

Interactive Spatial Auditory Display of Graphical Data

Timothy Neate

Master of Science (by research) in Music Technology

University of York
Electronics

November, 2013

Abstract

This thesis describes a series of experiments designed to test auditory display techniques for interactive image sonification on a tablet computer. The aim of these experiments was to evaluate new sonic methods developed for finding graphical features on a tablet computer screen for both regular size, and extended (larger than physical screen size) displays. A series of tests were designed to evaluate the techniques, and determine the effectiveness of binaural information in a series of goal-oriented searching tasks. The results show that, by using the techniques developed, users can locate graphical features on regular and extended displays, using sound alone. Additionally, it was found that users could improve their search times when using auditory information to improve their ability to search extended displays.

Contents

1	Introduction	1
1.1	The Primary Hypothesis	2
1.2	Thesis Structure	3
1.3	Reading this Thesis	4
2	Theoretical Literature Review	5
2.1	Sonification	5
2.2	Interacting with a Sonification System	7
2.3	Mapping Data to Sound	9
2.3.1	Auditory Icons	9
2.3.2	Earcons	9
2.3.3	Spearcons	10
2.3.4	Audification	11
2.3.5	Parameter Mapping	11
2.3.6	Model Based Sonification	12
2.4	Sonification Guidelines	13
2.4.1	Subjective Auralization	13
2.4.2	Ergonomics	15
2.4.3	Human Learning Capabilities	16
2.5	Spatial Audio	17
2.5.1	Loudspeaker Arrays	19
2.5.2	Binaural Audio	21
2.6	Image Processing Conventions	23
2.6.1	Digital Images (Raster Images)	23
2.6.2	HSV – Hue Saturation Value	25
2.7	iOS Development Outline	25
2.8	Core Graphics	27
2.8.1	Core Audio	27
2.9	Alternative Audio Processing Environments for iOS	28
2.9.1	libpd	28

CONTENTS

2.9.2	The Amazing Audio Engine	30
2.9.3	iOS-Csound API	31
3	Relevant Projects in the Area	34
3.1	Interactive Sonification for Task Monitoring	34
3.2	Sonification of Graphical Data	38
3.3	Spatial Auditory Display	45
4	Research Agenda	50
4.1	Hypothesis Discussion	50
4.2	Summary of ‘Relevant Projects in the Area’ Section	52
4.3	Refined Hypothesis	53
4.4	Aims and Objectives	54
4.4.1	Aims	54
4.4.2	Objectives	55
5	Preliminary Test	58
5.1	An Introduction	58
5.2	The Experiments	60
5.3	Initial Test Hypotheses	60
5.3.1	Hypotheses – Test 1	61
5.3.2	Hypotheses – Test 2	62
5.4	Designing the Sound Examples	62
5.4.1	Designing Test 1	63
5.4.2	Designing Test 2	64
5.5	The Experimental Procedure	65
5.6	The Demographics	66
5.7	The Results	67
5.7.1	Test 1 – The Results	67
5.7.2	Test 2 – The Results	69
5.8	Test Conclusions	73
5.8.1	Test 1	73
5.8.2	Test 2	75
5.9	Limitations of Pilot Test	76

6	Design and Implementation	78
6.1	Location of Image Features	80
6.1.1	Philosophy	80
6.1.2	Design	80
6.1.3	Initial Evaluation	81
6.1.4	Final Implementation	81
6.1.5	Discussion	85
6.2	Extending the Display	85
6.2.1	Philosophy	85
6.2.2	Design	86
6.2.3	Initial Evaluation	87
6.2.4	Final Implementation	88
6.2.5	Discussion	91
6.3	Touch Interaction	92
6.3.1	Philosophy	92
6.3.2	Design	93
6.3.3	Initial Evaluation	93
6.3.4	Implementation	93
6.3.5	Conclusions	95
6.4	Touch to Feature Mapping	96
6.4.1	Philosophy	96
6.4.2	Design	98
6.4.3	Implementation	98
6.4.4	Conclusions	101
6.5	The Audio Engine (Csound)	101
6.5.1	Sending Data from iOS to Csound	102
6.5.2	Receiving Data from iOS in Csound	103
6.6	Auditory Mappings	103
6.6.1	Pulse Train	103
6.6.2	Binaural Panning	106
6.6.3	Colour Sounds	107
6.6.4	Volume	109
6.6.5	Alert Sound	109

CONTENTS

6.6.6	‘Boing’ sound	110
6.6.7	Dealing with Interaction	111
7	Methodology and Testing Procedure	112
7.1	Test Methodology	112
7.2	Test Design	113
7.2.1	Test 1: regular screen size searching	113
7.2.2	Test 2: searching extended displays	116
7.3	Experimental Procedure	119
7.3.1	Allocation of Groups	120
7.3.2	Demographics and Ability to Perceive Binaural	120
7.3.3	Test Description and Practice Examples	121
7.3.4	Undertaking and Recreating the Tests	122
7.4	Technical Setup	122
7.5	Data to be Gathered	126
7.5.1	Tests 1.1, 2.1, 2.3, 2.4, and 2.5	127
7.5.2	Tests 1.2 and 2.2	127
7.5.3	Test 1.3	127
8	Results	129
8.1	Participant Demographics	129
8.2	Test Results	130
8.2.1	Test 1.1: finding a black dot [with/without binaural]	131
8.2.2	Test 1.2: three coloured dots [Group B not told colour mappings]	133
8.2.3	Test 1.3: picture identification [both groups with same mappings]	135
8.2.4	Test 2.1: black dot in a large image [with/without binaural]	137
8.2.5	Test 2.2: three coloured dots in a large image [with/without binaural]	140
8.2.6	Tests 2.3, 2.4 and 2.5: black dot in a large image [no visual restriction]	143

CONTENTS

8.3	Results Conclusions	146
8.3.1	Comparing Binaural and Monaural	147
8.3.2	Comparing Audio and Visual to Visual Alone	148
9	Conclusions and Further Work	149
9.1	Review of Hypothesis	149
9.2	Major Conclusions from Studies	149
9.3	Recommendations for Further Work	151
9.3.1	Additional Testing of Visual vs. AudioVisual Tests	151
9.3.2	Investigate the Effect of Binaural Audio Using Other Techniques	152
9.4	Potential Applications	152
9.4.1	Adapting Computer Interfaces for the Visually Impaired or Visually Engaged	152
9.4.2	Allowing Users to Search Vast Images Such as Cervical Cancer Data	153
9.4.3	Using Spatial Audio to Improve User-Experience	154
9.5	Summary of Findings	154
	Appendices	156
	Appendix A ISON 2013 Paper	156
	Appendix B Pilot Study Script	167
	Appendix C Csound for iOS API – A Beginner’s Guide	172
	Appendix D Test Script (Group A)	213
	Appendix E Test Script (Group B)	225
	Appendix F Test Coordinator’s Script	238

List of Tables

2.1	iOS devices	27
5.1	Table of Pure Data patches developed	64
6.1	Touch delegate methods	94
6.2	'hrtfmove2' parameters	107
7.1	Test 1.1 parameters	113
7.2	Test 1.2 colour – frequency parameters	114
7.3	Test 1.2 parameters	115
7.4	Test 1.3 parameters	116
7.5	Test 2.1 parameters	117
7.6	Test 2.2 parameters	118
7.7	Dimensions of tests 2.3, 2.4 and 2.5	119
7.8	Parameters for tests 2.3, 2.4, and 2.5	119
8.1	Results for Test 1.1	132
8.2	Results for Test 1.2	134
8.3	Number of times each picture was chosen	136
8.4	Results for Test 2.1	138
8.5	Results for Test 2.2	140
8.6	Results for Test 2.3, 2.4 and 2.5	146

List of Figures

1.1	Apple’s ‘Spaces’ system	1
2.1	Interactive sonification topic web	6
2.2	Interactive sonification paradigm	8
2.3	Presbycusis in men and women	15
2.4	Physical process of hearing	18
2.5	Pairwise panning	20
2.6	A KEMAR	22
2.7	The pixel coordinate system	24
2.8	RGB colour model	24
2.9	HSV colour model	25
2.10	iOS Devices	26
3.1	The Sofirow system being used	35
3.2	Metaphor-based sonification of arm movement	36
3.3	Haptic vs Pseudohaptic sketch accuracy	37
3.4	‘headcirc’ algorithm	40
3.5	Shapes the participants were tasked with sketching	42
3.6	Traced features of car	43
3.7	Image to sound mapping for The vOICe	44
3.8	The vOICe system running on an Android phone	45
3.9	A user’s perception of the map data	46
3.10	Priority zones within the context	48
5.1	Frequency of Light	59
5.2	Piano with colour	59
5.3	Castel’s piano	60
5.4	Notes in test	64
5.5	Preliminary test setup	66
5.6	Colours users associated with sounds	68
5.7	Colours users associated with sounds (synesthesia)	69
5.8	User’s preferences to sound	69
5.9	Note to colour association	70
5.10	Note to colour association (colours separate)	71

LIST OF FIGURES

5.11	Note to colour association (synaesthetic)	72
5.12	Note to colour association (pitch perfect)	72
5.13	Note to colour association (relative pitch)	73
5.14	Note to colour association (no musical training)	74
6.1	Design overview	79
6.2	Rastering algorithm	82
6.3	Value of colours in pixel	84
6.4	Extending the display	86
6.5	Vector – A to B	97
6.6	Vector – B to A	97
6.7	Magnitude calculation	99
6.8	Proximity zones	104
6.9	Saw-tooth and square-tooth wave	105
6.10	Time and frequency domain representation of colour synth . .	108
7.1	Four pictures used in Test 1.3	116
7.2	Progressively larger displays	119
7.3	The visual restriction device used during the tests	123
7.4	Test schematic	125
7.5	Picture of test setup	126
8.1	Boxplot of Test 1.1	132
8.2	Boxplot of Test 1.2	134
8.3	Boxplot of Test 1.3	137
8.4	Boxplot of Test 2.1	139
8.5	Boxplot of Test 2.2	141
8.6	Boxplot of Test 2.3	143
8.7	Boxplot of Test 2.4	144
8.8	Boxplot of Test 2.5	145
8.9	Comparison of binaural and monaural location times	147

Table of media examples on accompanying CD

Media example	Description
Audio Example 2.1	Apple 'putting in bin' sound
Audio Example 2.2	Apple 'emptying bin' sound
Audio Example 2.3	Apple screenshot sound
Audio Example 2.4	Ascending earcon example
Audio Example 2.5	Decending earcon example
Audio Example 2.6	Spearcon example
Audio Example 2.7	Binaural example
Audio Example 6.1	Pulse train example (sawtooth wave envelope)
Audio Example 6.2	Pulse train example (square wave envelope)
Audio Example 6.3	Filtered saw wave pulse train example
Audio Example 7.1	Medium pitch filtered saw wave
Audio Example 7.2	Low pitch filtered saw wave
Video Example 6.1	Demonstration of extending the bounds of the screen
Video Example 6.2	Demonstration of touch detection
Video Example 6.3	Demonstration of vector calculation
Video Example 6.4	Demonstration of alert zone
Video Example 6.5	Pulse train demonstration
Video Example 6.6	Binaural pulse train demonstration
Video Example 6.7	3D Sonification of three coloured dots
Video Example 6.8	Example of extended display search task
Video Example 9.1	Apple's 'spaces' paradigm

Acknowledgements

First of all I would like to thank my supervisor Andy Hunt for going far beyond the call of duty at every opportunity to make this thesis happen. Without his guidance, support, and passion I would not be the person I am today. I would also like to thank (in no particular order): Abigail Richardson and Nick Arner for helping me write the Csound for iOS Beginner's Guide, Jude Brereton for being my thesis advisor and helping me with some thesis-based structural issues, POWERCYCLE for the mid-MSc world tour, the good people of the Audio Lab at York for the cake-fuelled round table discussions, Lisa Burch at TFTV for helping me get decent video editing equipment, Browns of Heslington for their glorious sandwiches, the Stack Overflow community for having the answer to every error, Norberto Degara for taking me under his wing out here in Germany, and the people at Fraunhofer IIS for giving me the opportunity to work out here during my write-up period. Finally, I would like to thank my girlfriend Alena for her patience, her support, and for enabling me to retain some sanity while writing up this thesis.

Declaration

This thesis is the work of Timothy Neate, unless otherwise noted. The work was done under the guidance of Dr. Andy Hunt at The University of York, UK. Some of the material in this thesis was presented in the following paper (included in *Appendix A* accompanying this thesis):

Neate, T. and Hunt, A. (2013). Interactive Spatial Auditory Display of Graphical Data. *Proceedings of the 4th Interactive Sonification Workshop*, Erlangen.

For my parents.

1 Introduction

As the displays we interact with daily become smaller and more content rich due to the acceleration in computing power in recent times, it is evident that screen real estate is becoming increasingly limited. This means that the human-computer interaction community is constantly exploring new methods of fitting more content on small displays. Several attempts from the field of auditory display have recently been explored in an effort to expand the screen's visual content into the auditory domain [Meijer, 1992] [Sanchez, 2010] – effectively reinterpreting what we perceive from a visual display by means of sound. The recent growth in audio processing and interaction modes on portable devices means that spatial auditory display has the potential to become a viable means of extending the visual domain [McGookin and Brewster, 2001] [Heuten et al., 2006], especially in the context of smaller screen devices.

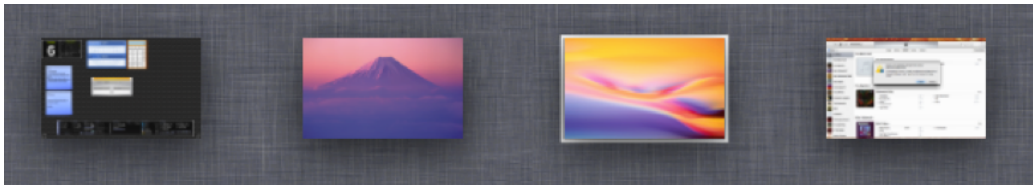


Figure 1.1: ‘Spaces’ – a system for creating multiple desktops

An increase in computing power, and advances in the world of materials technology have allowed for the power of desktop computers to be condensed into small hand-held devices such as smartphones and tablet computers. This introduces a new set of challenges. From an interaction standpoint, modern tablet computers typically offer more than the mouse and keyboard setup – elaborate touch-based gestures, voice activation, haptic feedback, and more. However, it is evident that we are restricted by the decreasing size of devices, and that we are constantly trying to design interfaces that virtually extend the screen size – for example, a multiple desktop paradigm, such as Apple’s ‘Spaces’ (Figure 1.1).

1.1 The Primary Hypothesis

This thesis aims to explore the development of techniques that allow a user to locate graphical features on a tablet-computer display by means of interactive auditory feedback. Additionally, these techniques can be applied to extended displays (displays that are larger than that of the screen) for use in aiding the visually engaged (looking elsewhere) or restricted (not being able to see), or to improve the location of graphical features on large displays by means of adding to visual cues with sound so that the techniques developed should offer a target-based approach in which the characteristics of the required features can be described. Then the target's location can be represented by sound mapping, such that the user can find it without visual cues by filtering out all information that does not match the initial description.

The approach described in this thesis is to first develop techniques, and then test them on a series of progressively more complex images. To begin, a simple black dot on a white screen is considered. This allows for the techniques to be assessed at a fundamental level, determining which work for users, and which do not. Then, the following tests are made more complex by adding more colours, larger images, and different types of image feature.

1.1 The Primary Hypothesis

A major focus of this work is the use of spatial audio in interactive sonification – does it make a significant difference to our ability to locate the image features? Or does it simply get in the way? The primary hypothesis addresses this more explicitly by singling out this parameter:

It is possible to improve a user's understanding of graphical data by using spatial audio to provide interactive auditory feedback.

The next sub-section describes the structured approach to tackling this research question in the rest of this thesis.

1.2 Thesis Structure

Section 2, the ‘Theoretical Literature Review’, covers the technical aspects behind the thesis. Different forms of auditory display are discussed, along with other relevant topics to this thesis, such as spatial audio, image processing, and some programming conventions.

Section 3, ‘Relevant Projects in the Area’, discusses appropriate literature to give a background to the work that follows. The section covers the main areas of this project; Interactive Sonification for Task Monitoring, the Sonification of file pal Data, and Spatial Audio Sonification. The work here is referred to extensively in the following sections, and the information gained provides insight into the sonification techniques developed.

Section 4, ‘Research Agenda’, allows for the knowledge gained in the previous two sections to be discussed, and for the hypothesis and research goals to be refined in light of that information.

Section 5 (Preliminary Test) describes a short test undertaken to determine our association between colour and sound. A series of subjective listening tests was run on participants to determine if people have preconceived ideas about how colour relates to sound. The results gained here inform sound design decisions made in the upcoming design-based sections.

Section 6, ‘Design and Implementation’, discusses the ideas, philosophy, and literature behind each technique developed, before giving a full description of its implementation in software. The methods developed in this section are

1.3 Reading this Thesis

then used to test the main hypothesis and several sub-hypotheses.

Section 7, Design and Methodology, describes the development of user tests to judge the success of the auditory display techniques developed, and to test the primary hypothesis. Each test's design and technical set-up is discussed in depth, along with the data to be gathered during the experiment.

Section 8, 'Results', presents and analyses the information gained from the above user tests. The participants' demographics are explained such that the gist of the group can be gained. Then the results for each test are presented and their implications discussed. The main patterns found in the results are then analysed in more depth such that conclusions may be made.

Section 9, 'Conclusions and Further Work', makes deductions based on the evidence gained from the results. The hypothesis is reviewed and compared to the results. The thesis concludes by listing the potential applications of the findings.

1.3 Reading this Thesis

Each section provides information for the following sections and so it is recommended to read in linear section order for a full understanding to be gained. Throughout the thesis several audio examples, video examples, and other resources (such as appendices) are referred to. Unless stated otherwise, it is possible to access these within the thesis itself, or on the CD that comes with the printed version of this thesis.

2 Theoretical Literature Review

This section describes the key concepts in the project such as interactive sonification, spatial audio and the programming languages and principles. It is written with the assumption that the reader has some knowledge of audio, programming, and signal processing. A theoretical background is provided to aid support the rest of the report. It covers the following topics:

- Sonification – 2.1
- Interacting with a Sonification System – 2.2
- Mapping Data to Sound – 2.3
- Sonification Guidelines – 2.4
- Spatial Audio – 2.5
- Image Processing Conventions – 2.6
- iOS Programming Outline – 2.7
- Core Graphics – 2.8
- Alternative Audio Processing Environments for iOS – 2.9

2.1 Sonification

This section discusses sonification. Beginning with a definition, it covers the topic's constituent parts, along with some theoretical background on each one.

2.1 Sonification

Sonification is defined as:

“The use of non-speech audio to convey information. More specifically, sonification is the transformation of data relations into perceived relations in an acoustic signal for the purposes of facilitating communication or interpretation.” [Hermann and Hunt, 2004, pg. 149]

or more recently:

“The data dependent generation of sound, if the transformation is systematic, objective and reproducible.” [Walker and Nees, 2011]

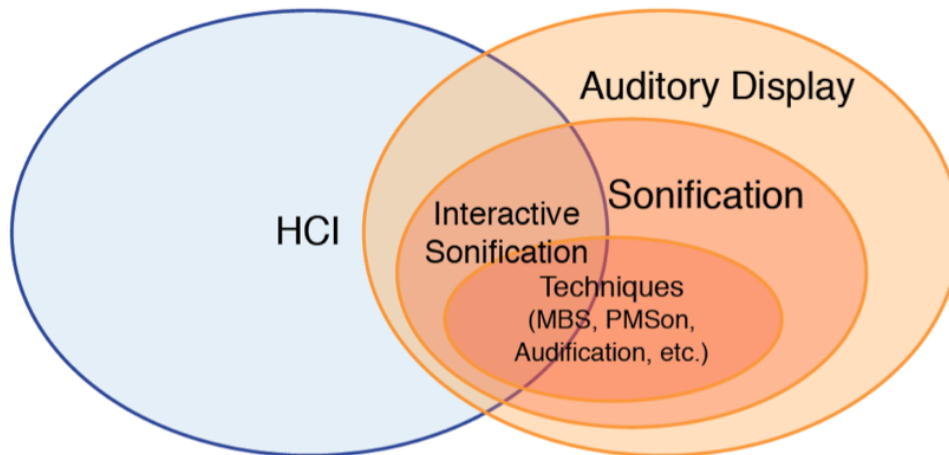


Figure 2.1: Interactive sonification topic Web – taken from [Hunt et al., 2011, pg. 274]

Interactive Sonification is a rapidly developing field in which the topic of Human-Computer Interaction (HCI) overlaps with that of Sonification. It allows us to interactively explore data sonically and receive real-time feedback from an operation. Figure 2.1 outlines the interactive sonification paradigm

2.2 Interacting with a Sonification System

by means of a Venn diagram which shows how the area of Interactive Sonification is formed by combining constituent parts of Auditory Display with the concept of Human Interaction with the interface. There are several areas of interactive sonification that can be studied. The key areas to focus on are the data transformation technique, the algorithmic implementation and the interaction itself. It is important that these features link in a logical and intuitive manner [Hermann and Hunt, 2011].

The data transformation technique is the method used to translate the data into its acoustic representation. Such techniques include Parameter Mapping Sonification, Audification and Model-Based Sonification. These can be combined dependent on the data set being sonified [Hunt et al., 2011, pg. 4]. The algorithmic implementation concerns the operation, computation, and performance, when rendering the data transformation as sound. This means that the quality of the interactive sonification process depends on the current hardware and technologies. The growing processing power of computers means that there are increasingly fewer issues with transforming large sets of data into sound, and also in being able to interact with that process seamlessly in real time. This implies that algorithmic implementations can become increasingly complicated, enhancing the process of sonification over time.

2.2 Interacting with a Sonification System

“It is essential that the bindings between the physical interactions with the interface tie into the acoustic relations in a logical and direct way. Much like interacting with physical objects, the element of instant feedback is essential to realise a believable and effective sonification.” [Hunt et al., 2011]

The interactive aspect of a sonification system largely relies on the human

2.2 Interacting with a Sonification System

involved and the technologies of the time. The control of an auditory display relies on many factors, including the user's needs and responses, the modes and means of control, and how the user and the auditory display form a feedback loop [Hermann and Hunt, 2004]. Figure 2.2 describes the difference between computer-driven conclusions (a) and the method implemented when sensory feedback is provided to the user (b).

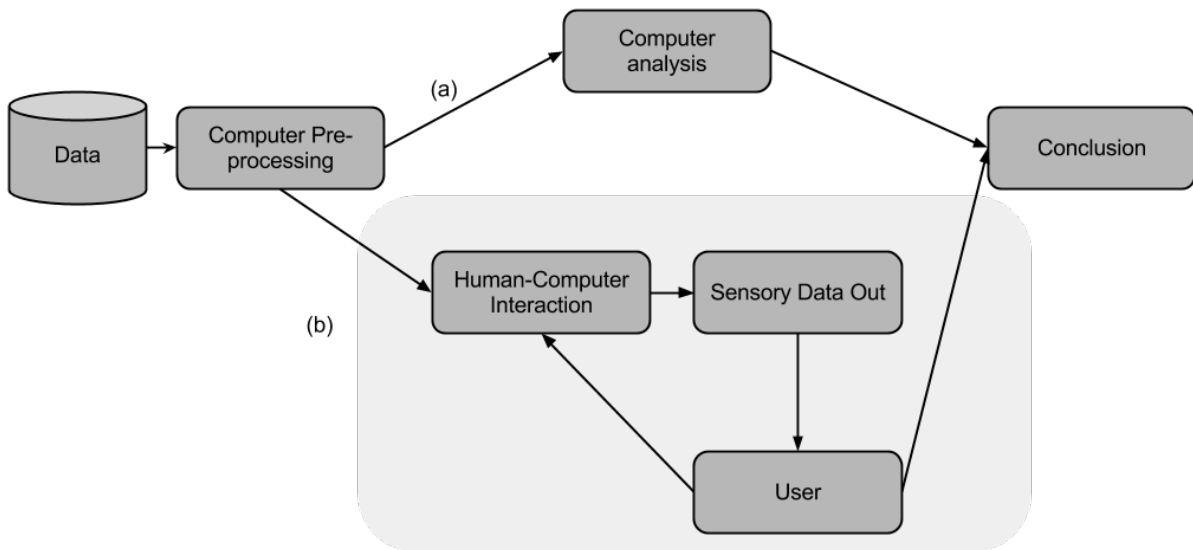


Figure 2.2: Interactive sonification paradigm - adapted from [Hunt et al., 2011, pg. 276]

As the power of computers has increased we have become more dependent on them to process data. In some fields it is apparent that there are simply too many parameters for a computer to process and reach an appropriate conclusion by itself. It is widely believed in the auditory display community that patterns can be revealed by putting a human into the feedback loop and allowing them to explore the data interactively [Pauletto and Hunt, 2009] [Hunt et al., 2011, pg. 276].

2.3 Mapping Data to Sound

The mapping technique used to transform data into sound can be customised according to the type and format of information within the data. It is the task of the system designer to choose appropriate sound mapping methods such that the data is clearly represented in the auditory domain. This section outlines some of the main mapping methods, along with some potential applications.

2.3.1 Auditory Icons

“Auditory icons are a popular method of displaying information by sound via the medium of using analogy to everyday sound-producing events.” [Buxton et al., 1994]

A good example of such an analogy is the auditory icon on the Apple Macintosh computer when a user puts a document into the recycling bin (*Audio Example 2.1*). This acts as an auditory skeuomorph of someone throwing a piece of paper into a bin. A rustling sound emulating paper being crumpled up is then used to assure the user that the bin has been emptied (*Audio Example 2.2*). Buxton notes that this method of feedback provides more information than visual stimuli alone, resulting in a more intuitive experience. This is said to reduce cognitive strain and improve the immersion of the user in the environment [Buxton et al., 1994]. Another example of the effective use of auditory icons is the use of a camera shutter sound (*Audio Example 2.3*) when a user takes a screenshot – it conveys the information better than any visual cue could, by means of analogy.

2.3.2 Earcons

“Earcons are brief musical melodies consisting of a few notes whose timbre, register, and tempo are manipulated systematically, to build up

2.3 Mapping Data to Sound

a ‘family of sounds’ whose attributes reflect the structure of the hierarchy of information.” [Walker et al., 2006]

Earcons are typically composed of motifs, which are short rhythmic sequences of pitches with variable intensity, timbre, and register. They can be used to convey some change in an interface to a user. They are particularly useful when they relate to the changes in a logical way. An example of this would be an ascending motif when inserting a device into a computer (*Audio Example 2.4*), and a descending motif when removing the device (*Audio Example 2.5*). In the paper ‘Earcons and Icons: Their Structure and Common Design Principles’ Blattner et al. outline some, almost grammatical, rules to be used when creating earcons [Blattner et al., 1989]. This allows for us to convey very specific messages about occurrences in a system via a series of hierarchical motifs.

2.3.3 Spearcons

“Spearcons are created by speeding up a spoken phrase until it is not recognised as speech.” [Walker et al., 2006]

Spearcons are used to convey textual information in a highly compact way to a user. It is a good method for portraying information to those with a visual impairment, or eyes that require attention on another task due to the fact that it can convey information explicitly. Evidence from a 2006 paper by Bruce Walker suggests that given some practice, spearcons can be more effective than other methods of navigating auditory based interfaces, leading to an overall improvement in performance and accuracy [Walker et al., 2006].

An advantage also noted by Walker is the fact that spearcons are able to provide information more definitively than other methods, such as auditory icons, or earcons. An example of this would be trying to convey more techni-

cal words such as ‘File’, ‘Edit’ or ‘View’ [Walker and Nance, 2006]. An example of spearcons being used to navigate a text-to-speech auditory menu can be heard in *Audio Example 2.6* (taken from [Hunt et al., 2011, pg. 24]).

2.3.4 Audification

“Audification is the most direct type of sonification technique. It plays ordered data values directly by converting them into instantaneous sound pressure levels.” [Dombois and Eckel, 2011]

Audification is the process of interpreting a data set as amplitude and playing it back in the auditory domain. Dombois and Eckel consider it the simplest form of sonification and state that it is normally neglected in the later developments of a sonification system [Dombois and Eckel, 2011]. Though simple, audification presents the data in its truest form, and often provides the most accurate representation of the signal. Audification is most useful when dealing with large amounts of data. If the data becomes too much for to perceive, it is possible to use our finely tuned auditory systems to perceive the patterns in the data. Dombois and Eckel note the fact that the human auditory systems require some amount of training and experience to perceive audification [Dombois and Eckel, 2011].

2.3.5 Parameter Mapping

“Parameter Mapping Sonification involves the association of information with auditory parameters for the purpose of data display.” [Berger and Grond, 2011]

As sound is inherently multidimensional, it is particularly well suited for displaying multivariate data [McGee, 2009]. Parameter mapping takes advantage of this by using characteristics of the data to control auditory descriptors

such as pitch, timbre, loudness or even how it is panned. The two types of parameter mapping sonification are outlined below:

Discrete Parameter Mapping Sonification – the notion of creating a sound event for each alteration in data. An example of this would be turning on and off an oscillator dependent on events occurring [Hunt et al., 2011, pg. 16].

