow into problematic levels.

There are three things which affect our chosen frequency audio range: Human hearing, recording equipment, and playback equipment. Typical human hearing range is between 20Hz and 20 kHz [39]. We wanted to create a system, which would be non-intrusive to human senses. To accomplish this, frequency bands were chosen outside of typical hearing range. Human hearing range is also reason why most mobile phones are optimized to use that range (or lower range cut the cost of unit). This also means most mobile phones have sampling rate of 44.1 kHz.

The sampling theorem is a fundamental bridge between continuous signals (analog domain) and discrete signals (digital domain). Time continuous signal can be perfectly reconstructed from samples if sampling rate is twice the highest frequency in the sample as represented in Formula 1. Using typical smart phone audio sampling rate on formula 1 (also known as Nyquist-Shannon sampling theorem) we get the Nyquist frequency, which is the highest frequency that can be reconstructed from samples.

$$Fs \geq 2f_{max} \tag{1}$$

This gives a usable frequency range of 0-22.05kHz. The requirement to use a frequency range that is not audible to humans limits the choice of frequency bands to two options. First, the lower range, which is below human hearing range is from 0 Hz to 20 Hz. The second option is to use the range which is higher than what a human can hear, *i.e.* from 20 kHz to 22.05 kHz.

Modern smart phones have a nominal frequency range 0-22.05 kHz. In reality, however, the frequency response of a given microphone is more complicated, as shown in Figure 2. The frequency response is variable, and the highest frequencies

are weakened. Of course, the quality of phone recording equipment varies considerably between models, but most high-end phones can record cd-quality sound with little distortion in the chosen band. The high-end range frequency band has more leeway than the low-end one, because most people cannot hear the upper 20 kHz range. This means that a frequency range of 19 kHz - 20 kHz can be utilized without too much of disturbance to humans.

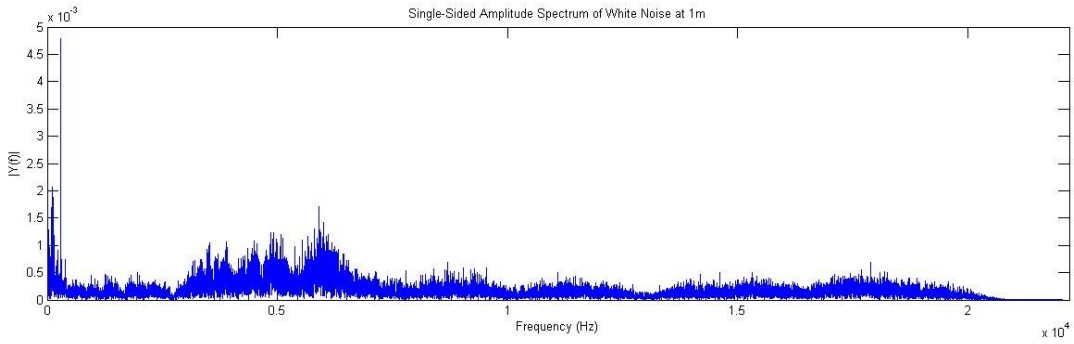


Figure 2. Frequency response of Nexus microphone.

3.1.3. Audio Signature Parameters

Most basic fact we must know from audio waves is that they lose power proportional to distance squared as shown in equation 2. We know that doubling amount bands halves energy of each band. From this information we can create a table to compare different parameters. Amplification from the transmitter is not taken into consideration, as we are only interested in the limits of signals creation and the framework. If needed, amplification can easily be adjusted later. Individual band energy is estimate from MATLAB simulation.

$$P \sim \frac{1}{r^2} \tag{2}$$

In Table 1, the first column is number of FFT bits needed for calculating sample (also our window size in bits). The next two columns are the number of samples and sample length in seconds. After that is frequency resolution of FFT calculated with Formula 1 in practice this means that with our sampling rate of 44100 Hz, shortest possible length sample is around 2^{10} which has frequency resolution of 43.03 Hz/bin. Fifth column is maximum number of bands with 1 kHz bandwidth with that resolution. Last two columns are energy of individual bands of Fourier transformed signal (assuming maximum number of bands is sent) when sent and when received at 5 meter mark calculated using Formula 2.

Table 1. Samples, frequency resolution and band energy.

Bit	Number of samples	Sample length in seconds	Frequency resolution (Hz/bin)	Maximum number of bands with 1 kHz band width	Individual band energy	Individual band energy at 5 m mark
8	256	0,005805	172,265625	5,804989	0,05168	0,002067188

9	512	0,01161	86,1328125	11,60998	0,02584	0,001033594
10	1024	0,02322	43,06640625	23,21995	0,01292	0,000516797
11	2048	0,04644	21,53320313	46,43991	0,00646	0,000258398
12	4096	0,09288	10,76660156	92,87982	0,00323	0,000129199
13	8192	0,18576	5,383300781	185,7596	0,001615	$6,45996*10^{-5}$
14	16384	0,371519	2,691650391	371,5193	0,000807	$3,22998*10^{-5}$
15	32768	0,743039	1,345825195	743,0385	0,000404	$1,61499*10^{-5}$
16	65536	1,486077	0,672912598	1486,077	0,000202	$8,07495*10^{-6}$
17	131072	2,972154	0,336456299	2972,154	0,000101	$4,03748*10^{-6}$

3.1.4. Doppler Effect

Doppler Effect comes into play when either or both transmitter or receiver is moving. In our system Doppler effects comes from moving receiver and unlike with radio waves where high speed of signal compared to velocity of receiver makes effect trivial in walking speeds, effect with audio is noticeable. Quick calculation using Formula 3 showed with baseband of 19 kHz and average human walking speed of 1.388 m/s, variation of sent and received frequencies values would be about +/- 80 Hz. We can also see that frequencies increasing while walking towards the source and decreasing while walking away (Formulas 4 and 5). For example if the sent frequency was 19160 Hz depending on whenever we are walking towards or away from source and speed of movement received frequency could be anything from 19081 Hz to 19238 Hz.

Base formula:

$$f = \left(\frac{c + v_r}{c + v_s}\right) f_0 \tag{3}$$

Receiver is moving towards the source.

$$f = \left(\frac{c + v_r}{c}\right) f_0 \tag{4}$$

Receiver is moving away from the source.

$$f = \left(\frac{c - v_r}{c}\right) f_0 \tag{5}$$

Where:

- c is the velocity of waves in the medium; in our case its speed of sound at sea level which is 340.29 m/s.
- v_r is the velocity of the receiver relative to the medium; positive if the receiver is moving towards the source (and negative in the other direction); (average human walking speed is 1.388 m/s)

- v_s is the velocity of the source relative to the medium is which positive if the source is moving away from the receiver and negative if it is moving towards the receiver in our case source will always be stationary (0m/s).
- f_0 is original frequency
- f is Doppler shifted frequency

Problems are the following: in order to correct frequency shift we need to know speed of user and direction of movement in relation to source to estimate Doppler shift or we have to have some sort of reference point like pilot signal to estimate the Doppler shift. As mentioned before we could use GPS, WLAN or Bluetooth to calculate our speed and direction but relying too much on these technologies will lead to heavier battery consumption and often they are not available or do not have the necessary resolution. We also have some other options for example new Nexus 5 has inbuilt support for step counter which could be used. We decided to use pilot signal and use one frequency as reference to calculate Doppler shift. Also we have to understand even after correction there are slight variations in Doppler shift while moving around the transmitter: For example Doppler Effect will only be uniform if user follows circle at static distance from source. But this is not realistic estimation of human behavior.

3.1.5. Transmission equipment

Fixed devices in the environment require certain equipment to be able to broadcast the audio signatures into the surrounding space. Most importantly, the fixed device must be equipped with an amplifier and a loudspeaker capable of transmitting the frequencies used by SONDI. Due to the limitations of human hearing discussed above, most loudspeakers have a frequency range well below 20 kHz.

Typical directional speakers have a frequency range of 150 Hz-16 kHz, loudspeakers with higher or lower frequency range being more expensive. Both directional subwoofers and directional tweeter loudspeakers are available. Typical consumer grade subwoofers can produce sounds in the 20-200 Hz range. However, achieving directionality with subwoofer elements requires the speakers to be arranged in large arrays of multiple elements. Such arrangements are impractical for our purposes. This leaves us with tweeter speakers, which are more manageable size even in directional versions. The availability and practicality of tweeter speakers point clearly to favoring the higher (19 kHz-22.05 kHz) range in building the SONDI audio signatures.

Delving deeper into the importance of directionality, we see that a typical loudspeaker is best described as a point source, from which audio waves typically spread in a longitudinal fashion spreading energy equally in all directions. From the systems perspective, this means that a user taking a straight path through the center of the cone will look like she or he first moved towards the audio source at a slow speed, and then away again also microphone receivers are far from ideal as seen from Figure 2. Factors like this will cause errors in received signal.

While moving past the transmitter, signature detection chance depends on time spent at the area covered by the audio cone (width of the cone at a given distance depends on the directionality of the loudspeaker, see Figure 3). A 45 degree angle cone is presented by green color and measures number of sample while user walks

past loudspeaker in 90 degree angle at set distance (from one end of the cone to another) and orange colored cone is 30 degree angle with same setup. Amplification should be used while transmitting a signal, as it allows us to utilize signals with lower individual band energies. Channel effects can be negated by buffing certain frequencies that are known to be affected by channel, but this would require channel models.

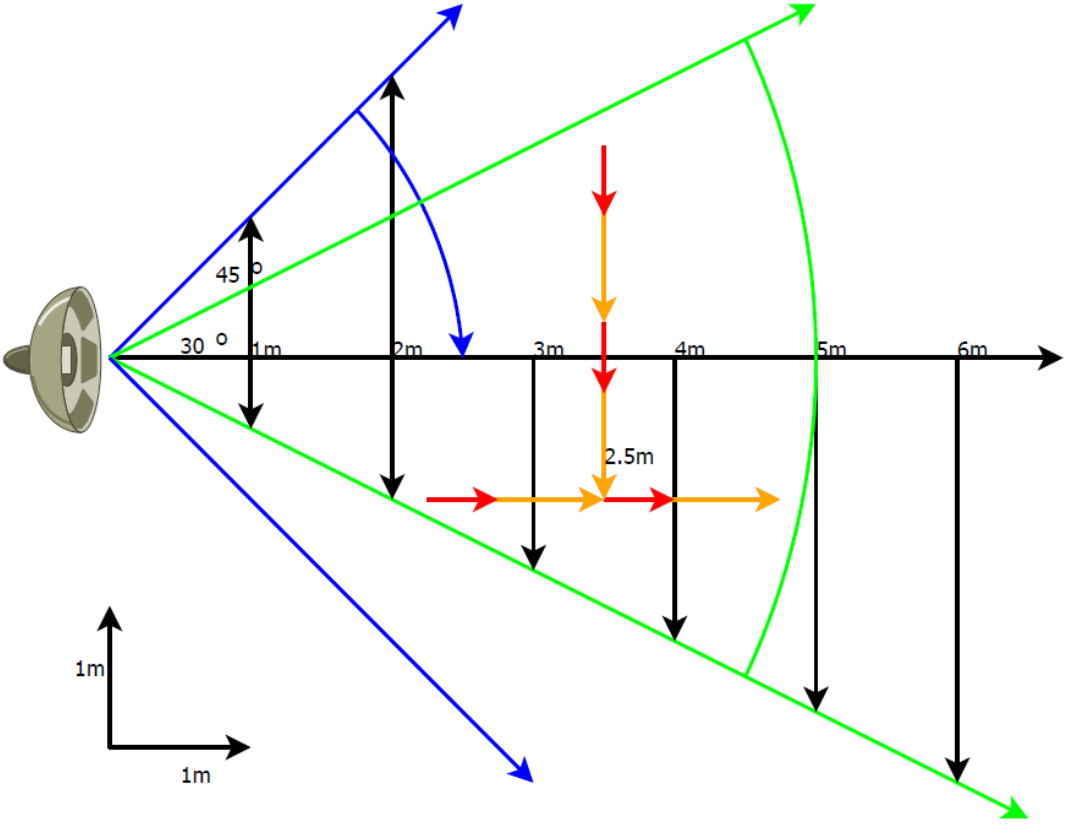


Figure 3. Chances to detect signals at angles of 90, 30 and 45 at walking speeds.

3.2. Audio Format

As one of the requirements is that a commercial speaker system should be used to transmit signal, different audio formats and playback devices (media players) have to be carefully considered. Audio format selection is limited by the capabilities of mobile phones, whereas computer devices support nearly all audio formats. The playback software imposes further limitations on the selected audio format, as for instance web-based implementations may not be able to support all available formats.

As all audio formats are not supported by all mobile devices, the selection should favor those formats that are most common. One of the most supported formats for cross-platform audio is MP3, which has the required sampling rate of 44100 Hz. However, MP3 is a lossy data compression format and, more importantly, it achieves the compression by reducing accuracy of frequencies that are on the fringes of human hearing. MP3 compression employs perceptual coding, an approach based on psychoacoustic models that permit the codec to discard or reduce the precision of

audio components that are less audible to human hearing [5]. Unfortunately, these are the frequencies that are most relevant to an implementation utilizing inaudible signals, and hence MP3 has to be ruled out both for signal transmission and data recovery.

3.3. Audio Signal Detection Methods

When detecting and decoding audio signatures such as the ones discussed in this thesis, it is important to understand how different detection methods work. We have two competing designs each based on FFT. Frequency transformation is simple method of examining frequency components of signal. Cross-correlation has been used in the past to recognize objects on pictures and measure similarity between signals [7]. It has also been used in DAW for watermark detection [16]. Modulated signal was also considered but ultimately it was decided against it. Problems rise from the fact that modulation and demodulation are complex systems and take time to create and perform. Most importantly it was not our goal use the SONDI system for actual data transferring but as quick way of establishing physical presence of devices. Modulation could be useful for efficient usage of limited signal bandwidth signal bandwidth and adding error resistance to transmission.

3.3.1. Cross-correlation

In cross-correlation, once a sample has been recorded by the phone, the mobile client uses cross-correlation to detect signatures from the audio recording. The recorded sample is correlated with the reference (target) signal, and the maximum peak is located. This maximum peak is concluded as the location of a signature if its cross-correlation value is significantly larger than that of background noise. Effectively, cross-correlation measures similarity of two waveforms shifted with time t (lag). Our cross-correlation method does not use signal-to-noise ratios to detect signatures, but rather uses a pre-set similarity threshold value against which the peak value is compared. Cross-correlation can be efficiently calculated using FFT.

Cross-correlating a signal with itself is referred to as auto-correlation. There, the peak value is always at zero (shifted with time t=0) and it represents the maximum value of correlation. By calculating cross-correlation we can find the time shift in which signals are most similar (peak location), and similarity (peak value). This means we get nice singular value which presents whole signal similarity this also holds true for periodic signals.

3.3.2. Frequency Transformation

Frequency transformation is method where mobile client uses Fourier transformation to find frequency information form the audio recording. First Doppler correction is applied to frequency search area this done so that correct frequency sequence. This information is then processed in order to find binary sequence which then decoded and the decoded sequence to corresponding source.

Frequency transformation is a method where the signal is transformed from time scale to frequency scale, much like is done with Shazam [55]. Basically, the

frequency range is divided into frequency bands where each band is considered as one bit. When a sample is captured by the receiver, a FFT transformed sample is scanned for active frequencies. In this sample, each active frequency band is considered as a logical 1 one and non-active bands are considered as logical 0. These methods differ in the number of potential unique signatures. Both methods have some merits: Cross-correlation has a wider range of codes (or more precisely signals) than frequency transformation, but it is considerably heavier on calculation and each different signal must be checked individually against the recording to be certain of sample identity. Frequency recognition is lightweight, but has a smaller range of codes which depends on channel band width. However, it can check every possible code at once making it better in multi-signal cases.

3.3.3. Binary Transmission and Error Correction

By using binary transmission instead of sending just an audio signature enables us to use digital signal coding methods to improve error resistance and data capacity, however this option is only available to one frequency transformation. This is the reason to favor it over cross-correlation. Binary also enables wide variety of error correction methods one such method is error correcting codes.

Hamming code is FEC-code (Forward error correcting), FEC-code were chosen due nature of communication (one directional sending). FEC-codes are group of codes which can correct errors in transmission without needing to resend the message. This is done by adding parity bits which add redundant information which can be used to reconstruct damaged transmission (similar to previously mentioned repetition code). Hamming or BCH codes are old and inefficient, but simple to implement.

The basic Hamming code can correct any one bit error and detect some of the two bit errors, but they may be corrected erroneously, as if they were single-bit errors. If extra parity bit is used we could detect all two bit errors. Adding parity bit to hamming code enables us to detect all 2 bit errors however this also further diminishes channel capacity. Quick calculation shows that adding parity would decrease amount of false positives by 7% while decreasing overall all channel performance by 7% (13/14) making the tradeoff in our case practically useless even worse for overall system. Some other hamming codes would also work and were briefly tested but hamming (12 4) (total 12 bits which 4 of them is parity bits) was deemed to be best compromise for our current system.

3.3.4. Fourier Transformation

At the heart of both methods is Fast Fourier Transformation, which is simply a fast way of doing transformation from time domain to frequency domain (aka Fourier analysis). The frequency resolution is dependent on the relationship between the FFT length and the sampling rate of the input signal. Higher sampling rates are not automatically better, as a higher rate means lower frequency resolution for the same FFT size. Frequency resolution df can be calculated with formula 6 where Fs is the input signal's sampling rate and N is the number of FFT points used.

$$df = \frac{F_s}{N} \tag{6}$$

3.4. Simple Majority Error Checking

All error-correcting codes are capable of correcting only a limited amount of errors thus is important to have some sort of error way to check for errors after transmission if we want to have completely error free transmission. With audio-based systems such as the one described here, receivers moving freely in a given space can cause errors in multiple frequency bands at same time, which in turn may result in data words being either falsely received or falsely corrected.

This may lead to false positive results. In order to mitigate this problem, a scheme where consecutive audio samples are also compared to each other was devised: a second recording is taken after the first recognized sample (similar in effect to resending message). If results are identical, the received signal is accepted as correct. Otherwise the detection cycle is continued as normal. This corrects random errors caused by channel, but not systematic errors. There are several ways in which this double-checking can be implemented, each of which affects detection time and error percentage differently.

There are three basic ways of doing simple majority error checking: *two consecutive samples*, *first two out of three samples* and *save last sample*. For the sake of simplicity, we assume that a signal is either *correct* or *false* and if decision "signal is found" is made all existing samples are discarded before next detection cycle. . With *two consecutive samples*, after comparing two samples both samples are discarded. The two consecutive samples method is both the simplest to implement and provides the best proofing against false positives, but it is also potentially the slowest and has the lowest chance of finding a solution. Chance of false positive is always P_{False}^2 while signal detection chance is $P_{Correct}^2$.

The *first two out of three samples* method compares the current sample to the last and previous to last sample, and if either of them matches with the current, the samples are accepted as correct. Two matching samples can have one non-matching sample in between, which guarantees decision after three samples (in a simplified case), which again increases recognition of correct samples by $2 * P_{False}^2 * P_{Correct}$ and amount of false positives by $2* P_{Correct}^2 * P_{False}$. This makes it the best of the three for finding correct results.

The *save last sample* approach does not discard the last sample if the previous conclusion was a mismatch. This saves time in cases where the last sample was correct despite being a mismatch (case of single error in uneven numbered sample). This sample saving increases recognition of correct samples by $P_{Correct}^2 * P_{False}$ and the amount of false positives by $P_{False}^2 * P_{Correct}$. At worst, saving last sample can be equally slow as the *two consecutive samples* method, and it is not guaranteed to arrive to conclusion. However, it does create less false positives than the *two out of three* method, and has more correct results than *two consecutive samples*. Figure 4 shows how many samples are needed for each method for odd and even numbered errors. Red color indicates error in recording and sampling results. Each recording block is one sample and lines below the blocks indicate results of checking and which samples were used. First is *save last sample*, which needs three samples to reach conclusion on odd error and four on even error. *First two out of three samples*

always needs three samples, while *two consecutive samples* needs four. Figure 5 summarizes the effects of different error checking methods to false positives and correct results. Horizontal axel is probability of correct results while vertical axel detection percentage after error checking.

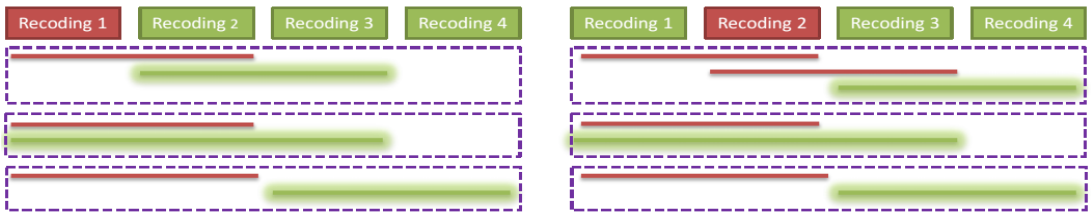


Figure 4. Simple majority error checking methods.