Continuous Parameter Mapping Sonification – changing the acoustic parameters of a continuous sound as a function of the data. An example of this would be changing the frequency of an oscillator to track the change in an external parameter [Hunt et al., 2011, pg. 17].

2.3.6 Model Based Sonification

“Model-Based Sonification is defined as the general term for all concrete sonification techniques that make use of dynamic models, which mathematically describe the evolution of a system in time.” [Hermann, 2011, pg. 403]

A specific system gained with model-based sonification is called a sonification model. They are generally physics-based models used to generate a reaction based on a user’s actions [Hermann, 2011, pg. 404]. The concept is largely about exploration – much like we explore a drum membrane with a stick, noting the different pitches and timbres across it. We can also do this with data; we can excite a virtual data object in model space and render it as sound. Hermann outlines the primary applications: the aforementioned exploratory data analysis, augmenting human computer interaction, and process monitoring [Hermann, 2011, pg. 405].

2.4 Sonification Guidelines

This section outlines some key guidelines when designing an interactive auditory display with regards to how we perceive sound, ergonomics, and human hearing capabilities.

2.4.1 Subjective Auralization

We all perceive auditory information in a different way. As we are all different the way we transform the input from our sensory organs into an intricate array of neural impulses and perceive them as sound will always be slightly different. Due to the complexity of our brains and auditory systems, we have developed individual perceptions with regards to the auditory domain. When sonifying a set of data, it is important that the features of the information are audible; therefore, when designing a system it is important to not exceed human hearing capabilities. To ensure that this does not occur, some psychoacoustic principles must be adhered to.

Although the way in which we perceive sound is ultimately subjective, there are some assumptions that can be made for the majority of people. The most obvious factor when considering frequencies for sonification is the range of the human hearing system. This is approximately 20Hz – 20 kHz for a human with healthy young ears [Angus and Howard, 2006]. For example, imagine that we want to examine a change in warmth by sonifying temperature data; mapping degrees Celsius to frequency. The values given are between -4°C and 20°C . It is evident from these values that they will not scale directly to the auditory domain because the lowest frequency a human can hear is 20Hz. It is then possible to use knowledge of the human hearing range and make considerations for the human hearing range to design the algorithm to alter the frequency accordingly. To solve this problem a simple algorithm can be used to ensure the frequencies are audible:

2.4 Sonification Guidelines

Step 1 – Ensure all frequencies are positive by adding 5 to all values (lowest possible value = 1, highest possible value = 25). *Step 2* – Ensure all frequencies are audible by multiplying all values by 100 (lowest possible value = 100, highest possible value = 2500) This small example now results in a 2400 Hz range in pitch. Not only are all the frequencies audible, but it also means that due to the large range, even small differences in temperature can be detected.

We must also consider why we react to certain sounds the way we do. As animals we have evolved to survive [Darwin, 1859]. There are numerous by-products of this, related to how we perceive sound, and it is important that these are considered when designing auditory display systems. The subjective auralizations we have are driven by our survival instinct; it is known that high pitched or loud sounds such as hissing or explosions are not comforting to humans, particularly infants and children [Valentine, 1930]. This is merely a by-product of millions of years of evolutionary development. Nature has favoured those who fear danger. It is the reason we find high frequencies alarming, and therefore apply them to alerting people when considering sound design.

As the systems should be designed for people of all ages it is important to note the process of presbycusis – the diminishing in the upper frequencies of the hearing range as we age. Figure 2.3 details the typical hearing depletion of an ageing individual.

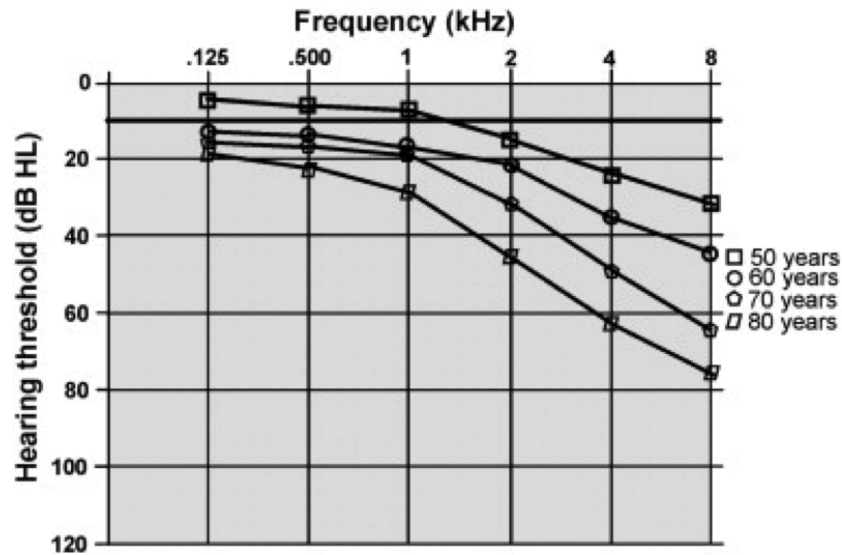


Figure 2.3: Presbycusis in Men and Women – from [Fetoni et al., 2011, pg. 414]

2.4.2 Ergonomics

“It is advisable to respect the bindings between physical actions and acoustic reactions that we have been familiar with since birth – or are possibly even coded into our sensory organs and brains.” [Hunt et al., 2011, pg. 295]

Hermann and Hunt outline some intuition and experience based conventions that designers would be misguided to break when making auditory display systems [Hunt et al., 2011, pg. 295]:

- The notion of a system being more reactive when a larger impulse is input;
- A system sounding higher pitched when under more tension;
- The expectation of sounds to become quieter when less energy is applied to them.

Though basing sounds and interactions around subjective human responses can lead to less numerically analytical outputs from systems, it is necessary to think about how the data relates to humans as they are the main sensory element in the interaction loop.

2.4.3 Human Learning Capabilities

“The effectiveness of any display will be influenced by the design of the display itself but also the characteristics in the data being analysed.”
[Bly, 1984]

Though Bly’s statement is valid, it has been suggested by Eldridge that important contributory factors are the abilities of the user, and the user’s familiarity with the particular display [Eldridge, 2005]. Perceiving patterns in data, through both auditory and visual stimuli, will differ from individual to individual. Take for example a Geiger counter; this device outputs an audible click representing a certain number of ionization events over some time period [Patel, 2006]. This is an easy concept for the user to understand. It is simple as it maps intuitively to our preconceptions of time and how the frequency of an event would map to it.

Comprehending an audio representation of multivariate data is more complicated and may require some learning. An individual may have to practise extensively to comprehend more complicated data sets. This is similar to the way we learn to understand graphical representations of data; the more complicated they are, the longer the learning process. Understanding the way humans comprehend information in sound is an important factor in producing a successful sonification. Walker and Kramer outlines the key features that should be worked towards to allow humans to comprehend data sets more effectively [Walker et al., 2006]:

- Parallel listening – the ability to perceive multiple audio channels.
- Rapid detection – the speed at which we detect information.
- Ease of learning and engagement qualities.
- Ability to discern relationships and trends in data and data streams.

2.5 Spatial Audio

“Everyday life is full of three-dimensional sound experiences. The ability of humans to make sense of their environments and to interact with them depends strongly on spatial awareness, and hearing plays a major part in this process.” [Rumsey, 2001, pg. 1]

This section discusses what is meant by spatial audio, giving an explanation of the key elements and outlining potential methods. It also describes each method’s potential advantages and disadvantages.

It is normally assumed that the term ‘spatial audio’ suggests the processing of signals such that they convey spatial content. Spatial audio theory is based on the human perception of sound, and it is essential to know some basic audio and physiological properties to understand the problems, and challenges, of 3D auditory display. How we locate sound is dictated by several factors, which are outlined below.

Interaural Time Difference – a sound not arriving from directly in front of, or behind a listener, will result in the sound reaching each ear at a different time. This is known as the interaural time difference, or ITD. The ITD for a listener depends on the angle of the source, as this affects the additional distance that the sound has to travel to the more distant ear. The maximum delay between ears is typically around 0.65ms. [Rumsey, 2001, pg. 22]

2.5 Spatial Audio

Level Differences and Spectral Cues – the head’s mass makes it a reasonable barrier to sound at high frequencies, and to a lesser extent at low frequencies. This imparts a level difference between the ears for off-axis sounds. More defined spectral information is filtered by our pinna; the outer section of the ear. This acts as a filter and provides complex spectral information to the ear drums, enabling us localize sound more effectively. Sounds that emanate from behind the head tend to give rise to a reduction in higher frequencies; this is due to the slightly forward facing shape of the pinna. [Rumsey, 2001, pg. 23] The physical system of hearing is outlined in Figure 2.4.

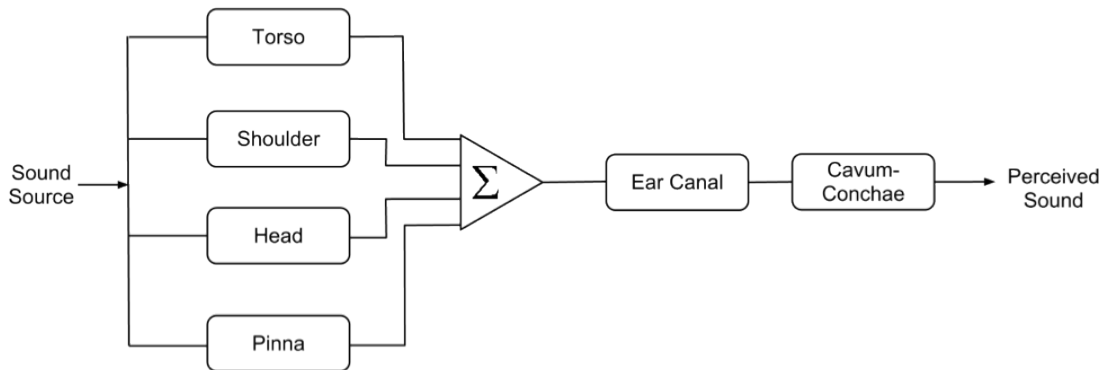


Figure 2.4: Physical process of hearing – adapted from [Krebbler et al., 1999]

The interaural level and time differences allow us a localization accuracy of approximately 10 degrees in the horizontal plane. Though worse in the vertical plane, this can still accurately provide a large amount of information to a person [Sandberg and Håkansson, 2006] There are numerous methods of displaying spatial audio to a user; some more effective, and some more convenient. This section will evaluate the main methods of spatial audio reproduction, outlining the benefits, drawbacks, and limitations of each method.

2.5.1 Loudspeaker Arrays

Multichannel audio is concerned with systems where there are more than 2 speakers; the systems are typically used when a large amount of auditory information needs to be displayed, for example – cinema, or art installations. The notion of projecting sound across several speakers is not a particularly modern concept; multichannel audio has been becoming increasingly prevalent in cinemas since the 1940s, mostly with the aim of immersing the audience in the auditory field; making them more involved in the movie [Miller, 2004]. However, to accurately represent a sound in space, more precise methods are required. To create realistic imaging with loudspeakers, a virtual source must be created. A virtual source is the perception of a sound source that does not coincide with any physical position of origin. This can be done using pairwise panning. Pairwise panning is the method of using pan control to create an audio image between two or more speakers by means of altering the amplitude of each channel [Pulkki, 1999]. It is common knowledge to most audio engineers, and it can be done in the horizontal or vertical plane.

Figure 2.5 depicts a simplified outline of the pairwise panning model, where ‘ p ’ is the panning direction of the virtual source, defined as a 2D unit vector, and the equation that governs ‘ p ’ is described in Equation 2.1 where ‘ g_n ’ and ‘ g_m ’ are gains which control the angle of vector ‘ p ’, and ‘ I_n ’ and ‘ I_m ’ are the Cartesian vectors for the channels. This can be easily extended over multiple speakers to implement Vector Base Amplitude Panning (VBAP).

$$p = g_m I_m + g_n I_n \quad (2.1)$$

VBAP is a general model for projecting a source anywhere around a user using pairwise panning in the x, y, and z planes. VBAP can project a phantom source provided the angles of the speakers and the position of the listener are provided. The more speakers used in the process, the more reliable the phantom source positioning will be [Pulkki, 1997]. A significant benefit of

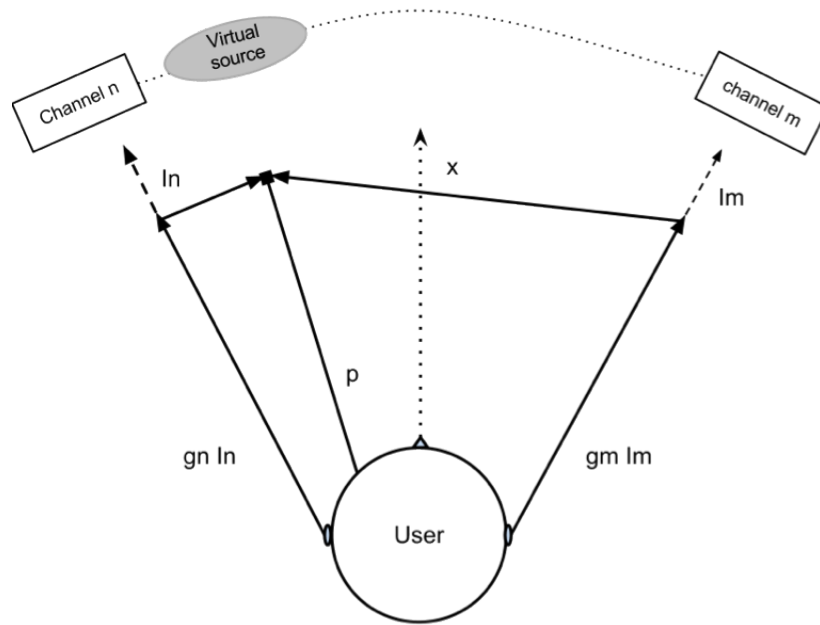


Figure 2.5: Pairwise panning – from [Pulkki, 1997]

VBAP is that it will work on the vast majority of people. As most of us have experienced 3D sound throughout our lives it means that we are already used to the idea of sources appearing around us in a 3D space.

An issue with VBAP is the scale of the physical system. To effectively represent sound using this method, not only must multiple speakers be used, but the user must also stand exactly in same place in the speaker arrangement throughout the duration of the sound experience. This leads to a lot of practical issues, primarily due to the fact that extensive speaker rigs are costly and large. The next section explores a potential solution to the logistical issues in the VBAP method, by trying to emulate this experience on headphones.

2.5.2 Binaural Audio

“Binaural approaches to spatial sound reproduction are based on the premise that the most accurate reproduction of natural spatial listening cues will be achieved if the ears of the listener can be provided with the same signals that they would have experienced in the source environment or during natural listening.” [Rumsey, 2001, pg. 65]

Binaural audio is normally produced for playback over headphones. This approach can reproduce most of the cues (interaural time difference, interaural level difference and spectral cues) that make us perceive realistic 3D sound over headphones. By ‘taking the place’ of our eardrums with microphones, it is possible to create a Head Related Transfer Function, or HRTF, the effect of which can be generated by using a dummy head, a real human, or modelling the head-related transfer of a human computationally to emulate the response of a human’s body.

A rudimentary approach for recording binaural audio is to place two microphones into the ear canals of a human. [Rumsey, 2001, pg. 65]. This approach can yield some strong results as it provides the spectral cues for this specific person, however, due to fact that placing a microphone inside a human’s head is inconvenient, a dummy head can be used as a compromise. The dummy head, or KEMAR (shown in Figure 2.6), is used to mimic the frequency response of the human head (and often torso) as accurately as possible. The KEMAR resembles the human head as closely as possible with regards to density, size and therefore, its frequency response [Rumsey, 2001, pg. 66]. The dummy head allows the microphone capsules to be placed inside the head of the ‘listener’ – capturing the interaural time and level differences, as well as the all important pinna cues.



Figure 2.6: A KEMAR – from [Gra, 2007]

Audio Example 2.7 demonstrates a recording made using a KEMAR.

It is possible to emulate an HRTF computationally. By approximating the sound incidence angle for an individual it is possible to synthesise signals with the appropriate time delays and spectral characteristics. Extensive work has tried to optimize head related transfer functions so that they work on a larger proportion of users. An example of refining HRTFs is the work done by Brian Katz and Gaetan Parseihian. This work involved creating a perceptually optimized HRTF by testing 45 individuals, and finding the best average HRTF [Katz and Parseihian, 2011].

There are, however, some disadvantages with a binaural audio implementation. The main issue is the fact that people have different HRTFs. Though a dummy head, or a computationally derived HRTF, can provide a good ‘average head’, many people have different psychoacoustic responses to the

effect. It is not as reliable as some other implementations [Rumsey, 2001, pg. 67]. It has also been noted that a disadvantage is the inability for a user to move their head to resolve directional confusion. In the real world, and most 3D listening environments, we can tilt our head to try and attain the source of a sound, with binaural audio we cannot. Some attempts at head tracking have proved successful, but ultimately this can lead to a more complicated implementation, detracting from the simple setup of binaural playback [Begault and Wenzel, 2000].

2.6 Image Processing Conventions

This section discusses the image conventions needed to understand the implementation in this thesis. It explains the fundamentals of digital images such as pixels, and colour models, describing how these are quantified and what standards govern their processing.

2.6.1 Digital Images (Raster Images)

The smallest graphical element is a pixel – the most fundamental element in most modern graphical displays. A pixel is a uniformly coloured element and, typically, thousands, or even millions are used to create most modern computer graphics. The majority of computer-based graphical images are shown on a raster display. A raster is a rectangular two-dimensional array of pixels that cumulatively make up an image [Shirley, 2002]. Each pixel is assigned a coordinate, with the origin in the top left hand corner, as outlined in Figure 2.7.

Computer graphics are generally defined in terms of the RGB (Red, Green, Blue) colour system. The concept relies on the additive mixing of the three primary light colours – red, green and blue. The resultant colour depends on the weighting of these three hues in the mixture [Collomosse, 2008]. Each

2.6 Image Processing Conventions

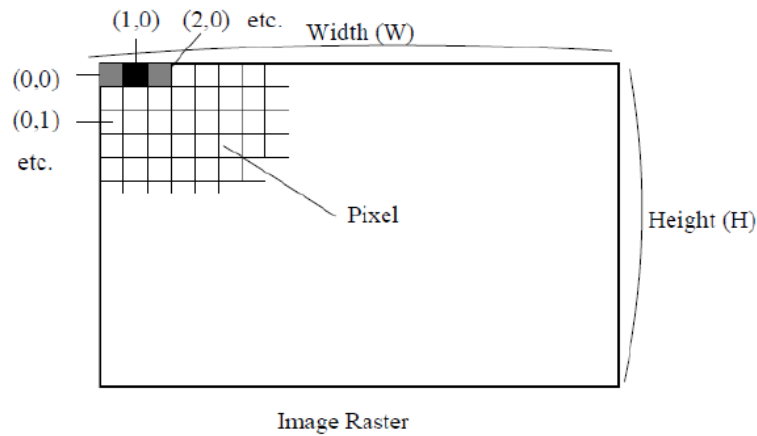


Figure 2.7: The pixel coordinate system – from [Collomosse, 2008]

channel of the RGB system has 256 shades; 0 being the lowest level (darkest) and 255 being the highest (brightest). Additionally, an alpha channel can be used. This channel governs transparency. When the value of this channel is 0, it is transparent, and when this value is 255, the channel is fully opaque. A 3D RGB colour model is depicted in Figure 2.8.

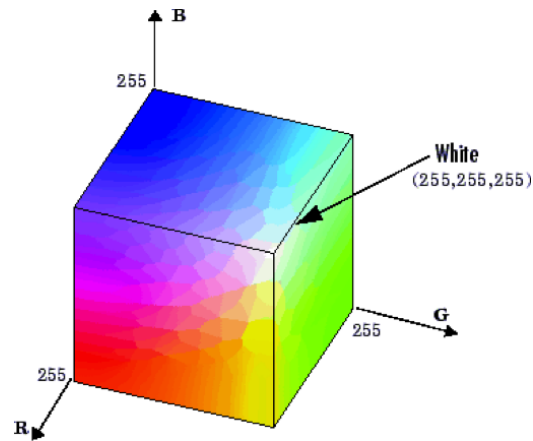


Figure 2.8: RGB colour model – from [Collomosse, 2008]

2.6.2 HSV – Hue Saturation Value

As not all colours can be represented by the RGB colour system – as shown by the tri-stimulus experiment [Orr, 2012], other colour systems have been defined with the additional aim to make them more accessible and intuitive. The HSV (Hue Saturation Value) colour system, outlined in Figure 2.9 is often preferred by artists as it is more intuitive – they are able to choose a colour, and select its brightness, as opposed to trying to mix Red, Blue, and Green to achieve the desired colour [Berk et al., 1982].

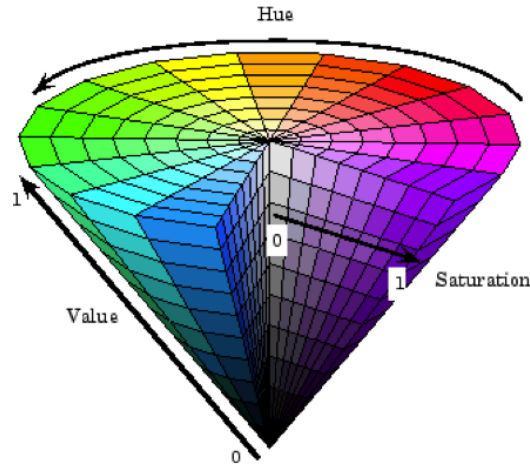


Figure 2.9: HSV colour model – taken from [Collomosse, 2008]

2.7 iOS Development Outline

With regards to the programming involved in this project, it is assumed that the reader has some background in software development, and understands most commonplace programming conventions. Although, this section explains most conventions used in this project the technicalities of the code will not will be described in depth. It is therefore suggested that the following link can be used to clear up technical details:



(a) iPad



(b) iPhone

Figure 2.10: iOS Devices – from the Apple website: <http://www.apple.com>

<https://developer.apple.com/devcenter/ios/index.action>

iOS is a mobile operating system developed by Apple Inc. The operating system, which runs on iPhone, iPad, iPod touch, and Apple TV, allows for a multitude of features to make the user experience as intuitive and enjoyable as possible. These include accelerometers, multi-touch gestures, and numerous other interactive techniques. Some of these devices are depicted in Figure 2.10. There are two groups of devices to consider when developing for portable iOS devices – small (iPod and iPhone), and large (iPad and iPad mini). The smaller devices have the advantage of being wholly portable – it is possible to carry them in the palm of one’s hand. However, the larger devices have larger screens, and therefore fit more visual content. The specifications of the most recent examples of these devices are outlined below in Table 2.1.

When making the choice between developing for large, or small devices, it is essential that the task the user is to undertake is fully understood. An interface that is too small can lead to a frustrating and fiddly interface, whereas an interface that is too large can be cumbersome, especially in situations where the user is active, for example, taking a jog.

iPad 4	iPhone 5s
Height: 241.2 mm	Height: 123.8mm
Width: 185.7 mm	Width: 58.6 mm
Depth: 9.4 mm	Depth: 7.6 mm
Weight: 662 g	Weight: 112 g
Pixels per inch: 264	Pixels per inch: 326 cm
2048 by 1536 pixels	1136 by 640 pixels

Table 2.1: iPad and iPhone dimensions from the Apple website:
<http://www.apple.com>

2.8 Core Graphics

Core Graphics is a C-based API (Application Programming Interface), based on the Quartz advanced drawing engine. Apple developed Core Graphics to provide multiple image processing techniques, such as: 2D rendering, gradients, patterns, off-screen rendering and the creation of PDF documents.

2.8.1 Core Audio

“Core audio is the engine behind any sound played on a Mac or iPhone OS” [Adamson and Avila, 2012]

Core audio is a low level audio processing API, developed by Apple, to govern how audio is processed in Mac OSX and iOS. It uses audio queues and audio units to process audio into a desired result. *Audio queues* are reusable buffers of audio that are passed to hardware. An audio queue typically does the following:

- Connects to audio hardware;
- Manages memory;
- Employs codecs for compressed audio formats; and

- Records and plays back audio.

Audio units are components such as mixers, distortions, or equalizers. They take in audio, and produce an audio output with some effect or alteration added to the input. iOS uses audio processing graphs linked together to process audio sample by sample in order to create the desired effect [iZotope inc., 2012]. Typically, to produce an audio unit that processes audio in a significant way, extensive development is needed. Some developers wish to cut out the audio units all together to avoid the low-level digital signal processing algorithms. Recently, several new APIs have been developed with the aim of overcoming this issue. The next section will outline the three most predominant of these external processing tools, describing their benefits and usability.

2.9 Alternative Audio Processing Environments for iOS

This section describes three alternatives developed to work around the complexities of Core Audio. The three alternatives (libpd, Csound-iOS, and The Amazing Audio Engine) each have their advantages. libpd and Csound-iOS use audio engines that already exist (Pure Data and Csound respectively), and The Amazing Audio Engine is a built-for-purpose audio engine that works in tandem with Core Audio.

2.9.1 libpd

‘libpd’ is an audio engine for mobile applications that uses the graphical programming language Pure Data (Pd) to deal with the sound aspect of an application by providing the user with powerful inbuilt functions that can be easily linked together via the graphical programming interface. It was written by Peter Brinkmann, along with a book called ‘Making Musical Apps’ [Brinkmann, 2012]. In the book Brinkmann gives a quick introduction to Pd,

before giving a thorough description of how to use libpd in an application. Examples such as a guitar tuner are described, and a full walkthrough of communication between iOS and Pd is given. libpd is available for download from:

www.puredata.info/downloads/libpd

2.9.1.1 Pure Data Overview

The Pure Data community site (<http://puredata.info/community>) describes Pd thus:

“Pure Data is a real-time graphical programming environment for audio, video and graphical processing. It is the third major branch of the family of patcher programming languages known as Max, originally developed by Miller Puckette and company at IRCAM. (Institut de Recherche et Coordination Acoustique/Musique)”

Due to the simple and interactive nature of Pure Data, it makes a great language for prototyping audio systems. It allows for those with little signal processing knowledge to make simple effects, basic instruments, and most things required to build the audio engine for a simple application. There are two main releases: Pd vanilla, and Pd extended. The main difference between the two is that Pd extended comes with a multitude of additional libraries, adding much more functionality to the software. Pd vanilla, though still useful, does not come with the more recent developments, built by third party developers.

2.9.1.2 Evaluation of libpd

libpd offers a way for those with little audio programming experience to control an audio engine. It can allow a developer to accomplish a lot of audio processing, with only a small amount of learning. It is evident that with this simple, less equipped version of Pd, applications such as Circle of Fifths [Brinkmann, 2012], and the numerous RjDj apps (available at <http://rjdj.me>) have shown that some significant apps can be built with libpd. However, numerous users of the library have reported issues processing data in a timely manner; latency times of over 500ms have been reported on a large number of devices [Kaufman, 2012].

It could be suggested that the most limiting factor of libpd is the inability to use the extra libraries that the Pd-Extended package brings, because a large part of the functionality available in Pd is from the libraries included in Pd-Extended. Technically, it is possible to run the missing libraries from Pd-Extended in libpd by building the binary files and packaging them for libpd. However, the majority of users settle for the compromise of using the compatible, but more simple ‘Pd vanilla’.

2.9.2 The Amazing Audio Engine

The Amazing Audio Engine is a framework designed for iOS by Michael Tyson, and released in March 2013. Tyson’s main aim was to produce an uncomplicated, functional framework that avoids the complexities of Core Audio. Tyson claims that it is very simple to use, while offering sophisticated audio processing. The amazing audio engine is closely linked to the Audio Bus app [Tyson, 2012]; a piece of software developed for iOS that allows users to route audio between several different apps. The amazing audio engine, and accompanying documentation, is available from the following link:

<http://theamazingaudioengine.com/>

2.9.2.1 Evaluation of The Amazing Audio Engine

The Amazing Audio Engine represents the current trend in the iOS-Audio field. Since its release the audio community have been developing applications with this engine as it provides more accessible audio programming to the iOS community. It represents a change in the audio app development world; moving away from the low-level programming typically associated with iOS based audio, and simply delegating the responsibility of this to an API. This seems logical; most app developers are not audio experts, and lack the knowledge to design and implement the intricate audio algorithms they need to achieve the results required.

2.9.3 iOS-Csound API

The iOS-Csound API was developed by Steven Li and Victor Lazzarini and released in April 2012 [Lazzarini, 2012]. The API allows for the audio programming language ‘Csound’ to run in the iOS environment, and for the two to communicate. iOS-Csound works in a very similar way to libpd; iOS and Csound can simply communicate data back and forth. This allows each system to complement the other; Csound dealing with the complex audio processing, and iOS dealing with the user interaction. The Csound-iOS API is available for download from the following link:

<http://sourceforge.net/projects/csound/files/csound5/iOS>

2.9.3.1 Csound Overview

“Csound is an incredibly powerful and versatile software synthesis program. Drawing from a toolkit of over 450 signal processing modules, one can use Csound to model virtually any commercial synthesizer or multi-effects processor.” [Boulanger, 2001]

Csound is a text-based programming language, written in C. It was written by Barry Vercoe in the late eighties, and has developed into one of the major audio-based programming languages. It is able to use hundreds of inbuilt, and third party functions called opcodes to produce a variety of sounds and effects, making it a highly versatile audio processing environment. Though Csound is not as interactive as many other audio based programming languages, such as Max and Pd, it allows for communication with numerous external devices and software. Csound can be downloaded from:

<http://www.csounds.com>

2.9.3.2 Evaluation of Csound-iOS API

The Csound-iOS API is a highly promising alternative to using Core Audio. An examples folder that comes with the API outlines its key features, showing how it is possible to embed Csound as a highly efficient audio engine while reaping the benefits of the iOS user interaction experience. Three significant applications made using this API are outlined below:

- csGrain – A real-time audio processing tool for iPad, that can take sound input and process it using granular synthesis. [Boulanger, 2012]
- Dandy – a simple sampler for iPhone. [Hepper, 2013]

2.9 *Alternative Audio Processing Environments for iOS*

- Density Pulsaret – an extensive granular synthesis engine for iPhone and iPad. [Petrolati, 2013]

These applications demonstrate the processing of the Csound audio engine in the iOS environment. Though mostly focused on granular synthesis, these applications suggest that Csound-iOS is capable of very high performance audio processing with no noticeable latency.

3 Relevant Projects in the Area

As there has been little prior work on this precise area of research it is important to break it down, and study its constituent parts. When combining any number of techniques, or technologies, it is essential that every part is well studied. This survey of the previous work has been done such that it is possible to add to the work of others and expand the knowledge in the area, but still ensure the project maintains some novelty. This chapter covers the following areas:

- Interactive Sonification for Task Monitoring (3.1)
- Sonification of Graphical Data (3.2)
- Spatial Auditory Display (3.3)

The work done is then discussed with an aim of exploring what data may be of use, and what information can be applied to this project.

3.1 Interactive Sonification for Task Monitoring

Interactive sonification is a wide varying field, which has led to a great diversity in the projects and applications associated with it. Notably, it has been used to provide real-time auditory feedback to allow for the improvement of tasks. For example, it has been used to improve the analysis of EMG (Electromyography Data) [Pauletto and Hunt, 2009], the generation of games for the blind [Papetti et al., 2008], and has provided insight into how a person moves, in the context of physiotherapy [Vogt et al., 2009].

By providing an extra modality, interactive sonification can allow for the user to monitor a task they are undertaking, and improve their performance,

3.1 Interactive Sonification for Task Monitoring

all while remaining eyes-free. An example of this is Schaffert et al's [Schaffert and Mattes, 2012] approach to using auditory feedback to keep visually impaired, and non-visually impaired rowers in synchronisation using sound. This was done by developing a system called 'sofirow' (Figure 3.1), which transforms the propulsive acceleration of a boat into sound – mapping the information directly to the tones on a MIDI-scale (between 0 and 127).

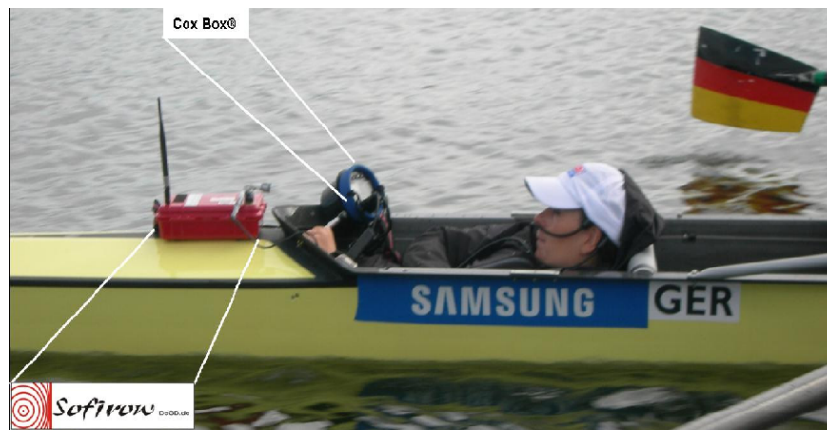


Figure 3.1: The sofirow system being used – from [Schaffert and Mattes, 2012]

In general, the users found the mapping aesthetically poor, stating that the sound was *“irritating and confusing”*. Initially this put the users off, but given some time to learn the system they found it less irritating, and that it helped them improve their overall performance. It is clear that when designing a sonification system, that the aesthetics of the proposed sounds should be tested on users beforehand to ensure that they are not irritating with extensive use.

However, it must be noted that the aesthetics of the sound should not be the only focus of the sound design in an interactive sonification system. In [Vogt et al., 2009] it is evident that the quality of the information provided was compromised for aesthetics. Vogt et al. describe an approach for interactive sonification for rehabilitation in which motion tracking was used to convey

3.1 Interactive Sonification for Task Monitoring

vital information about a user’s body shape – rewarding them with positive auditory feedback for performing the correct motions, and giving them negative auditory feedback for deviating from them.

Vogt et al.’s system, depicted in Figure 3.2, uses sound mappings such as ‘rustling leaves’ or ‘animal sounds’. It is assumed that sounds such as these were chosen for their aesthetic value – sampled sounds are generally less harsh to listen to than the aforementioned MIDI scale [Schaffert and Mattes, 2012], and other more pure-tone based approaches such as [Pauletto and Hunt, 2009] and [Meijer, 2013]. However, removing significant amounts of information by ‘filtering’ the data into categories beforehand may not only mean that some vital data may be lost in the process, but also that the sound mappings may be unintuitive, for example, we may not necessarily associate ‘wind in trees’ with ‘up’, as proposed by Vogt et al. [Vogt et al., 2009]. It is essential that there is a trade-off between providing all of the data, and filtering it for aesthetics or ease of use. This is something that is particularly relevant in the next section (Section 3.2)

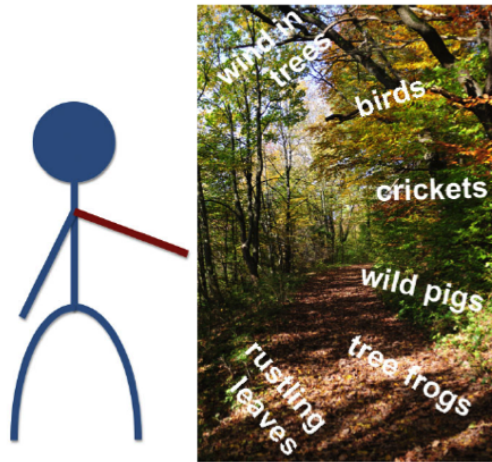


Figure 3.2: Metaphor based sonification of arm movement – from [Vogt et al., 2009]

3.1 Interactive Sonification for Task Monitoring

As well as monitoring tasks in the real world, some researchers have worked towards giving users feedback about the way they are interacting in a computer interface with sound. As mentioned in Section 2.3, auditory icons [Buxton et al., 1994] and earcons [Blattner et al., 1989] can be used to provide some feedback to a user about their interaction with a computer. More recently, work such as [Fernström et al., 2004] describe approaches to human-computer interaction that use interactive sonification to allow users to complete ‘eyes free’ tasks on a non-tactile interface. Fernström et al. describe a method for allowing a user to perceive ‘pseudo-haptic’ buttons from sound alone. This was done by trying to emulate the sound that friction would make by mapping a user’s touch pressure, and velocity to a sound engine that emulated the frequency characteristics of surfaces when touched. They note the importance of low latency between interaction and feedback as our auditory systems perceive temporal information significantly faster than our visual systems [Fernström et al., 2004].

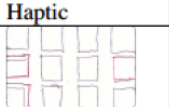
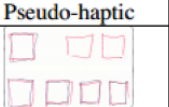


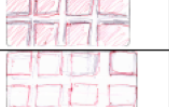

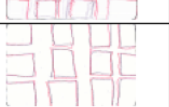



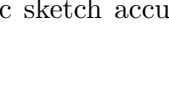
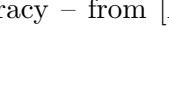
Layout	Participant	Haptic	Pseudo-haptic
A	1		
A	1		
A	2		
A	2		
A	3		
A	3		

Figure 3.3: Haptic vs Pseudohaptic sketch accuracy – from [Fernström et al., 2004]

By focusing on an exploratory approach [Gibson, 1961] to getting the gist of an interface, Fernström et al. report some interesting findings. The results

suggest that the users could determine where most of the interface elements were, as well as their approximate size and shape from sound alone (as shown in Figure 3.3). It is clear from the results in this paper that interactive sonification, when applied to understanding a computer interface (or any small visual field), is well suited by an exploratory search, in which the user gets real-time feedback about their interactions, and is not simply presented with a large quantity of data.

3.2 Sonification of Graphical Data

This section gives an overview of some major work done in the area of transforming graphical data into sound. Relevant work is described and the key techniques and concepts are discussed such that work may be built upon or embellished. When exploring the literature for the sonification of graphical data some trends emerged, and some key divergences in the methods used for different types of data was clear.

One of the most significant issues when it comes to the sonification of graphical data is the data itself. In terms of complexity it can range from a single black dot on a small screen, right up to a live video feed. Therefore, it is clear that different approaches should be explored for each case. It was found that, generally, still images favour an approach where the graphical data is filtered extensively, and the user explores the data in their own time (interactive sonification as opposed to audification).

The divide in this presentation of graphical data as sound is evident in several projects undertaken by the University of York that focused on the sonification of pre-cancerous cells, with an aim to reduce the cognitive strain of a lab technician when searching large amounts of graphical data (cell slides) for potentially dangerous cells. The first approach by Alyte Podvoiskis [Podvoiskis, 2004] focuses on filtering a specific colour range in the image that was

typical of a potentially dangerous cell when stained with a dye in accordance with the ‘Pap test’ [Trimble, 2009]. The user was then free to interact with the image, exploring it with the mouse until they were able to classify the cell clusters likelihood of being dangerous from sound alone – when the user ran the mouse over an area, the program deemed to be potentially dangerous the frequency and amplitude of an oscillator increased to alert them – creating an alarm-like sound. In general, the system was effective, and it was shown that users could detect potentially dangerous cells with good accuracy (despite some shortcomings when it came to ‘borderline’ cells) [Edwards et al., 2010]. This target-based approach offered a reasonable compromise between human processing, which is good for detecting patterns in such data (but often gets sensory overload), and computer processing.

On the other hand, as part of the same sonification of pre-cancerous cells project, Hines [Hines, 2007] and Lee [Lee, 2006] focused on a non-interactive approach in which the cells were categorised by the computer using more advanced image processing techniques than that of Podvoiskis. In this system, feature detection was used to determine the likelihood of a cell being dangerous, and a sound was used to classify them. It could be argued that this method involved too much computer pre-processing, and not enough human processing – the system was not an interactive sonification system as it took the user out of the loop. In this case it was evident that there appears to be a fine line between the computer doing all of the processing, and the human being in the loop. As humans can pick up on complex patterns in data (especially auditory data) that computers can not [Hermann and Hunt, 2011, pg. 21], it could be argued that extensive pre-processing hinders the effectiveness of the system.

When making considerations for the sonification of these cells Stammers [Stammers, 2006] also attempted a non-interactive approach in which the user was placed in a 3D sound field and the visual information was represented by playing sound samples around them. The user was not free to

interact with the data, but were a passive party as the sounds (representing the severity of the cells) were placed around them (as shown in Figure 3.4). Stammers notes that this system was not hugely successful because of the user’s inability to move around (in real time) the graphical data, and therefore the sound field – when transforming an image into sound, it is clear that sometimes they are too complicated, or large to take in at once. Moreover, Stammers’ final implementation used sample based auditory feedback based on quite unintuitive sounds (such as a ‘frog croaking’ to represent some level of severity), something that has proved relatively unsuccessful in other work [Vogt et al., 2009].

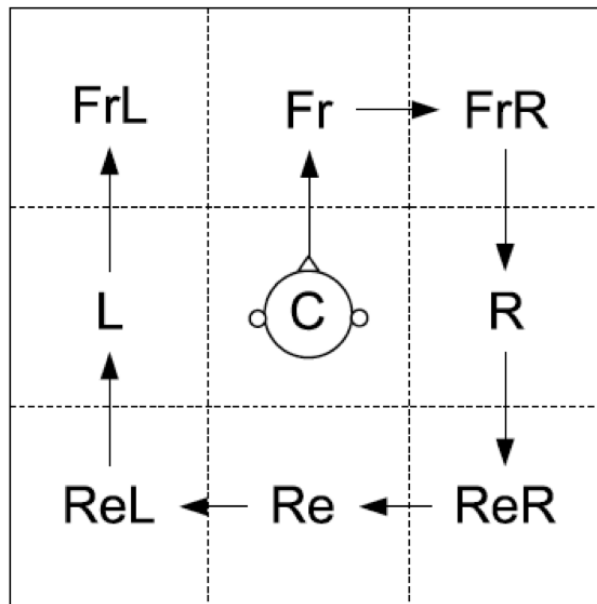


Figure 3.4: ‘headcirc’ algorithm – taken from [Stammers, 2006]

A similar pattern emerged with other work – when transforming graphical data on a screen into the auditory domain, it is clear that a more interactive approach is beneficial [Sanchez, 2010] [van den Doel, 2003] [van den Doel et al., 2004]. Less interactive approaches tended to falter, or required more intensive work on behalf of the user [Hines, 2007]. However, when transforming live video feeds it is not as clear cut, and depends more on the ap-

plication of the system. The literature implies that, in general, target-based approaches normally benefit from a filtered approach as with the aforementioned screen-based methods, but also with live video feeds, this is shown by the success of Bologna et al.'s 'See ColOr' project [Bologna et al., 2008] [Bologna et al., 2009] [Bologna et al., 2010] in which a system is devised to enable visually impaired persons to navigate some physical environment by means of transforming visual cues, such as a red path on the floor, into sound.

Bologna et al. first undertook an experiment [Bologna et al., 2007] in a virtual environment using a touch sensitive tablet to provide tactile feedback. In this work, they wanted to determine if it is possible for blindfolded persons to associate colours with the sounds of musical instruments. This was based on [Rossi et al., 2009], which involved an experiment based on how we can associate certain musical instruments with colours.

This then led onto them designing a test to see if users could undertake a real-world experiment [Bologna et al., 2008] – matching coloured socks into pairs. The evidence suggests that they were largely successful in this task, however, it could be argued that there are better mappings for colour as we need to commit to memory how each instrument relates to each colour.

[Bologna et al., 2009] and [Bologna et al., 2010] discuss a more complex real-world navigation task. They still opt for the filtered graphical pre-processing to undertake the tasks, however, in [Bologna et al., 2010] depth is sonified – so the users can determine with sound the distance of the target. In effect, they try and render our stereoscopic perception of the world into sound. The distance between the target and the user is measured with a pulse train – the rhythmic frequency of the instrument was increased as the user got closer to the target.

A similar distance representation is used by Yoshida et al. [Yoshida et al., 2011] when trying to communicate 2D shapes using auditory feedback. When tasked with representing the distance to a specific feature, Yoshida et al. used two sonification mappings in tandem – local area sonification to describe the contour of the shape, and distance to edge sonification to allow the user to determine the location of the feature. The distance to edge mapping was similar to the pulse train used in [Bologna et al., 2010]. The fact that both were successful in representing distance suggests that a pulse train is likely to be a good way of representing distance.

A test was produced to demonstrate the difference in using and not using local area sonification, in addition to distance to edge (pulse train) sonification. It was evident that there was a noteworthy improvement between local sonification being off, and on, when it came to determining the finer details of the shape – suggesting that multiple mappings may be needed to represent finer details of shapes. Figure 3.5 outlines the shapes the participants were aiming to sketch through auditory display.

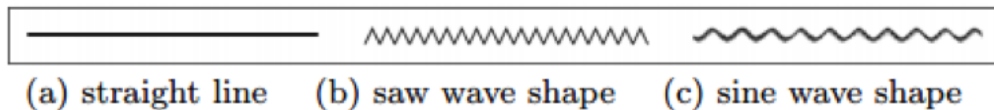


Figure 3.5: Shapes the participants were tasked with sketching – from [Yoshida et al., 2011]

Sanchez et al. [Sanchez, 2010] opt for a different distance mapping – they use the tuning of a radio as a metaphor for distance to aid a user in tracing a car (as shown in Figure 3.6) – as they get closer there is less static, and once they are on track they can hear the original sound clearly. This approach is interesting, however, one could argue that mappings such as the radio mapping is not a logical choice – the metaphor of tuning a radio is something that future generations won’t understand, and ultimately relies on some prerequisite knowledge. Using sonification metaphors that rely on some knowledge

that everyone may not have is always an area that must be approached with caution. The mappings should be as intuitive to us as possible. Moreover, the metaphor does not provide any directionality – the user will know how far they are away from the shape, but will not know what direction to go in. It is therefore advised that when trying to represent direction to a user in a target-based auditory display that they are provided with not only a sense of distance, but also a sense of direction. Approaches such as [Stammers, 2006], that represent directionality by means of spatialised audio, would provide an ideal mapping for distance in such a context, and will be discussed in the next section (Section 3.3).

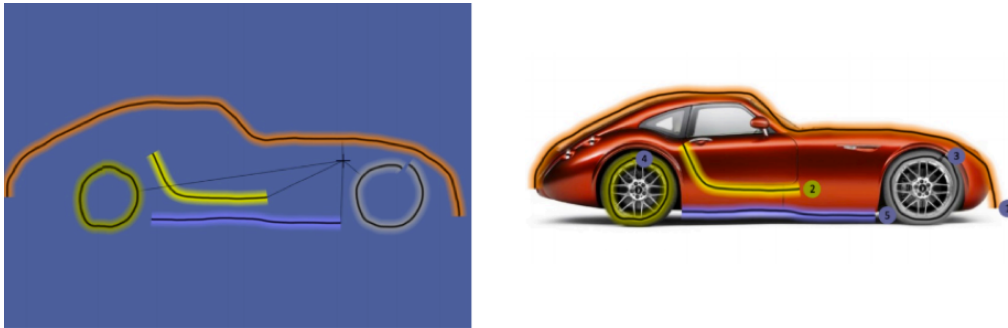


Figure 3.6: Traced features of car – from [Sanchez, 2010]

Contrary to these filtered approaches, a non-filtered approach was shown to benefit Peter Meijer’s ‘The vOICe’ system [Meijer, 1992]. This system transforms all of the visual information from a live video feed into the auditory domain by mapping the frequency of an oscillator to the height of the pixel in the display, and the brightness of the pixel mapped to its amplitude (Figure 3.7). It was shown, in neurological studies [Amedi et al., 2007] [Merabet et al., 2008], and in real-life test cases [Meijer, 2013], to have a significant effect after an initial intensive training course.

It is clear that some auditory displays will require some practice to learn. But, due to the complexity of the audio output, approaches such as Meijer’s [Meijer, 1992] require full courses of training. The training scheme Meijer

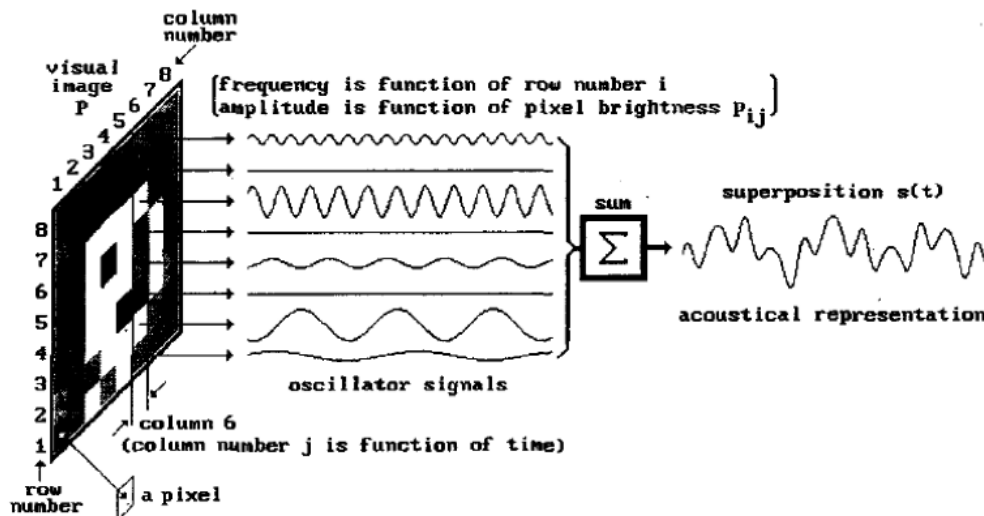


Figure 3.7: Image to sound mapping for The vOICE – from [Meijer, 1992]

provides is similar to the way blind individuals such as Daniel Kish [Arnott et al., 2013] are able to learn to determine their surroundings by means of echolocation. The users are taught to decode the subtle changes in the audio feed and interpret them as changes in their 3D environment. The current form of the system (shown running on the Android platform in Figure 3.8) and more information about this system is available from The vOICE’s website ¹.

A direct comparison [van den Doel et al., 2004] of The vOICE and an alternative system (SoundView) shows that The vOICE does not perform as effectively when sonifying still images. As in [Fernström et al., 2004], it is shown that the sonification of still images (SoundView [van den Doel, 2003]) benefits a simplified exploratory approach where the user is allowed to interact with the screen, as opposed to the holistic approach of sonification in The vOICE [Meijer, 2013]. The approach SoundView takes is to allow the users, much like in Podvoiskis’ approach [Podvoiskis, 2004], to interact with

¹www.seeingwithsound.com

3.3 Spatial Auditory Display

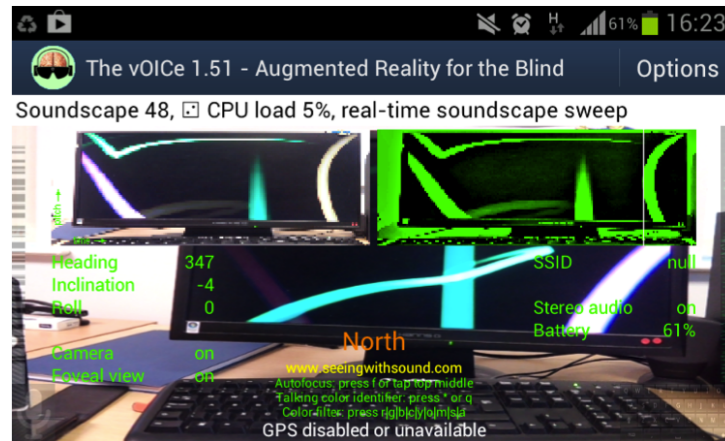


Figure 3.8: The vOICe system running on an Android phone – taken from [Meijer, 2013]

the still image (in this case with a stylus) and hear the auditory feedback to gain a mental model of the image they are interacting with.

3.3 Spatial Auditory Display

This section describes the key work done into the use of spatial audio in auditory display. As discussed in Section 2.5, the term ‘spatial audio’ encompasses a series of techniques to spatialise sound. This section focuses on applications of these techniques as a parameter used in sonification.

Generally it was found that spatial audio is most commonly used to represent a physical direction, and that it was a good parameter for doing so. It has been used to portray a physical direction on a screen [Heuten et al., 2006], and in a real life scenario [Michal Bujacz and Strumillo, 2012].

Heuten et al. propose a system that aims to allow visually impaired individuals to gain a cognitive model of a new place before going to it [Heuten et al., 2006]. To do this they created a virtual 3D sound room to convey directional

3.3 Spatial Auditory Display

information about geographical objects, such as lakes and parks (Figure 3.9). These features were represented by placing multiple sound sources around the user in a direct mapping, using binaural audio. For example, if a feature was close and to the right it would be loud and the sound of the source would be binaurally panned to the right.

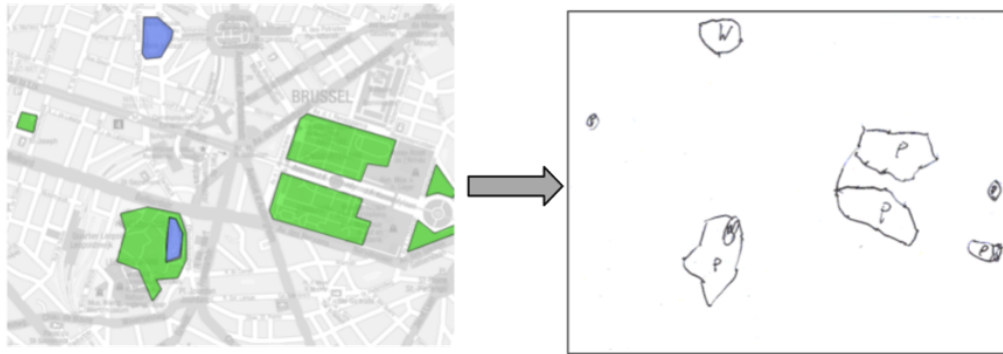


Figure 3.9: A user's perception (right) of the map data presented (left) – from [Heuten et al., 2006]

A key feature of this work is the fact that the user is provided with the ability to explore the space, therefore allowing them to gain a better cognitive model than in work such as [Stammers, 2006]. Additionally, the authors note the advantage of using physical devices such as digitizer tablets over a mouse, as the user gets tactile feedback about their position – they can feel where they are on a tablet by touching its borders and learning their relative position by means of the same exploration. Yet again, bringing to light Gibson's work on the exploration of every day objects [Gibson, 1961]. But more interestingly, adding to the work of [Podvoiskis, 2004] and [Fernström et al., 2004], as the use of 3D audio allows the user not only to get an idea of what they are interacting with, but also, it allows them to get an idea of what is around them at the same time. This is because 3D audio allows for multiple sources to be easily detected and differentiated.

This sort of exploratory interaction is even more relevant as tablet computers

3.3 Spatial Auditory Display

are becoming more ubiquitous – when we interact we can learn our relative position on the screen in a way we cannot with a mouse and keyboard, which has great implications in the field of assistive and eyes-free technology.

A similar approach [McGookin and Brewster, 2001], by McGookin and Brewster, proposes a method that hybridises visual feedback and spatialised audio to compensate for the decreasing screen size of modern electronic devices, and to allow for ‘eyes free interaction’. They propose the ‘focus and context’ method – the ‘*focus*’ is the information that the user is most interested in, and the ‘*context*’ is merely the non-important information on the periphery of the user’s interests. They use a similar paradigm to Spence and Apperley’s Bifocal Lens concept [Spence and Apperley, 2013], but with an additional modality (audio).

The *focus* element is the device’s visual display. Outside of this focus is the auditory domain; the *context*, which displays information by using 3D audio to produce different auditory cues around the user – extending the interface’s useable area. Approaches such as this, and Apple’s ‘Spaces’ paradigm (discussed in the Introduction (Section 1)), highlight the need to be able to present information off-screen. When condensing computers down to fit in the palm of one’s hand, the WIMP (Windows, Icons, Menus, Pointers) paradigm, that we are used to, often fails to display the information we require due to the physical constraints of having a small screen.

McGookin et al. suggest a notion of priority zones. This concept (shown in Figure 3.10 involves filtering the data such that it is not cluttered (as done in [Heuten et al., 2006], [Michal Bujacz and Strumillo, 2012], and [Sanchez, 2010]). For a sound to play, a feature must be within a priority zone that has a priority less than, or equal to, the feature itself. For example: a priority four ride would sound in zone three, but a priority three ride would not sound in zone four. This form of selective filtering allows for uninteresting peripheral information to be negated, and for relevant information to be presented

for the user's attention.

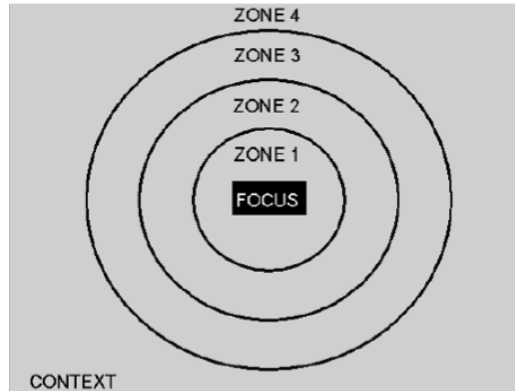


Figure 3.10: Priority zones within the context – from [McGookin and Brewster, 2001]

Bujacz et al. take the presentation of 3D space for navigation to the next level of detail [Michal Bujacz and Strumillo, 2012]. They discuss a significant topic when it comes to the presentation of 3D environments – ‘*how much data do we sonify?*’ This is an important question, that ultimately depends on the end user, and the task at hand. This approach aimed to provide ‘the best of both worlds’ to the users; ensuring that they make the complicated data simpler, without removing important information.

To ensure a non-complex auditory interface that also provided all the required information, 3D sound was used to represent spatial information. To prevent information overload for the user, an image-processing algorithm was developed that reconstructed the room into a simplified form. This was then transformed into sound – the distance to the obstacles was mapped to the pitch and amplitude of some assigned auditory icon, and the duration of the sound was relevant to the distance of the object and then its direction mapped directly to a binaurally panned sound source.

3.3 *Spatial Auditory Display*

The most significant part of this paper is the realization that often a compromise needs to be made when sonifying complex data, especially in 3D sound. Is it wise to sonify all the data, and assume that the human brain can decode it, in a similar manner to The vOICe [Meijer, 1992]?. Or is it wiser to opt for a more selective version of the data sonified, much like FISHEARS [McGookin and Brewster, 2001]? User-centered design allowed for this system to be effective – removing all redundant information a complex 3D sound field, in accordance with the user’s feedback, meant that they took less time to learn the system, and ultimately could make better use of it [Michal Bujacz and Strumillo, 2012].