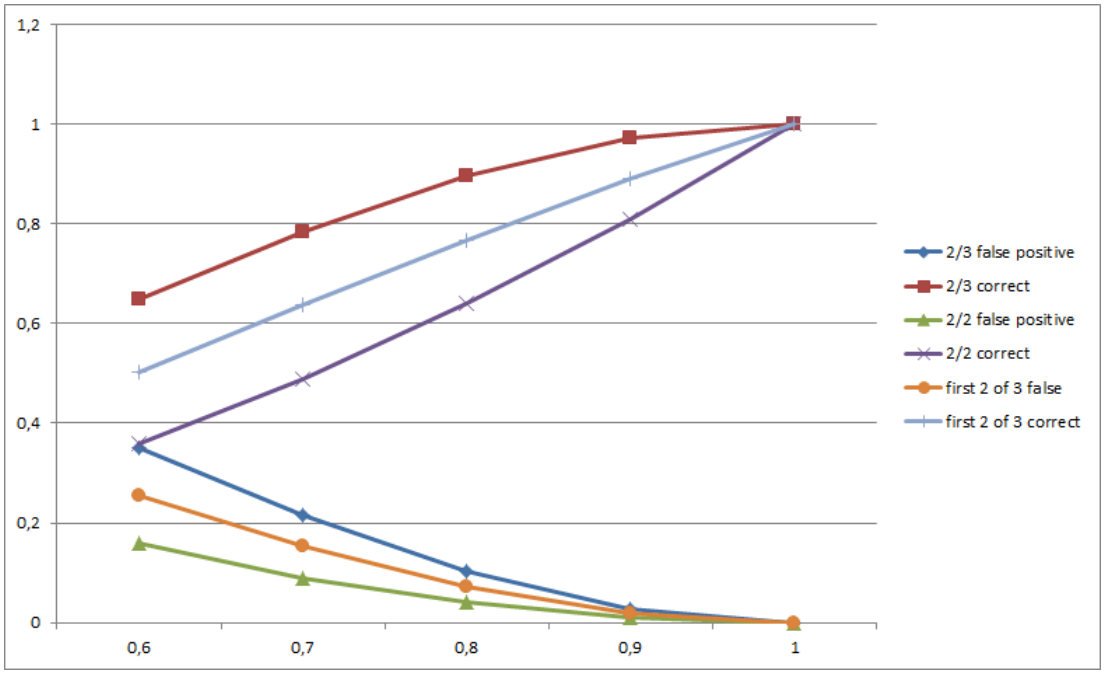


Figure 5. Various error checking methods with different probabilities.

In reality, error mechanics are not this simple. Not all errors have the same probability, so chance of a false positive actually depends on the probability of a particular error. For example, if the total chance of receiving an error is 10%, divided between errors P_a, P_b and P_c; then, for a false positive to occur, two consecutive errors would need to be the same (e.g. two. consecutive A errors) which has probability of $(0.1 - P_b - P_c)$ to power of two. Errors can extend over multiple samples (systematical errors), in which case they are more likely to cause false positive errors. Random error singular or pair-wise, cause the system to abandon results as inconclusive thus increases detection time of the signature.

Since the goal is secure pairing, it is better to discard uncertain samples rather than risk false positives. The best way to compare different error-correction methods is to take into account the associated costs (recovery from false positive, *etc.*), and average time taken after multiple samplings. False positives can be assigned with a time cost, effectively beginning when the user notices and resets the system after a false positive.

4. SYSTEM DESIGN

The conceptual design shown in the previous sections can be realized on actual hardware in several ways. Here, we present one possible implementation with two different possible audio signature detection methods. We selected relative simple setup one or more stationary unique sources of audio signatures and multiple moving receivers and central server. Idea is that we have array of multi-user client units each with unique signature monitored by central server and group of roaming users.

4.1. System Design

The SONDI system can roughly be divided into three main components (shown in Figure 6): *mobile device*, *server* and *audio barcode source*. The source element is attached to a fixed device or object in such a way that we can determine when the mobile device has a signal line of sight to the target object or device. The fixed device can be any smart device in the environment, such as an interactive public display, a push-based information delivery system, or even a printer.

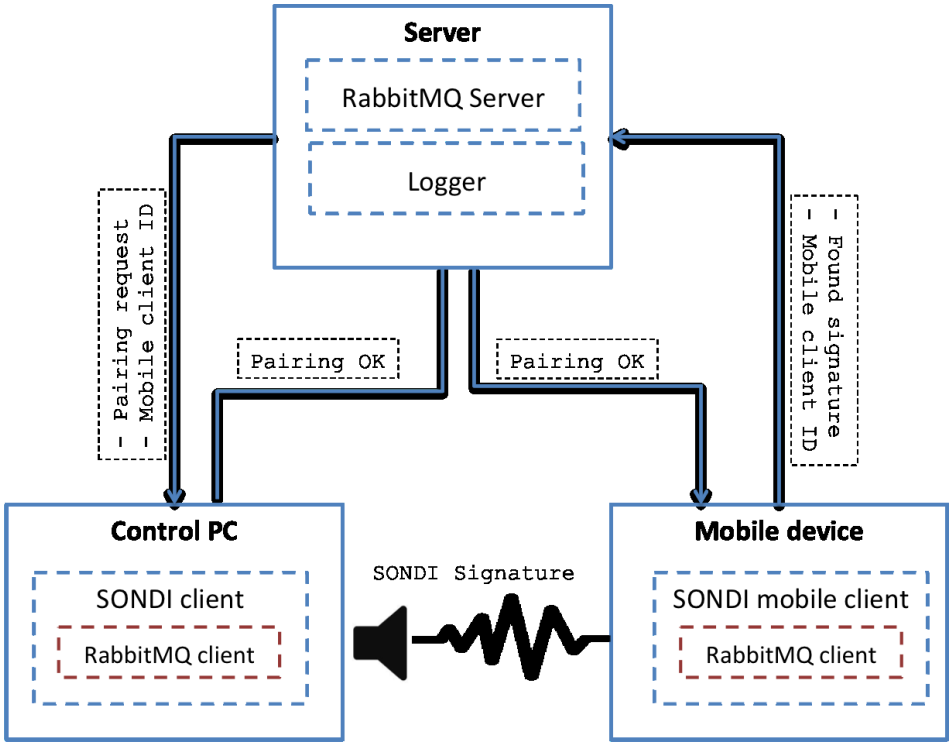


Figure 6. SONDI system component structure.

The server component is responsible for managing communications over the Internet between devices, and also handles logging activities. The server also keeps track of changes in the system. The mobile device runs a dedicated mobile application, and in the current implementation requires a WLAN connection. WLAN is used for rough position estimation, and for Internet connectivity when applicable. An Internet connection is required for communicating with the SONDI server. The system is unidirectional in a sense, meaning that the mobile device listens to the audio signal broadcasted by a SONDI-enabled device. When a signal is recognized, communication is switched to asynchronous messaging system.

In the current implementation, the mobile device must maintain a list of known target device signatures (audio barcodes). It also has to know the address of the SONDI server. This is an important consideration from a design point-of-view, as trust cannot be established unless the user is made aware of the origin of the audio signature his/her mobile device is picking up. A final requirement is that the mobile device has to be able to record cd-quality audio (sampling rate 44.1kHz), and have a microphone capable of recording frequencies as high as 22.05 kHz for system to work.

4.2. Goals

Regarding system accuracy, the design goal was set to 99.9% reliability within 5m of the audio source with a stationary receiver. With a mobile receiver (mobile device moving in a given space), the requirement was set to correctly catch a signal at least once while either moving towards or away from the audio source within a 5-meter distance. Three criterions, namely *distance*, *reliability* and *speed of signal recognition* were identified as relevant in measuring signature detection capability. For a moving receiver, the relationship between these criteria can summarized as follows: Formula 7 shows how the number of measurements (audio samples) depends on distance from the source to the receiver d_r, recording rate F_m and inversely affected by the velocity of receiver v_r. Formula 8 simply tells relation between detection chance of the signature and amount of measurements needed to find at least one signature given previously mentioned parameters and signature detection chance.

$$N_{measurements} = \frac{d_r * F_m}{v_r} \tag{7}$$

$$N_{measurements} * P_{detection\ chance} \geq N_{detections} \tag{8}$$

We can have partial control over two of the parameters presented here: measurement rate and detection chance. *Reliability* can compensate for slow signal detection, because with high reliability values we can take fewer samples. The measurement rate is dictated by our algorithm run length (and system latencies). To decrease minimum detection distance we must decrease time it takes to capture and detect signals.

4.3. Signal Design

Signal design depends heavily on the implemented detection method. However, some general rules and guidelines can be established to help select the correct parameters. Since we do not have established standards first we must establish framework for signal creation. We opted to use a simple signal creation method, where signal energy is divided equally amongst frequencies to get maximum power

for all frequencies; this also means that in a case with more frequencies, each individual frequency has less power.

We created several mock up signals to and used MATLAB to calculate amplitude responses to each of the unamplified mock up signals to see how the signals would behave. As mentioned earlier that doubling of number of bands roughly halves power of the single band (Figure 7a and c). As we learned frequency bands have limited resolution (limited by length of frequency band) basically shorter our signal is less frequency bands we can use. These observations are valid to both detections methods.

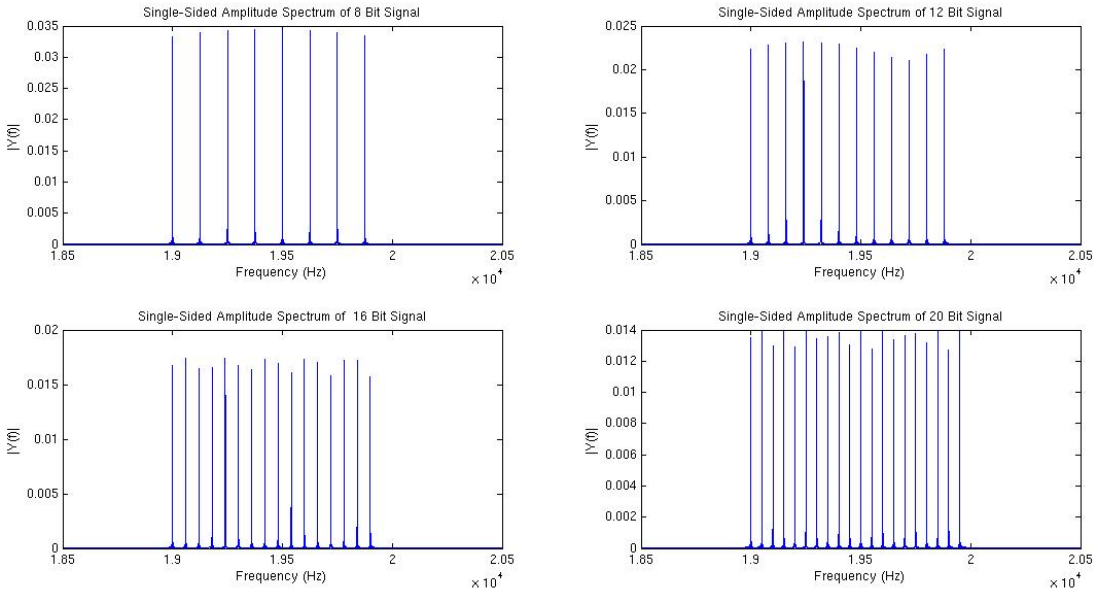


Figure 7. Amplitude Spectra of Signal with Different Bit Lengths a) 8, b) 12 c) 16 and d) 20.

Referring back to Table 1, combinations of parameters can be selected which are likely to work when taking into account the previously mentioned limitations with audio buffer lengths in android platform (anything lower than 200ms is most likely wasted). Further, samples longer than 1 second are too slow for real time usage. By mixing and matching values from the Table 1 we can get the framework for signal creation parameters that are most likely to work.

First we must take account of sample length which largely dictates run time. Next we must consider the range of the signatures. In order to reach our target range of 5 meters with unamplified signal while keeping above typical urban noise threshold of 0,001 we need to have individual band energy level at least equal or greater than that which means we should select resolution which at least good as 9 bit system. With 1 kHz bandwidth this gets us a practical upper limit of 12 for maximum number of data bands.

SONDI Audio signatures are created by segmenting a sound wave in case of cross-correlation into time slots each with its own frequency and in case of frequency transformation into frequency slots. Difference between signals in different detection methods can be best seen from Figure 8 and Figure 9. Where cross-correlation is signal where frequencies change in time scale in the frequency transformation method frequencies stay constant in the time scale. The signature files are created

using a sampling rate of 44.1 kHz (mono), which matches the typical recording capabilities of mobile devices.

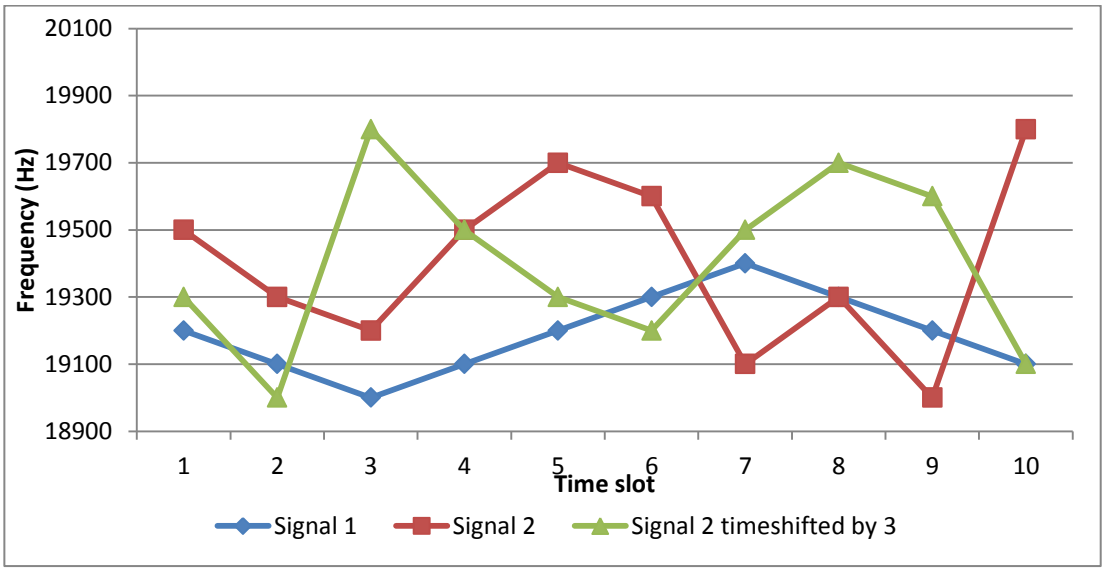


Figure 8. Cross-correlation signals in time domain.

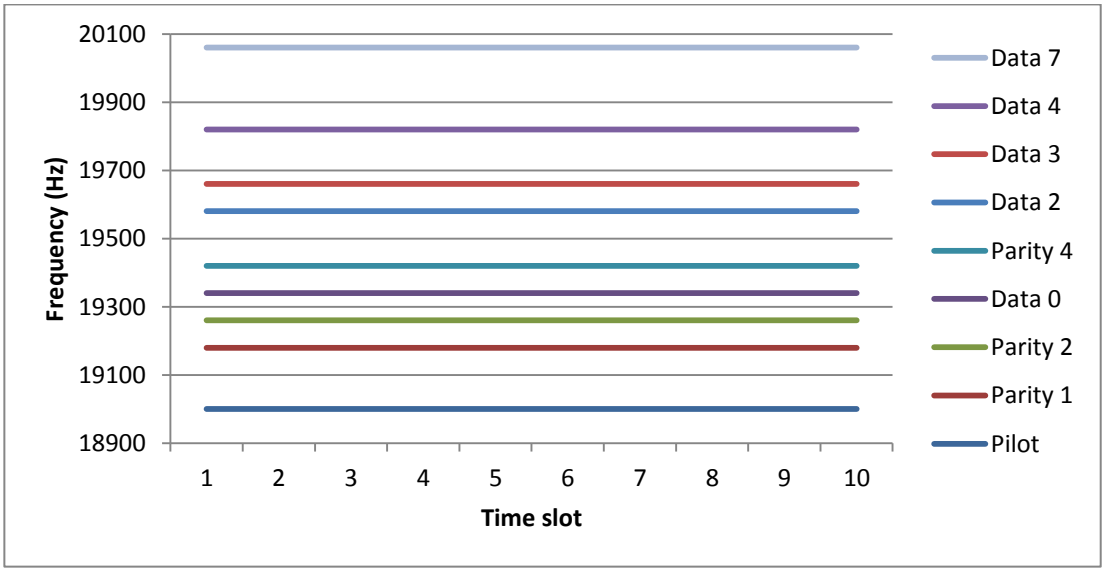


Figure 9. Frequency transformation signal in time domain.

As mentioned earlier with frequency transformation method we decided to use binary transmission and encode the information using hamming coding. We send all frequencies together at the same time which means each frequency band share the same transmission power. This in turn means depending number of active frequency bands, band starting level changes, worst possible case scenario is all data bits active and all parity bits active. Maximum number of active bits with hamming(12 4) encoding plus pilot is 1111 1110 which adds total of 7 active data bits and 4 active parity bits and one pilot which means total of 12 active bits. This means that even if are using 13 frequency bands at worst each transmitted code has equal transmission level to 12 band system (all bands active) this was done by selecting parameters so

that sum of parity bits and data bits would be minimized. Similar optimization of coding parameters and codes can be used to improve capacity while minimizing loss of transmission power per frequency.

5. IMPLEMENTATION

5.1. System Setup

The SONDI system was implemented using the following hardware components: the SONDI server was implemented with python scripts and deployed on a centos 6.5 server with 2G of memory and two 1.8 GHz processors. The SONDI mobile client was implemented as an Android application. The mobile application was installed on different phone models, namely nexus, s2 and s4 mini. The SONDI client is running on the control PC of the *target device*, in our implementation an interactive public display, was implemented with normal laptop running Firefox browser. The system logs all data into a MySQL database, as described in detail in section server side.

We also used panOULU WLAN [40] as form of rough location checking as knew that all *target devices* would be in the area of panOULU. This enables more sensible search distance of audio signatures. panOULU is free city wide wireless local area network. There are several public UBI-displays in downtown Oulu and campus areas which also works WLAN hotspots [40] and such it is good example of possible real life setup.

For data logging MySQL is used, MySQL is established and widely available system for creating and maintaining databases. As we are using already established platforms (android/windows/Linux) communication with devices over internet is also case of selecting some of already existing systems. In this case we opted to use RabbitMQ which we had already established infrastructure. This meant we had make RabbitMQ realization for android and design a communication scheme.

Loudspeaker (transmitter for audio barcode) is required to be able to produce sound in frequency range of 0-22kHz with directional sending capability. Directionality is needed to tell users orientation in relation the audio barcode source. Even though transmitter system is meant to be stationary transmitter elements should be small enough handled by one person, as the SONDI system is meant to blend into environment. Using large loudspeaker elements would make system impractical.

General structure of mobile application is shown on Figure 10. We have SondiMain which is basically our user interface. When application is started it checks if user is registered and if not it writes user data file (AubacoUserRegisteration) and asks server for user ID. Barcode_recognition does all signal detection and decoding. OnAlarmRecive does timed restarts of application as called other parts. We also have RabbitMQonAndroidSender and RabbitMQonAndroid which do sending and receiving messages with server and display element respectly using RabbitMQ.

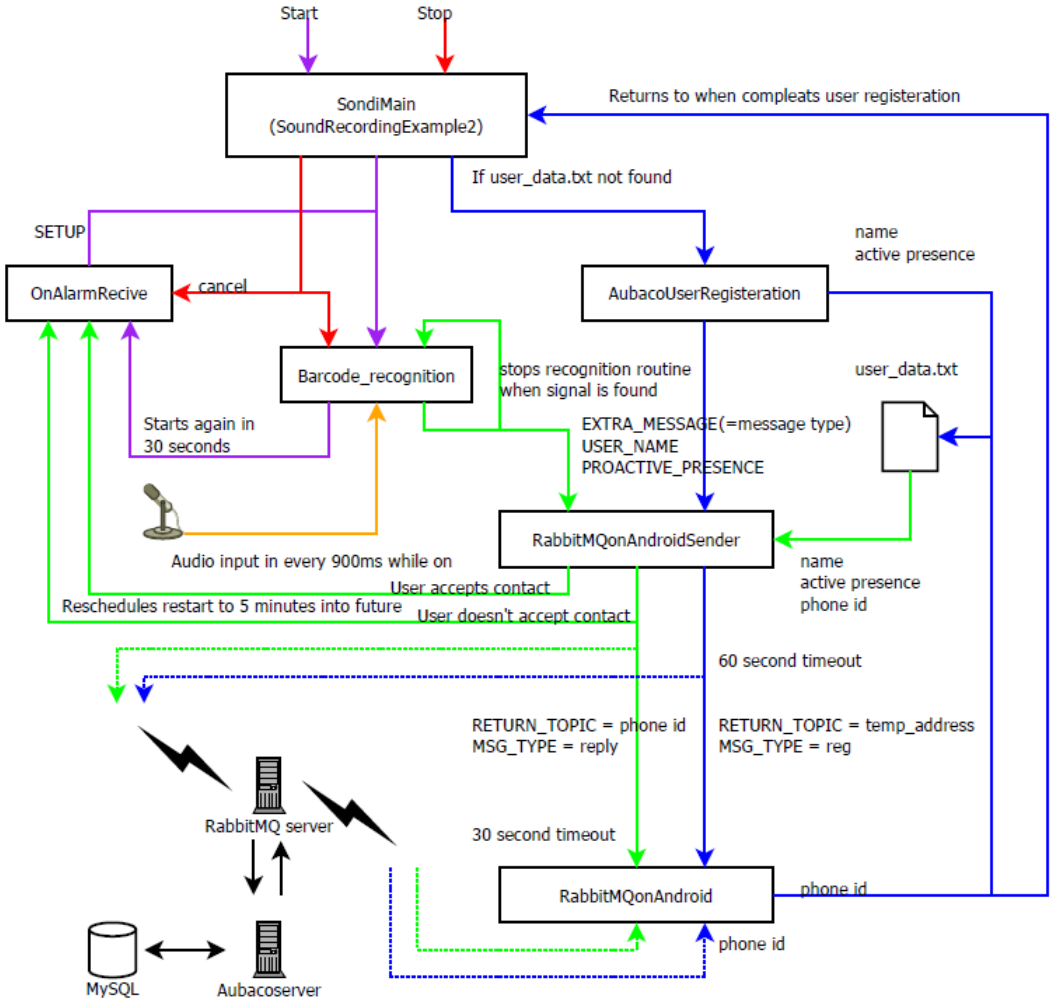


Figure 10. Program diagram for SONDI mobile application.