Throughout the literature a common trend emerged – the mapping of 3D sound to find a specific target, as a physical direction, was generally done directly, i.e. if the target was on the left, the sound was panned to the left. Moreover, of these implementations, the most successful were interactive, where the users were free to move around the 3D sound world and the superfluous [McGookin and Brewster, 2001], or distant information was filtered out [Heuten et al., 2006] [Michal Bujacz and Strumillo, 2012] . Approaches that were non-interactive [Stammers, 2006] or did not use appropriate filtering [Meijer, 2013] did not yield strong results when interacting with a screen, as opposed to a physical environment.

4 Research Agenda

This section restates and examines the hypothesis in light of the information gathered in the ‘Theoretical Literature Review’ and ‘Relevant Projects in the Area’ sections. As it is possible to interpret many of the terms from the primary hypothesis in a number of ways, these will first be discussed. Then, each term in the hypothesis is clarified, by using recommendations from the previous chapters to refine the finer details of the hypothesis. The aims and objectives are then redefined, influenced by the information gained in the previous two sections, such that tests can be developed to support, or refute, the primary hypothesis in the remainder of this thesis.

4.1 Hypothesis Discussion

As discussed in the introduction the primary hypothesis states:

It is possible to improve a user’s understanding of graphical data by using spatial audio to provide interactive auditory feedback.

The primary hypothesis is purposefully general; therefore, it is important to discuss some significant factors, and variables, that it encompasses. Each term in the hypothesis will now be clarified so that the it is more specific and can be quantified.

User – The term ‘user’ is generic, as different systems suit the needs of different operators. However, in the scope of this project, the hypothesis deals with the most important factor for the perception of graphical data – whether we can see it or not. This implies that there will be two clear types of user – the visually able and the visually impaired.

4.1 Hypothesis Discussion

Understanding – Understanding, in the context of this hypothesis, could mean that the user is able to understand the entire image, or smaller, more specific features of the image.

Graphical Data – In perceptual terms, graphical data means ‘what we can see with our eyes’. The complexity of the graphical data in question is extremely important to consider – if an individual were asked to locate a black dot on a screen, it would be simple to do so by visual means alone. However, if one were tasked with locating the colour value (R = 153, G = 99, B = 39) in the Mona Lisa portrait, this would be considerably more challenging.

Spatial Audio – This can refer to a number of techniques for altering the perceived point of origin of a sound – VBAP, panning, ambisonics, etc. A technique, influenced by the literature, will be used to spatialize auditory information around a user such that they can better understand it.

Interactive – This term depends largely on the mode of interaction. Modern technology provides us with a diverse range of options with regards to interaction with computers – mouse, keyboard, tablet computer, speech, etc. The interaction technique is important; a method that is highly instinctive for the user must be chosen, such that the interaction experience is as simple and enjoyable as possible.

Auditory feedback – This implies that there will be some auditory response to the user interacting with the graphical data. As little work has been done in the area of turning graphical data into sound, appropriate sound design is essential. The reviewed literature, and some preliminary research could help determine more effective sound design methods.

The next section discusses the key findings in the literature, such that the hypothesis can be refined and the aims and objectives can be stated.

4.2 Summary of ‘Relevant Projects in the Area’ Section

A key finding when looking at interactive sonification for monitoring a task (Section 3.1) was that the feedback needed to be as real-time as possible [Fernström et al., 2004] [Schaffert and Mattes, 2012]. As well as this, there should be a tradeoff between aesthetics, and content [Vogt et al., 2009]. One of the most important findings was found in [Fernström et al., 2004] – the notion of the free exploration of data to gain a mental model of it is something that will be highly relevant the sonification of graphical data in this project.

When looking at work done on the sonification of graphical data (Section 3.2) it became evident that there were two main approaches: filtered [Podvoiskis, 2004] [Sanchez, 2010] [van den Doel, 2003], and non-filtered [Meijer, 1992]. The general trend was that approaches that required the search of a small area, or where the user was searching for a target, were improved by some degree of graphical pre-processing [Podvoiskis, 2004], and that approaches that involved complex visual information were improved by a non-filtered approach [Meijer, 2013].

When making considerations for using 3D audio (Section 3.3) as a parameter, it became clear that mapping directionality within a dataset to 3D sound works well [Heuten et al., 2006] [Michal Bujacz and Strumillo, 2012]. Moreover, in an interactive sonification approach, it became evident that 3D audio can allow us to understand our surroundings and our immediate position on a screen, or environment, simultaneously [Heuten et al., 2006]. Additionally, McGookin and Brewster suggest that this approach may be improved by zone-based filtering [McGookin and Brewster, 2001].

4.3 Refined Hypothesis

The hypothesis is now refined with the information gathered from the previous four sections. The hypothesis is deconstructed into more specific constituent parts, inspired by the initial hypothesis, and the previous two sections.

The refined primary hypothesis now states:

It is possible to detect graphical features on a tablet display by means of real-time, interactive, binaurally spatialized audio.

Each of these terms is now described in context of what will be tested in this thesis. Each decision has been informed by the initial idea, trends in the literature, and the capabilities of modern technology.

Detect – By ‘detect’ it is meant that the user will be able to find graphical features on a display, without looking at the visual cues it provides. Methods will be developed such that a user, when in physical contact with a display, can discover the features aurally with the aim of finding the location of a specific feature.

Graphical features – The graphical features will differ throughout the tests and become gradually more complex. Initially, simple shapes and colours will be used; these will become more complicated as the tests progress with an aim to test different sonification methods.

Tablet Display – Something that occurred repeatedly in the literature review was a lack of technological capabilities; often there were projects that could not be fully realized due to the lack of computation, or interfacing. As discussed in the Theoretical Literature Review, using a tablet display (an iPad

in this case) will allow for reasonable processing power, with the added bonus of a highly sophisticated touch interface.

Real-time/Interactive – When undertaking data exploration by sound it was noted in the literature review that one of the most important contributing factors to a successful outcome was an interactive auditory display that works in real-time. Real-time implies that there is minimal perceivable latency. Latency, in interactive interfaces, can be highly frustrating to the user. It is therefore suggested that the interface should have minimal lag between interaction and audio output.

Binaural Spatialized Audio – The decision has been made to focus on binaural spatialized audio for numerous reasons; the initial project hypothesis aimed to incorporate spatial audio, and from the Literature Review and Theoretical Background it became evident that binaural audio offers a highly effective, portable, and cheap solution to spatialized audio.

4.4 Aims and Objectives

This section describes the aims and objectives of this research. Which clearly lay out the steps needed to verify the newly refined hypothesis such that techniques can be developed, tested, and finally evaluated in the remainder of this thesis.

4.4.1 Aims

- 1) To meaningfully represent graphical data, on standard and extended displays, using spatialized audio.
- 2) To develop interaction techniques that are real-time, responsive, and abide by the interactive sonification human-computer loop paradigm.

3) To gain insight into the success of the techniques and tools developed, and make considerations for how they may be improved, or applied.

4.4.2 Objectives

1) Develop image processing techniques to allow for the detection of specific graphical features.

a) Create an algorithm that can detect specific colours in an image.

b) Develop an algorithm that can find the average location of an image feature such that its location may be determined.

2) Develop methods that allow a user to interact with the graphical information in a meaningful real-time manner.

a) Use iOS touch detection methods to allow for the detection of fingers, track their locations and determine when the user has removed their fingers.

b) Calculate the vector between the touch co-ordinates and the image features detected.

3) Develop an audio engine that responds to the information gathered from the image and the information derived from it by the program.

a) Experiment with a series of audio engines to determine the best option for this work.

- b) Experiment with sonification methods to best represent the graphical information.
 - c) Make use of binaural audio to indicate graphical features to the user.
 - d) Conduct a subjective preliminary test into the best way to represent certain graphical characteristics.
- 4) Devise a series of progressively more challenging tests to determine the effectiveness of the implementations, and establish whether these methods can be improved, or whether they can be applied in some areas.
- a) Develop two test procedures, each with minor differences, to determine the effectiveness of the parameter mapping in the sonifications, and the effectiveness of the interaction techniques.
 - b) Develop increasingly complex tests that challenge the test subjects to find specific graphical features, with varying degrees of auditory assistance, and with varying sizes of displays.
 - c) Conduct and document testing procedure in detail with modern techniques such as video documentation and screen capture.
- 5) Analyse the video and screen capture of the tests described in objective 4 and draw conclusions with regards to the effectiveness of the interaction, and auditory display techniques developed. These conclusions will allow for the verification, or evaluation of the hypothesis in light of the new information gathered.

4.4 *Aims and Objectives*

Before progressing to implement these tests, a preliminary experiment was done. The following chapter describes an test devised to determine if people associate specific colours with certain sound characteristics. This was carried out before any system implementation such that the sound design choices could be better informed.

5 Preliminary Test

This chapter discusses an experiment to determine the best way to map colour to sound in an auditory display. It gives an introduction to the topic, outlining some significant work and some preconceptions about how we relate sound, specifically frequency, to colour. A small-scale (15 person) experiment is then described and discussed, providing insight into how colour should be mapped to sound such that it the information can be used in the context of sonification.

5.1 An Introduction

When transforming images into sound it is important to consider what different colours ‘sound’ like. Notable work done into the mapping of colour to frequency has been carried out over centuries. Isaac Newton [Newton, 1730] and Louis-Bertrand Castel [Hutchinson, 2012] explored the concept of how we can relate colour to the pitches across the notes of a piano. Colour is to light what pitch is to sound; the smaller the wavelengths of the signal, the higher the resulting frequency. It is evident that a potential method of mapping colour to frequency directly is possible; red being the lower frequency; purple the highest, as outlined in Figure 5.1.

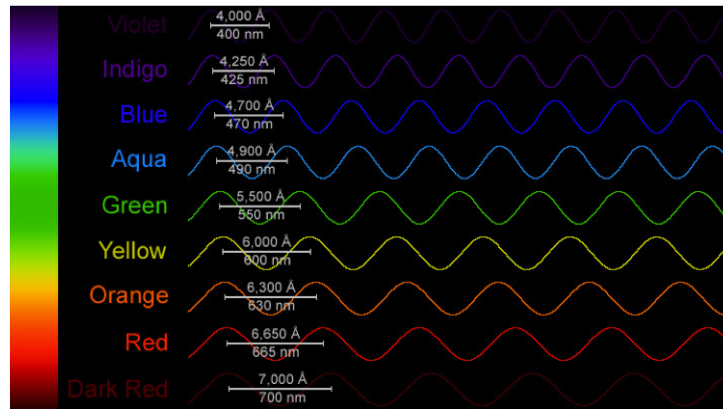


Figure 5.1: Frequency of Light– from [Hutchinson, 2012]

When mapping the piano to colour, Newton mapped frequency inversely to colour; violet the lowest and red the highest. This is shown in Figure 5.2.



Figure 5.2: Piano with mapped to colour spectrum – from [Hutchinson, 2012]

Castel decided that the primary colours were related to the chord C major, as he believed the most fundamental colours (blue, yellow and red) used by artists could be related to the common chord C major. He considered blue the darkest of the three, so he assigned this to middle C. He then assigned E to the colour yellow, and G to the colour red based on their frequency content inspired by Newton’s work. He then placed the colours he saw as transitional colours, i.e. the colours between the blue, yellow and red in the spectrum, in-between the C, E and G [Hutchinson, 2012]. This is shown in

Figure 5.3.

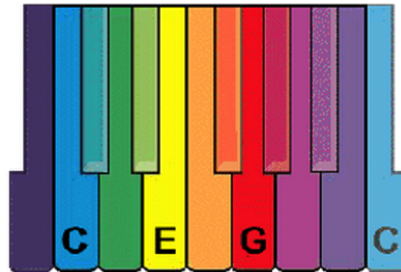


Figure 5.3: Castel’s interpretation of colour to sound mapping – from [Hutchinson, 2012]

5.2 The Experiments

It was decided that two experiments could be devised to provide information with regards to the best way to map colour to frequency content – one that tested a range of synthesis methods to determine the ‘pleasantness’ and meaning of synthetic sounds. And one that tested the tones across an octave of a piano to explore how pitches in a scale relate to colour. These tests were accompanied by a short demographics questionnaire in which the participant had to fill in some information about themselves. This was done to look for extra patterns related to age, culture, musical training, and whether the user has previous ideas about how colour and sound are related.

5.3 Initial Test Hypotheses

Test 1 was devised to assess which side of the spectrum the participants associated with high or low frequencies. An additional goal was to determine if any of the considered sonification techniques were unpleasant to listen to. This was to ensure that when designing an auditory display the users were not annoyed by the sounds. The main questions to be answered were:

5.3 Initial Test Hypotheses

- What side of the colour spectrum would people associate different sounds with?
- What sounds do people like and dislike? And are these related to the frequency content?

Test 2 was devised to test the relationship between pitch, and colour. As discussed previously, Isaac Newton and Louis Bertrand Castel had theories about representing colour on the notes of a piano. This test was designed to determine if any patterns emerged from testing individuals listening to nine random notes in a scale. The main areas of interest were:

- Is there any correlation between the colours and the pitches we hear?
- What colours do we associate with specific pitches? For example, is red a high, or a low pitch?

To answer the questions associated with each test some testable hypotheses were written. These are outlined, and discussed below.

5.3.1 Hypotheses – Test 1

For Test 1 of this study it was believed that *participants would inversely associate the colour spectrum with frequency* – red colours would imply high frequencies, and they would associate violet colours with low frequencies. This hypothesis was drawn because we typically associate red colours with danger, and therefore high frequency content, as outlined in Section 3.5. Additionally, it was believed that *participants would rate the high frequency sounds as sounding worse than the low frequency sounds*. This hypothesis was based on the notion that we associate the colour red with danger, and therefore unpleasant sounds.

5.3.2 Hypotheses – Test 2

For Test 2 of this study it was believed that *participants would associate red with high pitches, and purple with low pitches*, the reasoning being that if a pitch is of high frequency it is more alerting than a lower frequency pitch. Additionally it was thought that *those with musical training would try and relate the colours to notes in a scale*; detecting intervals between specific sounds and making judgments from this information. Those without musical training would not have the skills to be able to contextualize the pitches' relation to each other. Those with perfect pitch, or relative pitch, would be able to associate colours with pitches better; they could fill in the gaps for pitches they did not instantly associate with a colour.

It was thought that those with some degree of timbre-based synaesthesia (associating certain timbres of instruments with certain colours) would associate all sounds examples with one or two colours, and those with some degree of pitch-based synaesthesia will associate specific areas of frequency with specific frequencies. Those with some degree of timbre-based synaesthesia would associate specific instruments, such as the piano (in this case), with a certain colour. Those with pitch-based synaesthesia would associate each pitch with a different colour.

5.4 Designing the Sound Examples

Some sound samples were designed for the tests. The sounds for Test 1 were generated in Pure Data, and the sounds in Test 2 were generated using a sampled piano. Each sound was played three times to ensure that the participants were able to hear the sound enough times to make a decision.

5.4.1 Designing Test 1

When designing this test, a series of sounds were generated in Pure Data. An overview of the how each sound was generated is outlined in Table 5.1. Each patch features a sound that is ramped up and down in amplitude using Pd's 'vline' object. All samples were recorded in stereo and bounced down as .wav files to ensure consistency.

Sound	Description
1.1	This patch filters white noise with a high frequency band pass filter. The centre of the band pass filter was chosen to be 1500 Hz, as this was representative of a relatively high frequency. It produces a high-pitched wind like sound.
1.2	This patch filters white noise with a low frequency band pass filter. The centre of the band pass filter was chosen to be 500Hz, as this was representative of a low frequency. It produces a low-pitched wind like sound.
1.3	This patch produces a burst of noise across all frequencies using the 'noise' object. It produces a harsh broadband noise sound.
1.4	This patch produces a high pitched sinusoidal beep. The frequency 1000 Hz was used as this was considered a high, but still easily audible frequency.
1.5	This patch produces a low-pitched sinusoidal beep. The frequency 300 Hz was used as this was considered a low frequency, but high enough to be heard on most speakers.
1.6	This patch filters an 80Hz phasor with a band pass filter with a high centre frequency. A value of 1500 Hz was chosen as a fitting high frequency as it was easy to hear, but is still noticeably high pitched. The phasor's higher harmonics are emphasized, but it is possible to hear its lower harmonics too.

5.4 Designing the Sound Examples

1.7	This patch filters an 80Hz phasor with a band pass filter with a low centre frequency. A value of 400 Hz was chosen as a fitting low frequency as it was easy to hear on all speakers, but still noticeably low pitched. This sound features less of the high frequency content mentioned in the description of <i>Audio Example 1.6</i> .
-----	--

Table 5.1: Table of Pure Data patches developed

The patches outlined in Table 5.1. have been combined into one patch for easier navigation. This patch is available on the project CD associated with this thesis and is labelled '*Preliminary Test Patch*' in the '*Audio Examples*' folder. Additionally, the sound examples made for this test are available in the same folder.

5.4.2 Designing Test 2

For the second test some piano notes were generated. To do this, nine notes between C3 and B3 were chosen and randomized. The notes used, along with their respective sound example number, are outlined in Figure 5.4.

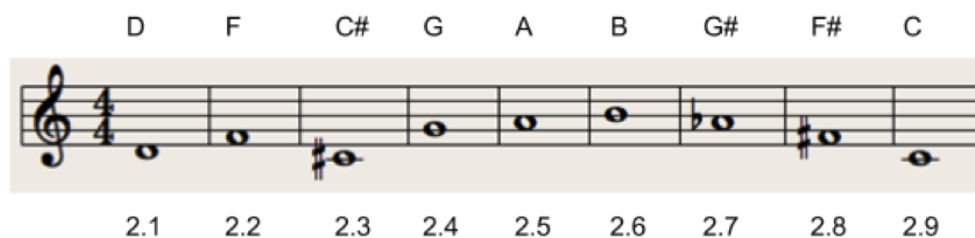


Figure 5.4: Notes of piano used in user testing

The digital audio workstation Logic Pro 9 (more information about Logic can be found here: <https://www.apple.com/uk/logic-pro/>) was used to create the samples, its inbuilt sampling engine (the EXS24) was used as it offered relatively consistent sampling across the note range. Each sound was repeated three times to ensure that the participant was able to hear the pitch, and any other nuances when listening. The samples were then bounced down as stereo .wav files ready for use in Windows Media Player.

5.5 The Experimental Procedure

The experiment took place in the Genesis 6 Teaching Room, Audio Lab, Department of Electronics, at the University of York. The room was chosen because of its sound treatment and low people-traffic – it was unlikely that anyone would disturb the experiment. For each participant it was important that they were not provided with any information before the test, so they were given the participant handout (see *Appendix B*) as they entered the room. The participants were given the same set of instructions, and the setup (outlined in Figure 5.5) did not differ between individuals.

Once the participants were familiar with the testing procedure, and had filled out a short demographics form, they were asked to complete both of the tests by playing the sounds in a Windows Media Player playlist. The subjects were instructed to listen to all the sounds related to the test. This was done to ensure that they were informed of all the sounds before they began filling in the questions. After doing this they were free to begin marking the colours they believed to be related to the sounds in any order. The participant handout informed them that they were allowed to go back and re-listen to the sounds, so that they could make comparisons as they went along. This ensured that they were not influenced by the sound that preceded it; reducing bias as much as possible.

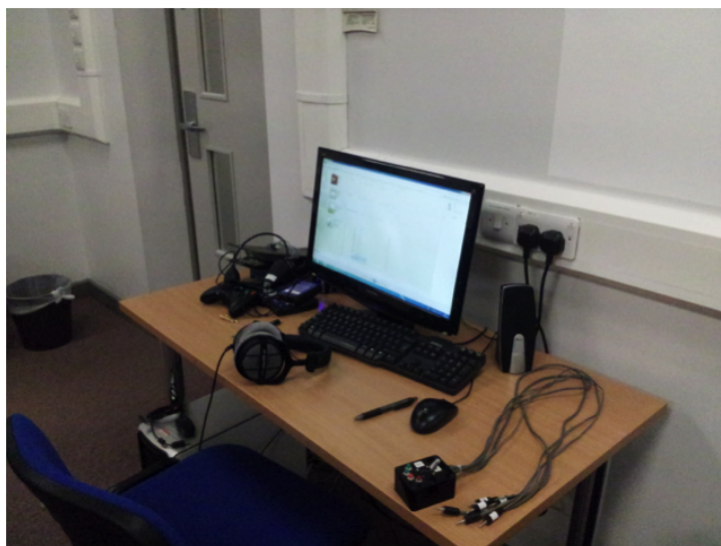


Figure 5.5: The setup used for the preliminary test

The participants typically took between seven and ten minutes to complete the experiment. During this time the interaction with the tester was minimal. Occasional questions were asked to clarify technical issues, problems with the media player, what way to put on the headphones, etc.

5.6 The Demographics

As this was a small preliminary test, a large number of participants was not needed as the results were not required to be significant, but to merely give an indication of some trends. Overall, 15 participants took part in the experiment. They were mostly students from the Department of Electronics at the University of York (10 people), however, there were two students from other departments; Computer Science, and English and Related Literature. Additionally, there were three non-students; these were audio lab staff or technicians. The participants' ages ranged from 22 – 40, with an average age of 26.7. The participants represented several nationalities (British, Chinese, American, Belgian, Greek, Dutch and Russian). Out of the 15 participants,

10 were male, and five were female.

Due to the fact that the subjects were mostly students, and largely from an audio-based research group, there were significantly more musicians than an average group of people; all but one participant considered themselves some level of musician. In the group, 4 people considered themselves to be professional/ex-professional musicians. 13 out of 15 considered themselves to have some degree of relative pitch, and two out of 15 considered themselves to have perfect pitch. A third of the participants considered themselves to have some degree of synesthesia, most stating that they related certain pitches, or timbres, to specific colours. Notable comments included:

“I associate frequency with colour; if I’m recording some music, I label the lower frequency instruments such as bass guitar and bass drums with colours like purple, and as the frequency of the instruments increases I begin to use colours like orange and red.”

“It’s difficult to describe with words, but for instance: I usually would associated a saxophone playing jazz with some orange”

5.7 The Results

The collected results were put into an Excel spreadsheet so that they could be analyzed. The results for each test is now explained and compared with the hypotheses.

5.7.1 Test 1 – The Results

From Figure 5.6 it is possible to see that sounds with predominantly high frequency content, such as 1.1, 1.3 and 1.4 are associated with the colour red more than blue, with the exception of Sound 1.1, in which they are the same,

5.7 The Results

and Sound 1.6, in which it is associated more with purple. Moreover, sounds with low frequencies, such as 1.2, 1.5, 1.6 and 1.7 are strongly associated with violet. Notably 13 out of 15 participants associated the high-pitched pure tone with red. It was found that both sounds that involve a phasor were strongly associated with the colour purple.

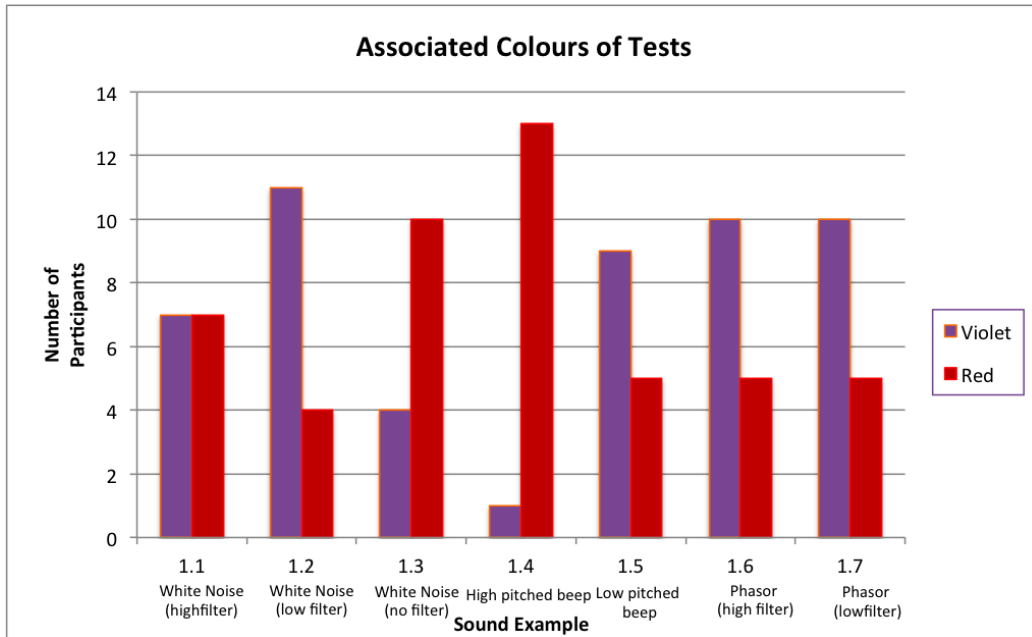


Figure 5.6: Colours the users associated with sounds

Those with synaesthesia showed similar results to the average; however, the correlation seemed to be stronger. This is outlined in Figure 5.7.

With regards to sound preference; it was found that sounds that focused on low frequencies, such as 1.2, 1.5, and 1.7 were the most enjoyed sounds by the participants, and the sounds that used high frequencies were the least enjoyed. The most disliked sound was the pure noise sample (Sound Example 1.3), with an overall score of 1.5. This is outlined in Figure 5.8.

5.7 The Results

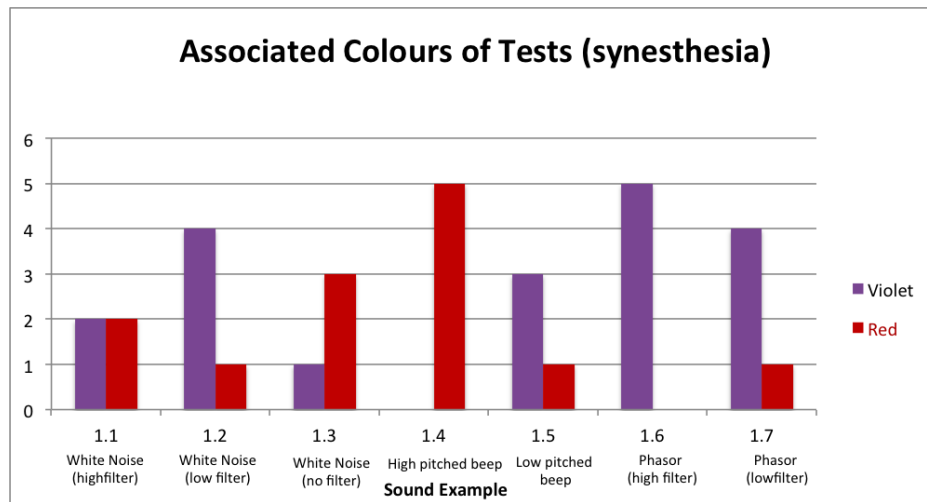


Figure 5.7: Colours the users associated with sounds for synesthesthetic listeners

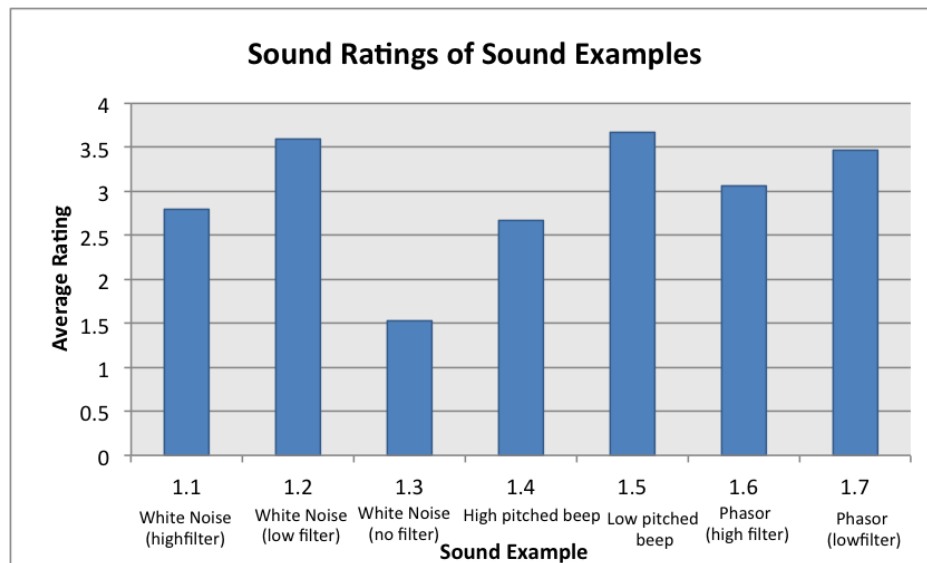


Figure 5.8: User's preference to sounds in Preliminary Test

5.7.2 Test 2 – The Results

The results for Test 2 were more clear than Test 1. Although there was no strong correlation between the participants' results (Figure 5.9), the pat-

5.7 The Results

tern partially resembled the hypothesised pattern (low-pitched sounds being closer to purple, and high-pitched sounds being closer to red). in Section 5.3 – for example, the low pitches, such as C, C# and D all have high end spectrum colours as their most frequently associated, and G#, A and B all have low end spectrum colours as their most frequent. These results are also outlined for each individual colour in Figure 5.10. From this, it is evident that the users associated certain notes with certain colours more than others. For example, just under half of the participants associated the note ‘D’ with the colour blue.

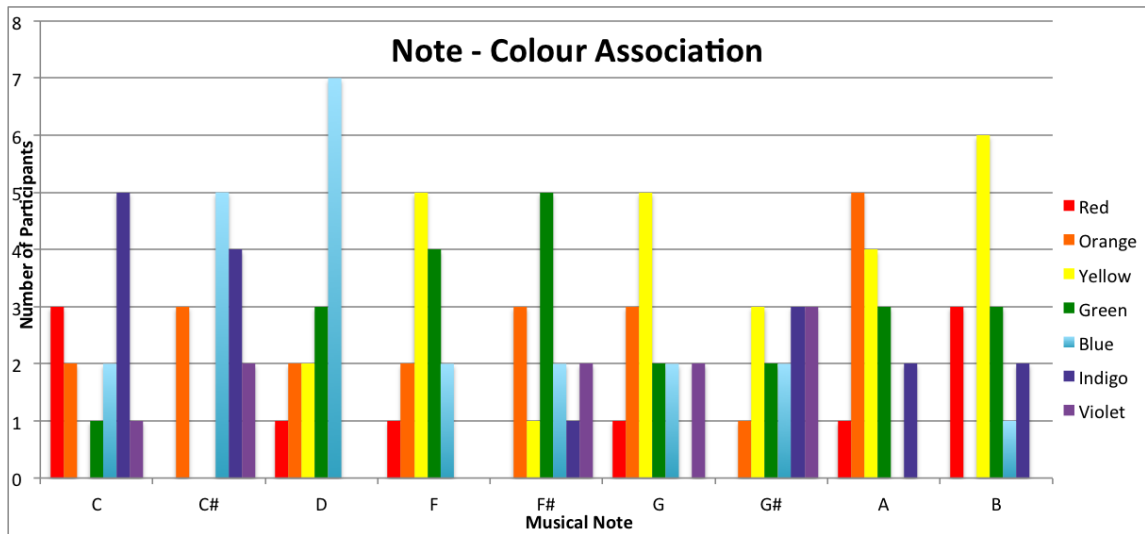


Figure 5.9: Note to colour association in Preliminary Test

Those who considered themselves synaesthetic to some degree appeared to show stronger correlation with the hypothesis. By excluding the participants who considered themselves non-synaesthetic a slightly stronger trend (though only four participants) appeared, as shown in Figure 5.11. Note colours such as violet and purple occur slightly more often at the lower end of the scale, however, although colours such as red appear more often at the higher end of the scale, there are many inconsistencies.

5.7 The Results

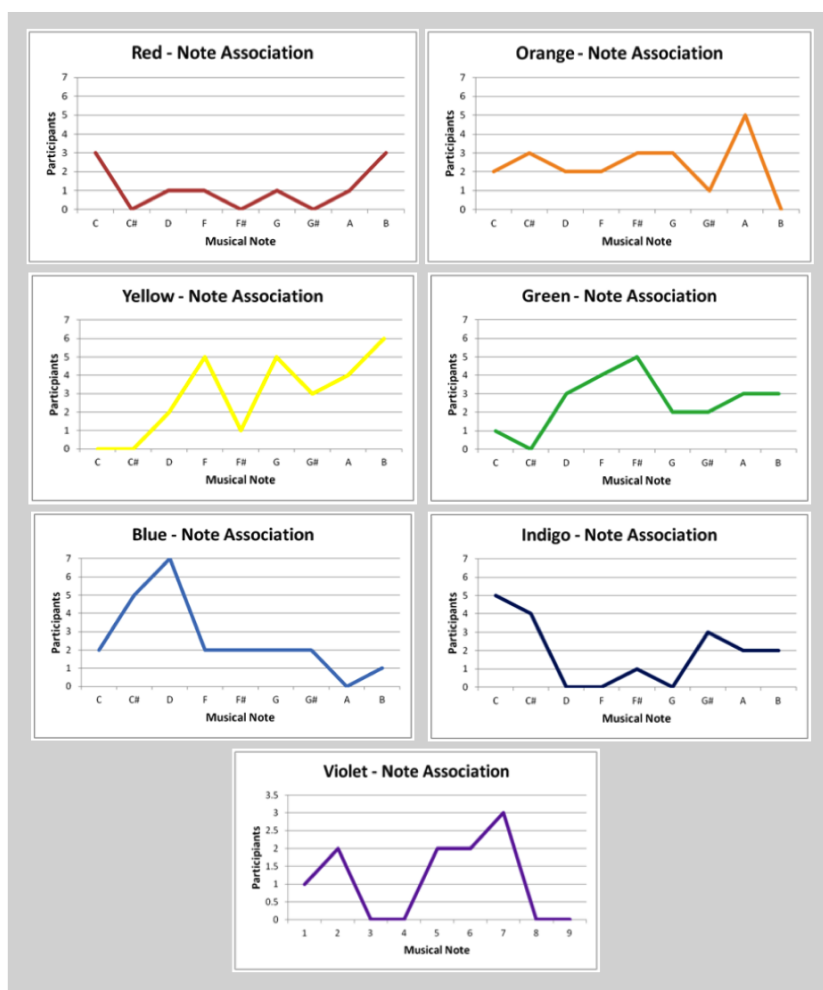


Figure 5.10: Note to colour association for individual colours

To investigate if a person being pitch perfect made a difference with regards to the trend, those without perfect pitch were negated (13 out of 15 participants). It was evident that those with perfect pitch strongly associated the higher notes with red, orange and yellow, and the lower notes with the colours violet and blue, bar a few anomalies. This is outlined in Figure 5.12.

5.7 The Results

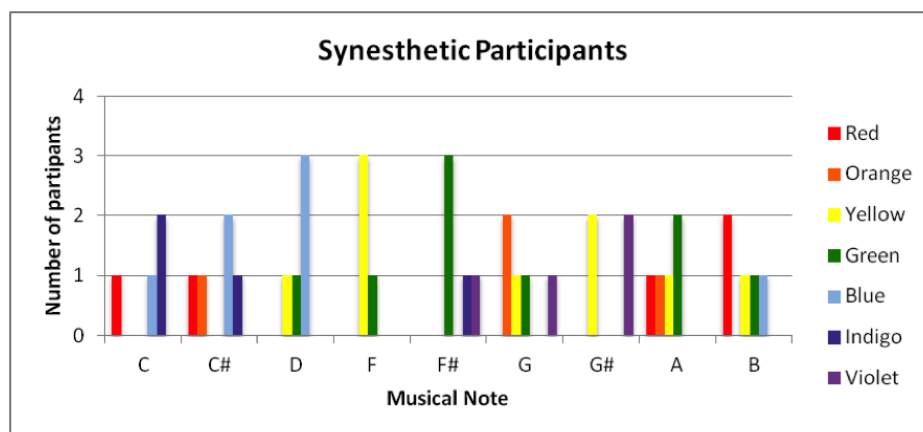


Figure 5.11: Note to colour association for synaesthetic people

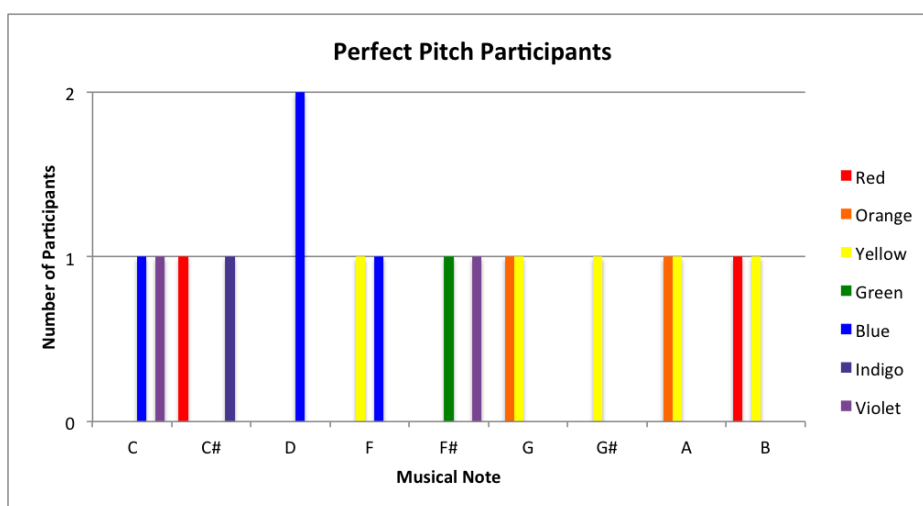


Figure 5.12: Note to colour association of pitch perfect people

Those who expressed some degree of relative pitch were then tested – this was the majority of the participants and is shown in Figure 5.13. There appears to be less correlation than in the perfect pitch participants with a significant anomaly; red being the joint most frequent colour associated with the lowest note.

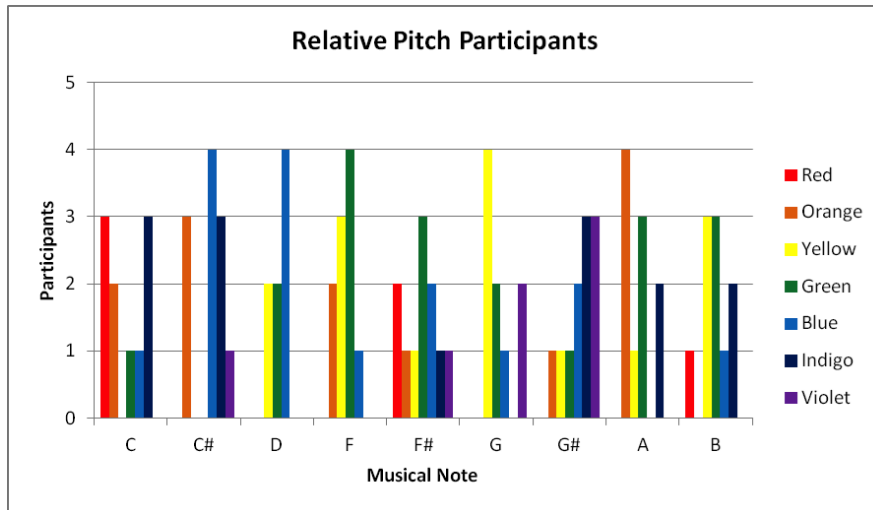


Figure 5.13: Note to colour association for those with relative pitch

To test if such strong correlation was only evident in those with perfect and relative pitch, those with no sense of relative, or perfect pitch were negated – leaving only two participants with no pitch training. The results showed strong correlation between the colour and the pitch the two individuals associated it with, as shown in Figure 5.14.

5.8 Test Conclusions

This section discusses the results and compares them to the preliminary test hypotheses. The implications of these results is also described regarding how the results could be used to design an auditory display that transforms graphical data into sound.

5.8.1 Test 1

It was hypothesised that people would inversely associate the colour spectrum with audio frequency, i.e. they would associate low frequency sounds

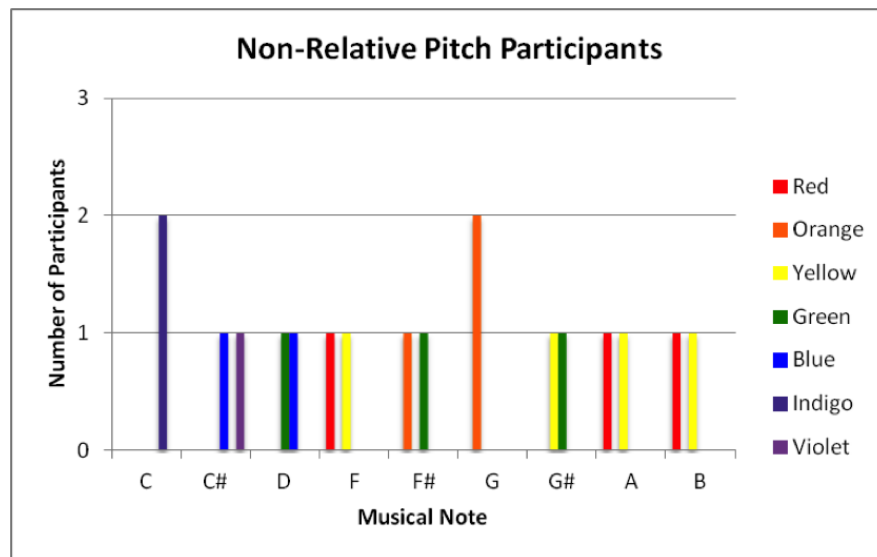


Figure 5.14: Note to colour association for those with no musical training

with high pitched notes, and vice-versa. The results support this; it is evident that the sounds with higher frequency content were regarded as ‘red sounding’, and sounds with lower frequency content were thought to be ‘violet sounding’. This information implies that when transforming graphical data into sound, the most intuitive mapping for colour is to inversely map the spectrum to auditory frequency – red being the highest auditory frequency, and violet the lowest.

An additional hypothesis in this study was that people would rate high frequency sounds as worse sounding than low frequency sounds – and again the data seems to support this. It was evident that the most universally disliked sound was broadband noise, followed by sounds that have high frequency content; these also often coincided with the sounds that were marked as ‘red sounding’. This implies that when designing an auditory display, high pitched sounds should be associated with ‘bad’ occurrences – much like the alarm sound devised by Alyte Podvoiskis (described in Section 3.2) when the user moved the mouse over potentially dangerous cervical cells. It was found

that the filtered phasor sounds were always associated with the colour purple. There are two potential explanations for this: either that the complex tone of a phasor is ‘purple sounding’, or that the users associate the low frequency fundamental tone (80Hz) with purple, regardless of the filtering on the upper harmonics.

5.8.2 Test 2

Prior to conducting the experiment it had been suggested that participants would associate high piano notes with red, and low piano notes with violet. There was not as much correlation in the results as Test 1, but there was still some evidence to support this hypothesis. It was evident from observing the results that some individuals did not associate the colours with pitches at all; numerous participants associate the timbre of the piano with a specific colour, causing the results to have less correlation than in Test 1.

The non-synaesthetic participants were negated to test if they supported the hypothesis more than the overall group. After doing this a stronger pattern emerged. These results appeared to show a slightly stronger correlation; with the high end of the colour spectrum being predominant with the lower piano notes, and the low end colours in the higher notes. The perfect pitch and musically untrained participants’ results were then compared. Apart from some inconsistencies both groups seemed to associate violet, blue and indigo with the lower pitched notes, and yellow, orange and red with the higher pitched notes.

While some people associated the pitch of the note with the colours, others related colours to the timbre of the note. This is evident from some individuals choosing only two colours, as they associated the piano with these colours. This was also outlined by some participants noting that they experienced synaesthesia, but only with timbres, instruments, and even genres of music. Though there was some correlation in those with synaesthesia, some

5.9 Limitations of Pilot Test

anomalous results were evident – for example, individuals choosing blue and green as high frequency colours, and some associating red with the lowest colour. It was thought that this was due to some participants associating the timbre of the piano with a particular colour, and not the pitch.

It was noted that as well as red, yellow was a colour that was associated with the high pitched sounds. It was also featured as the most common colour chosen by individuals; being the most associated colour with the notes F, G, G# (joint) and B. It could be suggested that this is to do with how people associate specific colours with certain instruments – a bright sounding instrument such as a piano could be associated with the colour yellow because of its timbre. However, tests on more instruments would be needed to be done to support or refute this.

5.9 Limitations of Pilot Test

This experiment could be expanded in several ways to verify its accuracy and improve its results. These are outlined below:

Randomize the colour choices – in the experiment the participants were asked to choose from a series of colours. These colours were always in the same order. Perhaps this could have influenced the decisions they made. It is recommended that future experiments negate this possibility by randomizing the order of the colours presented to the participant.

Use notes from multiple octaves – to see if the participants associate the specific notes outside of this octave, a larger range could be used. It would be interesting to find out if a participant associated the same notes from different octaves with the same colour.

5.9 Limitations of Pilot Test

Use a variety of instruments – some participants noted that they associated the sound of a piano with specific colours. To attain a better understanding of how we associate frequency and colour more instruments could be used.

The most useful finding in this experiment was that the participants associated the colour spectrum inversely with the auditory spectrum – they believed that colours such as red and orange should be associated with high pitched sounds, and that colours such as purple and blue should be associated with low frequency sounds. Though this experiment did not prove this definitively, it provides enough information to inform the sound mapping design. The association of sound and colour can now be tested further, in the context of interactive sonification.

6 Design and Implementation

This section describes the thought processes, design, and implementation behind the development of interactive sonification techniques on the iPad. The implementations are described in depth with regards to the philosophical choices behind the design, the design itself, and some initial feedback from users. The implementation of the techniques is described such that others may understand the test procedure associated with this thesis, and replicate its results.

All the techniques developed in this section are included in the final tests, discussed further in the next two sections. The tests are available in the form of Xcode projects, written in Objective-C, on the CD accompanying this project under ‘Xcode Projects’.

The techniques developed in this section should support users in finding image features on the iPad screen with no visual cues. An additional aim was to allow a non-visually restricted user to locate image features faster than without auditory feedback. The methods developed support a goal-oriented searching approach, where the user is searching for something specific, e.g., a black dot on a white background, or a specific shade of blue in a complex set of colours. An overview of the procedure to locate graphical features, and present them as audio is displayed in Figure 6.1. In the following sections, each step is broken down and its development discussed and documented. An outline of this section is described below:

Section 6.1 describes the initial image processing algorithms developed to determine where the image features of interest are.

Section 6.2 outlines how the ‘extended display’ screen was developed.

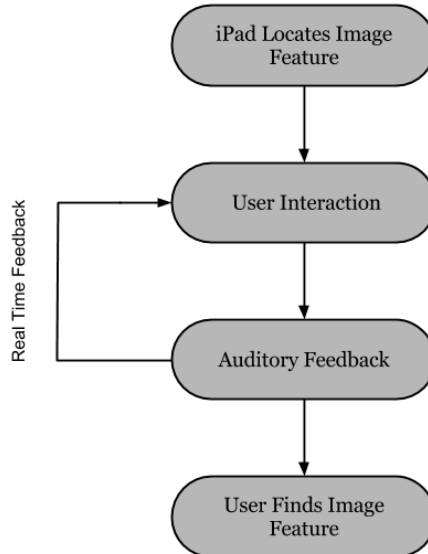


Figure 6.1: Overview of interactive sonification system to be developed

Section 6.3 describes how the touch interaction was implemented.

Section 6.4 discusses the calculations for determining the relationship between the touches and the image features.

Section 6.5 discusses the method of linking the audio engine (Csound) and iOS.

Section 6.6 outlines how the mapping between the processing in iOS relates to the sound. This section documents the development of the sonification mapping strategies in depth.

6.1 Location of Image Features

An algorithm was developed to locate the average point of a specific colour in an image on the iPad. The philosophy behind the design is described, along with the design itself. A full description of the implementation is then given, accompanied by a section evaluating its success.

6.1.1 Philosophy

To locate a specific image feature we must know what characteristics are desired, and find a way to negate the unwanted aspects of the image. In the context of this system, the desired characteristic of the image is a specific pixel range. Much like the work of Podvoiskis [Podvoiskis, 2004] and Stammers [Stammers, 2006] (as described in Section 3.2), by searching for a specific colour it is possible to home in on what we are searching for, as opposed to trying to ‘hear’ a specific colour out of a complex auditory field. As discussed in Section 4.2, the more minimalistic filtering of the graphical data before processing into sound seems to benefit still images, more so than a complex auditory field.

Feature detection was considered for use in this project – a process which can provide a highly ‘filtered’ approach to analysing the data. However, if scaled-up to search for multiple features, it has been shown to be cumbersome in the context of sonification [Hines, 2007].

6.1.2 Design

The system developed to detect specific colours should act as a filter – only allowing pixels of some RGB range through when rastering the image. The location of these pixels can then be registered, as shown in Figure 6.2 (next page). Once the filtered pixels’ locations have been logged, it is possible to find the average ‘x’, and ‘y’ co-ordinates of the image feature using Equation

6.1, where ‘P’ is the filtered pixel’s coordinate, ‘d’ is the direction of the rastering algorithm (‘x’ or ‘y’) and N is the number of pixels filtered in this axis.

$$\Delta_d = \frac{\sum_{P_d=0}^{N_d} P_d}{N_d} - t_d \quad (6.1)$$

6.1.3 Initial Evaluation

Initial testing with simple images shows that the algorithm is able to find the average location of simple coloured dots. By creating custom images in which the values of the pixels are known, it is possible to make sure that the algorithm is finding the specific features. By changing the statements on the simple filter, it is possible to search for any colour, providing its RGB values are known.

6.1.4 Final Implementation

The raster-based system outlined in the design section was implemented using the Core Graphics API. The code in the algorithm is now described and depicted using a flow chart in Figure 6.2.

The ‘locateImageFeature’ method was written to implement this algorithm. The following two lines load an image (in this case “example2.jpg”), which is stored in the project file, and extracts its pixel data:

```
1 UIImage *myPicture = [UIImage imageNamed:@"example2.jpg"];
2 CFDataRef pixelData=CGDataProviderCopyData(CGImageGetDataProvider(
    myPicture.CGImage));
```

The second line creates a CGImage (Core Graphics Image) and removes the CFData’s information from it. CFData is a data object that allows access to the values of the image. This stores the red, green, blue and alpha values of

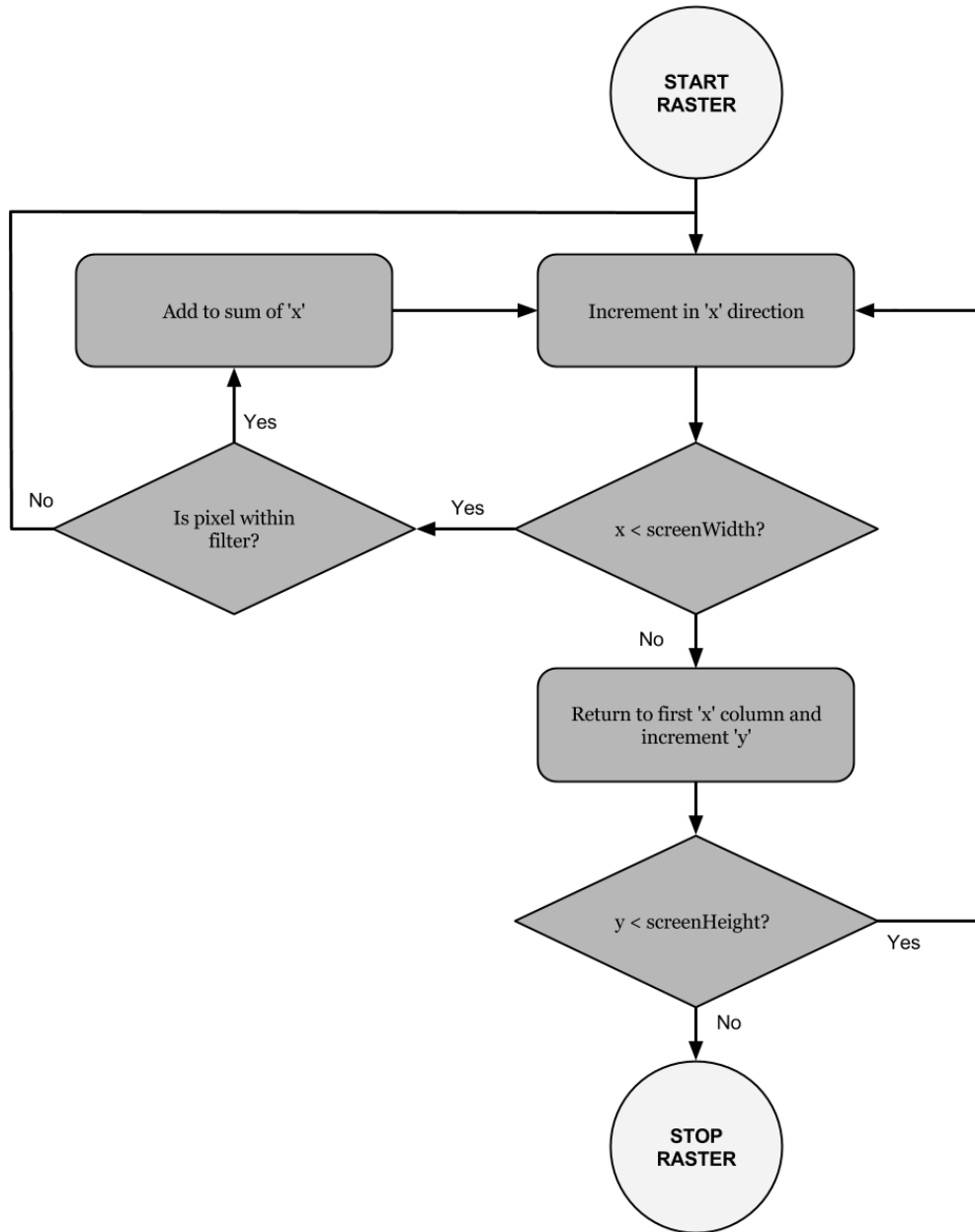


Figure 6.2: Rastering algorithm to store pixel data

6.1 Location of Image Features

the entire image. The width and height of the image is then obtained using the following code:

```
1 myWidth = CGImageGetWidth(myPicture.CGImage);
2 myHeight = CGImageGetHeight(myPicture.CGImage);
```

‘CGImageGetWidth’ and ‘CGImageGetHeight’ are methods that return the dimensions of the image in pixels, which allows the algorithm to be used on any given image size. A read-only pointer is then set up to return the values from the CFData’s stored data:

```
1 const UInt8 *pixels = CFDataGetBytePtr(pixelData);
```

It is then possible to implement the rastering algorithm:

```
1 for( xCoordinate = 0; xCoordinate < myWidth; xCoordinate++)
2 for( yCoordinate = 0; yCoordinate < myHeight; yCoordinate++)
3 {
4     int pixelStartIndex = (xCoordinate+(yCoordinate*myWidth))*
        bytesPerPixel_;
5     UInt8 redVal = pixels[pixelStartIndex]
6     UInt8 greenVal = pixels[pixelStartIndex + 1]
7     UInt8 blueVal = pixels[pixelStartIndex + 2]
```

The double ‘for’ loop (lines 1 and 2) allows for the rastering of the image. Once the algorithm reaches a new pixel, its red, green, blue, and alpha values are extracted by using ‘CFDataGetBytePtr’; a method that returns the information within the CFData object. The ‘pixelStartIndex’ is merely an offset of 4 to skip every 4 values in the CFdata – the ‘redVal’, ‘greenVal’ and ‘blueVal’ are then extracted. Figure 6.3 describes a pixel’s value, and names them in the context of this algorithm.



Figure 6.3: Pixel values in CFData

Now that the colour values for each pixel have been accessed, an 'if' statement can be used to filter the pixels. An example of searching for a green pixel is shown below:

```
1 if (blueVal < blueThreshold && redVal < redThreshold && greenVal >
   greenThreshold)
2 {
3   sumX += xCoordinate;
4   countX += 1;
5   sumY += yCoordinate;
6   countY += 1;
7 }
```

For every pixel located, the 'x' and 'y' co-ordinate is added to a sum and the variable 'countX' is incremented. The average of the co-ordinates is then calculated by dividing the sum of the co-ordinate values by the number found, for both 'x' and 'y' respectively, as outlined in Equation 6.1. This is shown below:

```
1 averageY = sumY/countY;
2 averageX = sumX/countX;
```

From this, a value, or series of values (dependent on the number of filters), is determined as the average for a specific colour.

6.1.5 Discussion

This algorithm allows any size of image to be loaded in and for the average position of a specific colour to be found. It is ideal for simple images, allowing for the average position of any shape to be located. However, in situations where there are multiple shapes, and the code has not been adapted manually to encompass this, the algorithm will falter. Therefore suggested further work may include adding extra functionality such that a more ‘universal’ approach can be adopted. This would involve a method that determines how many ‘image features’ there are, and then the existing algorithm would be adapted to incorporate a more flexible method for the amount of features it can intercept.

6.2 Extending the Display

As noted in sections 4.3 and 3.3, the notion of ‘extending’ the graphical display with auditory information can allow a user to sense an off-screen presence, which is ideal for large-scale images, or interfaces where there is more than one screen. Figure 6.4 describes a potential scenario – a user, represented by a face, is searching an image that is larger than the device’s display. The user is looking for specific bits of information to analyze – some of which are on-screen, some of which are not. Spatial sound can be used to extend the visual domain and aid them in travelling towards certain features. As the extended display will be larger than the iPad screen itself, the method to move around it must be both intuitive, and effective.