5.2. The Mobile Application: Audio Encoding

Mobile phones typically use an audio data format with a sample rate of 44.1 kHz in mono and 16 bit PCM encoding, which results in a microphone buffer size of 3904 sample points. In the android audio format implementation, this is actually 1952 samples each buffer, lasting a total of 44.2 milliseconds. A variable called recording rate denotes how often audio samples are recorded to find signals. Our recording rate is limited by following factors: sample length, the time it takes to record the sample, and the running time of the algorithm.

It is a known fact that android devices suffer from severe latency with attempts at real-time audio on current unmodified Android devices using the provided APIs due to fairly long audio buffers. This affects equally both cross-correlation and frequency transformation methods. As an example, in frequency transformation method the sample is 354 ms long, but recording takes typically about 450-650ms depending on model of the phone. This is called *input latency*. For a typical android phone, input latency is between 100ms and 250ms. There is very little that can be done to change this, which mandates that recording rate be set at 900ms. This is to ensure that the system will work with most phones without sampling between cycles overlapping each other and negatively affecting the error checking process. There is some hope

that in future versions of android operating system this would be fixed as it was announced at recent session [52] in *"Google I/O 2014 titled "Building great multi-media experiences on Android"*

On the mobile device, the android platform specifications [44] show that an application that will be supported by most smart phones should use a sampling rate of 44100 Hz, mono format and 16 bit encoding. This fits with our requirements (see section *human hearing*). When take look at supported media formats [11] in android system we get following list of file types (Table 2).

Table 2. Android smartphone audio file type.

Format/file type	**Available to operating system**	**Lossy encoding**
3GPP (.3gp)	4.1+	Yes
MPEG-4 (.mp4)	4.1+	Yes
MP3 (.mp3)	4.1+	Yes
WAVE (.wav)	4.1+	No
ADTS raw AAC (.aac)	4.1+	Yes

Various typical smartphone audio saving formats were tested for recording data before settling on wav format. This was mostly due same reasons as with MP3-format, most supported formats either employ perceptual coding or have low sampling rate. It is important to make distinction between different supported formats; most android phones (from 4.1+) can play files in wav-format and record PCM but not save it without additional work.

WAV-format is lossless non-packed format, as mentioned earlier WAV-format recording is not fully supported so, we first we had create software for wav-recording for testing and ultimately for signal detection .In the final version of program data is read directly from the microphone buffer. This means that recorded samples do not have to be saved, however for debugging and demonstration purposes saving audio data is invaluable. Most modern browsers support direct playback of wav-files but the native support is poor and the same html5 code behaved differently from computer to computer thus for testing we settled for using windows media player to play sound files. Later wav-files embedded were to webpage by using QuickTime player.

5.3. Error Checking

Illustrated in table 3 is variety of costs for false positives with different simple majority error checking method presented earlier in conceptual design. With several time values each showing a different scenario ranging from quick reset to worst case scenario where system does through full pause cycle. We assume that it takes to 0.9 seconds to find signal it will then take 1.8 seconds or 2.7 seconds depending on method to compare samples and assume that discarding results takes twice as long as finding correct signal. All results are calculated with simplified model (again presented earlier in conceptual design) were correct signal chance is 90% and false positive is 10%. Initially it appears that two consecutive samples is best all round but when interpolated to same amount of samples we can see that saving last sample is best overall error checking method for our system when recovery from failure takes less 300 seconds.

Table 3. Costs of error checking with false positives and other errors.

Minimum number of samples	**10s**	**30s**	**60s**	**300s**
2 of 2	1,9	2,1	2,4	4,8
2 of 3	2,242	2,802	3,642	10,362
overlap	2,1529	2,5329	3,1029	7,6629
Interpolated to same number of samples (6)				
2 of 2	5,7	6,3	7,2	14,4
2 of 3	4,484	5,604	7,284	20,724
overlap	4,3058	5,0658	6,2058	15,3258

5.4. Loudspeaker System

First we experiment with sound shower which are good directional speakers but lack necessary frequency range (400 Hz -16kHz) to send signal outside of human hearing range (20Hz-20kHz). Problem was even though signal receiving and sending worked it could still be heard by humans. Thus we had to find something else to work with. We managed to acquire for testing Genelec 1029a loudspeaker which has required frequency range and enough directional characteristics to enable us to test system fully. From the technical manual we learn at angle higher frequencies are dampened significantly for example at 1 meter mark with angle of 45º, 1029a is 10-15 decibels lower than reference level 0^0 angle which is enough for us to separate cases if transmission power levels and thresholds are set accordingly (see Appendix 1). Possibility of using whole available frequency range enabled us to hide the audio barcode better to the point that it was undetectable to humans.

5.5. Messaging

SONDI uses an asynchronous messaging broker called RabbitMQ [42]. The Android platform does not have native support for RabbitMQ messaging, which mandated a custom solution based on the RabbitMQ Java Client Library (available on the RabbitMQ homepage [42]). Previous studies have demonstrated that RabbitMQ can handle the required frequency of messages mandated by the SONDI system design [22].

All messaging will pass through the RabbitMQ server and the SONDI server. The SONDI server is running a python script with a RabbitMQ message receiver and sender along with MySQL updater. The main functionalities of the server are updating databases and directing messages. The basic way of sending messages between devices is illustrated in Figure 11 and Figure 12.

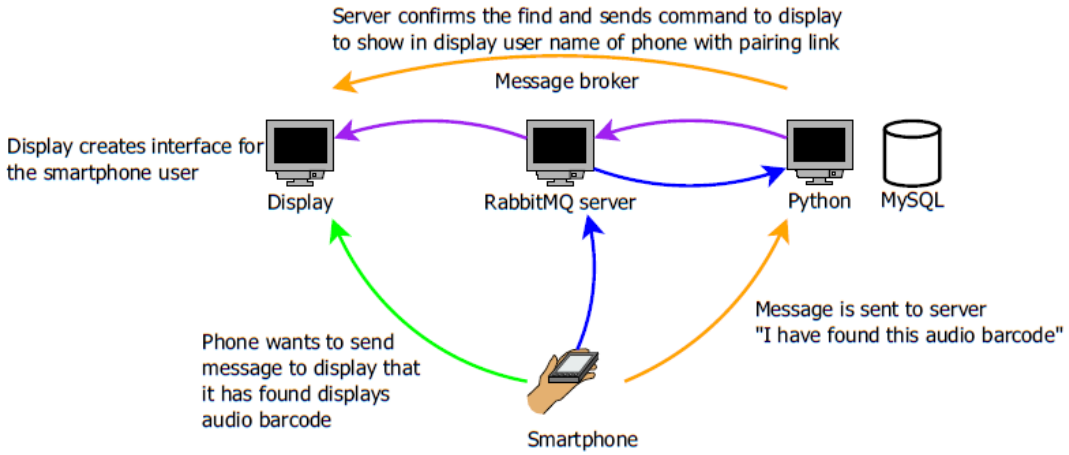


Figure 11. Phone to display communication.

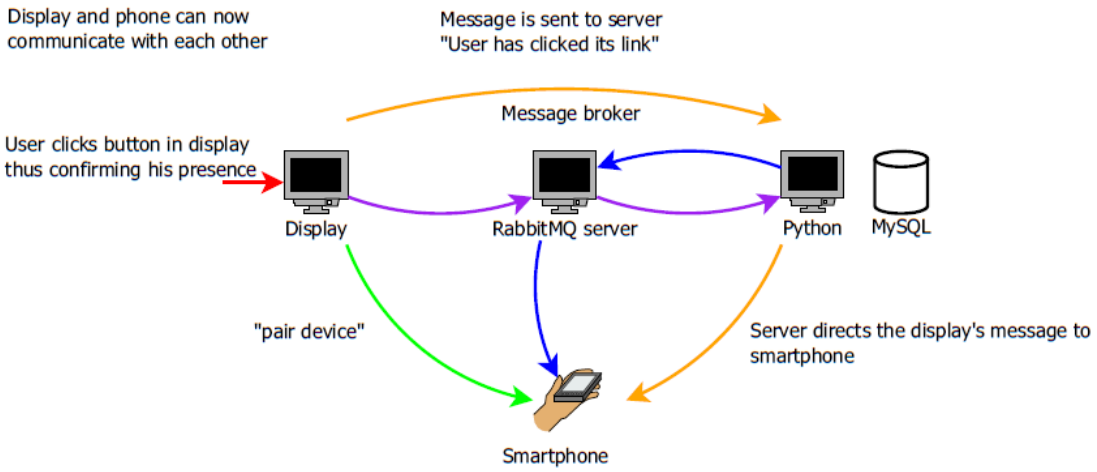


Figure 12. Display to phone communication.

The implemented messaging solution uses *topic exchange*. In topic exchange, a routing key decides the message destination. The routing key is a list of words, delimited by dots; a message sent with a particular routing key will be delivered to all the queues that are bound with a matching binding key. Hence, messages can be directed from a single producer to multiple consumers, which enables the decoupling and decentralization of server components where needed. This enables directing traffic easily by using keywords. The routing key structure is shown in table 4.

Table 4. Routing key structure.

Routing key	**Explanation**
aubaco.display_name_in	Messages sent to particular display element (from server).
aubaco.display_name_out	Messages sent from particular display element (to server).
aubaca.reg	Common registration address
aubaco.reg.common_session_id	Temporary return address for phone registration before username is established.
aubaco.userid	Particular phones address.

The mobile client uses two topics: *reply*, used to communicate with the SONDI server when a signature is received; and *reg*, used for registering the mobile device as a SONDI client identified with a unique ID. Each fixed device running the SONDI client has a dedicated topic (aubaco.display_in_name), where all messages sent to that particular device go; and another topic (aubaco.display_out_name) where all messages from that device go into.

For security and power saving reasons, the SONDI mobile clients RabbitMQ sender and receivers have timeouts. This also ensures that inactive users do not cause the system to freeze. All messages use a JASON template (Appendix 6), where the first part is used by RabbitMQ server for data logging, and SONDI-specific data is encapsulated inside a data object (JSON). All messages use this standardized format. The template is explained in detail in table 5.

Table 5. Explanation for each data tag used in template.

Tag	**Explanation**
common_session_id	32 characters long unique identification string used to identify users session id stays same during whole communication chain.
session_stop_time	This is always last timestamp and is updated each time message is opened.
username	User chosen call name used in machine to human communication.
user_id	Unique user id generated in registration phase used to identify users and as return address for messages it is also used to send real name in to server during registration.
ubi_name	Name of the target which we contacting.
pairing	Message type data/keyword.

We use message type select or start corresponding service in the smart device each type of message also shows short pop-up message using toast to inform user of what is happening right now between display and the device. Display can also send message to server which is used to monitor display component state. Since demo implementation required multiuser interface using single interface unit.

Queue system was implemented message is sent to notify user when display is free and user can finish rest of pairing procedure else user is sent wait message. Login is sent when user finishes pairing and is logged in to display. Demo message starts demonstration routine in smart device (more of this latter, simple debug routine without contacting server). Queue structure of webpage interface is shown on Figure 13.

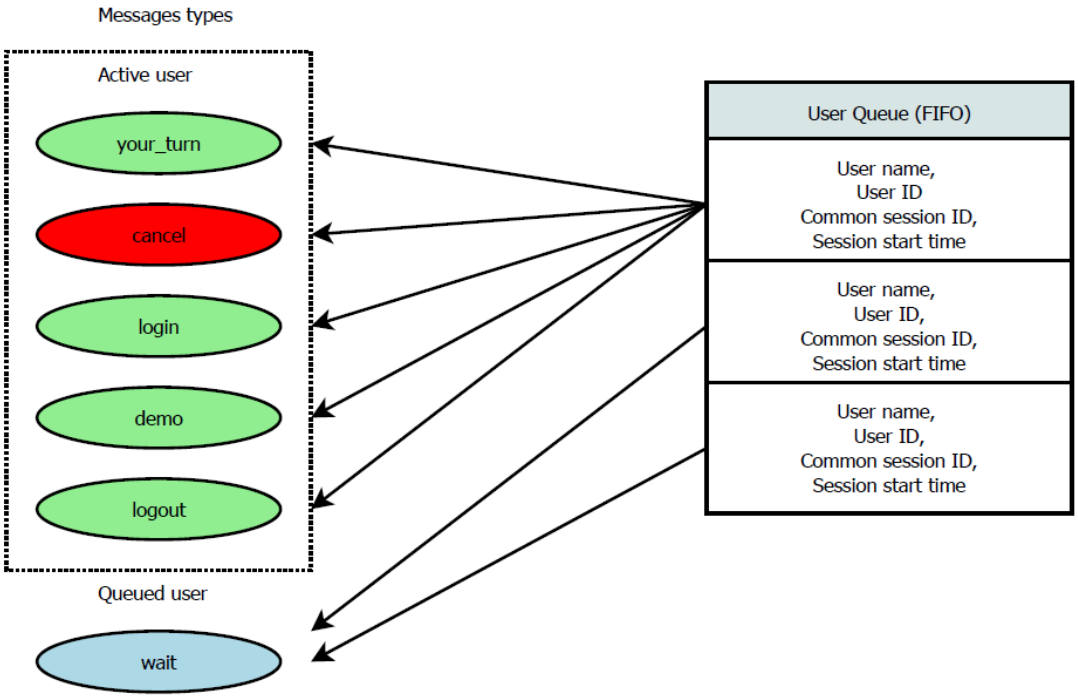


Figure 13. Message types and display user queue.

5.6. Signal Detection Methods

To achieve set goals of reliable near real time detection speed in ranges up to 5 meters, two methods of audio barcode recognition were tested, namely *cross-correlation* and *frequency transformation*. Both methods utilize FFT (Fast Fourier transformation), making them comparative. Figure 14 presents an overview of both methods.

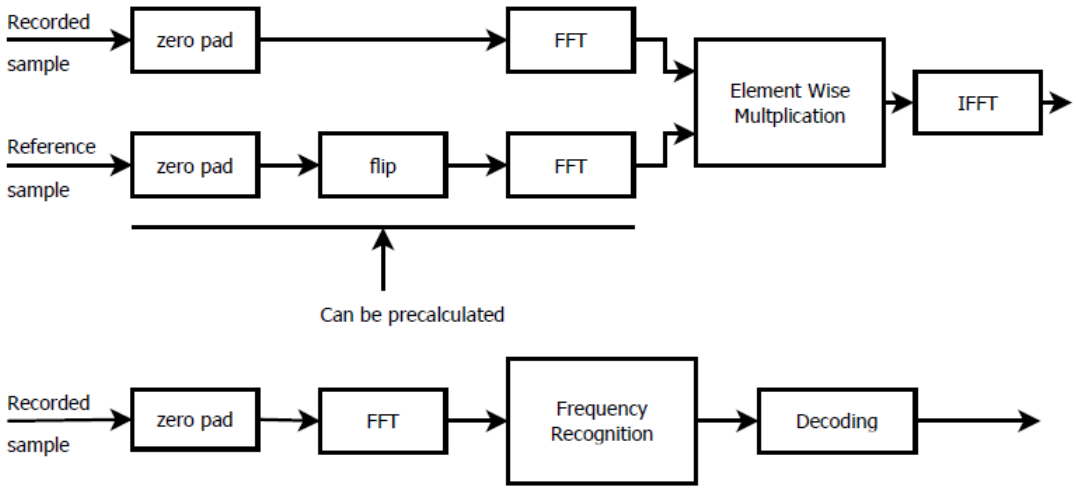


Figure 14. Block diagram cross-correlation (top) and frequency transformation (bottom).

Two methods of signal recognition were chosen for implementation: Cross-correlation and frequency transformation (FFT based frequency recognition).Neither

of these methods mandates using a synchronization signal, which would allow us to detect the beginning and end of a signature. Rather, the entire recorded sample is treated as one continuous entity, which causes some variation in cross-correlation results as the signal can be captured in different phases of transmission. However, the continuous transmission of the signal mitigates this effect to a minimum.

Cross-correlation was used to establish a valid baseline for further experiments. Cross-correlation was chosen as starting point for few reasons: First of all it is simple method which requires two FFT's, one IFFT and finding maximum value. Second reason was adaptivity, in case where we would have to mask signal or wanted to use natural sounds, with cross-correlation utilizing these possibilities is trivial matter of recording sample and loading it to system as reference.

For FFT we opted to use radix-2 which is simple but memory efficient FFT algorithm. Radix-2 is also known as Cooley-Tukey algorithm and is probably most common way to compute the discrete Fourier transform (DFT). Radix-2 takes O(N log N) time to calculate N length sample. Both methods were tested using the same realization of radix-2. This removes one variable when comparing the two methods, thus signal detection routine execution time is only affected by number of FFT's and how long it took algorithm to find results.

5.6.1. Cross-correlation

Cross-correlation uses signature lengths of 650ms; a minimum of 28706 samples (preferably more) is required to capture the entire signature. For cross-correlation to work, both the recorded signal and the reference signal need to be zero-padded, so that at least half of each signal is set to zero. For computational efficiency, the zero-padded signal lengths should be selected so that they are some power of two. Hence, SONDI uses a capture length of 16 microphone buffers (i.e. 31232 samples, which is just below 2^{15}). This provides us with a good compromise between signature length and processing time. This also results in a frequency bin of 1.34Hz/bin (Formula 1), which is more than enough to differentiate between the 100 Hz sub-bands used to create SONDI signatures. Recoding cycle is show in Figure 15. With the cross-correlation method we decided to divide the audio signature into 10 different time slots, each of which can have a separate frequency. This leads to 10^{10} different combinations which is enough for almost any use case (in reality, some of the combination are redundant, i.e. not useful or difficult to differentiate).

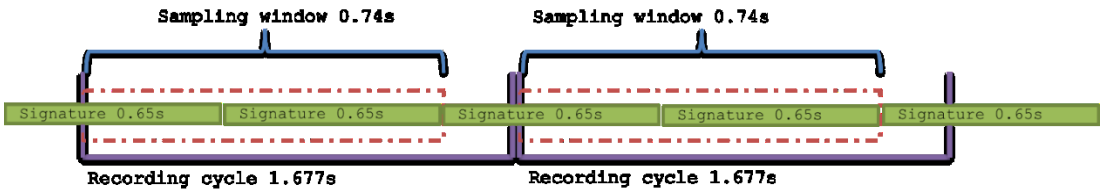


Figure 15. Cross-correlation sampling.

In the final phase of correlation, one of the zero-padded signals (either the recorded sample or the reference signature) is reversed, and Fourier transformation using the radix-2 algorithm is applied to both. For sake of optimization reverse is

done to reference signal so that we can pre-calculate FFT for reference signal during signal creation. This leaves us only one FFT and one IFFT, both of which are performed in the mobile device. Next, element-wise multiplication of the two transformed signals is done, and inverse Fourier transformation is applied on the result using radix-2 algorithm. The end result is shown in Figure 16.

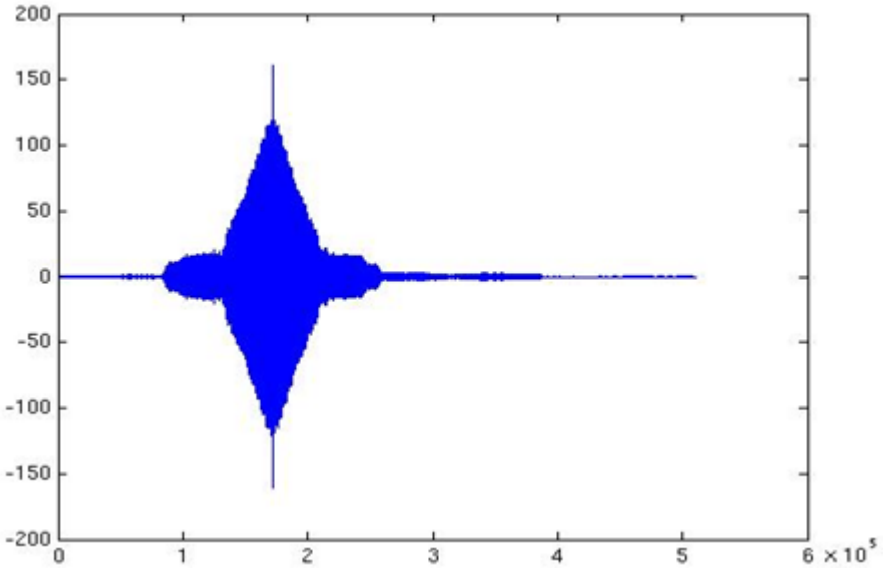


Figure 16. Spectrum of cross-correlated signal.

5.6.2. Frequency Transformation

Second tested method frequency transformation is little bit more complicated than cross-correlation. Frequency transformation has recording length of 354ms which translates to 15616 samples Recording length is result of experimenting with different code lengths to find reliable and short signature length (audio signature length is the main contributing factor in detection routine runtime). This recording length results in a frequency bin of 2.69 Hz/bin (using Formula 1).

Frequency transformation method uses audio signatures that are created segmenting a sound wave with the duration of 354 milliseconds into 13 frequency bands. 19 kHz frequency band is allocated as pilot signal used to correct Doppler Effect, pilot signal frequency has guard band of 180 Hz used keep pilot signal from mixing with data. Rest of the bandwidth is allocated to data where we have 12 frequencies with 80 Hz gaps between each frequency. Each frequency presents one bit, Data word is encoded using hamming (4 12) coding on data so from 12 bits, 8 are data and 4 are parity bits used for error correcting. Active frequency indicates a logical 1 and absence of frequency implies logical 0. Frequencies are combined into one single signal where all active frequency bands are played together.

When audio signature is received by the mobile client the signal spectrum is calculated by using single 15 bit FFT next pilot bit searched from the spectrum. Pilot bit has search area of 180 Hz (as explained in below), pilot bit frequency is saved and used as offset for searching data bits. The search area for data bits is adjusted by offset of pilot signal then active frequencies are identified from the spectra. Data bits have small threshold values and search area of 80 Hz around the expected/main frequency (frequency bandwidth of 80 Hz and search area of +/- 40 Hz) and the band

is counted as active if threshold value was surpassed and found band is within +/- 25 Hz of expected (this to migrate problem caused by frequency crawl which can be caused for example slight movement of phone or calculation and rounding errors in software). Recording cycle is shown in Figure 17. Also since each bit is represented by different frequency it is expected that different signal are affected differently by channel.