6.2.1 Philosophy

When considering methods to navigate such an interface it is important to separate the different modes of interaction. One mode involves operating some functionality within our visual field; when we interact, we produce some stimulus or action. This event is normally the end goal – for example,

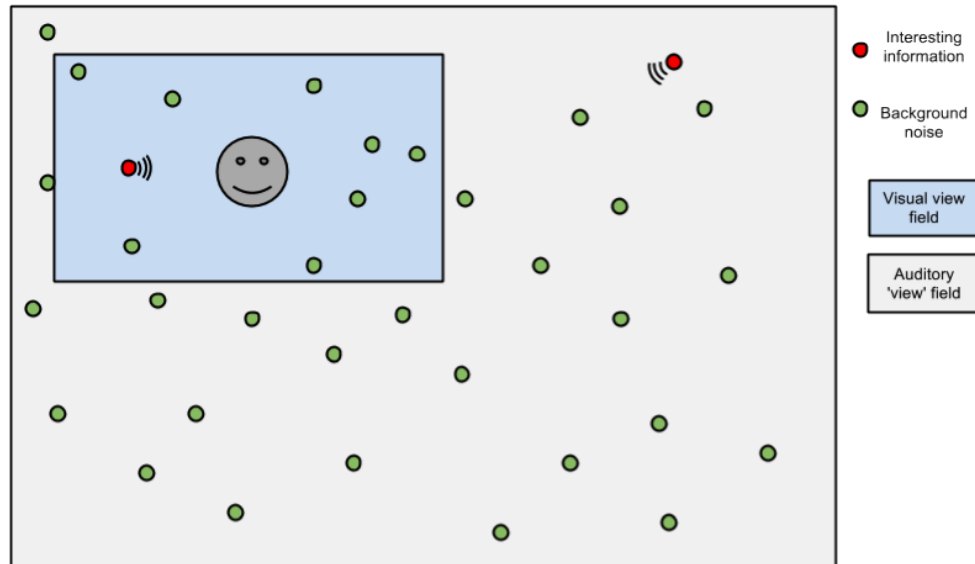


Figure 6.4: Extending the visual domain with auditory information

clicking an icon on a screen to open a program. Another mode of interaction involves locating an object to interact with, which may currently be off-screen. Swiping across a tablet display, typing text into a search-bar, or even speaking into a device are all examples of this. When developing for tablet display we are provided with a multitude of potential interaction modes. It is important that these are easily differentiated, such that each interaction mode the user learns works consistently.

6.2.2 Design

To create an extended display to navigate a large image, an interface must be created that allows the user to move freely around the image using gestures alone. The initial design used Apple's gestures in an attempt to produce intuitive interaction modes. It was decided that if a user wishes to navigate the extended display, a 'panning' gesture should be used. The panning gesture can be used to scroll around a display, or drag objects around a screen.

A minimum of two fingers should be used to operate the panning gesture. If single finger gestures are then used to gain auditory feedback the use of multiple finger gestures ensures differentiation when the user is trying to change their location within the scroll-view.

Another choice to be made was the directionality of the scroll-view; inverted, or non-inverted? When interacting with technology, specifically touch screen devices, many of us have opinions on whether our control should be inverted or non-inverted. As mentioned in Section 2.4.2, Hermann and Hunt note that ergonomics are very important when designing auditory displays, and that we must “*respect the bindings between physical actions and acoustic reactions that we have been familiar with since birth*” [Hermann, 2006]. So when making the decision between non-inverted and inverted it is important to consider ergonomics above all else.

It can be argued that inverted scrolling (or natural scrolling) should be used because it is what we are used to in the real world when interacting with a physical object to gain visual information from it – if one places a piece of paper on a table and focuses on a particular point, to view the information above this point they must move the paper down, and to view the information below this point, they must move the paper up. It is this ergonomic principle that fuelled the design choice to implement inverted scrolling in both the ‘x’ and ‘y’ directions.

6.2.3 Initial Evaluation

An informal initial evaluation suggested that the scrolling gestures were intuitive – the mapping was tried on three people, and all three found the inverted scrolling more understandable than non-inverted scrolling. They also found the differentiation between the two interaction modes instinctive. However, it was noted by two participants that if they were visually restricted and searching for information by sound alone, they would not know when they

hit the edge of the image. It was suggested that a sound could be used to indicate when a user hits a boundary. Suggestions included a ‘boing’ sound – resembling a springing motion – as a way of communicating intuitively that the user had gone too far. It was also noted that the scroll-view took too long to decelerate – the users wanted to be able to stop the scroll-view, so they could interact instantly with the information.

6.2.4 Final Implementation

The first step of the implementation of the scrollview in iOS was to drag a UIScrollView from the library of UI objects and place it on the storyboard in Xcode’s interface builder. As the size of the scrollview cannot be stretched beyond the size of the iPad screen in the interface builder, it needs to be enlarged programmatically. A UIScrollView, 9 (3²) times the size of the iPad screen (1024x768), was made using the following code:

```
1 [myScrollView setFrame:CGRectMake(0,0,3072,2304)];
```

The ‘setFrame’ method changes the frame of the UIScrollView, and the CGRectMake makes it a rectangle. Parameters 1 and 2 specify the origin (in terms of x and y coordinates), and parameters 3 and 4 specify the size of the scroll-view. The size of the contents that can be scrolled must then be set using the following code:

```
1 self.myScrollView.contentSize = CGSizeMake(3072, 2304);
```

The content size is set by setting the ‘contentSize’ property of the UIScrollView instance to change the content size of the scroll-view to that of a specified shape and size – in this case a rectangle nine times the size of the iPad screen. Once the scroll-view had been created, a method of adding an image to it was devised. A custom image-view class (discussed later) was created to display an image. The following code creates an instance of the class, tells the compiler to allocate memory for this, then initializes the view – a rectangle with origins (0, 0) and dimensions the same as the scrollview.

6.2 Extending the Display

```
1 CustomImageView *myImageView = [[CustomImageView alloc]
    initWithFrame:CGRectMake(0, 0, 3072, 2304)];
```

The ‘myImageView’ object’s ‘userInteractionEnabled’ property was then set to ‘YES’. This was done as follows:

```
1 [myImageView setUserInteractionEnabled:YES];
```

The UIImageView was then set as a sub-view of the scrollview:

```
1 [myScrollView addSubview:myImageView];
```

It was then possible to change the image the UIImageView displays programmatically:

```
1 myImageView.image = [UIImage imageNamed:@"test2.1.jpg"];
```

The image, in this case ‘test2.1.jpg’, must be stored in the main bundle. The property ‘image’ of the class UIImage is, with this code, changed to allow ‘myImageView’ to access the image stored in the main bundle.

As discussed in Section 2, Apple’s gesture recognizers allow for gestures to be registered, and then for the developer to implement code such that the appropriate action takes place. The following code uses the UIGestureRecognizer class to create a ‘gestureRecognizer’ object for the scrollview. The ‘UIPanGestureRecognizer’ is used to implement the scrolling functionality, described in Section 2.7.

```
1 for (UIGestureRecognizer *gestureRecognizer in myScrollView.
    gestureRecognizers)
2 {
3     if ([gestureRecognizer isKindOfClass:[
        UIPanGestureRecognizer class]])
4     {
5         UIPanGestureRecognizer *panGestureRecognizer = (
            UIPanGestureRecognizer *) gestureRecognizer;
6         panGestureRecognizer.minimumNumberOfTouches = 2;
```

6.2 Extending the Display

```
7     magnitude = 0;
8     }
9 }
```

The ‘minimumNumberOfTouches’ property ensures that the gesture recognizer does not trigger when the user is simply trying to interact with the device for auditory feedback – this is an important way to differentiate between the two aforementioned different modes of interaction. The variable ‘magnitude’ refers to the volume level of the audio engine – it was essential that the audio engine stopped when the user was not trying to get auditory feedback. The scroll-view’s deceleration rate was then altered in accordance with the information gathered in the initial evaluation. A faster deceleration rate was then attained using the following code:

```
1 myScrollView.decelerationRate = UIScrollViewDecelerationRateFast;
```

The ‘decelerationRate’ property of the UIScrollView class was used to change the time it took for the scrollview to slow down. The variable ‘UIScrollViewDecelerationRateFast’ is an iOS defined constant (float) that allows for a reasonably fast deceleration rate. In order to access the touch delegate methods of the UIImageView, a custom view had to be created. The interface was created in the .h file using the following code:

```
1 @interface CustomImageView : UIImageView
2 {
3
4 }
5 @end
```

Then the touch delegate methods, discussed in more depth in the next section, were coded within the implementation of the ‘CustomImageView’ class. The methods outside of the ‘CustomImageView’ class were then accessed using the following (example) technique:

```
1 FBPViewController *callMethod;
2 callMethod = [[FBPViewController alloc] init];
```

```
3 [callMethod someMethod];
```

6.2.5 Discussion

This implementation allows a user to navigate a large image in a smooth and intuitive manner. An image of any size may be loaded, and with minimal adjustments to the code, a user can scroll around the screen, and the functionality remains the same. It should be noted that the iOS `UIScrollView` class allows for additional functionality such as a touch stopping scrolling smoothly, and ‘stretching’ beyond the bounds of the image, and it bouncing back.

It was decided from user feedback that that there should be a ‘boing’ sound when the user has scrolled too far. To this end, a method was implemented that determined when a user had exceeded the bounds of the screen, and by how much, such that an appropriate auditory response could be produced. The following method was written to determine how far the user had exceeded the bounds of the view:

```
1  
2 -(void)scrollViewDidScroll: (UIScrollView*)scrollView  
3 {  
4     float scrollOffsetY = myScrollView.contentOffset.y;  
5     float scrollOffsetX = myScrollView.contentOffset.x;  
6  
7     if (scrollOffsetY <= -3)  
8     {  
9         boingPitch = abs(scrollOffsetY);  
10        boingTheta = 0;  
11    }  
12    else if...
```

The `UIScrollView` delegate method ‘`scrollViewDidScroll`’ was used to determine when the scroll-view was being moved by the user. Two float values

6.3 Touch Interaction

were then set to the ‘x’ and ‘y’ ‘contentOffset’ property of the scrollview. By doing this it was possible to gather how much the user had scrolled. Then a series of if/else statements were written to determine what happens if a user exceeds a certain boundary – in this case beyond the bounds of the screen. If the user goes beyond the bounds of the screen, the amount is then calculated and used in the variable ‘boingPitch’ to be sent to an oscillator to make sound. Additionally, binaural panning was added – when the user exceeds the left boundary it pans left, when it exceeds the bottom barrier it pans behind them, etc. If none of the boundaries are breached, the value ‘boingPitch’ is simply set to zero to ensure it makes no sound.

Video Example 6.1 shows interaction with the iPad simulator triggering ‘boing messages’

6.3 Touch Interaction

There are two main modes of interaction in this design. The method of interaction for navigating the interface was discussed in the previous section, and now it is important to discuss how the interaction for auditory feedback was implemented.

6.3.1 Philosophy

For any interactive sonification system, it is essential that its operation is seamless and intuitive. The design must allow a user to search the interface and gain auditory feedback, therefore, as discussed in Section 2, it should be real-time, reactive, and adhere to the interactive sonification paradigm. As considered in the previous section, the two modes of interaction should be fully distinguishable, yet complement each other when used simultaneously.

6.3.2 Design

To enable an intuitive touch-screen experience it was decided that the system should have the following functionality:

- The system should track the user's touches when they interact with the screen;
- The system should stop tracking touches when the user removes their touches from the screen; and
- When the user moves their touch around the screen, the system should update the user's touches coordinates in real time.

6.3.3 Initial Evaluation

The initial evaluation of this design concluded that, as the two modes of interaction were clearly distinguishable, the design held up to the requirements.

6.3.4 Implementation

To implement the system described in the design section, three main methods are required; one that tracks the user's touches when they interact with the screen, one that tracks the user's touches when they move them, and one that is called when the user removes them. There are three touch-based delegate methods in the 'UIResponder' class, provided with the iOS SDK, which can handle the touches in the way required. These are outlined in Table 6.1.

6.3 Touch Interaction

Method name	Description
touchesBegan	This method triggers when touches begin – it allows the developer to use the point touched as a set of Cartesian coordinates.
touchesMoved	This method is called when the touches are moved. The Cartesian coordinates then update in real time.
touchesEnded	This is called when the touches are removed.

Table 6.1: Touch delegate methods and their descriptions

The following code was used to implement the ‘touchesBegan’ method:

```
1
2 -(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
3 {
4     UITouch *touch = [touches anyObject];
5     if(touch.view == myImageView)
6     {
7         CGPoint position = [[touches anyObject] locationInView:
8         self.view];
9         xTouch = position.x;
10        yTouch = position.y;
11    }
```

The method uses an instance of the ‘UITouch’ class to tell when a specific area on the screen (myImageView) has been touched. If this area is touched, a CGPoint (Core Graphics Cartesian coordinate system) value is assigned to the object ‘position’. It is then possible to test this by printing the variables to terminal, to verify that the touch point is being tracked. A similar method can then be used to detect when the touches have been moved:

6.3 Touch Interaction

```
1 -(void) touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event
2 {
3     UITouch *touch = [touches anyObject];
4     if (touch.view==myImageView)
5     {
6         CGPoint position = [[touches anyObject] locationInView:
7             self.view];
8         xTouch = position.x;
9         yTouch = position.y;
10 }
```

When the touches are moved, the coordinates update in real time. It is then possible to use a similar technique to implement the ‘touchesEnded’ method, as shown below. Anything can be included in the method body, and will be run when the touches have been removed.

```
1 -(void) touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event
2 {
3     UITouch *touch = [touches anyObject];
4     if(touch.view == myImageView)
5     {
6         //Perform actions
7     }
8 }
```

A video of the final implementation of this system running on an iPad simulator can be found in *Video 6.2*.

6.3.5 Conclusions

The code written in this chapter allows for the tracking of the user’s touches when operating the sound feedback. As mentioned in the initial evaluation section, the design held up to the design requirements – there is a clear differentiation between the user touching the screen, moving their touches,

and the touches being removed. The touch information gathered can now be used by other methods to perform the calculations to drive the audio engine.

6.4 Touch to Feature Mapping

This section describes the development of algorithms to calculate the relative position of the target graphical feature and the current touch position. The calculations should then be used to communicate this information with the audio engine such that an appropriate sound mapping can be used.

6.4.1 Philosophy

When designing a method of determining a relationship between touch positions and a graphical feature a vector is used. An angle and a magnitude can be used to determine the direction, and the distance to an image feature. There are, however, some important things to consider with regards to the way the feature is mapped to the touch:

- (a) Should the image feature be the central point, and therefore the centre of the auditory field? (an exocentric approach) (Figure 6.5); or
- (b) Should the point touched be the centre of the sound field? (an egocentric approach) (Figure 6.6).

Of the two designs, design (a) allows for an approach where the user is the centre of the sound field. They would be free to explore the auditory world and sounds around them will relate to physical positions on, or off, the screen. Design (b) would allow for an approach where the target is the centre of the auditory world. The user would then try and aim for this point with their touch. Design (b) has the following drawbacks: The inability to track

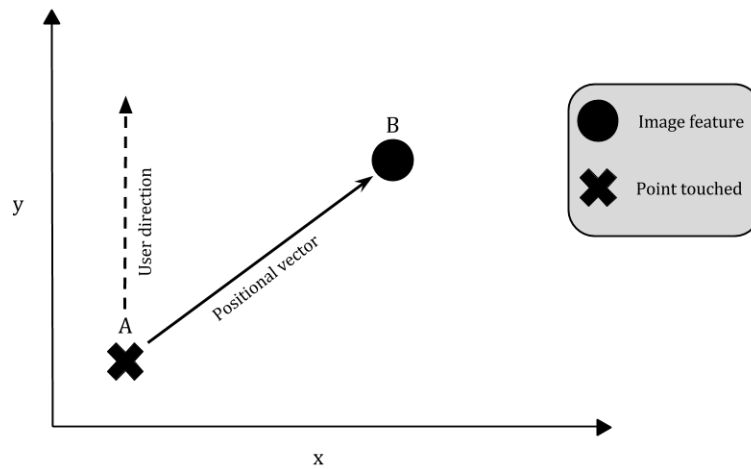


Figure 6.5: Vector between point touched and image feature

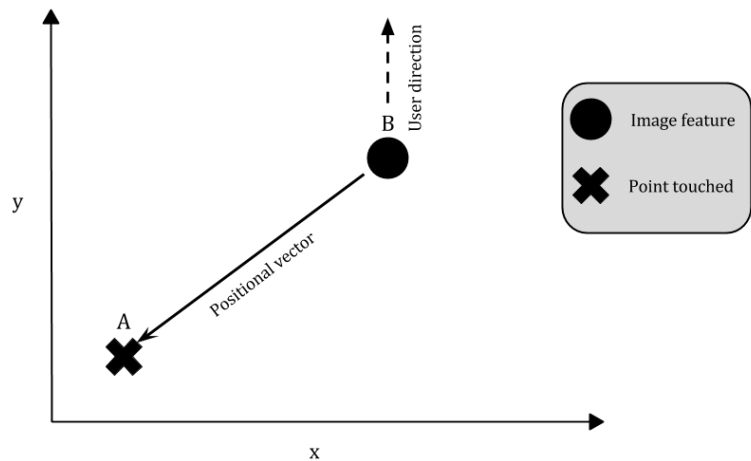


Figure 6.6: Vector between image feature and point touched

multiple sources – if the central point of the auditory field is a target, then it's impossible to track multiple sources simultaneously. A non-user centred design means that an exploratory approach would be less logical – the user would need to work backwards to determine the point of their own finger, not to determine the position of the source.

6.4.2 Design

As discussed in the previous section, a user-centred design will benefit an exploratory approach for searching images with sound. This means that all calculations should be from the perspective of the user's touch. To determine a vector between the touch of a user, and the image feature, Equation 6.2 and Equation 6.3 (Equation 6.1 written in terms of the 'x' and 'y' direction) can be used to calculate the change in 'X' and 'Y' positions between the touch and the image feature, where 't' is the touch point – its respective direction denoted by the subscript 'x', or 'y'.

$$\Delta_X = \frac{\sum_{P_{x=0}}^{N_x} P_x}{N_x} - t_x \quad (6.2) \quad \Delta_Y = \frac{\sum_{P_{y=0}}^{N_y} P_y}{N_y} - t_y \quad (6.3)$$

It is then possible to use the following result to calculate the angle (Θ) of the vector in degrees using Equation 6.4.

$$\Theta = \frac{180}{\pi} \arctan \left(\frac{\Delta x}{\Delta y} \right) - \frac{\pi}{2} \quad (6.4)$$

To complete the positional vector its magnitude ($|M|$) must be found. This can be done by rearranging the Pythagorean Theorem as shown in Equation 6.5.

$$|M| = \sqrt{\Delta x^2 + \Delta y^2} \quad (6.5)$$

Then it is possible to calculate the magnitude of the vector, as shown in Figure 6.7.

6.4.3 Implementation

To implement the functionality described in the design section, the average of filtered pixel coordinates for each dimension ('averageX' and 'averageY')

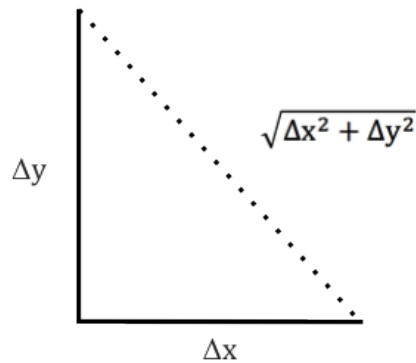


Figure 6.7: Calculating the distance between touch and image feature

must be known. As the knowledge to calculate this was acquired in Section 6.1, we must find a way determine the direction of the image feature relative to the user's touch. From Equation 6.4 it is possible to develop the following method to calculate the angle:

```

1 -(void) calculateAngle
2 {
3   deltaX = xTouch - averageX;
4   deltaY = yTouch - averageY;
5   theta = ((atan2(deltaY, deltaX) * (180/(M_PI))))-90);
6 }

```

Lines 3 and 4 calculate the difference between the touch point and the average point of the image feature, then line 5 uses Equation 6.4 to calculate the angle from the perspective of the touch. With this information, using Equation 6.5, it is possible to calculate the magnitude using the following method:

```

1 -(void) calculateMagnitude
2 {
3   float dx = xTouch - averageX;
4   float dy = yTouch - averageY;
5   magnitude = sqrt(dx*dx + dy*dy);
6   magnitude = abs((magnitude/100) - 10);
7 }

```

6.4 Touch to Feature Mapping

In this code, line 5 uses Equation 6.5 to calculate the difference in pixels between the touch point and the image feature. Here it uses a system called proximity zones to divide up the distance into smaller parts – this will be discussed in more detail in Section 6.6.1.

The final system devised in this section can be shown working in *Video Example 6.3*.

It was decided that if the user were to find the image feature, an alert sound should be used. If the user’s touch is within the shape, a message can be sent to the audio engine to trigger an appropriate sound; this was done in iOS by using the following method:

```
1 -(void) soundAlertFrequency
2 {
3     if(alertValue < alertZoneSize)
4     {
5         alertFrequency = 600;
6     }
7     else
8     {
9         alertFrequency = 0;
10    }
11 }
```

This method checks if the variable ‘alertValue’ is within ‘alertZoneSize’ pixels of the center point of the image feature. ‘alertValue’ being the value in pixels between the user’s touch, and the center of the image feature. It is calculated in the same way as magnitude. The ‘alertZoneSize’ should be set to be the radius of the desired image feature in pixels.

A video example of this working can on the iPad simulator can be seen in *Video Example 6.4*.

6.4.4 Conclusions

The methods developed find the vector between the user's finger and the shape in real time. The mapping appears to be intuitive. By providing this vector to the audio engine (discussed in the next two sections) the algorithm can now 'place' the user in the center of the auditory field representation of the image, and allow them to interact in real time to explore it. An area for improvement is the way the alert zone is triggered. Further work should involve finding a way to detect what colour pixel the user is touching, then checking to see if it corresponds to the colour the image feature detection algorithm is searching for.

6.5 The Audio Engine (Csound)

Now that a series of positional vectors have been calculated to drive the sound creation, the audio engine itself must be designed. It was decided, from the information gathered in Section 2.9, that the iOS-Csound API offered the best functionality for this project, as it allows for messages to be sent between iOS and Csound with ease. This means that the implementation can use the superior touch interaction and processing of iOS in tandem with the powerful Csound audio engine.

This section describes the code used to send data from iOS to Csound. However it will not describe how this API works, or how it can be used in a project. An in-depth tutorial, that was written (along with two other students) to facilitate this masters thesis, has been included in *Appendix C*. Additionally, its latest version is available from the following link:

<http://www-users.york.ac.uk/~adh2/iOS-CsoundABeginnersGuide.pdf>

The following two subsections explain the main methods and opcodes that the iOS-Csound API uses to allow the two platforms to communicate. The ‘calculateAngle’ method is used as an example to demonstrate how the information is sent using the API.

6.5.1 Sending Data from iOS to Csound

The ‘setup’ method (iOS-Csound delegate method) is used to inform Csound that it will receive a variable using the following code:

```
1 -(void) setup:(CsoundObj *)csoundObj
2 {
3     NSString *azimuthString = @"azVal";
4     azimuthChannelPtr = [csoundObj getInputChannelPtr:azimuthString];
5 }
```

Line 3 creates a string called ‘azVal’ (azimuth value) to give a name to the communication channel via which iOS will send values to Csound. Line 4 then creates a channel pointer such that iOS can write information to it. This is done using the ‘getInputChannelPtr’ method – a iOS-Csound API method. This pointer is then updated by assigning the variable ‘theta’ to the pointer’s memory address using the ‘updateValuesToCsound’ method. This method then sends this value to Csound at intervals dependent on the control rate. This is shown below:

```
1 -(void) updateValuesToCsound
2 {
3     *azimuthChannelPtr = theta;
4 }
```

It is then possible to repeat this process for all variables we wish to send from iOS to Csound.

6.5.2 Receiving Data from iOS in Csound

To receive the information in Csound the ‘chnget’ opcode is used to retrieve the information from a channel allocated by iOS. This is written in the Csound .csd file as follows:

```
1 kAz chnget "azVal"
```

The Csound opcode ‘chnget’, or ‘channel get’, takes the value from the channel ‘azVal’ and assigns it to a k-type variable in Csound – a value that updates in accordance with the control rate. Then it is possible to use this variable within Csound to control parameters of opcodes.

6.6 Auditory Mappings

This section documents how the auditory mappings used in the tests were developed. Each mapping’s theoretical design is discussed, and then its implementation in Csound is described.

6.6.1 Pulse Train

A pulse train, also known as a pulse wave, is a regularly occurring pulse in a signal. It can take the form of a number of types of wave – square, saw-tooth, triangle, etc. [Howard, 2005, pg. 471]. Generally this is used in digital electronics to regulate a timed process. However, it has also been shown that pulse trains can be successfully applied to auditory displays (as discussed in Section 3.2). Yoshida’s [Yoshida et al., 2011] method of increasing a pulse-train as a user gets closer to an image feature has been shown to be an intuitive method of helping a user locate the approximate region of a shape.

Rather than using Yoshida’s method of increasing the mapping linearly, it

was decided that a series of proximity zones should be devised so that the change in the frequency of the pulse train was more discrete (see Figure 6.8). Each proximity zone (partially inspired by the work in [McGookin and Brewster, 2001]), used a different speed of pulse train – the higher the proximity zone, the faster the pulse train. On an image the same size as the iPad screen, typically 9 proximity zones were used. For larger images, the number of proximity zones was scaled up accordingly so that the mappings remained consistent.

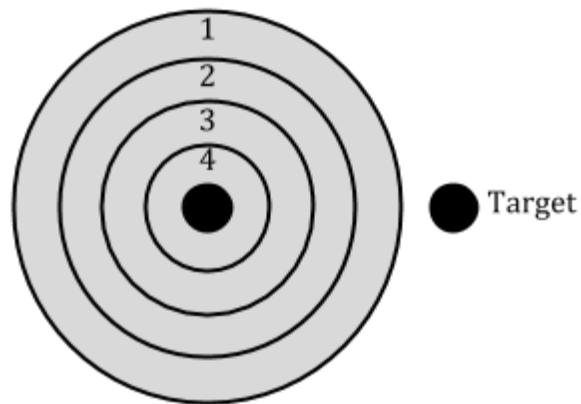


Figure 6.8: Proximity zone system used to represent distance

An initial evaluation concluded that small variations were hard to detect, and that larger and more discrete changes gave a better indication of how far away the feature was. After some experimentation it was found that nine proximity zones per screen was a good compromise. It was decided that using a pulse train to control the volume parameter of a white noise generator would be a good place to start when considering a binaural pulse train – white noise is an ideal source for inferring spatial cues to a user as it contains all frequencies. It was concluded that a saw-tooth wave should be used to drive the pulse train, as opposed to a square wave. A square-wave offers sudden bursts of signal, whereas a saw-tooth quickly ramps up the volume – offering a less harsh pulsing sound. Two waveforms describing these signals are depicted in Figure 6.9.

6.6 Auditory Mappings

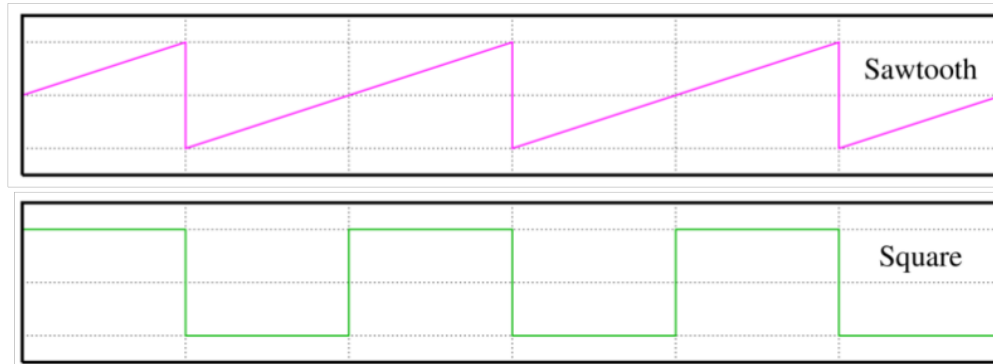


Figure 6.9: Depiction of a saw-tooth and a square-tooth wave – taken from <http://commons.wikimedia.org/wiki/File:Waveforms.png>

A comparison of the two sounds, made in Csound, can be heard in *Audio Example 6.1* (saw-tooth) and *Audio Example 6.2* (squarewave).

To make the pulse train in Csound an amplitude parameter was ramped from 0 to 1 using the following code:

```
1 kphs phasor kSpeed
2 asignoise noise kNoise * ( kphs > 0.1 ? 0 : 1 ), 0
```

‘asignoise’ is an audio sample variable that is used to output the process to the next section in the Csound code. ‘noise’ is the opcode used to generate the broadband noise, and its amplitude parameter uses the table index of a phasor opcode to ramp the amplitude from 0 to 1. The value 0.1 is the time (in seconds) that it takes for this to happen – this was set to a low value such that the pulse train can get to a high speed without overlapping with other pulses. The value kNoise is the amplitude value of the pulse train – if the user is touching the screen this is automatically set to a value deemed a good volume for the listener (assuming a medium volume level on the iPad). The value ‘kSpeed’ is sent from iOS in accordance with the method described in Section 6.5 – this value is controlled by the magnitude of the vector between the user’s touch and the image feature, as described in Section 6.4.

Video Example 6.5 shows this algorithm working with the previously developed image processing algorithms to depict the distance from a feature on the iPad simulator.

6.6.2 Binaural Panning

As discussed in Section 6.4, the angle of the vector is the value sent to Csound to control the binaural panning parameter. This parameter is the angle between the touch of the user's finger and the graphical feature, assuming the angle's origin is directly in front of the user. To implement this binaural panning functionality in Csound the following code was written:

```
1 aleft, aright hrtfmove2 asig, kAz, 0, "hrtf-44100-left.dat", "  
    hrtf-44100-right.dat"
```

'aleft' and 'aright' are the outputs – one for each channel. The 'hrtfmove2' opcode is based on the Woodworth spherical head model [Woodworth and Schlosberg, 1962, pg. 349 – 361] and was used as it has efficient and highly effective interpolation – ideal for a binaural implementation on iPad. Table 6.2 describes each of the arguments used in the implementation, along with their typical presentation in italics. By sending the angle of the vector from iOS, it is possible to tell the opcode to update the azimuth value of the opcode (*kAz*). By doing this it is possible to then output a binaurally panned version of the pulse train dependent on the angle of the vector coming from iOS.

Video Example 6.6 depicts the binaural panning working along with pulse train to provide auditory feedback on an iPad simulator.

Argument	Explanation
asig	The opcode's input signal (<i>asrc</i>).
kAz	The azimuth value (<i>kAz</i>), in this case the angle variable sent from iOS.
0	The elevation of the source (<i>kElev</i>). In this case the source is only around the azimuth.
"hrtf-44100-left.dat"	The file (<i>ifilel</i>) that the opcode gets its HRTF spectral data file from – in this case, the left.
"hrtf-44100-right.dat"	The file (<i>ifiler</i>) that the opcode gets its HRTF spectral data file from – in this case, the right.

Table 6.2: Table of parameters the opcode 'hrtfmove2' takes as arguments

6.6.3 Colour Sounds

From the preliminary test (Section 5) it was evident that when dealing with simple colours, the sound should be mapped in a similar fashion to the colour spectrum – blue/purple colours should be represented by low sounds, green/yellow colours should be represented by medium sounds, and colours such as red and orange should be represented by sounds with higher frequency content. It was found that, in general, users preferred low frequency sounds, and that their preferred high frequency sound was the filtered phasor. Based on this information it was concluded that to represent different pitches a filtered phasor should be used to represent the colours red, blue, and green.

Some experimentation was done into choosing the right type of filtered sound. The final approach adopts a method similar to that of formant synthesis – using bandpass filters to cut notches into the frequency spectrum. For the red sound, a low pitched phasor with low pitch bandpass filters cutting notches into its spectrum was used, for the green a higher pitched phasor with higher

6.6 Auditory Mappings

bandpass filters, and for the red an even higher pitched phasor was used, with even higher bandpass filters. The implementation of the red synth, in Csound, is shown below:

```
1 aredPhasor phasor kredPhasorVal * (kphs > 0.6 ? 0 : 1 ), 0
2 aredBandPass1 butterbp aredPhasor, 700, 80
3 aredBandPass2 butterbp aredPhasor, 2000, 50
4 aredBandPass3 butterbp aredPhasor, 4000, 80
5 asigRedBands = aredBandPass1 + aredBandPass2 + aredBandPass3
```

The first line uses a phasor as the source of the pulse train using the same method as described in Section 6.6.1 – its fundamental frequency set by the ‘kredPhasorVal’. The next three lines filter this source using the ‘butterbp’ opcode. Values of 700, 2000, and 4000 were chosen for the center frequencies of the bandpass filters after some experimentation. The same experimental technique was used for the q-values – 80, 50, and 80. The outputs of the filters are then summed into a single output signal ‘asigRedBands’. By looking at the sound’s (*Audio Example 6.3*) visual representation (Figure 6.10), in the time domain, it is possible to see the pulse train repeating, and the energy at the fundamental (180Hz), at the first bandpass filter (700Hz), the second bandpass filter (2000Hz), and the third (4000Hz).

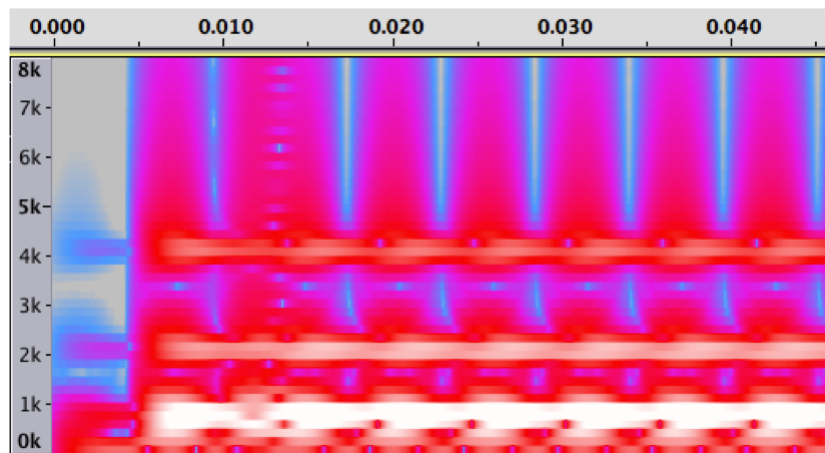


Figure 6.10: Time and frequency domain representation of colour synth

6.6.4 Volume

When the image-detecting algorithm seeks and detects more than one colour, multiple synths can be used to represent each feature. This presents a new challenge – how to best differentiate the sound sources. When multiple sources are being tracked, and played together they tend to clash unless some volume parameter is added. For this implementation the volume mapping described in Equation 6.6 was used.

$$Outputvolume = \left(\frac{Originalsignal}{Scalingfactor} \right) Magnitude^2 \quad (6.6)$$

The volume is first divided by a scaling factor – to prevent clipping – and it differs from synth to synth and should be independently calculated by a different function for more complex sources. This value is then multiplied by the magnitude squared – this was done such that the volume maps in a parabolic fashion to the image feature. This was found to be a good way of separating the sources as linear mapping did not separate close sources' volume enough.

6.6.5 Alert Sound

The function of the alert sound is to make the user aware that they have found something. Therefore, it should be attention-grabbing and easy to differentiate from the other sounds. As mentioned in Section 2.4.1, we generally associate high frequency content with danger, therefore alerting sounds are generally designed to be high-pitched. It is therefore suggested that a high-pitched sound should be used to alert the user. An oscillator was used as it is a simple, recognizable, sound that can be easily distinguished from the other sounds used in the auditory display.

To make the alert sound even more distinguishable from the other sounds in

6.6 Auditory Mappings

the auditory display, it was decided that it should have amplitude modulation – much like a siren. It was therefore suggested that the sinusoidal oscillator should have a carrier signal to alter its amplitude. The carrier signal was changed in proportion to the alert signal, so that if a higher-pitched alert sound is used, its amplitude modulation will increase. This was implemented in Csound as follows:

```
1 aoscil oscil 0.3, kAlert , 1
2 kVibFreq = kAlert/40
3 kvib vibr 0.8, kVibFreq, 2
4 aalertOut = aoscil * kvib
```

The first line creates an oscillator for the alert sound. Its amplitude is set at a regular level such that it can be altered for different purposes, allowing the vibrato sound to be added independently. The vibrato rate was calculated by dividing the alert sound (`kAlert`), taken from iOS, by a scaling factor – in this case 40 (this could be altered dependent on the application). The ‘vibr’ opcode was then used to provide a carrier frequency, such that it could control the amplitude of the oscillator in the final line, producing the output ‘aalertOut’.

Video Example 6.7 shows the alert sound mapping, along with the other mappings developed in the previous sections.

6.6.6 ‘Boing’ sound

As discussed in Section 6.2 , the sound to represent a user scrolling beyond the bounds of the display should be proportional to how far they have scrolled past it. It was decided that an oscillator should be used to portray how far the user has scrolled off the screen – its frequency mapping to the proportionate size. This mapping was created using the following code in Csound:

```
1 aboing oscil 0.1, kboing, 3
```


6.6 Auditory Mappings

This line uses the oscillator opcode ‘`oscil`’ – its amplitude value set to a fixed value (in this case 0.1), and its frequency controlled by the offset value (discussed in Section 6.2) sent from iOS. This oscillator then reads from an f-table – in this case f-table 3. This f-table contains the fundamental frequency, and some harmonics to emphasize some higher frequency content. This sound was then panned binaurally using the method outlined in Section 6.6.2 – and its azimuth angle sent from iOS dependent on what side of the screen the user exceeds. The following code shows how this was implemented in Csound:

```
1 aboingOutL, aboingOutR hrtfmove2 aboing, kboingAz, 0, "hrtf-44100-  
left.dat", "hrtf-44100-right.dat"
```

An example of a simple searching task on a screen 9 times the size of the iPad is shown in *Video Example 6.8*. The ‘boing’ synth is demonstrated at the end of the video.

6.6.7 Dealing with Interaction

As in Section 6.3, the system was designed to react to the user’s touch, and this must now be reflected in the audio. When, the user’s touch is removed in iOS, the variables that control the repeating sound are set to zero. This functionality was implemented in Csound thus:

```
1 if(kSpeed > 0) then  
2     kNoise = 0.2  
3     else  
4     kNoise = 0  
5 endif
```

This takes the form of an ‘if’ statement – if the variable ‘`kSpeed`’ (sent from iOS) is greater than zero, the volume of the pulse train synth is set to 0.2. Otherwise, it is set to zero. This means that when a user removes their touches, or leaves the proximity zones, the pulse train is silenced.

7 Methodology and Testing Procedure

This section describes the methods used to test the techniques developed in the Design and Implementation section, along with the primary hypothesis (outlined in Section 4). The test methodology is discussed – outlining what is being tested, and why. Then the design for the test is defined – describing the plan for an experiment with participants to test the hypothesis, and the auditory display techniques developed. The actual testing procedure is then discussed, describing the technical setup, and the testing conditions.

7.1 Test Methodology

To evaluate the effectiveness of the techniques developed, and the primary hypothesis, there should be two groups; a ‘treatment’ group (Group A) and a ‘control’ group (Group B). Group B should be provided with auditory display parameters to find a specific feature in the image. Group A should have additional parameters, with the aim of testing the primary hypothesis, and other questions that have arisen during the thesis. For example, Group A were given binaural audio as an additional parameter. Then, to judge the parameter’s effectiveness, we compare the variation in the results when using the extra sound parameter in assisting a participant in achieving their goal against the control group who do not have that parameter.

The number of participants should be dictated by the amount of time allocated for the testing procedure – in essence, there should be as many as possible, as this helps when it comes to statistically validating the data. Additionally, each group should have the same number of participants – and a method to randomly allocate the participants into a group must be devised in accordance with this.

7.2 Test Design

This section describes the design of each test and how it relates to the primary hypothesis, and some additional questions that have risen during the work in this thesis. Each test is discussed and the auditory mappings described in simple tables. For each test, it is noted whether or not the participant is visually restricted, i.e., prevented from seeing the iPad screen.

7.2.1 Test 1: regular screen size searching

Test 1 is a series of sub-tests that examine the primary hypothesis – whether spatial auditory display can be used to increase the subject’s ability to locate specific graphical features on the screen of a tablet computer. To do this, there should be a control group, and a treatment group. Each group should then undergo the same basic test; with the parameters we wish to test missing from the control group.

7.2.1.1 Test 1.1: finding a black dot [with/without binaural]

A simple image feature is used in this test – a small black dot on a white screen (shown in the folder ‘*Test Images*’, along with all other images used in this test, on the CD associated with this project). The only difference between the two groups of participants should be that only the treatment group has binaural auditory feedback – allowing us to test the primary hypothesis. Participants had the parameters outlined in Table 7.1.

<i>Visually Restricted</i>	Binaural Audio	Pulse Train	Alert Sound	Mono Audio
Group A	Yes	Yes	Yes	No
Group B	No	Yes	Yes	Yes

Table 7.1: Parameters used for each group in Test 1.1

The user is then tasked with finding this black dot, and their attempt is timed. The time difference between the mean of the two groups will indicate the effect of the binaural audio. This allows us to directly test the primary hypothesis in a simple, yet unambiguous manner.

7.2.1.2 Test 1.2: three coloured dots [Group B not told colour mappings]

As discussed in Section 5, a preliminary test concluded that there is evidence of a preference for how we associate sound, specifically frequency content, to colours. Test 1.2 was designed with the aim of finding out whether participants have a particular preference in the context of interactive sonification. Participants have to find, and describe, three coloured dots (red, green, and blue) that use the same mappings as described in Section 6.6.3. However, to establish whether participants have an inbuilt preference for specific colour-to-sound mappings, Group B should not be told the parameters. Based on the experiment in Section 5, the chosen colour mappings are shown in Table 7.2.

Colour	Sound
Red	High frequency (<i>Audio Example 6.3</i>)
Green	Medium frequency (<i>Audio Example 7.1</i>)
Blue	Low frequency (<i>Audio Example 7.2</i>)

Table 7.2: Colour to frequency mapping parameters for Test 1.2

This not only tests the users' colour-to-sound association, but more importantly the system's ability to portray a more complex graphical field, this time involving coloured objects. To ensure that the participants knew what they were looking for, both groups were played the sounds over headphones before beginning the test. In accordance with the aforementioned test procedure, at this point Group B was not told which colour was represented by

which sound, and Group A were. The parameters used are outlined in Table 7.3.

<i>Visually Restricted</i>	Binaural Audio	Pulse Train	Alert Sound	Mono Audio	Told Parameters?
Group A	Yes	Yes	Yes	Yes	Yes
Group B	Yes	Yes	Yes	Yes	No

Table 7.3: Parameters used for each group in Test 1.2

As Table 7.3 shows, the mappings between the two groups are the same. The only difference is the knowledge of the participants – Group A knows the colour mappings and Group B does not. The additional ‘volume mapping’ parameter was added because it was decided in the Design and Implementation Section that when representing multiple sources they often clashed, and were hard to differentiate. The volume parameter mapping reduces interference between different sound sources – the further away the user is touching from the dot, the quieter it is. This allows for the local dots to be more prominent, and for dots that are further away not to interfere, but still be heard.

7.2.1.3 Test 1.3: Picture Identification [both groups with same mappings]

The next logical decision was to determine whether a user could, with their ears alone, determine if they were touching a specific picture. To this end, four simple ‘minimalist’ (Figure 7.1) pictures were created and then participants chose the one they believed they were interacting with. The same mappings were used as Test 1.2 (as outlined in Table 7.4), but this time Group B were told the colour-sound mapping as well as group A.

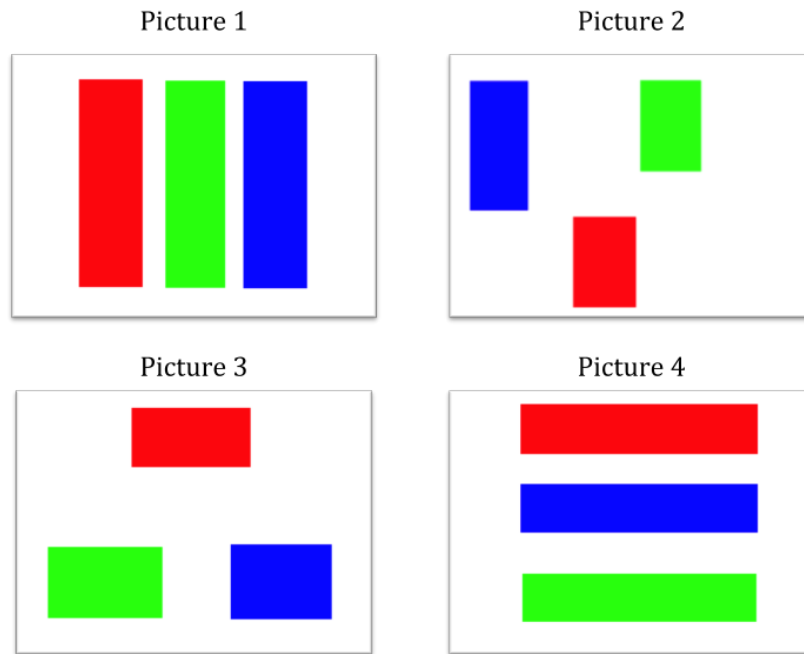


Figure 7.1: The four pictures the user had to choose between

<i>Visually Restricted</i>	Binaural Audio	Pulse Train	Alert Sound	Mono Audio
Group A	Yes	Yes	No	Yes
Group B	Yes	Yes	No	Yes

Table 7.4: Parameters used for each group in Test 1.3

As shown in the Table 7.4, the mappings are the same as in Test 1.2, but without the alert mapping. There is no comparison between the two groups in this test – it is simply about determining if participants can accurately detect a whole image from sound.

7.2.2 Test 2: searching extended displays

Test 2 is a series of tests that challenge the system’s ability to present graphical features on an extended display by means of the techniques examined

in Test 1. It features similar challenges to Test 1, however, the participants are asked to locate features on images that are much larger than the display, using variations of the techniques in Test 1, therefore often making it more challenging. The techniques are again tested using the same two groups – A and B.

7.2.2.1 Test 2.1: black dot in a large image [with/without binaural]

Test 2.1’s subjects were simply asked to find a black dot on the screen. Much like in Test 1, this gives a good impression of how well the techniques developed scale up to larger displays, and how big an impact the binaural audio makes. When deciding on a reasonable sized image for the user to be asked to navigate, it was concluded that nine times the size of the iPad screen (each dimension multiplied by three) would be a good choice – as from some initial user tests it was found that four was too simple, and 16 was a little excessive for the first test. The mappings of the sounds are outlined in Table 7.5.

<i>Visually Restricted</i>	Binaural Audio	Pulse Train	Alert Sound	‘Boing’ Sound	Mono Audio
Group A	Yes	Yes	Yes	Yes	No
Group B	No	Yes	Yes	Yes	Yes

Table 7.5: Parameters used for each group in Test 2.1

For the sound mappings, the same parameters as Test 1 were used, with the exception of an additional ‘boing synth’, as discussed in Section 6.6.6, to tell the user when they had exceeded the bounds of the scrollable view.

7.2.2.2 Test 2.2: three coloured dots in a large image [with/without binaural]

Test 2.2, much like Test 1.2, involves finding three dots of different colours in a relatively complex auditory field, however, Test 2.2 involves finding these dots on a scrollable image nine times the size of the iPad screen. As with Test 2.1 and Test 1.1, this examines the effectiveness of using binaural audio to locate the graphical features, however, it was expected to be seriously challenging to the participants – especially those in Group B (the group without the binaural audio) because of the larger virtual image. The auditory mappings used are outlined in Table 7.6.

<i>Visually Restricted</i>	Binaural Audio	Pulse Train	Alert Sound	‘Boing’ Sound	Mono Audio
Group A	Yes	Yes	Yes	No	Yes
Group B	No	Yes	Yes	Yes	Yes

Table 7.6: Parameters used for each group in Test 2.2

7.2.2.3 Test 2.3, Test 2.4, and Test 2.5: black dot in a large image [no visual restriction]

Tests 2.3, 2.4 and 2.5 compare how well users can detect simple graphical features with visual and auditory cues, compared to visual alone. Therefore, for the purpose of this experiment, one group (Group A) could see the iPad, and use the auditory display techniques developed to locate specific features, and the other group (Group B) was only given visual cues. To test the effect of increasing the display size, the participants have to locate features on scrollable images of progressively increasing size – the details of which are outlined in Table 7.7. This is represented graphically in Figure 7.2.

7.3 Experimental Procedure

Test	Image size
2.3	9 x iPad screen (3072x2304 pixels)
2.4	16 x iPad screen (4096x3072 pixels)
2.5	25 x iPad screen (5120x3840 pixels)

Table 7.7: Dimensions of tests 2.3, 2.4, and 2.5

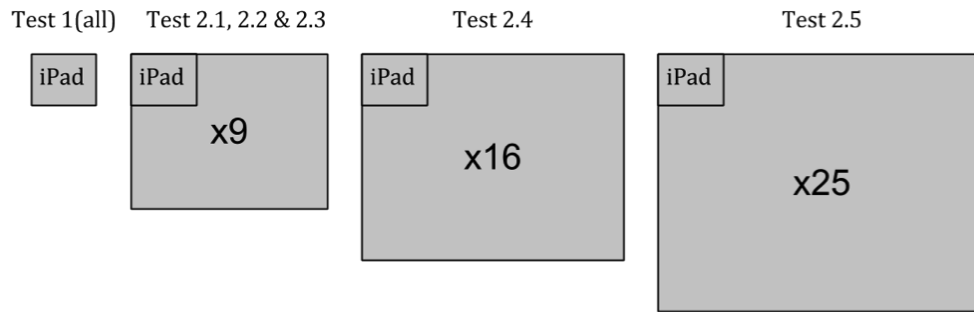


Figure 7.2: Progressively larger displays for extended desktop tests

As the task is simply to find a black dot on the screen, the auditory parameters are similar to Test 2.1. However, Group B was deprived of any auditory feedback – they were asked to take off the headphones. The parameters for tests 2.3, 2.4, and 2.5 are outlined in Table 7.8.

<i>No Visual Restriction</i>	Binaural Audio	PulseTrain	Alert Sound	Boing Sound
Group A	Yes	Yes	Yes	Yes
Group B	No	No	No	No

Table 7.8: Parameters used for each group in tests 2.3, 2.4, and 2.5

7.3 Experimental Procedure

This experimental procedure was designed to be repeatable, fair, and for the participants to enjoy the experiment. The experiments were designed to be

a series of goal-oriented tasks that became increasingly challenging, but still not so hard that the user could not do them. The following sections describe the test procedure chronologically, such that the tests may be produced, and reproduced. Additionally, decisions behind the test choices and creation of test materials are discussed in depth. Now that each test has been discussed individually, the experiment as a whole is described. An overall aim is to ensure that each participant has as close to ‘the same experience’ as possible. This will help to ensure non-biased results, and a consistent and complete set of data for each participant.

7.3.1 Allocation of Groups

In the beginning of the experiment the user is asked to take a seat. At this point, to ensure the participants are randomly allocated into a group the subject is asked which group they want to be in – Group A, or Group B. This is a simple way of ensuring that the selection of participant’s group is random, since the participants know nothing about the details of the experiment. If the participant does know something about the experiments, to be sure of fairness, a coin can be flipped as an alternative. It must be noted that only the test coordinator knows the contents of the experiments, and the test coordinator should not undertake the experiment.

7.3.2 Demographics and Ability to Perceive Binaural

The user is then given the script from the group they have been allocated and they are asked to read through the introduction to the experiment. The script is written in simple English and explains any technical terms to ensure that no participant is at a disadvantage. It is also essential to give the participant a consent form outlining the terms of their participation in this experiment – making it clear what happens to their data, what is expected from them during the experiment, and informing them that their participation is entirely voluntary. After reading through the introduction to the

test, and completing the consent form, the participant is asked to fill out a demographics form. It is important that some parameters are known about the participants to look for potential anomalies. It is essential that users are allowed to elaborate on any of their answers to these questions, therefore some space is left in the questionnaire such that they can write about their experiences.

A test is carried out to assess how well users perceive binaural imaging. A binaural sound example (*Audio Example 2.7*) is played, and the participant asked where the man speaking is in the room. Additionally, they should be asked where they believe the rats are in the room. This will allow for any anomalies (where people simply do not perceive the spatial content of the binaural audio) to be ruled out in the post-test data analysis process.

7.3.3 Test Description and Practice Examples

For each test it is essential that participants know not only what they are doing, but also why. They should not feel as if there is information they are not being told. For each of the two main sections of the experiment (Test 1 and Test 2) there is a detailed description of what the following sub-tests will involve. This is done with minimum jargon and technical language. To accompany this, the test co-ordinator can answer questions as the tests unfold, provided that they do not give the participant any clues about the image they are visually restricted from. If questions of this nature are asked the test co-ordinator would simply remain silent, or say “Sorry, I cannot answer that question”.

Due to the complexity of some of the tests, and the fact that the user would not have experienced anything similar before, it was important that the participants were given a practical example before they undertook the tests. This allowed them to become accustomed to the various mappings in a more relaxed environment. Additionally, without visual restrictions they can see

how the auditory display parameters map directly to the image before them. It also allows familiarity with the gestures required, for example, to operate the scrolling views in Test 2 – something that is hard to conceptualize from text alone. The test co-ordinator ensures that the participant is familiar with all of the parameters prior to starting the test.

7.3.4 Undertaking and Recreating the Tests

Each test script aims to unambiguously state the test to the user and make it evident that they can ask questions throughout. It is important that the participant is not rushed through the test and that they understand what they need to do. To recreate the tests, the Xcode projects of the tests have been included in the disk with this thesis under ‘Code for Experiments’. These tests should be run on an iPad and adhere to the protocols described in the next section. Additionally, the following scripts, designed and described in this section, should be used to administer the experiments. Digital versions of these scripts can also be found on the disk under ‘Scripts for Experiments’.

<p>Group A – <i>Appendix D</i> Group B – <i>Appendix E</i> Test Coordinator’s Script – <i>Appendix F</i></p>
--

7.4 Technical Setup

For the tests to be undertaken and documented properly there are a series of technical challenges: how the participant is visually restricted; how the actual tests are documented and carried out; and how they are programmed. The actual auditory display methods are discussed in the Design and Implementation section (Section 6). This section focuses on overcoming the technical obstacles behind the test’s design.



(a) Box with cloth down

(b) Box with cloth up

Figure 7.3: The visual restriction device used during the tests

Visual Restriction – visual restriction by means of a blindfold can often be a uncomfortable experience for participants in an experiment, and this would be a detriment not only to the participants, but also to the experimental procedure. It is evident that alternatives such as Fernström’s technique of using a box would alleviate the issue [Fernström et al., 2004]. Therefore a similar, robust method, was developed to allow the user to see freely, except for the area inside a box where they interact with the device.

To implement this, a simple cardboard box with a hole cut in either end was used – a hole for the participant’s hands, and another so that the interaction may be recorded on camera. Then the iPad is placed inside, and holes cut in the side so an audio feed can be taken from the iPad. However, it was evident on initial inspection that the participants would be able to see parts of the iPad. So the box was covered with cloth on one side so that the user could not see in, and not covered on the other side, such that the test could be recorded. The final version of the visual restriction device is shown in Figure 7.3a and Figure 7.3b.

Documenting the tests – A fully numerical evaluation of the tests would involve writing code to describe the user’s interactions with the device, then

printing it to terminal, or storing it in memory. For example, “The user is 30 degrees off axis and 40 pixels away from the target”. This was initially done, but was found not to give a good impression of the user’s performance – it did not capture the more subtle aspects of the users’ performances. An alternative numerical method of determining how well a user is finding features would simply be time how long they take to complete the tasks. This is good for comparing two groups, however, it does not give a good impression of why a user took a certain amount of time, or what methods they used to find the dot.

On the other hand, a more descriptive method could be used to analyse the data. Simply explaining what is happening, and presenting it by means of video can give us a good impression of the user’s performance. The downside of this is the fact that it is hard to quantify differences between tests. Eventually a compromise was chosen that allows for the descriptive method, alongside a numerical method (timing how long it takes to complete a given task). To implement this, the following specifications were written such that a design could be realized:

- The system must allow for the recording of the participant’s hands, and their interaction with the device at all times.
- The system must allow for the audio feed to be recorded, as well as the user to hear it simultaneously.
- The system must incorporate the previously discussed visual restriction device.

From these specifications, the schematic shown in Figure 7.4 was devised.

This schematic was then realized using a headphone amplifier to route the signal to the recording device (a laptop with an audio interface), and the

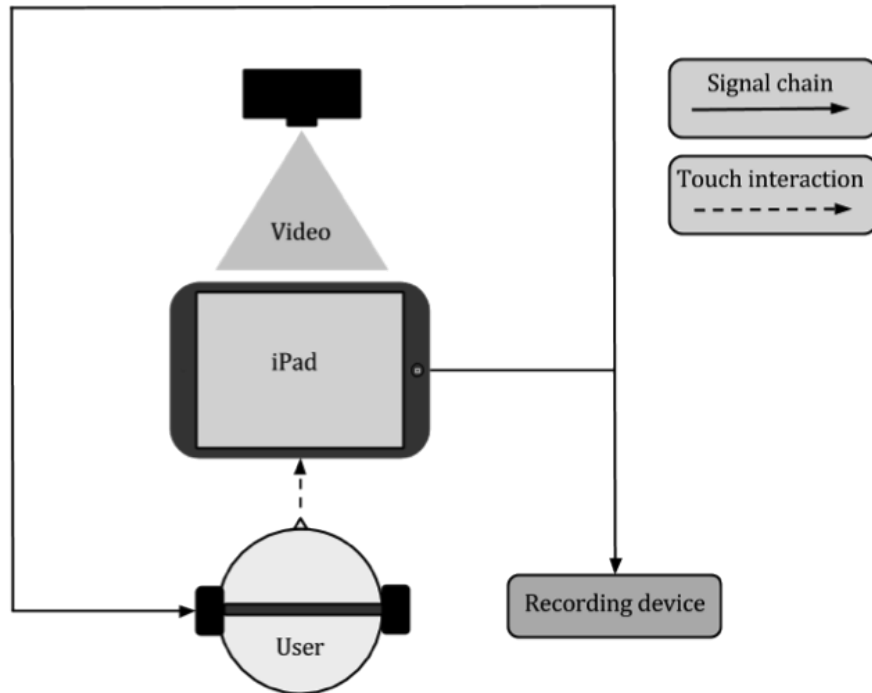


Figure 7.4: Schematic to undertake and record testing procedure

participant's headphones (a high quality set of studio headphones). This then meant that the audio feed could be recorded on the laptop using the freeware audio editor 'Audacity', linked below:

<http://audacity.sourceforge.net/>

The camera was positioned such that the user's hands and the iPad could be filmed. The lights in the room were adjusted to ensure the best recording quality throughout the tests. For Tests 2.3, 2.4 and 2.5 it was decided that the lights should be dimmed further such that the participant was not reflected in the iPad screen while undertaking the test. Additionally the screen of the iPad was dimmed to ensure that the participant could not see through the cloth of the visual restriction device. The technical setup is depicted

7.5 Data to be Gathered

in Figure 7.5. The seat at 12 o'clock in the photo is where the participant sat, and seat at three o'clock is where the test coordinator sat. The Test Coordinator had a copy of the participant's hand-out, as well as a copy of the Test Coordinator's script.