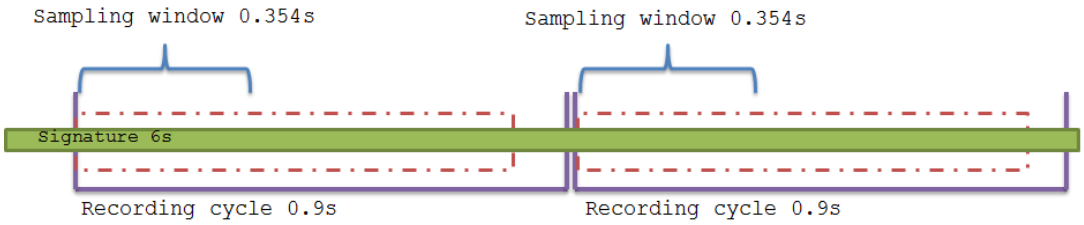


Figure 17. Frequency transformation sampling.

We first experimented with system that would look for highest peak at given band but this caused false positives and threshold values for band recognition were soon added as it became apparent that increased frequency band recognition chances were not worth of loss of precision, without some sort of checking of signal level to ensure that we actually have an active frequency in band.

Next detection criterion was focused on precision; recording was combed for active frequencies in small bands if the highest found frequency within band was within 10 Hz of expected frequency signal was seen considered one. Curiously this caused unexpected number of errors simply due frequencies not being strictly where expected. This was corrected by expanding target frequency range to be within 25 Hz of expected frequency.

Solution was to find balance between precision and signal strength after certain point, lessening the precision or threshold requirement does not increase detection chance. This can be explained, when we take closer look at received bits we notice that problem is not falsely active bands but received bands that are not active even if they should be this simply due other phenomena like destructive interference this can be compensated by increasing volume (transmission power).

Confirmation cycle (as descripted in simple majority error checking) was implemented because SONDI is designed for mobile devices which are naturally moving devices. While receiver is moving transmission errors multiply significantly to point where our error coding cannot keep up, due nature of communication we cannot use traditional methods of detecting and correcting false signals therefore to detect a signature without false positives. We resulted to taking multiple samples and made decision with simple majority. As whole this system has minimum detection time of 1.3 seconds it final configuration.

This modification in theory doubles barcode checking time with frequency transformation method while minimizing chance of false positives. However, real effect to detection time t_d is in practice less than double, when signal detection chances are also taken into consideration (Formula 9). We can calculate basic relationship which basically tells us that on average, bigger the difference in detection probabilities (in favor of double checking) less it has affects overall detection time t_{od}.

$$\frac{2*P_{single}}{P_{double}} * t_d = t_{od} \quad (9)$$

5.6.3. Doppler Effect Compensation

Solution to Doppler Effect was to add single pilot frequency set apart from other frequency bands in such way that signals would not mix up with data signals with average walking speeds. This additional frequency band was added to the beginning of each signature, and set as always active. The pilot frequency band is set apart from other bands by 180 Hz (Calculated maximum shift to both directions with average human walking speed plus extra small guard band of 20 Hz).

Location of pilot signal would be used as estimated starting point where to start looking for signal frequencies. The pilot signal is set at the low end of the utilized signal spectrum (19 kHz), because earlier testing has shown that these frequencies are least affected by recording and channel. A more robust solution for future work would be to use multiple pilot frequencies to make sure one rogue frequency would not break the entire system.

5.6.4. Signal Encoding

Barcodes are pre-encoded when a signal is created, and the mobile device does decoding run-time. The time required to decode signals has a negligible impact on overall detection time. The pilot frequency is not hamming coded, as coding it would make locating the pilot more difficult. Table 6 illustrates a hamming coded signal with marked data, parity and pilot locations.

Table 6. Parity bit coverage and encoding.

Bit position		**pilot**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Encoded data bits		no	p1	p2	d0	p4	d1	d2	d3	p8	d4	d5	d6	d7
Example		1			1		0	1	1		1	0	0	1
Parity bit coverage	p1	1	1=		1		0		1		1		0	
	p2	1		1=	1			1	1			0	0	
	p4	1				1=	0	1	1					1
	p8	1								0=	1	0	0	1
Encoded signal with pilot		1	1	1	1	1	0	1	1	0	1	0	0	1

5.7. Demo Implementation

In order to demonstrate the capabilities of SONDI, a demo application was created. As mentioned in chapter 2 our goal was to create a proactive pairing system. This idea of simplicity and quickness is extended into user interface. All communication is hidden from user, and the user is only prompted to confirm pairing when an audio signature is successfully received and decoded. An example use case scenario could be as follows:

A user is walking past a SONDI-enabled information display, which can provide a map of the building the user is currently in. The user's mobile device picks up the audio signature broadcasted by the information display, and alerts the user to the

possibility of receiving contextually relevant content from the display. The user decides to pair his/her device with the display, and downloads a map augmented with important location-based information from the display.

In order to realize type of scenario presented above, a demo implementation with two user interface components (the mobile client and the public display interface) was built. In this implementation, pairing is realized as a two-step process where the user is required to acknowledge both UI-elements in order to access the service (*i.e.* to confirm his/her presence physically). The system is intended to work when the user and display have a visual line of sight of each other. For this reason, the signature broadcast power was limited to a 5-meter zone extending from the speaker outwards.

5.7.1. Communications and User Interface

The basic interaction sequence is presented in Figure 18. As the user walks towards the display, his/her phone picks up and decodes the signature broadcasted by the display. After the signature has been successfully decoded, the user is prompted to pair his/her device with the display. The user must then press a button on the display to confirm his/her presence near the display (secure pairing).

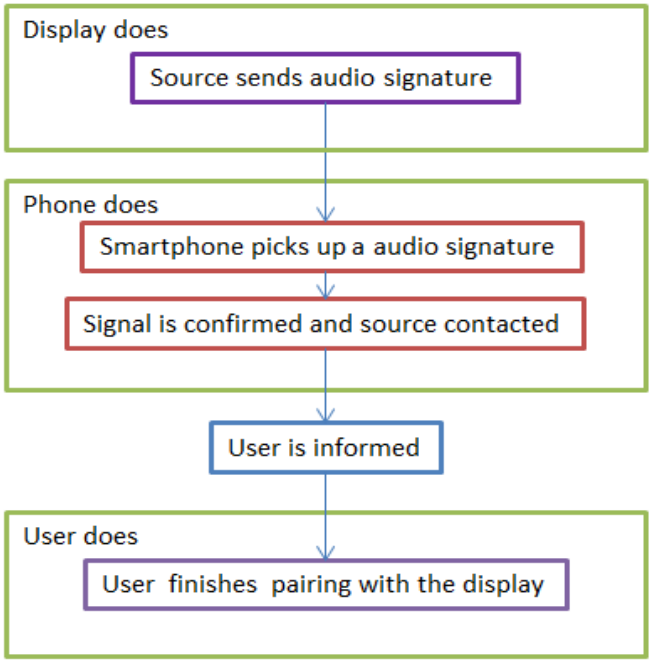


Figure 18. Pairing process.

The SONDI display element broadcasts back-to-back signature waves, each lasting 6 seconds. When in active scanning mode, the mobile element continuously runs so-called recording cycles, each lasting 0.9 seconds. During each cycle, the mobile client records one sample lasting 0.354 seconds from continuous signal. The rest of the cycle (0.546 seconds) is used to analyze the captured sample, perform signature detection and, in case a signature is captured second cycle is made for

confirmation, after which contact to SONDI server is initiated for the pairing procedure. If no signature is captured, a new recording cycle is initiated.

5.7.2. Application UI

When the user launches the mobile application for the first time, he or she is prompted to select a user name. The mobile client then sends a registration message to the server, which answers with a unique user id for that user name. This user id is used internally by the system, and the selected user name is used when communicating with the user.

The mobile application interface is very simple, consisting of two buttons (Start and Stop). The Start button initiates the audio signature detection routine, and the Stop button terminates it. There is also automatic demo mode which just detects codes without contacting server which can be started either from phone or from the display after user has been paired with the display. The detection routine is invisible to the user until an audio signature is found. The routine runs in the background, so it does not interfere with other running programs and the device can be used as normal.

While application is active it automatically checks every 30 seconds if phone is near display unit by checking which WLAN it is connected to (in our case panOULU). When the mobile device is determined to be general area of a SONDI-enabled display, the audio signature detection routine is initiated. When an audio signature is successfully found and decoded, the user is notified through tactile (vibration) feedback and a notification pop-up is presented on the mobile device prompting user to either accept or decline a pairing request from a SONDI-enabled device. In order to avoid continuous alerts while in the vicinity of the same device, the signature detection routine is suspended for five minutes after the initial pairing request has either been accepted or declined. The mobile client interface is shown in Figure 19.

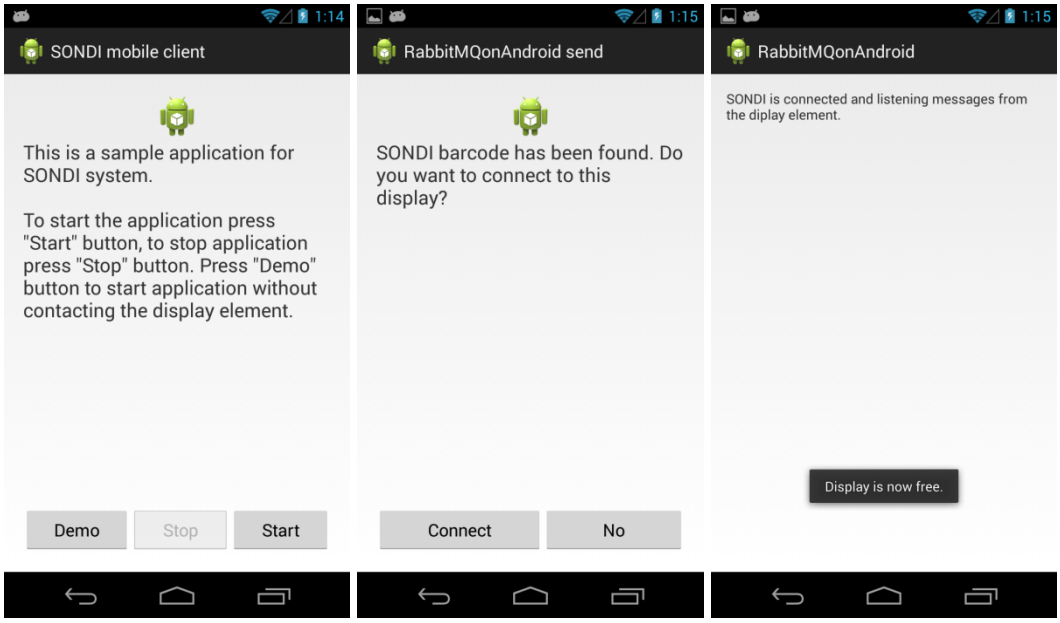


Figure 19. SONDI mobile client

5.7.3. Display UI

The interface shown on the public display is realized as a web page. When a mobile client discovers and decodes the audio signature broadcasted by the display, and the pairing request is accepted, the display responds with a greeting message displaying user's name and prompting the user for a confirmation to finalize the pairing sequence. This dialog-window and login session will timeout after 30 seconds if user does nothing. After the user has completed the pairing sequence, s/he can change SONDI mobile applications mode to from the display, or log out.

If another user is already paired with the particular display, incoming requests are placed in a queue. A communication channel between the queued users and the display is kept periodically alive, until the display is available for the next user. The display utilizes a two second buffer between users, mainly to clearly separate each session and avoid confusing users.

The display maintains a list of users who have been detected within the last 30 seconds, *i.e.* users with a SONDI mobile client that has decoded the audio signature broadcasted by this particular display within the past 30 seconds. The public display UI is shown in Figure 20.

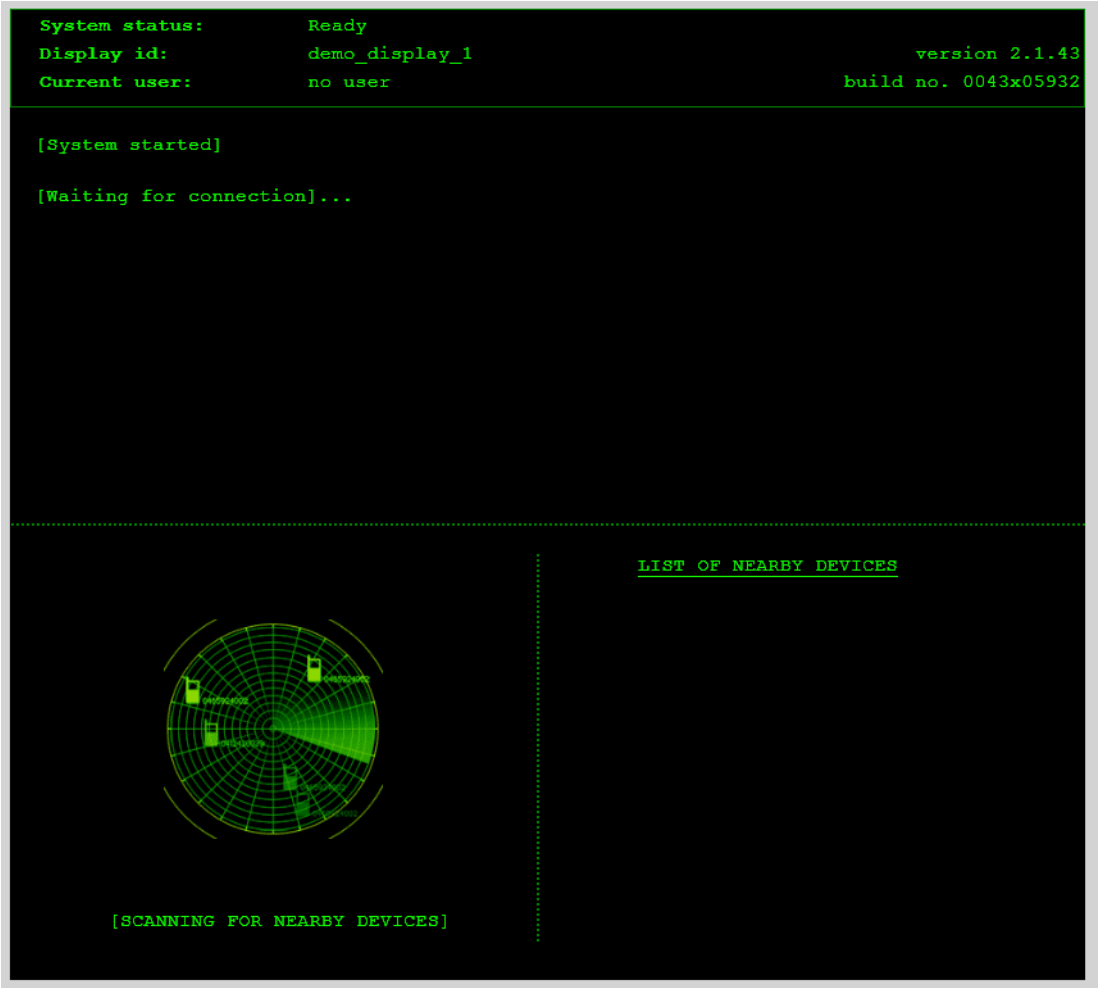


Figure 20. Basic layout of display element UI.

5.8. Server Side

For the purposes of the demo implementation, a single centralized server controlling logging and messaging between devices is used. Server scripts are created using Python 2.7, and interfacing with the RabbitMQ asynchronous message broker is done using Pika for RabbitMQ. Pika is a pure-Python implementation of the AMQP 0-9-1 protocol, and it is fairly independent of the underlying network support library. MySQLdb-library was used for MySQL database. A server script sorts' messages according to topics (as would they would with a pure RabbitMQ implementation), which means that messages are opened and logged before redirecting them to correct receiver topics.

Finally for standardized creation of audio signatures python script was implemented for server side. This script will encode any given 8 bit data word with hamming code (4 12) and add pilot signal before creating the audio file. Creation date and all details like original data word, devices broadcasting the created code etc… are saved to separate MySQL database for future reference, Figure 21 shows the server implementation for the SONDI demo, including relevant scripts and the MySQL database structure; below it in Figure 22 are MYSQL tables for each of databases.

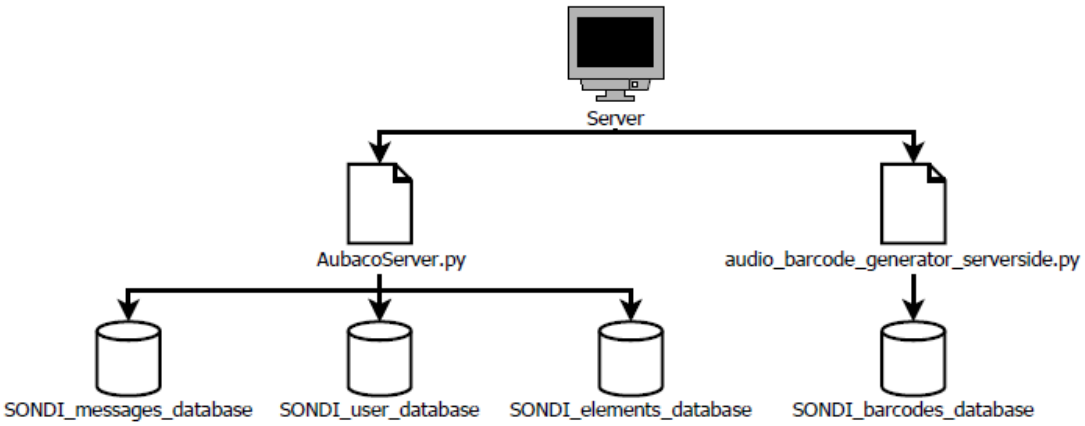


Figure 21. Server and databases.

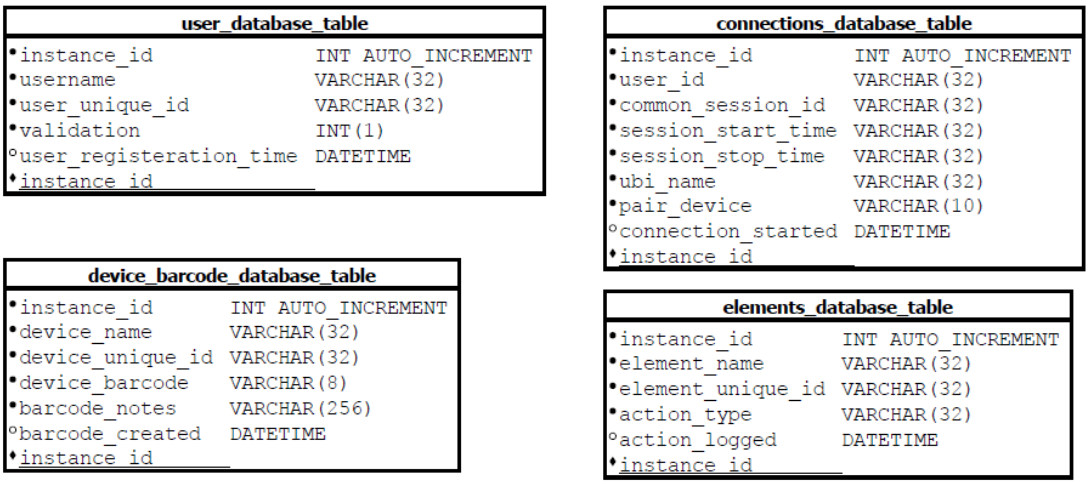


Figure 22. Database tables a) user database, b) messages database, c) audio signatures database and d) elements database.

6. EVALUTION

The two separate SONDI audio signature recognition methods, namely *cross-correlation* and *frequency transformation*, were both evaluated in an indoor setting. Further tests were carried out by simulating ambient urban noise (*cross-correlation* method), or actually deploying the system outdoors (*frequency transformation* method). The indoors testing area was a large shared office space (13x5.5 m in size, Figure 23). This area was devoid of other noises, except of those of normal office sounds such as hum from air conditioning, footsteps, conversations, keyboard clicks, etc. The ambient noise level in this condition was measured to be around 47 decibels. The same space was used for the simulated urban condition with the cross-correlation method, where pre-recorded urban sounds such as traffic noise *etc.* was introduced through a loudspeaker system.

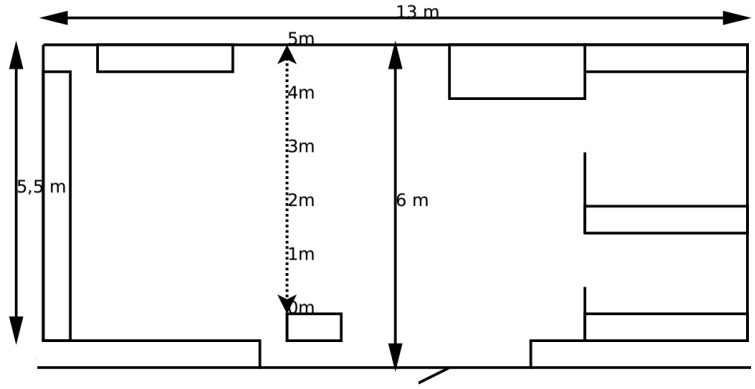


Figure 23. Indoor testing area

With the *frequency transformation* method, SONDI was deployed outdoors next to a busy road and car park (Figure 24). The site featured both automotive and pedestrian traffic, and additional loud noises from a nearby construction site including heavy construction vehicles driving past. Smart phones were placed on movable rig while control PC and loud speaker remained stationary. We did same stationary and movement as in laboratory tests in cross-section near university at busiest time of the year. Test setup and area are shown in Figure 24. In these tests we used only transmitter loudspeaker otherwise setup was identical to laboratory tests. Ambient noise levels in the area were same as previous measurements in city center.

Figure 24. Outdoors testing rig and area.

Tests were carried out using multiple modern smartphone models. Table 5 lists the used phone models with operating systems and recording latencies. Models where recording latency is not reported were used in "black box" testing, with the purpose of determining whether the system works reliably on different devices and models. System worked with all listed devices.

Table 7. Used phones and their operating system with recording latency.

Phone name	**Manufacturer**	**Operating system**	**Recording Latency**
Nexus	Samsung	4.3	100ms
Nexus 5	LG	4.4.3 and 4.4.4.	-
S2 plus	Samsung	4.1	60ms
S3 tab	Samsung	4.3	-
S4 mini	Samsung	4.2	100ms
S5	Samsung	4.4.2	100ms

6.1. Initial Stationary Testing

Testing of SONDI system was started with stationary testing, mainly in order to find out to effects of distance and transmission power. This was done to find out how increasing distance from audio source affects detection chance. We also tested if we could compensate distance by increasing transmission power (sound volume) and generally how transmission power affects signal detection. Phone(s) were stationary during whole measurement and distances up to five meters from the speaker were measured and marked on the floor, and tests were conducted at half meter intervals starting 0,5 and went up to 5 meter distances.

First phase of testing was done without any additional noises, in the second phase (simulated urban condition) we introduced ambient noise recorded at the downtown area. The test setup was otherwise identical in both phases, consisting of a speaker

(Genelec 1029a) situated in a fixed point, and a movable rig housing a decibel meter and a holder for the phone (Figure 25). Prerecorded traffic noise was played back from another Genelec speaker.

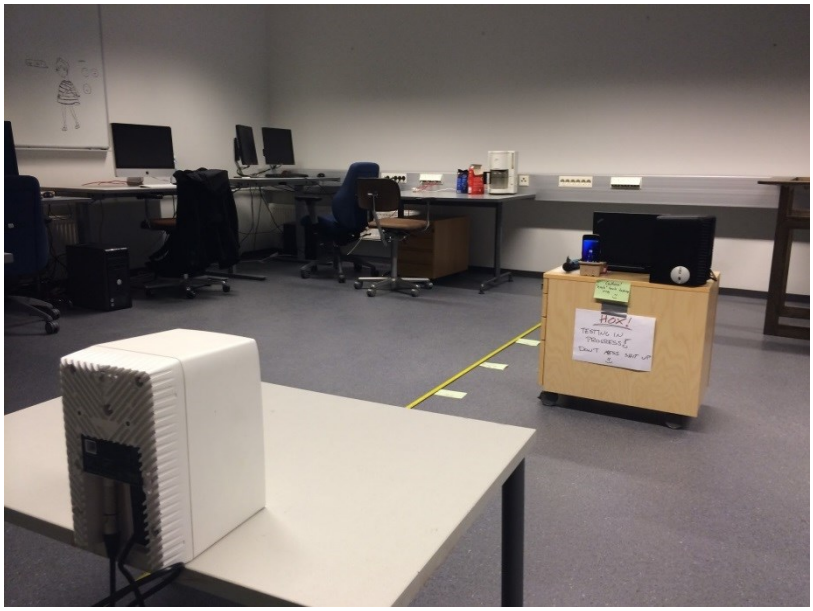


Figure 25. Test rig for first two stages of testing.

Ambient noise was recorded at peak rush hour using high-quality microphones to accurately capture the entire frequency spectrum, and included traffic, people and construction noises. While recording, we also measured the decibel level of downtown area (66.15db) so we could emulate a noisy urban setting as authentically as possible. Decibel meter was used as part of the rig to monitor noise level and to see if noise level would have any effect.

6.1.1. Cross-correlation Testing

At first two phases of testing for cross-correlation we logged 100 measurements at each distance, using cross-correlation similarity (cross-correlation maximum) threshold values of 60, 70 and 80. These PCM threshold values were selected based on experiment, where we selected lowest maximum value of cross-correlation which reliably worked at distance of 2m at 60.6 dB transmission power then we incrementally changed values to see what effect it would have on detection rate and reliability. Further, the volume of the broadcasted signature was manipulated to account for the effect of distance. We chose three levels of transmission power, corresponding to 50%, 60% and 70% of the maximum volume of the computer running the SONDI client. The three levels are: 60.6dB (50%), 62.9dB (60%) and 65dB (70%). We carried out 100 measurements for each similarity threshold value and each volume level at each distance, resulting in an extensive data set.

6.1.2. Frequency Transformation Testing

After stationary initial testing with cross-correlation third and fourth phases of we introduced in preparation to final real world tests also second method of signal creation and detection was implemented (see frequency transformation). For frequency transformation we also increased number of samples to minimum of 500 to get better statistical sampling.

As part of design process we wanted to test different error correction codes and methods. For testing less transmission power was used to introduce meaningful error to channel in order to test different error correction methods. We tested several different kinds of error correcting codes starting from simple repetition codes. We went through several different patterns of code lengths and number and order of repetitions. Afterwards we tested if we could improve system using more advanced error correcting codes in this case we used different hamming codes (12 4) and (16 4)

6.2. Non Stationary Testing

In third and fourth phases phones were directed at the loudspeaker and tester followed route from 0.5 to 5 m mark back and forth to simulate user walking towards the source and away from it in straight line. Also as part of clothing testing one phone was kept in pocket to test most likely use scenario where phone is in pocket and user is walking past the transmitter.

In fourth stage same routine was repeated but with added ambient noise like in phase 2 with the difference speaker placement as we had no way to move loudspeaker along with the walker so loudspeaker was directed along the route of the walker. In advanced testing transmission power was kept constant level throughout the testing. Two detection methods we compared each other after all four phases were done to see which would be the final version which would be tested in the wild and be the basis of demo version.

6.2.1. Frequency Transformation Testing

It was soon discovered using pilot signal alone was not enough to correct all caused errors since human movement is rather sporadic which meant even if signal is short, practically each individual band can have different Doppler shift. One way to counter Doppler effect entirely would to use frequency bands far enough for each other (about 160 Hz) that there would not be a chance to confuse frequency bands with walking speeds but this is not practical as it eats already premium channel capacity. But if signal is kept short enough Doppler shift should be relatively uniform for whole signal length. To counteract this, we implemented and tested error checking method to get rid of random errors and to eliminate any chances of false positives.

6.2.2. Cross-correlation Testing

In addition to movement test we did testing to see if we could reliably differentia audio signals and how long it would take problems. Also we briefly tested different barriers and conditions to see how the system would behave in the more realistic testing environments. Unfortunately it did come apparent that our solution had

several limitations thus we decided in favor of frequency transformation and abandoned cross-correlation.

6.3. Obstructed Receiver

Lastly we also tested SONDI system when obstructed by clothing. There were two different kinds of testing done with clothing, first moving receiver obstructed by normal indoor clothing using same setup (see above) as other indoor moving receiver testing and second using same setup as stationary testing using heavy autumn clothing (Figure 26). For stationary testing measurements were taken at 1 meter intervals. Goal was to test behavior in typical use cases where phone is obstructed by clothing (light indoors and heavy outdoors).

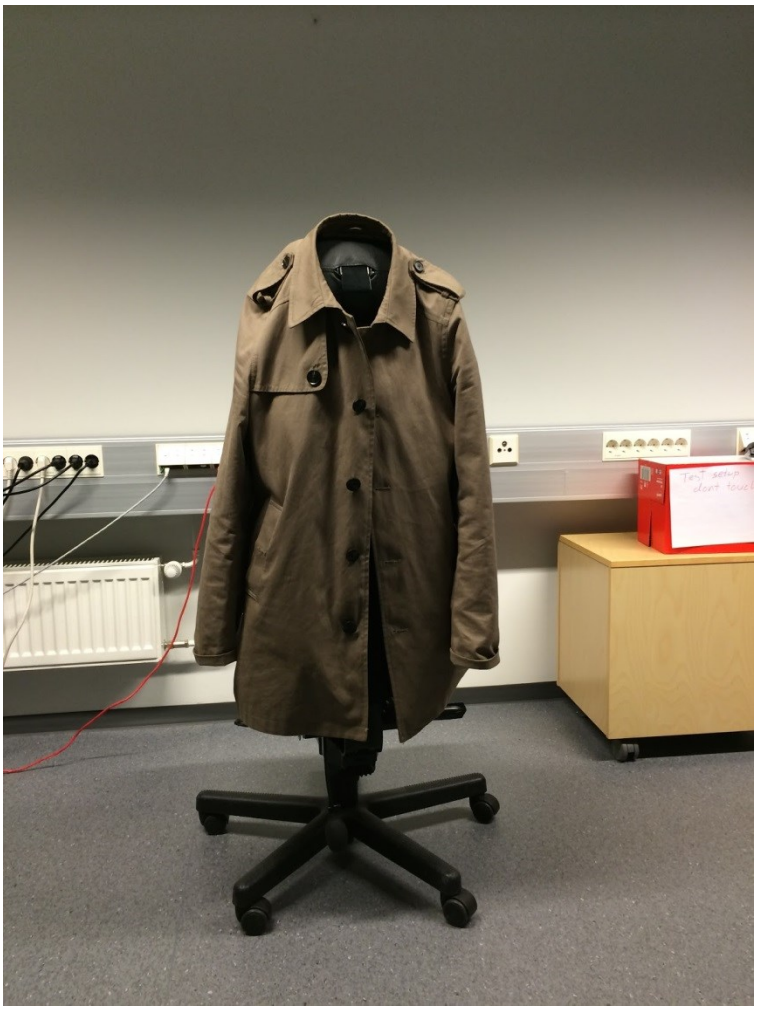


Figure 26. Stationary clothing test setup.

6.4. Demo Client Power Consumption Testing

Finally we measured power usage and estimated communications roundtrip time. For roundtrip we measured time it would take signal to be found and user input on display element. Natural this depended on user alertness, experience level and

distance of detection (walking time to display). Roundtrip time for alerted and experienced user was 5-6 seconds.

Battery consumption was measured by having the SONDI mobile client continuously scanning for signatures for 60 minutes in two separate conditions. First, measurements were carried out with no signatures being broadcasted (background condition), i.e. the device was continuously listening to signatures, but since none were present, no further actions were carried out. Subsequent measurements were taken with signatures being broadcasted, recorded and analyzed continuously (active condition). In this condition pairing messages were also created, sent and replies received via the RabbitMQ message broker every 30 seconds, amounting to a total of over 2000 messages.

7. RESULTS

The results section will cover main findings from evaluations carried out for both the cross-correlation and the frequency transformation methods. As mentioned earlier tests were done in phases and we will select one of those methods based on these evaluations. Our main interested is in the signature audio detection chance and error probability.

7.1. Effects of Distance and Transmission Power

As discussed earlier we know that transmission power will decrease as distance increases from the source. Thus we are interested how different methods compare to each other and to how distance affects detection chances and error probability of audio signatures. We are also interested if we could counter act harmful effects of increased distance with increased transmission power.

7.1.1. Cross-correlation

We can conclude that cross-correlation method works in case of single audio signature when receiver is stationary. Appendix 2 shows the results from testing cross-correlation method with both the indoor condition (no added noise) and the simulated urban condition (added traffic noise) using similarity threshold values of 60, 70 and 80 and transmission levels of 50%, 60% and 70%. Average detection time is calculated from recording cycles (1.677 seconds): for example, given a detection success of 100% at 1-meter distance, the average detection time per signature is 1.677 seconds. In other words, when the first recording cycle does not return a successful signature detection, another cycle is initiated until a signature is found. Detection time is hence increased as detection percentage is decreased.

Detection percentage refers to the percentage of individual signatures detected, and does not measure the success of creating a connection between the SONDI mobile client and the SONDI client; (except where detection % is 0) – detection percentage simply measures the number of successful detections in a series of 100 measurements. As we suspected increasing transmission power leads to increased detection chance and increased threshold value decreases detection chance. As we can see from Figure 27, signal shape is generally preserved while signal level diminishes. Similar effect can be seen from Figure 28 where average value of cross-correlations decreases while distance from source increases, indicating that due power loss signal has started to loose detectability. We also can see small valley of death around 3.5 meter mark where detection chance unexpectedly drops slightly.

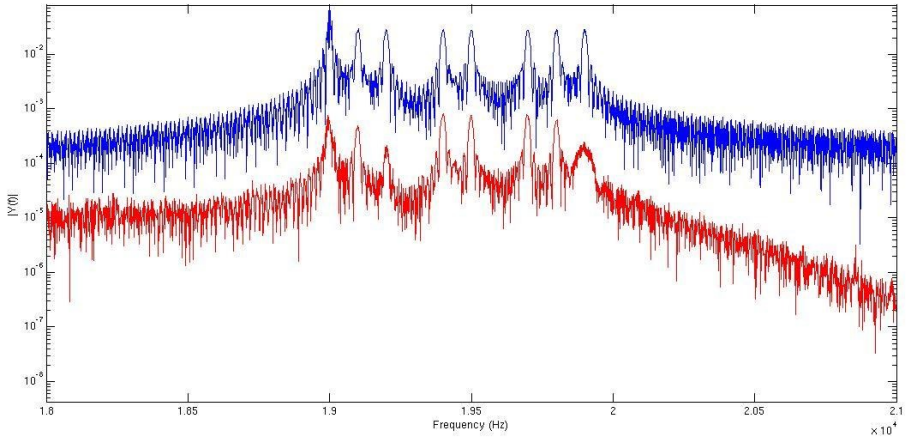


Figure 27. Cross-correlation signal imposed with recording taken at 1 m mark.

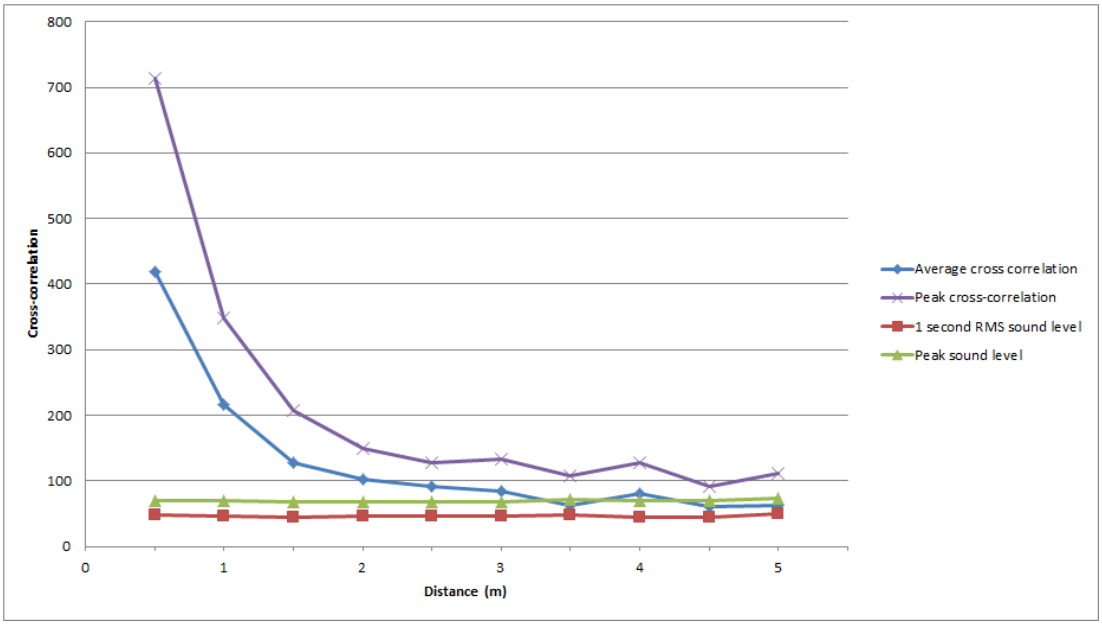


Figure 28. Effect of distance to cross-correlation PCM maximum values.

7.1.2. Frequency Transformation

Appendix 3 has results for frequency transformation testing. We tested final frequency transformation based solution with several different types of phones results which are summarized in Figure 29. Appendix 4 shows detection chance for frequency transformation method in more detailed form. We can see that previously mentioned valley of death effect can also be seen with the frequency transformation method but error correcting code can compensate most of as we can see from same figure (channel total detections and corrected). Figures also show percentage of false positives and discarded samples.

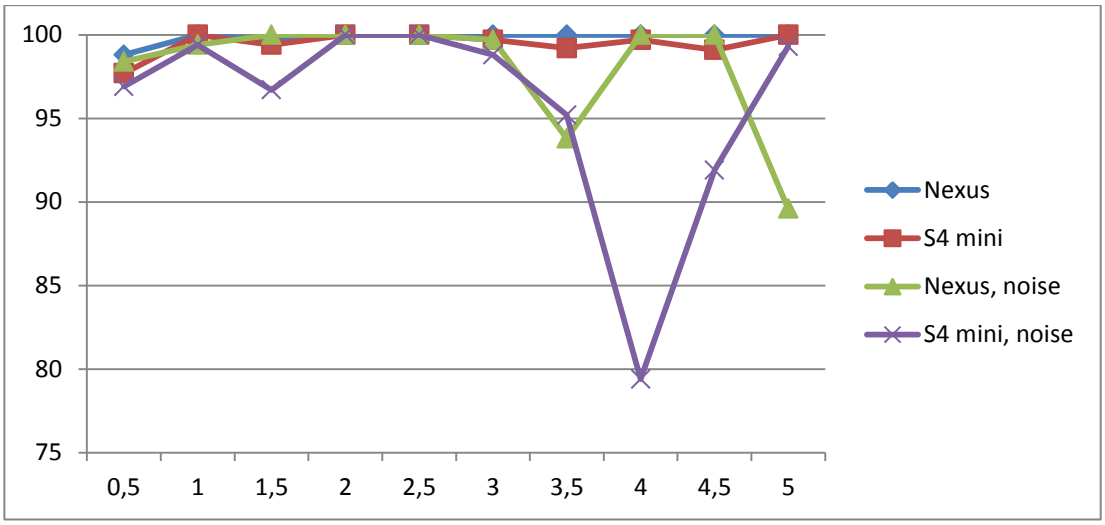


Figure 29. Stationary detection chance with frequency transformation at 0.5 meter intervals.

7.2. Movement and Barriers

We tested effects of moving receiver first in indoors with and without added noise as described in the testing plan. For testing we used three models nexus, S2 plus and S4 mini. Two of the phones were kept in hand while third was in pocket of a sweater while tester followed line in the ground moving towards and away from source. Each test was 10 minutes long each phone recording as many samples as possible during that time.

7.2.1. Cross-Correlation

After initial success with stationary testing cross-correlation, had disappointing results in both moving receiver and signal impaired by barrier test cases. We could not get the method work reliably or fast enough to be used in real time. Basically barriers either caused signal not to be found or false positives moving receiver in other hand slowed detection time to unpractical levels. We also did with multiple signatures with variety of similarity between them and target signature. Summary these results can be seen below in Figure 30. Histogram has cross-correlation values with different kinds of signal similarities. As we can see cross-correlation method works statistically but that is not fast or reliably enough for real time solutions.

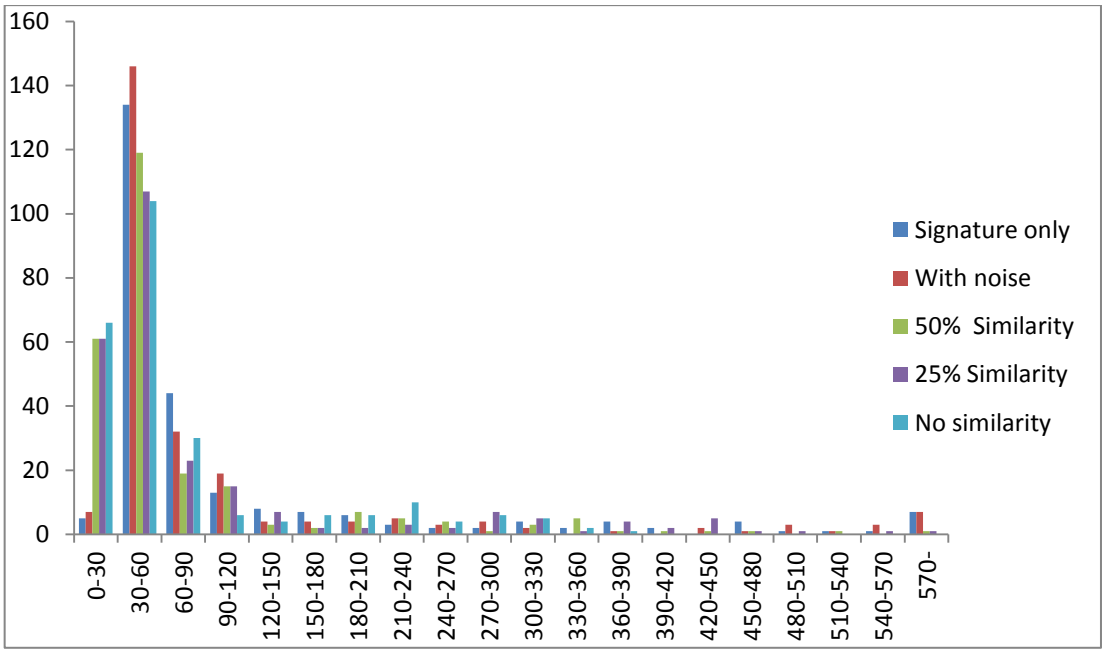


Figure 30. Cross-correlation maximum values with multiple different signatures at 2 meters from receiver in linear scale.

7.2.2. Frequency Transformation

Results are presented in Table 6. We learned that light clothing does not interfere with the system but movement itself will cause about 10% loss in signal detection. Some false positives are observed, depending on phone model false positive chance is 4-5% on better phones (s4 mini and nexus) to 20% on older phones (S2 plus). After error checking is added we see very little false positives, typically 0%. Figure 31 summarizes all movement tests for two phones of the used phones.

Table 8. Results of indoors walking tests with frequency transformattion method.

Phone	**Samples**	**Correct**	**False positives**	**Correctly received**	**Corrected**	**Discarded or null**	**False positives received**
Office		**Results**		**Channel data**			
nexus	352 (704)	314 (89.2%)	0 (0%)	555 (78.8%)	106 (15.1%)	10 (1.4%)	33 (4.7%)
nexus in pocket	336 (672)	301 (89.6%)	2 (0.6%)	540 (80.4%)	86 (12.8%)	11 (1.6%)	35 (5.2%)
S2 plus	440 (880)	252 (57.3%)	0 (0%)	539 (61.3%)	81 (9.2%)	214 (24.3%)	46 (5.2%)
S2 plus in pocket	462 (924)	139 (30.1%)	15 (3.2%)	341 (36.9%)	66 (7.1%)	331 (35.8%)	186 (20.1%)
S4 mini	336 (672)	301 (89.6%)	1 (0.3%)	547 (81.4%)	82 (12.2%)	14 (2.1%)	29 (4.3%)
S4 mini in pocket	320 (640)	278 (86.9%)	0 (0%)	514 (80.3%)	74 (11.5%)	12 (1.9%)	40 (6.3%)
Ambient city noise		**Result**		**Channel data**			
nexus	320 (640)	297 (92.8%)	0 (0%)	557 (87%)	58 (9%)	12 (1.9%)	13 (2%)

nexus in pocket	320 (640)	295 (92.2%)	0 (0%)	506 (79.1%)	106 (16.6%)	5 (0.8%)	23 (3.6%)
S4 mini	320 (640)	275 (85.9%)	0 (0%)	485 (75.8%)	101 (15.8%)	13 (2%)	41 (6.4%)
S4 mini in pocket	336 (672)	311 (92.6%)	1 (0.3%)	574 (85.4%)	64 (9.5%)	13 (1.9%)	21 (3.1%)
S2 plus	462 (924)	424 (91.8%)	2 (0.4%)	787 (85.2%)	90 (9.7%)	11 (1.2%)	36 (3.9%)
S2 plus in pocket	418 (836)	279 (66.7%)	8 (1.9%)	506 (60.5%)	128 (15.3%)	52 (6.2%)	150 (17.9%)

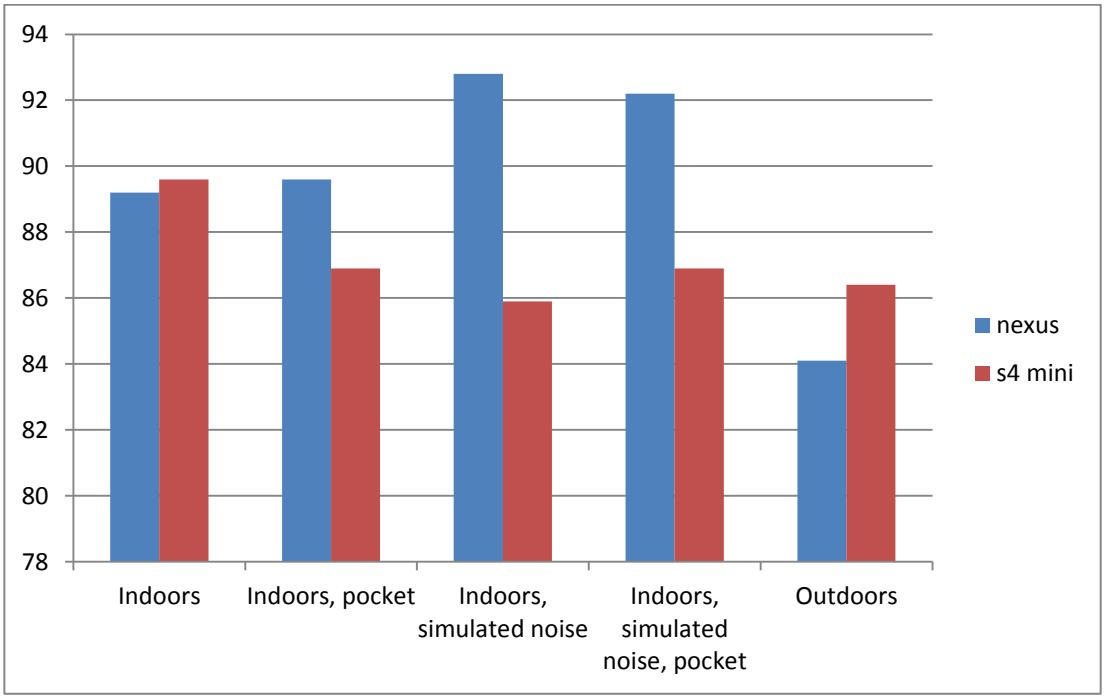


Figure 31. Frequency transformation detection chance moving receiver.

Results shown in Table 7 shows stationary testing with smartphone covered by heavy winter jacket. System worked while but was impaired, basically reliable detection range was halved due weakening signal. Testing results with moving receiver and different clothing resulted with very different results, with light clothing signal was practically unaffected showing the fine line in barrier penetration with audio signals.

Table 9. Stationary testing while obstructed by clothing.

Phone	**Samples**	**Correct**	**False positives**	**Correctly received**	**Corrected**	**Discarded or null**	**False positives received**
S4 mini		Results		Channel data			
Stationary 1m	111 (223)	109 (98.1%)	0 (0%)	215 (96.4%)	5 (2.2%)	0 (0%)	3 (1.3%)
Stationary 2m	115 (231)	64 (55.7%)	2 (1.7%)	20 (8.7%)	129 (55.8%)	47 (20.3%)	35 (15.2%)
Stationary 3m	127 (254)	113 (89%)	0 (0%)	208 (81.9%)	27 (10.6%)	13 (5.1%)	6 (2.4%)
Stationary 4m	170 (341)	112 (65.9%)	2 (1.2%)	192 (56.3%)	45 (13.1%)	28 (8.2%)	76 (22.3%)

Stationary 5m	139 (279)	8 (5.7%)	12 (8.6%)	17 (6.1%)	3 (1.1%)	211 (75.6%)	48 (17.2%)
Nexus		Result		Channel data			
Stationary 1m	120 (241)	87 (72.5%)	0 (0%)	22 (9.1%)	158 (65.6%)	50 (20.7%)	11 (4.6%)
Stationary 2m	122 (244)	31 (25.4%)	0 (0%)	6 (2.5%)	89 (36.5%)	138 (56.6%)	11 (4.5%)
Stationary 3m	127 (255)	1 (0.7%)	41 (32.2%)	3 (1.1%)	0 (0%)	86 (33.7%)	166 (65.1%)
Stationary 4m	175 (351)	0 (0%)	2 (1.1%)	1 (0.2%)	0 (0%)	328 (93.4%)	22 (6.3%)
Stationary 5m	129 (259)	0 (0%)	2 (1.5%)	0 (0%)	0 (0%)	249 (96.1%)	10 (3.9%)

7.3. Error Correction Codes and Detection

Even using simple repetition code offers significant improvement especially in amount of false positives. While testing, we also noticed that errors were usually only in one or two bands. Next task was to try and see if we could improve system using more advanced error correcting codes in this case we used hamming (12 4) coding (8 data bit and 4 parity bits) which to lead about 20% increase in correct signal detection compared to plain 8 bit system. Repetition code show better correct signal percentage but has only 1/16 of capacity compared to hamming coding (16 codes versus 256 codes respectively).

We also calculated how error checking affected the system from the Table 8 we can see real effect of different error checking methods calculated from same measurement data. We can see that amount of false positives is not increased as much as simplified model predicted but errors are still being created. From Table 9 we can see even though double checking in theory doubles detection time reality is that effect is usually less than that due increased amount of correct results. This best seen when we calculate time to find correct results and compare singular checking to double checking, we notice that in reality difference is less than double.

Table 10. Real effects of error checking methods to detection probabilities.

Phone	Samples	Correct	False positives	Correctly received	Corrected	Discarded or null	False positives received
S4 mini		Results		Channel data			
S4 mini [air con] 2 consecutive	304 (608)	151 (49.7%)	6 (2%)	269 (44.2%)	131 (21.5%)	65 (10.7%)	143 (23.5%)
S4 mini [air con] save last sample	304 (608)	168 (55.3%)	7 (2.3%)	269 (44.2%)	131 (21.5%)	65 (10.7%)	143 (23.5%)
S4 mini [air con] 2 of 3	304 (608)	188 (61.8%)	10 (3.3%)	269 (44.2%)	131 (21.5%)	65 (10.7%)	143 (23.5%)

Table 11. Increase in detection time and ration between methods.

Phone name	**Singular checking**	**Double checking**	**Ration**
Nexus office ambient	0.9s/0,788 = 1,142s	0.9s*2/0.892 = 2.017s	1,766
S4 mini in pocket	0.9s/0,442 = 2,036	0.9s*2/0.542= 3.321s	1.285
S2 plus in pocket	0.65s/0.605 = 1,07438s	0.65s*2/0.667= 1.949s	1.814

7.4. Outdoors

Table 10 summarizes outdoor measurements. Since we had already established effect of distance to SONDI system in laboratory conditions we took measurement with only 1m intervals. We also measured maximum range for system which is around 10m. Testing was done using two different models of phone: Nexus and S4 mini. We tested both moving receiver and stationary receiver. Afterwards we compare results to laboratory results and confirmed that system worked also in the outdoors.

Table 12. Stationary and moving receiver testing in outdoors.

Phone	**Samples**	**Correct**	**False positives**	**Correctly received**	**Corrected**	**Discarded or null**	**False positives received**
S4 mini		Results		Channel data			
Stationary 1m	207 (415)	201 (96.9%)	0 (0%)	394 (94.9%)	14 (3.4%)	3 (0.7%)	4 (1.0%)
Stationary 2m	219 (438)	217 (99.1%)	0 (0%)	434 (99.1%)	2 (0.5%)	0 (0%)	2 (0.5%)
Stationary 3m	201 (403)	200 (99.5%)	0 (0%)	401 (99.5%)	1 (0.2%)	0 (0%)	1 (0.2%)
Stationary 4m	231 (462)	231 (100%)	0 (0%)	459 (99.4%)	3 (0.6%)	0 (0%)	0 (0%)
Stationary 5m	234 (469)	216 (92.1%)	2 (0.9%)	379 (80.8%)	65 (13.8%)	6 (1.3%)	19 (4.1%)
Moving receiver walking speed	206 (413)	178 (86.4%)	0 (0.0%)	340 (82.3%)	36 (8.7%)	9 (2.2%)	28 (6.8%)
Nexus		Result		Channel data			
Stationary 1m	205 (411)	200 (97.3%)	0 (0%)	377 (91.7%)	29 (7.1%)	0 (0%)	5 (1.2%)
Stationary 2m	205 (411)	205 (100%)	0 (0%)	405 (98.6%)	6 (1.4%)	0 (0%)	0 (0%)
Stationary 3m	202 (405)	202 (100%)	0 (0%)	405 (100%)	0 (0%)	0 (0%)	0 (0%)
Stationary 4m	240 (481)	240 (100%)	0 (0%)	480 (99.8%)	1 (0.2%)	0 (0%)	0 (0%)
Stationary 5m	235 (471)	234 (99.4%)	0 (0%)	465 (98.7%)	5 1.1%)	0 (0%)	1 (0.2%)
Moving receiver walking speed	208 (416)	175 (84.1%)	0 (0%)	280 (67.3%)	92 (22.1%)	11 (2.6%)	33 (7.9%)

We also run tests in outdoor environments without transmitter to find out how common high frequency sounds are in real life. We recorded and analyzed several samples around Oulu city center and university. We found out that only in 6 cases out of 84 ambient noises had frequencies loud enough to trigger our pilot signal threshold also only 2 cases out of all samples had actually frequencies in data band first one was corrected by error correcting coding and another was caught by double checking system and deemed as false positive. From Appendix 5 we can see frequencies which triggered pilot frequency error were just over our minimum threshold and thus can be dealt with increasing threshold value. From this testing we can deduce that high frequency audio components are uncommon and should not possess problem for our system.

7.5. Power Usage

In the background condition, the test device expended a total of 1% of battery during the 60 minutes measurement period. According to Android's built-in battery interface, SONDI was responsible for around 2% of the total expenditure, with total CPU time of 4 minutes and 8 seconds. In the active condition, total battery consumption during 60 minutes was 5%, out of which SONDI was responsible for 7%. These measurements would indicate that SONDI is rather battery friendly, and even with messaging occurring every 30 seconds, the time to completely drain a battery would exceed 12 hours.

We also run mixed usage power testing to simulate more realistic usage. With nexus phone (screen on most of the time) 15% of total battery capacity was used which of our applications power usage according phone was 5-6% Testing was mixed usage of light (5 minute pauses for 25 minutes), medium usage (demo mode for 20 minutes) and heavy load (5 minutes worth of constant pairing requests, messaging etc). In total runtime of 60 minutes, according phone statistics CPU total time was 5 min 15s and CPU foreground was 2 min 55 s and keep awake 1s. Thus percentage of total battery usage of our application was 0.75% Testing was also replicated with s4 mini with similar results (8% of total battery capacity was consumed with screen mostly off during 60 minutes) and 1-2% total consumption for our application.

8. DISCUSSION

8.1. Testing Process

After establishing basics with stationary tests: We could conclude that with both used methods detection percentage increases relation to transmission power and decrease when distance increases. Both methods are comparative in the stationary single audio barcode tests. Issue can arise with the audio signal reflecting off surfaces, especially in indoor locations. This reverberation can create a so-called multipath effect, where a reflection from a secondary path may overlap with the signal from the line-of-sight path. In this case, the received signal is in fact the combination of signals from all the possible paths that intersect at the position of the microphone.

After stationary testing cross-correlation based detection method was found to be lacking in both speed and accuracy thus frequency transformation method was selected as better of the two tested method and worth of future development. The problem was that we could not separate signatures fast when we had several different possible signatures. There are also some problems with frequency transformation for example when smartphone is covered by clothing range of detection is limited by weakening signal. Effect depends a lot on type of clothing and where phone is kept, increasing volume somewhat counteracts on this type of error as we are dealing with signal attenuation.

Testing also revealed that system is sensitive to phone orientation and placement, as in stationary testing phone signal detection percentage could change dramatically when placement was changed. This was less of problem with moving receiver and frequency transformation method. As mentioned earlier this is mostly due placement as single microphone setups are susceptible to error caused by multipath.

During the outdoors testing biggest obstacle was amount of people moving past test area overloaded the panOULU-WLAN network which we used establishing rough location and communicate with server. Despite it, results were quite good stationary receiver test results were better than indoors testing results and even moving receiver test results were quite similar to indoor walking tests. Depending on place outdoors can actually be easier environment for the system than indoors due lesser multipathing of audio signal.

8.1.1. Effects of Distance

As seen in Appendix 2, distance became a confounding factor with cross-correlation when transmission level kept at 50%, we saw remarkable deterioration in detection percentages at distances of 3 and especially 5 meters, up to a point where SONDI was unable to detect a single signature at 5 meters in the urban condition with the strictest similarity threshold value (80). This was to be expected, as sound dispersers in the air at rate represented with Formula 2.

Raising the transmission power helped negate the effect of distance, and at 70% power cross-correlation performed very well at all distances in both conditions. Increasing transmission power has same effect on both signal detection methods but frequency transformation required less transmission power for similar performance. Both tested methods worked very well in stationary, single signal testing and achieved average accuracy of over 99,0% in all distances as seen from Appendix 2

and 3. Differences between methods come apparent when we move away from stationary system.

From testing we know that 0.001 (normalized MATLAB values for PCM) is about small as we can reliably to discern (Figure 32) and that at 5m mark, signal frequencies that are hardest affected by channel and recording characteristic have lost about 97.5% of its power (from 0.04 to 0.001). This is in line with Formula 2 which predicted power loss of 96% at 5m mark. System could be improved somewhat by selecting frequencies below 20 kHz where dampening is smaller and moving frequencies closer to each other. But then we would risk mixing frequency bands while receiver is moving. We can see symptoms of this in Figure 32d where there are 8 peaks over 0.001 even though only 7 frequency bands were active in original signal. From pictures we can also see effects of band attenuation and limitations of recording equipment. The higher frequencies are suffering more loss than lower frequencies this due microphone response (as we already saw in Figure 2).

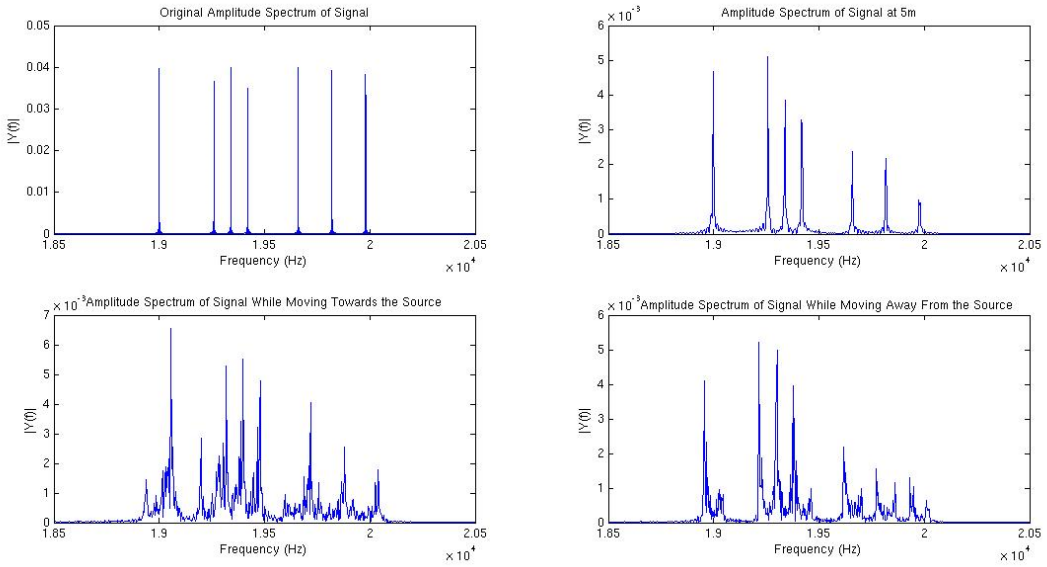


Figure 32. Effects of moving receiver a) audio barcode (7 bit), b) audio barcode as received at 5m, c) and d) audio barcode moving (7 bit).

8.1.2. Effects of Environment

In an enclosed space, factors besides distance may come into play. In real world conditions, there are several common noise sources in the high frequency range that may interfere with signature detection. For example, the air-conditioning unit in the office where SONDI was tested produces noise at the frequencies SONDI signatures use. These high frequency sources can alter or drown the signature, and in worst cases even add false positives. However as we tested these error sources are quite rare and usually have limited range. General observations are valid for both tested signal detection methods.

Cross-correlation measures signal similarity, so everything that can alter or mask the signal can fool the system. Higher similarity threshold values help to migrate this problem, but may also reduce the number of detected signatures and thus increase the average time it takes to find a signature. Figure 33 shows the air conditioning noise imposed on a SONDI signature. From the figure we can see that there is a clear spike

on a frequency just above 19 kHz, which could potentially interfere with the signature if similarity threshold was not set high enough in cross-correlation method. For frequency transformation method same error would falsely flag one the bands as one if zero was sent.

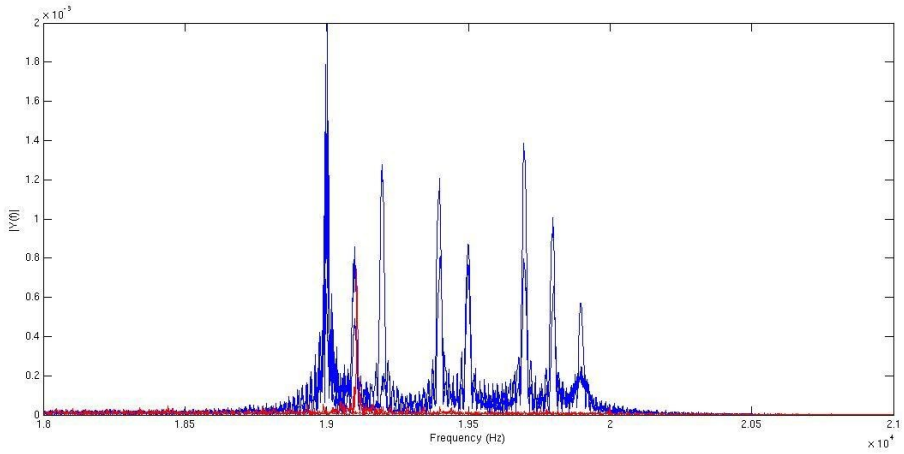


Figure 33. Error source (red) imposed on signature (blue).

During the testing of cross-correlation method, we saw multipath effect at the distance of around 3.5 meters, where a sharp decline in signature detection, a ”valley of death”, was observed as illustrated in Figure 34. For both signal detection methods, added noise amplified this effect. Same problems were encountered with frequency transformation method we can use few error correcting techniques to migrate these problems.

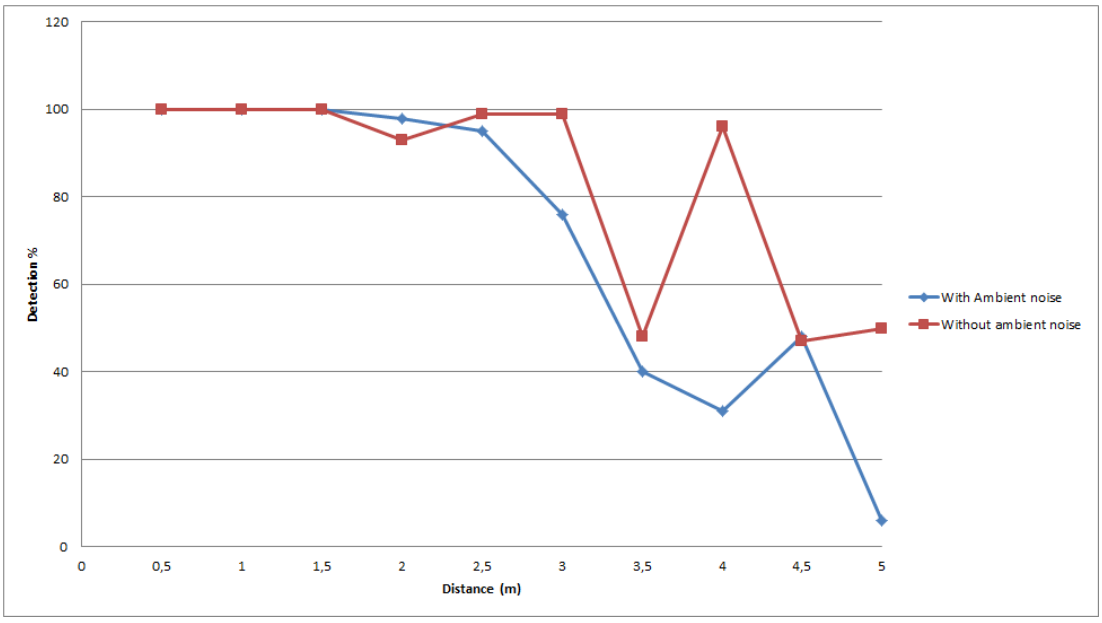


Figure 34. Cross-correlation Signature detection and ”valley of death” with 50% transmission power.

Error sources can be divided into two categories temporal where temporary source like whistle or passing car generates error, then there are fixed error sources which constant and have long duration like previously mentioned air condition noise in laboratory and quiet 19kHz frequency in rotuaari walking avenue. Both error types require different methods to deal with as discussed in error correction chapter. We can catch random false positive by comparing result to either previous and/or following samples. Systematic errors are best countered by adapting signals to channel.

8.2. Method Comparison

After laboratory testing, we sat down and weighted pros and cons of each tested method. While the cross-correlation system worked and recognized signals when user was moving and unlike frequency band method as default does not require anything special to work while receiver is moving problems we obvious. With both methods received signal strength depends on distance and transmission power this is where frequency transformation methods plus sides come to their own. Frequency transformation method can instantly separate between different codes (we always get string of bits) while cross-correlation requires multiple calculations to be sure which signal is received. With everything else equal cross-correlation is slower and more cumbersome of the two methods. After weighing pros and cons of each method we decided to go to final phase of testing with frequency transformation method.

8.2.1. Cross-correlation problems

During the moving receiver and multiple barcodes testing we run into severe problems with cross-correlation method. We discovered that cross-correlation based system was rather difficult to calibrate and produced large number of false positives and errors. With cross-correlation we get value that represents a similarity between received signal and sample this problematic since strongly correlating signal in distance and lightly correlating wrong signal near the source can have same results.

For example: We tested two signals which had 50% similarity (half of the signals were same) at 2.5m mark we got average peak value of 62.1 and highest peak value of 212.3 for false signal which meant that result were non-discernable from result for comparing correct signal to correct sample at the 5m mark. We can make some decision based on statistical analysis but in case of multiple signals when we do not know the detection distance it is impossible to tell signals apart. Only way to make difference between these two results is to test signal against both samples and pick correct result after comparing all results which is highly time consuming.

At the Table 11 we listed several signals with different levels of similarity and values of average cross-correlation and largest value of measured cross-correlation for moving receiver. We can see that on average correct sample has largest correlation values but even the non-matching signal can have larger values of cross-correlation than average value of correct match.

Table 13. Cross-correlation signals values and similarity, moving receiver.

Test file	Similarity	Average cross-correlation	Max cross-correlation
spectre_cc_WaveTest_CC2	100%	111,4485	742,3379345
spectre_cc_WaveTest_CC3	50%	79,1652	834,112505
spectre_cc_WaveTest_CC4	25%	89,65726	580,6916652
spectre_cc_WaveTest_CC5	0%	75,77086	370,0746599

Cross-correlation method was deemed unsuitable mainly due difficulty of handling multiple signals. Using synchronization would help to correct some of the problems we encountered. Using straight cross-correlation as method of signal recognition (or some other kind of signal matching) for this kind of system is not total out realms of possibility but system is rather slow and cumbersome. We could work around some of the problems in the cross-correlation. For example we could use location awareness to select reference sample we think the application is going catch before we start checking so that we do not have to make so many correlations. On the other hand treating audio channel same way as the radio channel showed a lot of potential even relatively simple system just encoding audio signal and snapshotting frequency scale produced usable system.

8.2.2. Frequency Transformation problems

With frequency transformation method had entirely different set of problems for example previously mentioned Doppler compensation, while method worked pretty well after Doppler correction was implemented. System still had significant number of false positives when receiver was moving as we can see from Appendix 4 received columns. As we do not have powerful enough coding to catch all errors, some sort of additional error checking had to be implemented. We cannot ask for transmitter to resend (as typical done in telecommunications when transmitting data) but we can try to catch another recording.

Results from our laboratory testing show that frequency transformation method performs well under normal office and outdoor conditions and the system is fast of enough to work in real time. SONDI works reliably up to distances of 5 meters from the audio source, likely further, but we limited measurements to 5m. There is something we must consider while calibrating system. System usefulness can diminish if signal is found too far from display device and user cannot visually connect the two. Typical scenario for this sort of incident would if device detects transmitter over 5m away or something blocks the visual line of sight for the user.

Fastest we got the work consistently (without double checking) was 650ms for signal recognition. If we managed to improve reliability to levels where double checking is not need or had better error correction. We could use system in shorter ranges which would in turn open wide range of possible applications for system. After taking account all latencies we know that system has detection cycle roughly length of 1.8 seconds or two samplings, we can see that our system in practice our system has minimum detection distance of 2.5 meters when walking straight at the transmitter (worst case scenario).

With average signal detection chance of 93.9% this means when closing in from 5 meter mark we have 1.1% chance to not find signal at all and 98.9% chance to find correct signal at least once. In practice matter is bit more complicated as errors are

more likely to occur when user is far from transmitter also as discussed earlier state of receiver (recording, checking etc...) is random when device enters the cone. Natural if user spends less time in the cone of transmission less chance we have to detect signal (example if user walks in the cone for 2.5 meters she or he 11.8% chance of not finding signal and finding false signal 0.4% chance).

8.3. Audio

As said before our signal is for practical reasons still within human hearing range and some consideration had to be had when creating signals. Unobtrusive signal creation proved to be difficult to realize as frequency changes create human audible pop or noise in the speaker, problem arise from phase discontinuation when frequency is changed and phase is not preserved. This was big problem with cross-correlation where signal is formed from time bands, each with its separate frequency. In worst case scenario playback of signal created 11 audible pops in quick succession which made it sound as loud white noise to human ear. This was corrected doing crossfading between each time band, cross fading is audio mixing technique where old frequency is faded to zero while new frequency is ramped slowly to full power. But even that left some popping effect due short length of sample. Since samples are looped back to back when transmitted which means if our samples repeats every 650ms popping effect can be heard every time recording loops, effect is less annoying than previously but still noticeable when near speaker.

For frequency transformation method as we did not have temporal component other than overall sample length which could be anything from minimum of 350ms to infinity. We could create much simpler system where all used frequencies were played at the same time thus creating only one small popping sound at the end of sample. If we keep sample length long enough, signal is practical silent to humans.

One of the problems is that for transmitting audio barcode outside of human hearing requires fairly high-end loudspeakers are needed. Loudspeaker tend to be tedious to setup or move around as they require separate power, again this is when range is required. System works perfectly fine from any transmitter device as long as it is capable of producing high enough frequencies: Cd-players, other phones, mechanical devices, etc…. Best practice would still be to use a transmitter which the control element can monitor in case of errors: like when signal transmission stops for some reason.

Audio channel should be treated similarly to radio channel and exploration of possibilities should be started by looking into options used in telecommunications field. Next step should be of testing and adding modulation to increase system capability and possibly error resistance. Real time signal receiving and decoding would also increase system usefulness in short ranges. It is also possible to transfer data over audio channel in both ways using same methods but range of communication is severely limited by available hardware in smart phones.

There is wealth of methods available in telecommunications field which can also be used in audio channel and we have barely scratched the surface. For example if we wanted better error correcting capabilities instead of repetition or hamming codes, turbo coding could be used as done in [36] Also powerful enough coding we could be able to get rid of separate error checking but that would require way detect all errors or at least most of them. Also as it would be possible transfer data using audio and thus eliminate need for Bluetooth, internet or WLAN connection.

8.4. Other Problems and Improvements

A persistent problem with multi-device interactions in augmented smart spaces is that of pairing. Pairing is required in all cases where two (or more) previously unfamiliar devices wanting to exchange information serendipitously encounter each other while roaming in a given physical space. In order for two devices to interact and exchange data, they must first somehow become aware of each other this has been major frustration with mobile devices, how to get two device which are in same room to be aware each other.

Typically this device detection is done using some sort of short-range communication scheme like Bluetooth or RFID. We propose another way of pairing devices system, which we call SONDI, which can be seen as complement to the other methods mentioned previously (see related work). Firstly, similarly to Bluetooth or RFID, using an audio channel offers a strong localization component, i.e. the two devices have to be physically close to each other for pairing to be possible.

8.4.1. Location Awareness

Unlike Bluetooth, audio signatures can be used to determine the user's position in relation to the target device by using a directional speaker, whereas Bluetooth is typical omnidirectional. Also we can easily limit the detection area to one room as there is little chance of over penetration of audio signals. Audio signatures are also very device agnostic when compared to RFID. Additional components are not need as the only hardware requirement for SONDI is a microphone which all smartphones have by default.

As a proactive pairing, SONDI can also help make people aware that they are in the presence of a smart device that supports pairing. Current system of using WLAN names to identify target area is not robust or secures enough for real applications and more precise methods should be used if system is to be used in less controlled environment as we cannot really guarantee that phone is connect to particular WLAN. Some other more advanced system should be designed but in general combination of GPS and WLAN should be enough for most applications of this sort.

8.4.2. Performance

Overall performance of SONDI system was satisfactory, system works in real time as best detection took less 1.5 seconds with better smart phone models with less than 1% chance of false positive and over 80% chance to find correct signal while user was moving in noisy environment. There are faster ways of doing FFT but when compared to FFT calculation time to recording time itis hardly biggest problem of the system.

With FFT we have plenty of frequency resolution left so we could expand codes problems arise mainly from required transmission power which can start rise to dangerous levels if we want to maintain range for audio signals. One of the main problems of code design was that there are no established models for channel behavior for audio signals; models would be helpful for creating future systems.

8.5. Future Work

Improvements to the SONDI system could be done in three areas: Barrier penetration, it has less penetration than radio based systems which means SONDI requires some effort from user as heavy clothing can block signal. Some additional development should be done to improve signal penetration in small scale. Two other faults are currently limited scope of signals and signal detection time while it is one fastest secure pairing methods to date there is still room for improvement.

Biggest innovation was not the new pairing method but the medium used to do it, while the chosen realization has its drawbacks as an example using human imperceptible channel prevents people themselves confirming authenticity of pairing request by listening signatures, loss of this functionality is not great concern due its unreliability and irritating nature repeating sounds. Audio signatures could have wide variety of different uses due easily available equipment. Most likely use scenario for SONDI or similar systems will be as part of some other system. For example in addition to previously mentioned examples we could create apply several sound sources and create grid system for local location system.

Next step in tests for this concept would user test where several users would test functionality in different environments possibly with several different unique signatures. Another interesting avenue would be test functionality of user interface in several different kinds of applications. Also effects transmitter placement in relation to environment and target device should be tested.

8.5.1. Platform

Most smartphones have two microphones which allows to record audio in stereo this could be used to improve system: If we record samples in stereo and then separated channels. We could do error checking only after one recording of sample by matching different channels together, however this method is vulnerable to burst and systematical errors since basically both samples would be affected with same errors however stereo recording would prevent some of the multipath errors.

In a real-world setting, people mostly carry their mobile devices in their pockets or inside bags and other containers, especially thick fabrics or full bags may muffle sound waves. One possible way to circumvent the muffling of sound would be to integrate SONDI to wearable technology such as Google Glass or smartwatches, most of which come with integrated microphones for hands free calling. A downside to this approach is, of course, that such wearable devices are still somewhat rare. However, SONDI might offer an interesting use case for emerging wearable technology. Outside wearable computing, the SONDI mobile client could be built to adapt the signature detection algorithm to these conditions using e.g. the ambient light sensor; when there is no ambient light, we can assume that the phone is put away in pocket or bag, and adjust the algorithm accordingly to account for the muffled audio.

System was built and tested in android platform which is still in constant state flux and is missing easy access for programmers to some key features which make harder utilize full potential of system. Previously mentioned audio latency issue is example of this kind of problems. Some improvement to performance could be done by using native code and using more advanced coding of the signal. One of the important avenues research would be signal detection methods which work with modern

compressed audio formats or to see if it is possible to create low disturbance system using lower frequencies that possibly overlap with human hearing.

It is obvious that 256 codes are not enough for all uses. If we have general service like “campus news” which is same in every location we could geographically recycle codes when out of line of sight but if we have specialized or customized service which would require as to differentiate each source as unique this is not possible and we could run out of codes pretty fast. For example: eight directional info point which could differentiate each direction would require eight different codes if we would want to for example cover whole campus are with info points we could do only 32 info points unless we use some additional ways to differentiate code areas like checking location more precisely. We have some unused channel capacity which we could use to extend system 15 data bits plus pilot and use hamming (15 4) encoding then we would have 2048 codes which is probably enough for most use cases.

If we want to expand to using hamming code (15 4) we run to some problems as we are sending more bits we cannot be certain that amount errors will be limited only to 1 or 2 bits. Golay-codes would be better but with they would require minimum of 23 bits plus pilot and/or parity bit. Thus we would have to send even more bits and more bands we have more likely we are having errors in multiple bands and due increased amount of bands require more transmission power. Other possibility could be to use turbo codes. Thus more advanced transmission scheme is needed. Another problem is that if similar systems gain popularity interference from other system can grow into problematic. As mentioned earlier; one possible avenue of interest would modulation, some of the different basic modulation choices are: frequency and amplitude.

8.5.2. Security

Since this is proof of concept project and not intended for more than limited use, in many places security is barebones. In our case security is based on idea “seeing is, believing” if we receive signal and can see the target device we can safely assume that communication request is real and not done by imposter. After initial contact and verification is made, all communication between devices are made through internet which makes it harder to intercept messaging from outside by masquerading as the target device.

Typical form of attack against this kind of security would be hijacking the audio signal, as audio signals of this sort are easy to capture and to replay in different location. Example scenario would be to record audio signature and play it in either same place with bigger volume or different place entirely thus creating fake pairing requester. This would not in no way compromise the system as we use audio as autonomous initiator of communication and connect only to known requesters; at most it would cause confusion to user as form false request. As invader would also need to tap into radio communication while simultaneous taking over or fake whole display setup.

Attack of this kind is not impossible but impractical, however security against this sort of attacks and generally could be enhanced if phone had list of device ID’s as we could use phone to tell user which to device, ID code phone has just picked up. User could then make informed decision based on device name and other clues like proximity of device. For future use better phone location awareness would improve security as we could compare results of audio signature detection to expected

results. We could also do confirmation of physical presence by having phone send a response signal to display element using audio channel thus confirming physical presence of both participants. This would make system extremely hard to attack with man-in-the-middle attacks.

8.5.3. Alternative Setups

We have basically two choices: An active system (loudspeaker) which require outside energy source or passive systems like (for example whistle or wind bell systems) which does not require energy source Passive system have several distinct advantages and disadvantages. Advantages include no need for electricity which enables to deploy system in locations which do not a have power infrastructure, for example outdoor parks. Other advantages are inborn weatherproofing but disadvantages are slow development cycle, poor adjustability and difficult controllability thus it was decided to go ahead with active system which enables more options and quicker adjustment.

With some modification system could be adapted to produce additional information: We already could give estimation of users walking speed based on Doppler correction. Also we could measure time difference between arrival of two signals and calculated distance from source from the time difference. Also discussed earlier we could create positioning system using grid array of transmitters or divide room into sectors.

9. CONCLUSIONS

Modern off-the-shelf consumer technology enables the creation of systems that utilize high-frequency audio for device discovery and pairing. Using such high frequencies that typically fall outside of the human hearing range enables us to use efficient and simple solutions when dealing with sound waves as pairing method. Most importantly, the solution proposed in this thesis can be retrofitted to existing mobile devices without requiring any hardware modifications. Further, as the audio-quality supported by mobile devices continues to increase in quality, we can also expect a wider usable audio bandwidth in the future. Similar solutions to this finding and pairing problem have not been concerned about both distance and real time requirements but we were able create system which works reliably even if receiver is moving and up distance of 5 meters. We tested two methods of audio signal detection and recognition and found frequency scale encoding to be more suited to our needs.

10. REFERENCES

1. Aalto, L., Göthlin, N., Korhonen, J., & Ojala, T. (2004). Bluetooth and WAP push based location-aware mobile advertising system. *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services,* 49-58.
2. Alt, F., Kubitza, T., Bial, D., Zaidan, F., Ortel, M., Zurmaar, B., . . . Schmidt, A. (2011). Digifieds: Insights into deploying digital public notice areas in the wild. *Proceedings of the 10th International Conference on Mobile and Ubiquitous Multimedia,* 165-174.
3. Bassia, P., Pitas, I., & Nikolaidis, N. (2001). Robust audio watermarking in the time domain. *Multimedia, IEEE Transactions On, 3*(2), 232-241.
4. Boney, L., Tewfik, A. H., & Hamdy, K. N. (1996). Digital watermarks for audio signals. *Multimedia Computing and Systems, 1996., Proceedings of the Third IEEE International Conference On,* 473-480.
5. Brandenburg, K. (1999). MP3 and AAC explained. *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding,*
6. Breen, C., & Dahlbom, C. (1960). Signaling systems for control of telephone switching. *Bell System Technical Journal, 39*(6), 1381-1444.
7. Brown, L. G. (1992). A survey of image registration techniques. *ACM Computing Surveys (CSUR), 24*(4), 325-376.
8. Carroll, A., & Heiser, G. (2010). An analysis of power consumption in a smartphone. *USENIX Annual Technical Conference,* 271-285.
9. Chu, S., Narayanan, S., & Kuo, C. (2009). Environmental sound recognition with time–frequency audio features. *Audio, Speech, and Language Processing, IEEE Transactions On, 17*(6), 1142-1158.
10. Cohen, P. R., & Oviatt, S. L. (1995). The role of voice input for human-machine communication. *Proceedings of the National Academy of Sciences of the United States of America, 92*(22), 9921-9927.
11. Core media formats. (2014). Retrieved from http://developer.android.com/guide/appendix/media-formats.html#core
12. Dearman, D., & Truong, K. N. (2009). BlueTone: A framework for interacting with public displays using dual-tone multi-frequency through bluetooth. *Proceedings of the 11th International Conference on Ubiquitous Computing,* 97-100.
13. Fukuju, Y., Minami, M., Morikawa, H., & Aoyama, T. (2003). DOLPHIN: An autonomous indoor positioning system in ubiquitous computing environment. *WSTFEUS,* , 53-56.
14. Goodrich, M. T., Sirivianos, M., Solis, J., Soriente, C., Tsudik, G., & Uzun, E. (2009). Using audio in secure device pairing. *International Journal of Security and Networks, 4*(1), 57-68.
15. Google. (2015). Google chromecast guest mode. Retrieved from https://support.google.com/chromecast/answer/6109292?hl=en
16. Haitsma, J., van der Veen, M., Kalker, T., & Bruekers, F. (2000). Audio watermarking for monitoring and copy protection. *Proceedings of the 2000 ACM Workshops on Multimedia,* 119-122.
17. Harrison, C., Schwarz, J., & Hudson, S. E. (2011). TapSense: Enhancing finger interaction on touch surfaces. *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology,* 627-636.

18. Harrison, C., Tan, D., & Morris, D. (2010). Skinput: Appropriating the body as an input surface. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* 453-462.
19. Harrison, C., Xiao, R., & Hudson, S. (2012). Acoustic barcodes: Passive, durable and inexpensive notched identification tags. *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology,* 563-568.
20. Harter, A., Hopper, A., Steggles, P., Ward, A., & Webster, P. (2002). The anatomy of a context-aware application. *Wireless Networks, 8*(2/3), 187-197.
21. Haselsteiner, E., & Breitfuß, K. (2006). Security in near field communication (NFC). *Workshop on RFID Security,* 12-14.
22. Heikkinen, T., Luojus, P., & Ojala, T. (2014). UbiBroker: Event-based communication architecture for pervasive display networks. *Proc.PD-Apps' 14,*
23. Juang, B., & Furui, S. (2000). Automatic recognition and understanding of spoken language-a first step toward natural human-machine communication. *Proceedings of the IEEE, 88*(8), 1142-1165.
24. Jurmu, M., Kukka, H., Hosio, S., Riekki, J., & Tarkoma, S. (2011). Leasing service for networks of interactive public displays in urban spaces. *Advances in Grid and Pervasive Computing,* , 198-208.
25. Kaltenbrunner, M., & Bencina, R. (2007). reacTIVision: A computer-vision framework for table-based tangible interaction. *Proceedings of the 1st International Conference on Tangible and Embedded Interaction,* 69-74.
26. Kukka, H., Kruger, F., Kostakos, V., Ojala, T., & Jurmu, M. (2011). Information to go: Exploring in-situ information pick-up "In the wild". *Human-Computer Interaction–INTERACT 2011,* , 487-504.
27. Kukka, H., Kruger, F., & Ojala, T. (2009). BlueInfo: Open architecture for deploying web services in WPAN hotspots. *Web Services, 2009. ICWS 2009. IEEE International Conference On,* 984-991.
28. Kumar, A., Saxena, N., Tsudik, G., & Uzun, E. (2009). Caveat eptor: A comparative study of secure device pairing methods. *Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference On,* 1-10.
29. Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., & Campbell, A. T. (2010). A survey of mobile phone sensing. *Communications Magazine, IEEE, 48*(9), 140-150.
30. Lange, B. M., Jones, M. A., & Meyers, J. L. (1998). Insight lab: An immersive team environment linking paper, displays, and data. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* 550-557.
31. Lawton, B. W. (2001). Damage to human hearing by airborne sound of very high frequency or ultrasonic frequency.
32. Lazuka, K. (2012). Retrieved from http://circlewithme.tumblr.com/post/25893923940/proximity-pairing-sound-waves
33. Lee, S., & Ho, Y. (2000). Digital audio watermarking in the cepstrum domain. *Consumer Electronics, IEEE Transactions On, 46*(3), 744-750.
34. Liu, Y., Yang, J., & Liu, M. (2008). Recognition of QR code with mobile phones. *Control and Decision Conference, 2008. CCDC 2008. Chinese,* 203-206.

35. Lopes, P., Jota, R., & Jorge, J. A. (2011). Augmenting touch interaction through acoustic sensing. *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces,* 53-56.
36. Loytynoja, M., Keskinarkaus, A., Cvejic, N., & Seppänen, T. (2008). Watermark-enabled value added services to broadcast audio. *Digital Ecosystems and Technologies, 2008. DEST 2008. 2nd IEEE International Conference On,* 388-396.
37. Memarovic, N., Langheinrich, M., Cheverst, K., Taylor, N., & Alt, F. (2013). P-LAYERS--A layered framework addressing the multifaceted issues facing community-supporting public display deployments. *ACM Transactions on Computer-Human Interaction (TOCHI), 20*(3), 17.
38. National Coordination Office for Space-Based Positioning, Navigation, and Timing. (2014). GPS accuracy. Retrieved from http://www.gps.gov/systems/gps/performance/accuracy/#difference
39. National Physical Laboratory. (2014). Acoustics. Retrieved from http://www.npl.co.uk/educate-explore/factsheets/acoustics/
40. Ojala, T., & Kukka, H. (2009). A digital city needs open pervasive computing infrastructure. *Proc.of Digital Cities, 6*
41. Peng, C., Shen, G., Zhang, Y., Li, Y., & Tan, K. (2007). Beepbeep: A high accuracy acoustic ranging system using cots mobile devices. *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems,* 1-14.
42. RabbitMQ - messaging that just works. (2014). Retrieved from http://www.rabbitmq.com/download.html
43. Radio-Electronics.Com. (2015). Bluetooth security. Retrieved from http://www.radio-electronics.com/info/wireless/bluetooth/security.php
44. Reference: AudioRecord class. (2014). Retrieved from http://developer.android.com/reference/android/media/AudioRecord.html
45. Rekimoto, J., & Ayatsuka, Y. (2000). CyberCode: Designing augmented reality environments with visual tags. *Proceedings of DARE 2000 on Designing Augmented Reality Environments,* 1-10.
46. Sharifi, M., Payne, T., & David, E. (2006). Public display advertising based on bluetooth device presence. *Mobile Interaction with the Real World (MIRW 2006),* , 52.
47. Smith, A., Balakrishnan, H., Goraczko, M., & Priyantha, N. (2004). Tracking moving devices with the cricket location system. *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services,* 190-202.
48. Sozer, E. M., Stojanovic, M., & Proakis, J. G. (2000). Underwater acoustic networks. *Oceanic Engineering, IEEE Journal Of, 25*(1), 72-83.
49. Stansfeld, S. A., & Matheson, M. P. (2003). Noise pollution: Non-auditory effects on health. *British Medical Bulletin, 68*(1), 243-257.
50. Stojanovic, M., & Preisig, J. (2009). Underwater acoustic communication channels: Propagation models and statistical characterization. *Communications Magazine, IEEE, 47*(1), 84-89.
51. Swanson, M. D., Zhu, B., Tewfik, A. H., & Boney, L. (1998). Robust audio watermarking using perceptual masking. *Signal Processing, 66*(3), 337-355.
52. Toombs, C. (2014). Android "L" promises to drastically reduce microphone latency and boost maximum audio quality. Retrieved from

http://www.androidpolice.com/2014/06/30/android-l-promises-to-drasticallyreduce- microphone-latency-and-boost-maximum-audio-quality/

53. Uzun, E., Karvonen, K., & Asokan, N. (2007). Usability analysis of secure pairing methods. *Financial cryptography and data security* (pp. 307-324) Springer.
54. Vallidis, N. M. (2002). WHISPER: A spread spectrum approach to occlusion in acoustic tracking.
55. Wang, A. (2003). An industrial strength audio search algorithm. *ISMIR,* 7-13.
56. Weaver, A., & Newell, N. (1954). In-Band Single-Frequency signaling. *Bell System Technical Journal, 33*(6), 1309-1330.
57. Wiley, R. H., & Richards, D. G. (1978). Physical constraints on acoustic communication in the atmosphere: Implications for the evolution of animal vocalizations. *Behavioral Ecology and Sociobiology, 3*(1), 69-94.

11. APPENDIX

Appendix 1. Extract from Genelec 1029 user manual.

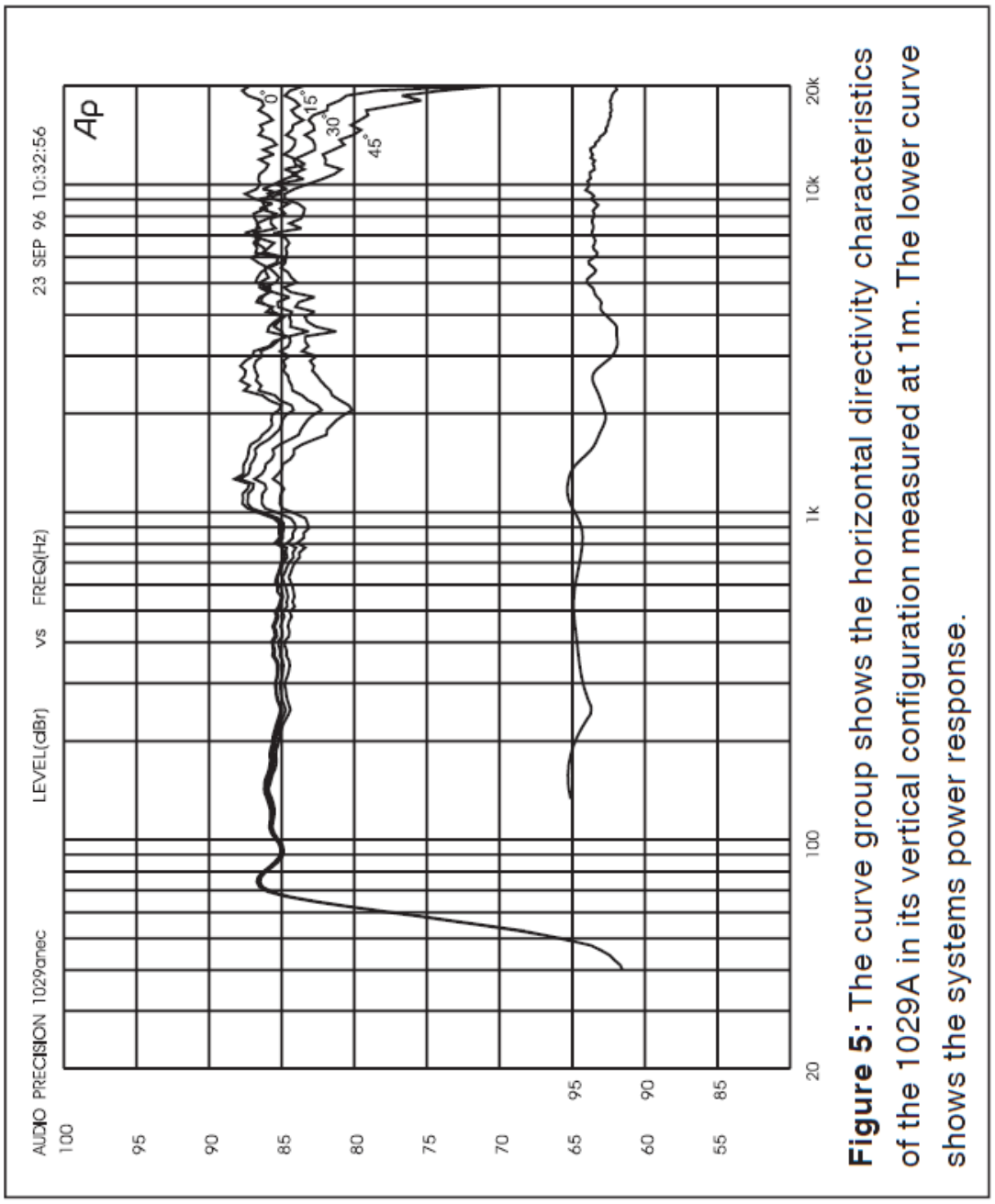


Figure 5: The curve group shows the horizontal directivity characteristics of the 1029A in its vertical configuration measured at 1m. The lower curve shows the systems power response.

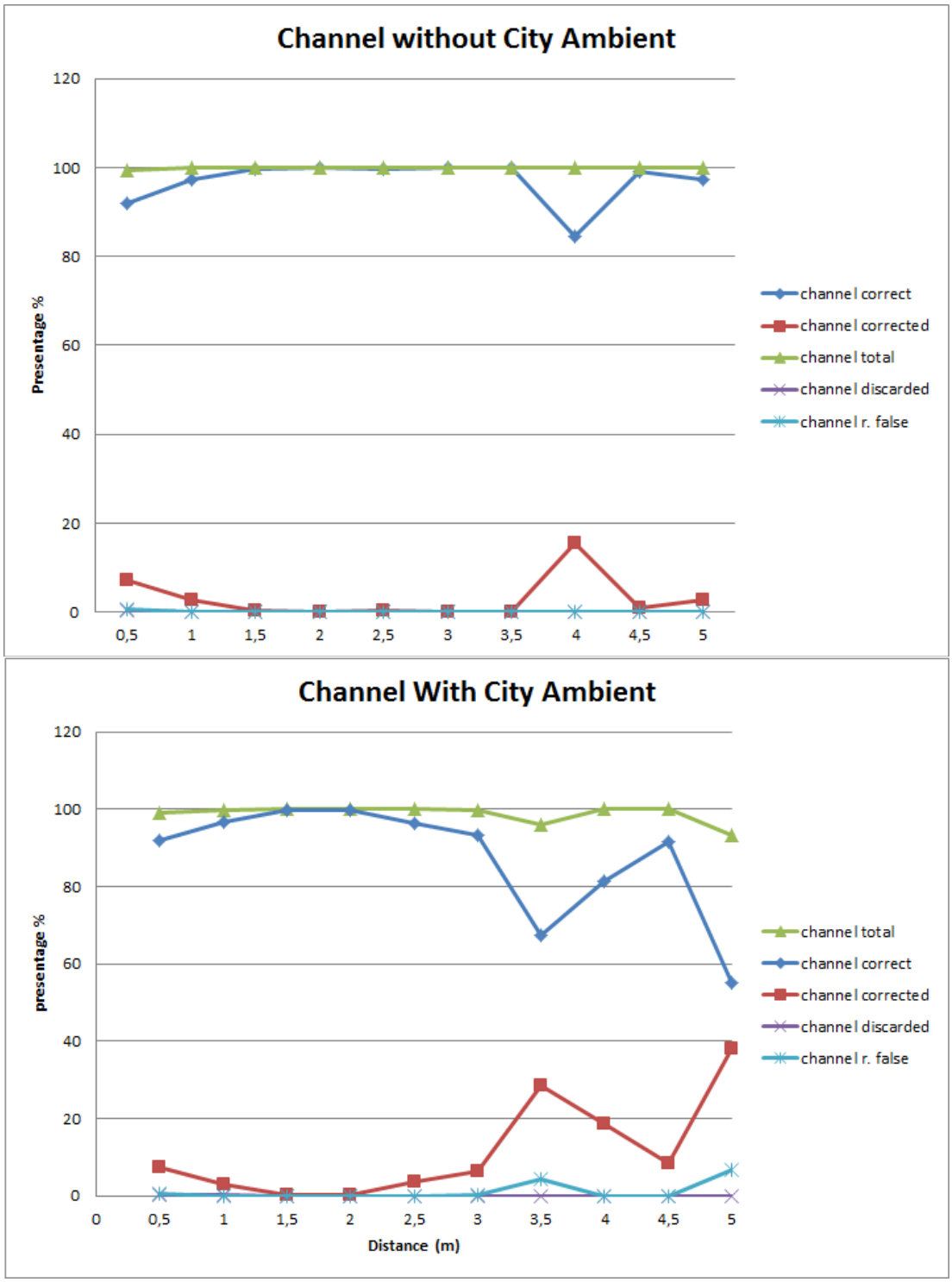
Appendix 2. Cross-correlation stationary testing.

Distance	Power	Detection success %	Average detection time	Detection success %	Average detection time
Threshold value 60		Without noise		With noise	
1m	50 %	100	1,677	100	1,677
	60 %	100	1,677	100	1,677
	70 %	100	1,677	100	1,677
3m	50 %	99	1,693939	76	2,206579
	60 %	100	1,677	99	1,693939
	70 %	99	1,693939	99	1,693939
5m	50 %	50	3,354	6	27,95
	60 %	99	1,693939	82	2,045122
	70 %	100	1,677	100	1,677
Threshold value 70		Without noise		With noise	
1m	50 %	100	1,677	100	1,677
	60 %	100	1,677	100	1,677
	70 %	100	1,677	100	1,677
3m	50 %	82	2,045122	52	3,225
	60 %	96	1,746875	96	1,746875
	70 %	97	1,728866	95	1,765263
5m	50 %	29	5,782759	1	167,7
	60 %	89	1,88427	69	2,430435
	70 %	99	1,693939	99	1,693939
Threshold value 80		Without noise		With noise	
1m	50 %	100	1,677	100	1,677
	60 %	100	1,677	100	1,677
	70 %	100	1,677	100	1,677
3m	50 %	70	2,395714	22	7,622727
	60 %	75	2,236	71	2,361972
	70 %	89	1,88427	86	1,95
5m	50 %	16	10,48125	0	never
	60 %	79	2,122785	28	5,989286
	70 %	98	1,711224	98	1,711224

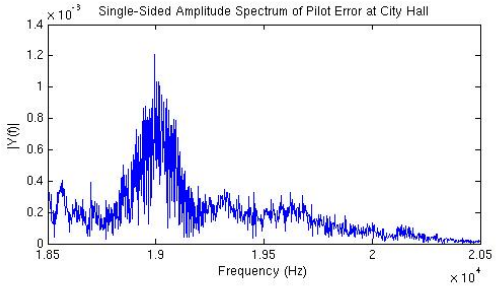
Appendix 3. Indoors stationary testing results, frequency transformation.

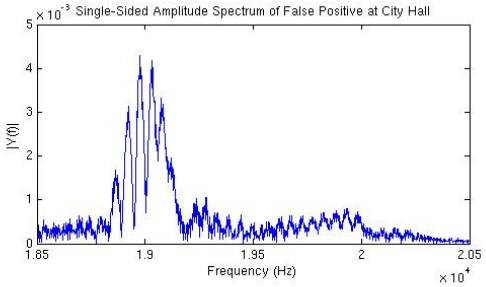
Phone	Samples	Correct	False positives	Correctly received	Corrected	Discarded or null	False positives received
S4 mini		Results		Channel data			
Stationary 0.5m	304 (608)	297 (97.7%)	0 (0%)	567 (93.3%)	34 (5.6%)	2 (0.3%)	5 (0.8%)
Stationary 1m	352 (704)	352 (100%)	0 (0%)	703 (99.9%)	1 (0.1%)	0 (0%)	0 (0%)
Stationary 1.5m	334 (672)	332 (99.4%)	0 (0%)	621 (92.4%)	49 (7.3%)	1 (0.1%)	1 (0.1%)
Stationary 2m	336 (672)	336 (100%)	0 (0%)	667 (99.3%)	6 (0.7%)	0 (0%)	0 (0%)
Stationary 2.5m	320 (640)	320 (100%)	0 (0%)	637 (99.5%)	3 (0.5%)	0 (0%)	0 (0%)
Stationary 3m	336 (672)	335 (99.7%)	0 (0%)	384 (57.1%)	287 (42.7%)	0 (0%)	1 (0.2%)
Stationary 3.5m	384 (768)	381 (99.2%)	1 (0.3%)	636 (82.8%)	128 (16.7%)	1 (0.1%)	3 (0.4%)
Stationary 4m	320 (640)	319 (99.7%)	0 (0%)	556 (86.9%)	83 (13%)	1 (0.2%)	0 (0%)
Stationary 4.5m	336 (672)	333 (99.1%)	1 (0.3%)	414 (61.6%)	254 (37.8%)	0 (0%)	4 (0.6%)
Stationary 5m	256 (512)	256 (100%)	0 (0%)	502 (98%)	10 (2%)	0 (0%)	0 (0%)
Nexus		Result		Channel data			
Stationary 0.5m	320 (640)	316 (98.8%)	0 (0%)	589 (92%)	47 (7.3%)	1 (0.2%)	3 (0.5%)
Stationary 1m	352 (704)	352 (100%)	0 (0%)	686 (97.4%)	18 (2.6%)	0 (0%)	0 (0%)
Stationary 1.5m	336 (672)	335 (99.7%)	0 (0%)	669 (99.6%)	2 (0.3%)	0 (0%)	1 (0.1%)
Stationary 2m	336 (672)	336 (100%)	0 (0%)	671 (99.9%)	1 (0.1%)	0 (0%)	0 (0%)
Stationary 2.5m	320 (640)	320 (100%)	0 (0%)	633 (98.9%)	7 (1.1%)	0 (0%)	0 (0%)
Stationary 3m	336 (672)	336 (100%)	0 (0%)	672 (100%)	0 (0%)	0 (0%)	0 (0%)
Stationary 3.5m	384 (768)	384 (100%)	0 (0%)	768 (100%)	0 (0%)	0 (0%)	0 (0%)
Stationary 4m	320 (640)	320 (100%)	0 (0%)	541 (84.5%)	99 (15.5%)	0 (0%)	0 (0%)
Stationary 4.5m	352 (704)	352 (100%)	0 (0%)	698 (99.1%)	6 (0.9%)	0 (0%)	0 (0%)
Stationary 5m	256 (512)	256 (100%)	0 (0%)	498 (97.3%)	14 (2.7%)	0 (0%)	0 (0%)

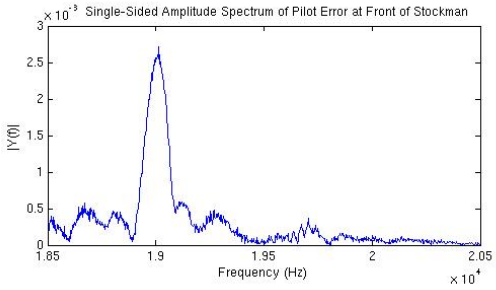
Appendix 4. Frequency transformation channel (Nexus).

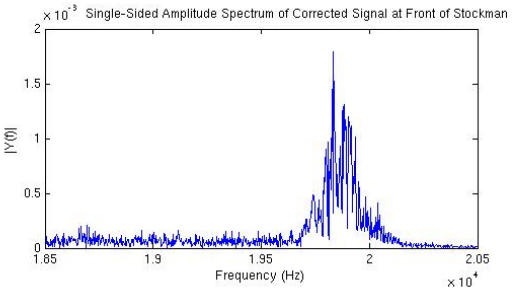


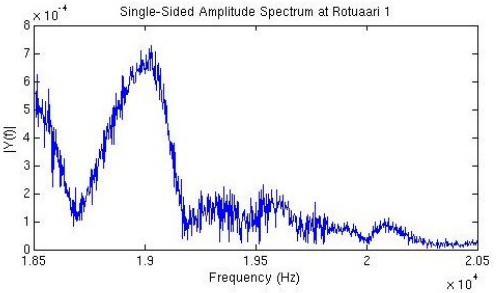
Appendix 5. Channel errors caused by real ambient noise in city environment.

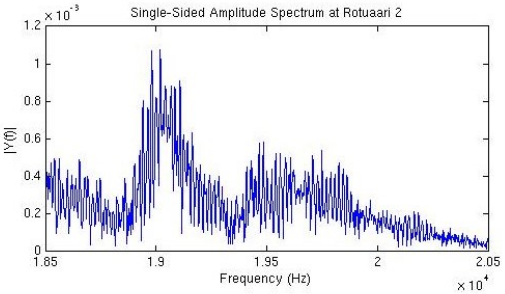


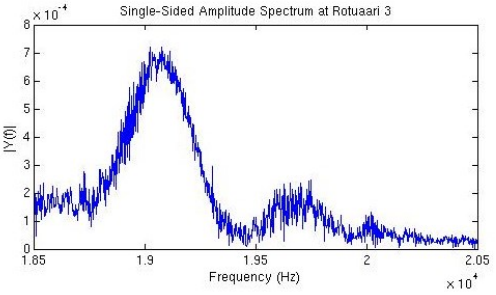


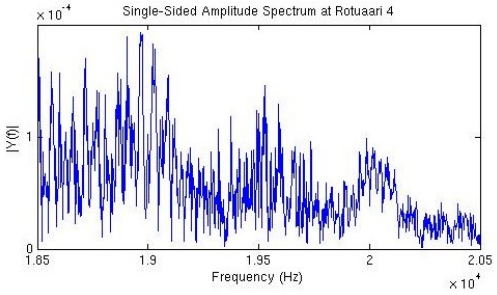












Appendix 6. RabbitMQ message template.

Message template


```
{
    "target_id":"campus-1",
    "origin_id":"ubi-hotspot-15",
    "session":"yTrEUtQM1f2gtnNiIas5iAy0JeDMQWr4",
    "timestamp":"2014-08-28T11:51:47.746Z",
    "log":"true",
    "type":"type_of_the_message",
    "data":{
            “common_session_id”:”rMQ1f2gtrHisDonCltyTa5yJeEUtSDA4”
            “session_start_time”:”2014-08-28T11:51:47.746Z”
            “session_stop_time”:”2014-08-28T11:52:12.512Z”
            “username”:”teppotesti”
            “user_id”:” MQr1fegtrHisDCEUtSDAltyPu9yonJe4”
            “ubi_name”:”ubi-1”
            “pairing”:”yes”
        }
}
```