Figure 7.5: Picture of test setup

7.5 Data to be Gathered

Certain information is required to determine any patterns in the data. This section describes the additional data requirements for each test, and outlines their significance, and what possible conclusions can be drawn from them. This is done such that the results can later be evaluated, and conclusions can be made in light of the experimental hypothesis. The data to be gathered from each test is described in the following subsections.

7.5.1 Tests 1.1, 2.1, 2.3, 2.4, and 2.5

Success rate of finding the black dot – it will be evident from the video footage if the participant found the dot or not. The success rate of each group can then be compared.

Time it takes to find the black dot – the time the user takes to find the dot will be a good indicator of how effective the binaural audio is at helping them find the dots, when compared to the time of the group without the binaural audio. It will be possible, by looking at values such as the mean time in a group, to compare the effect of providing the participants with binaural audio.

7.5.2 Tests 1.2 and 2.2

Number of coloured dots successfully identified – the number of dots whose colour is correctly identified by Group B, compared to Group A, should signify if they had any preconceptions about what colour they associate with what frequency in Test 2.1. In Test 2.2, it is a good way of judging the overall success of the mapping on an extended display when the both groups are informed about the colour mappings.

Time taken to identify all dots – the time each group takes to identify all dots should give an impression of the effectiveness of the system. The difference between Group A and Group B should signify the extra time Group B have to try and figure out the colour-sound mappings.

7.5.3 Test 1.3

Amount of pictures guessed correctly – the amount of times the participants guess the pictures correctly is indicative of how effective the mappings are at

7.5 *Data to be Gathered*

representing simple images.

Time taken – the average time taken by the group is of interest –it gives us an impression of how quickly the participants are able to make a decision when exploring the image.

8 Results

This section presents the results from the experiments described in the previous section. The participant's demographics are described, and the test results are presented. The noteworthy results from each test are described and discussed, then these results are then considered as a whole in a review of the key patterns found in the chapter.

8.1 Participant Demographics

18 participants took the experiment – nine in Group A, and nine in Group B. The participants had an average age of 25.8 (standard deviation = 4.35) and consisted of British, Chinese, Dutch, Greek, American, Belgian, and Russian nationals. Of the 18 participants the majority were male (12), and six were female. Most of the participants recruited for the experiment were from the Department of Electronics (13), predominantly the Audio Lab, and the remaining five were from the Department of Computer Science. Most of the participants were musicians (12 out of 18), due to the fact that a large number of participants were from the Audio Lab.

Participants were asked questions related to the perception of sound and colour, binaural audio, and the use of tablet devices. Three out of the 18 participants claimed to have some form of sound-to-colour synesthesia – relating specific pitches/timbres to colours. Of the 18 participants, only one had not experienced binaural audio before. When played a short binaural sample and asked to identify where they believed the source of the sound to be emanating from, 15 participants said that they knew where the sound was coming from at all times, and the other three said they knew where it was most of the time – it was clear that nobody in the group was unable to perceive the spatial information from the sound. With regards to the participants' technology backgrounds – 16 people owned some form of touch-screen

device, and the other two had some form of experience with them.

8.2 Test Results

The results were gathered from the observation of the videos – each participant was timed undertaking each task, and the time was rounded to the nearest second. It is possible to view the videos of each participant in whole at the following link:

Playlist of all 18 participant's videos:

<http://www.youtube.com/watch?v=zD9XiN6i3lY&feature=share&list=PLjVqjt929nNSoOZUtMnwrDlEtc7nCRsCW>

Deductions were made from the user's time if the participant stopped searching for a short period for reasons out of their control, for example – a technical fault. The times were then entered into Microsoft Excel for analysis. Additionally, the results were statistically validated using Chi-Squared tests and the t-test for two independent samples. The Chi-Squared test (χ^2) is a test often used for non-numerical, categorical data. It allows us to compare observed frequencies with theoretically predicted frequencies. To statistically validate the results, Equation 8.1 was used to ascertain the Chi-Squared statistic of a data set. This was then compared to the Chi-Squared distribution table to judge the result's significance level [Howell, 2011, pg. 502] where 'E' is the expected result – the result that would be attained by chance, and 'O' is the observed result.

$$\chi^2 = \sum \frac{(O - E)^2}{E} \quad (8.1)$$

8.2 Test Results

The t-test for two independent samples was used to determine if there was a significant difference between the two groups – i.e. a difference that could not have happened by chance. The ‘t’ statistic was attained using Equation 8.2. It was then possible to observe the significance level by comparing this value to the required significance level in a ‘t distribution table’, which can be found in Howell’s *Fundamental Statistics for the Behavioural Sciences* [Howell, 2011, pg. 596], or most books/websites on descriptive statistics.

$$t = \frac{X_a - X_b}{\sqrt{S_p^2 \left(\frac{1}{n_a} + \frac{1}{n_b} \right)}} \quad (8.2)$$

Where $S_p^2 = \frac{(n_a-1)S_a^2 + (n_b-1)S_b^2}{n_a+n_b-2}$, X_a is the observed mean of the control group, X_b is the mean of the treatment group, n_b is the number of participants in the control group, n_a is the number of participants in the treatment group, and s_a and s_b are the variances of the groups, respectively. The results were then verified using the statistical programming environment ‘R’. It allows for statistical tests to be run, and for an exact ‘p value’ to be attained. More information about the R programming language can be found here:

<http://www.r-project.org/>

8.2.1 Test 1.1: finding a black dot [with/without binaural]

In this test every participant managed to successfully complete the primary objective – finding the black dot. In accordance with the primary hypothesis, the alternative hypothesis was that the group with the binaural audio (Group A) was expected to outperform the group without the binaural audio (Group B). The null hypothesis was that Group A would perform the same as, or worse than, Group B.

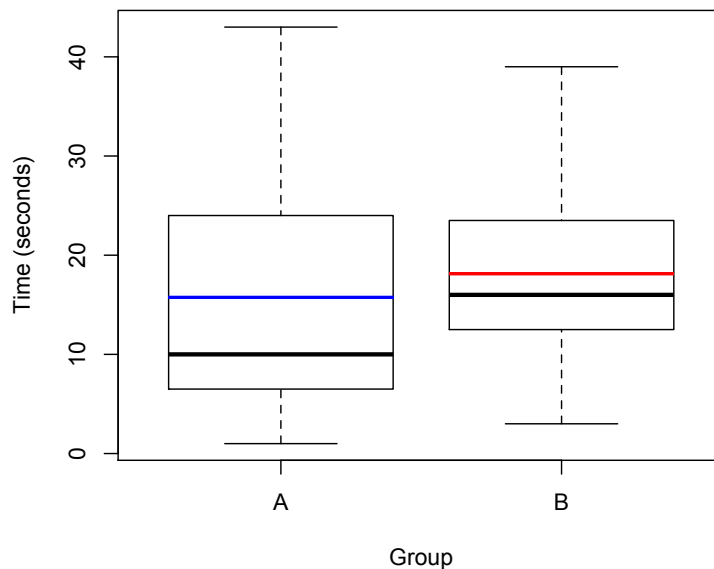


Figure 8.1: Test 1.1 – box plot for groups A and B when searching for a black dot

Group A, on average, found the dot in 15.62 seconds (blue line in Figure 8.1), with a Standard Deviation of 14.66. Group B completed the test with a larger mean time of 18.38 (red line in Figure 8.1), and with a Standard Deviation of 10.68. A t-test was run for the two independent samples, attaining a p-value of 0.6745 – failing to reject the null hypothesis at all reasonable confidence levels – suggesting that this could have happened by chance. These results are summarised in Table 8.1

Group	Mean	Standard deviation
A (sound)	15.62	14.66
B (no sound)	18.38	10.68

Table 8.1: Results for Test 1.1

Discussion

From this test it was evident that the binaural audio, on average, sped up the participants in finding the dot – the mean time of Group A was 17.7% faster than the mean time of Group B. However, it is important to note that without a strong level of significance, this cannot be said for sure. The low number of participants could be responsible for the low levels in significance, as well as the great variation in techniques used to locate the shape – resulting in relatively high Standard Deviations.

Two participants were not fully ready when they undertook the tests – these were classed as statistical outliers. However as they progressed through the rest of the tests, their results were much closer to the average – making it evident that the data was an anomaly, and that they were not generally performing poorly. These anomalous results underline the need for proper practice sessions before the test. Each anomalous participant undertook the practice session, but it was noted that neither of them tried the example while not using visual cues. This lack of training is likely to be the cause for their irregularly slow location of the image feature.

8.2.2 Test 1.2: three coloured dots [Group B not told colour mappings]

In this test Group A were told the colour-sound mapping parameters. With this knowledge they were able to get on average 2.44 out of three dots correct. Group B, by means of instinctive guessing were able to get 2.33 out of three dots correct. The null hypothesis was that Group B would guess, on average, one out of three dots. This is what they would guess by chance. The alternative hypothesis was that Group B would guess more than one out of three correct, because they had preconceptions about how the three different sounds relate to the three different colours. A Chi-Squared test was run to determine the chances of group B guessing 2.34 out of three dots by chance. A confidence interval of $p = 0.09687$ was attained, therefore reject-

8.2 Test Results

ing the null hypothesis at $\alpha = 0.1$. Of those who identified all three dots, Group A took an average time of 75.83 seconds and Group B took an average time of 122.67, with standard deviations of 46.27 and 92.1 respectively. The key results are outlined in Table 8.2, and Figure 8.2 is a box plot of the participant's times.

Group	Amount correct	Time taken	Standard deviation
A	2.44	75.83	46.27
B	2.33	122.67	92.1

Table 8.2: Results for Test 1.2

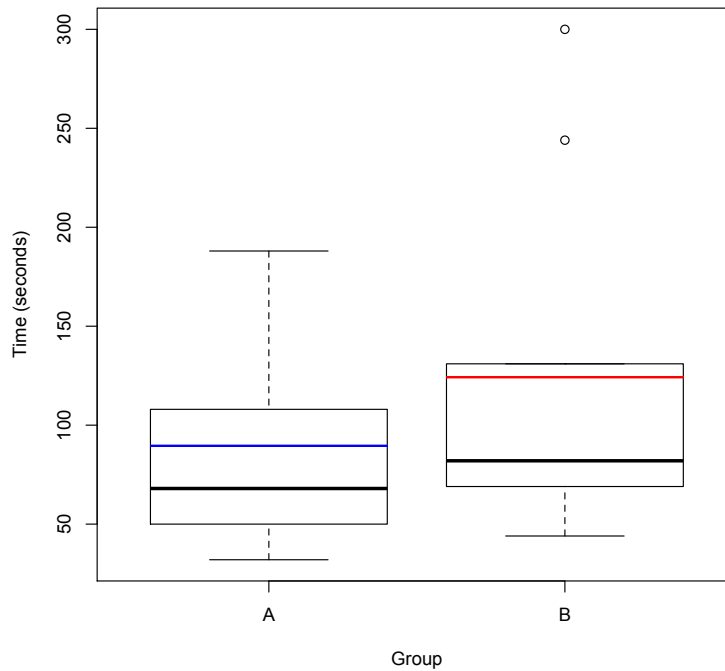


Figure 8.2: Test 1.2 – box plot for groups A and B when searching for three coloured dots

Discussion

The fact that the participants were able to guess the colours of the dots better than chance suggests that they believed that red sounds are best represented by high-pitch sounds, green by medium-pitch sounds, and blue by low-pitch sounds – supporting the work described in Section 5. The significance of this (rejecting the null hypothesis at $\alpha = 0.1$) implies a relatively low possibility of chance, however, it is not a strong enough confidence level to say this definitively.

Group A did not get a perfect score despite being told the mappings beforehand. This is because some participants either did not find all of the dots, or they got confused and forgot the mappings. It was clear that Group B were slower because they took a lot longer to think about the mappings they associated with each colour – often going back and forth between colours once they had found them. This extra search time also explains the larger standard deviation in Group B as some participants took the extra time to think about the sound mappings, whereas others guessed quicker. Group A generally tended to name them as they went along and made the decisions with regards to what colour they were touching as they already knew the mappings.

8.2.3 Test 1.3: picture identification [both groups with same mappings]

In this test the participants were tasked with guessing which one of the pictures they were touching based on the auditory feedback alone. If the auditory feedback did not help the participants find a picture they would get it right, on average, one quarter of the time (as there were four pictures). Therefore, the null hypothesis was that each picture would be chosen 4.5 times by the 18 participants. The alternative hypothesis was that, with the assistance from the auditory feedback, they would guess Picture Three, the correct answer, more than 4.5 times. The pictures they chose in the experi-

8.2 Test Results

ment are described in 8.3 where ‘Expected’ implies the expected amount of times chosen (what would be expected by chance), and ‘Observed’ denotes the observed number of times chosen.

Picture	1	2	3	4
Observed	0	1	17	0
Expected	4.5	4.5	4.5	4.5

Table 8.3: Number of times each picture was chosen by the participants

It is evident that picture three, the correct answer, was selected considerably more than the other pictures by the participants. A Chi-Squared test was run to test if this could happen by chance. A p value of $4.562e^{-10}$ was attained – rejecting the null hypothesis at $\alpha = 0.05$. With regards to time taken, the participants took between 11 and 51 seconds to identify the picture, with an average time of 27.34 seconds and a standard deviation of 12.85. The spread of times in this test is depicted in Figure 8.3.

Discussion

It was clear from this test that the participants were able to detect a picture from its sound representation without guessing – such a low p-value suggests a very low chance of the subjects attaining this score by means of guessing alone.

The large variation in times can be attributed to the searching techniques of the individuals. When discussing the test with the participants after the experiment it became evident that some participants looked at the pictures beforehand and made a mental model of what they believed they were looking for. Meanwhile, others investigated the screen, and then by process of elimination chose their answer. The subjects with the faster times generally had more logical searching patterns, and adhered to the first approach previously described. The video examples below shows the participant who got the fastest times in this test.

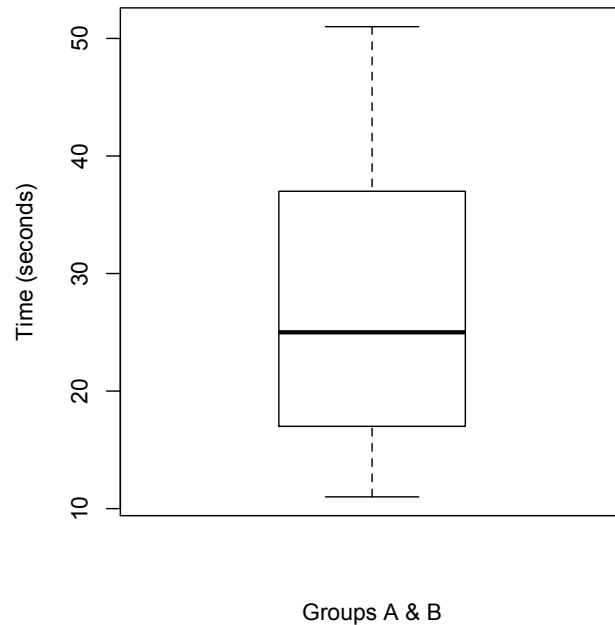


Figure 8.3: Test 1.3 – boxplot of time taken to complete inspection of picture

Participant with the fastest time on Test 1.3:

<http://youtu.be/7rpAev5eYeM?t=1m14s>

8.2.4 Test 2.1: black dot in a large image [with/without binaural]

The participants were largely successful in finding the black dot in this test – 17 out of 18 were able to find it. The null hypothesis was that the additional binaural parameter that A was given would not speed up their performance, and the alternative hypothesis was that Group A would take less time to

complete the task than Group B.

In the experiment it took Group A on average 108.25 seconds (blue line in Figure 8.4), with a standard deviation of 68.16. Group B took longer – 131.50 seconds (red line in Figure 8.4), with a Standard Deviation of 71.66. A one tailed t-test for two independent samples was used, attaining a p value of 0.5036. There was a large range of times in each group – ranging from 31, up to 231 seconds in Group A, and from 48 up to 215 seconds in Group B. It is clear from looking at the boxplot (Figure 8.4) that A had a large distribution of results, but the interquartile range for Group B was larger. The key results from this test are summarised in Table 8.5.

Group	Time (seconds)	Standard deviation
A (sound)	108.25	68.16
B (no sound)	131.50	71.66

Table 8.4: Results for Test 2.1

Discussion

It is clear from the times that Group A performed better than Group B – they located the dot 21.5% faster. However, due to the relatively low level of significance it is not possible to say this could not have happened by chance. From observation of the standard deviations, it is clear that Group B had slightly more disperse times than Group A. Upon watching the videos, it becomes evident that, in general, Group B were far more sporadic in their search. Group A tended to begin by touching the screen, and then moving in the direction they believed the source to be. It normally took B longer to get into the same area as the dot, and longer to locate it locally.

The large interquartile range suggests that even though Group B had a diverse set of times, there were not a lot of anomalous pieces of data – but a good representation of the group’s performance. It is clear that with Group A that this was not the case – the interquartile range appears small, but the

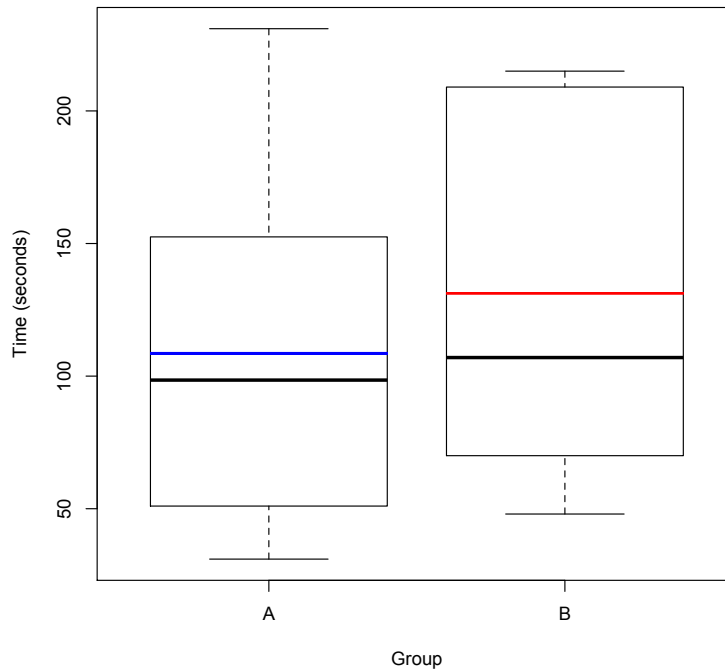


Figure 8.4: Test 2.1 – box plot for groups A and B when searching extended display

large whiskers suggest some participants scored very low, and others very high times – suggesting some participants fully understood the mappings and performed well, and others had a hard time with the mappings. It is evident from observation of the videos that the slowest times from each group were mostly people who got mixed up with the interaction system. Often they were trying to scroll when they had already scrolled up against a wall – it was evident that the ‘boing’ alert was not prominent enough to alert the user to this fact when making smaller movements (as shown below).

Video example of ‘boing’ sound being ineffective:

<http://youtu.be/uYvJJDKPztU?t=20m53s>

8.2.5 Test 2.2: three coloured dots in a large image [with/without binaural]

In this test, seven out of the nine participants in Group A found all three dots, and one participant located two dots. One participant, who was classed as an exception due to giving up after 163 seconds found no dots. The group, on average found 2.56 out of three dots. In Group B, four out of nine participants found all three dots, two found two dots, two found one dot, and one located no dots. The group, on average, found two out of three dots.

The fastest time to find all three dots successfully in Group A was 155 seconds, and the slowest time was 309 seconds. In Group B, the fastest time was 159 seconds, and the slowest was 691 seconds. The null hypothesis was that Group A, with the binaural audio, would perform the same as Group B. The alternative hypothesis was that Group A would be able to find the dots quicker without the binaural audio. The average time to finish for all members of Group A was 242.75 seconds (blue line in Figure 8.5), with a standard deviation of 65.71, and for Group B, a slower time of 391.34 seconds (red line in Figure 8.5) was observed, with a much larger standard deviation of 249.81. A one tailed t-test for two independent samples produced a p value of 0.1244, indicating a relatively high level of significance in the results.

Group	Time (seconds)	Standard deviation
A (sound)	242.75	65
B (no sound)	391.34	249.81

Table 8.5: Results for Test 2.2

The average time for those who found all three dots in Group A was 254.85

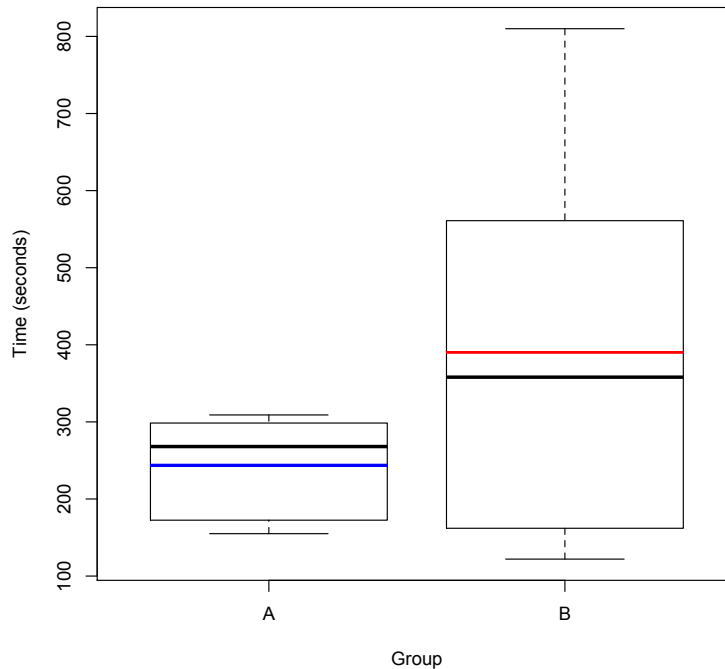


Figure 8.5: Test 2.2 – box plot for groups A and B when searching extended display with three colours

seconds, with a standard deviation of 60.58. In Group B this was higher at 312 seconds, with a standard deviation of 255.16 seconds. A one tailed t-test for two independent samples produced a p value of 0.5867 – a low level of significance.

Discussion

It is clear that, overall, Group B had a harder time locating and identifying the dots – they scored worse than Group A when correctly identifying the dots. This can be attributed to a number of factors – the extra concentration they put into locating the dot when close to it often led to some confusion – even if they knew they were on a dot, they were not able to spatially dif-

8.2 Test Results

ferentiate the sources in the manner Group A were able to. Additionally, a potential explanation is that Group B did not have the informed practice of interacting with a multi-colour auditory field that Group A had in Test 2.2 – this may have slightly altered the outcome of the test.

As with Test 1.1 and 2.1, Group A appears to have a faster overall finish time, with a more statistically significant result. Group A were able to finish the task, regardless of whether they correctly located all the dots, 61.21% faster than Group B, and were generally more successful in finding the dots. However, the difference between those who actually found all three dots is not as great – only 22.42%, with a lower level of significance due to the lower number of participants included when calculating this test statistic.

There was a very large standard deviation in Group B – 3.8 times that of Group A when considering all results, and 1.99 times that of Group A when considering only right answers for both groups. This is largely due to some participants taking very long to complete the test, often because they had decided to give up using the auditory display techniques. A good example of this would be the participant that simply chose to raster the screen with their finger until they heard the alert (as shown in the video example below). It was evident that some of the group without the binaural audio found it extremely challenging to complete the task using the audio.

Example of participant rastering the screen and not relying on the sound to locate the feature at all:

<http://youtu.be/JIfyP1041J4?t=15m>

8.2.6 Tests 2.3, 2.4 and 2.5: black dot in a large image [no visual restriction]

In these tests all participants found the dots. This was expected as they were searching with their eyes and eventually this was going to happen regardless. In Test 2.3, Group A (with the auditory feedback) found the dot in 10.55 seconds (blue line in Figure 8.6) with a standard deviation of 10.56. Group B (without the auditory feedback) took longer, with an average time of 14.33 seconds (red line in Figure 8.6) and standard deviation of 10.26. The fastest participant in Group A took six seconds, and the slowest took 24 seconds. In Group B the fastest participant also took six seconds, and the slowest took 36 seconds.

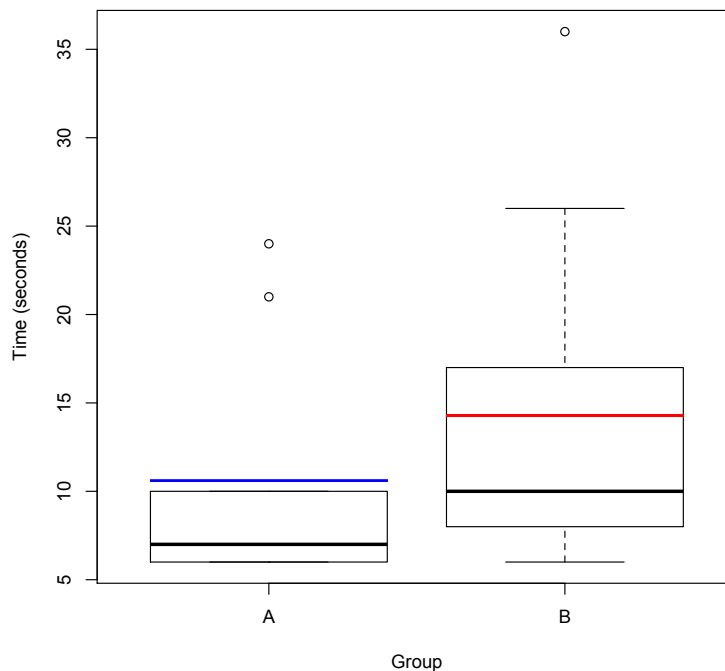


Figure 8.6: Test 2.3 – box plot of Group A (audio and visual) and Group B (visual alone) on a screen 9 times the iPad screen

8.2 Test Results

In Test 2.4, Group A (with the auditory feedback) found the dot, on average, in 12.44 seconds (blue line in Figure 8.7) with a standard deviation of 5.98. Group B (without the auditory feedback) took 15.22 seconds, on average (red line in Figure 8.7), with a Standard Deviation of 11.13. The fastest participant in Group A took three seconds, and the slowest, 22. In Group B the fastest participant found the dot in just two seconds, and the slowest took 39 seconds.

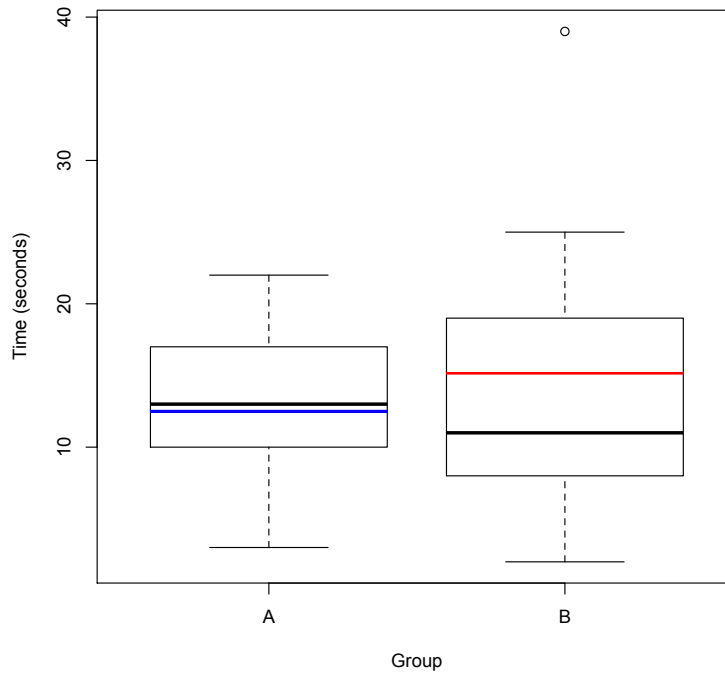


Figure 8.7: Test 2.4 – box plot of Group A (audio and visual) and Group B (visual alone) on a screen 16 times the iPad screen

In Test 2.5, Group A (with the auditory feedback) found the dot, on average, in 15.56 seconds (blue line in Figure 8.8), with a standard deviation of 5.6. Group B (without the auditory feedback) took under half this time – 7.22 seconds (red line in Figure 8.8), with a standard deviation of 3.99. The fastest participant in Group A took 9 seconds, and the slowest took 27 seconds. In

Group B, the fastest participant found the dot in just four seconds, and the slowest in 15 seconds.

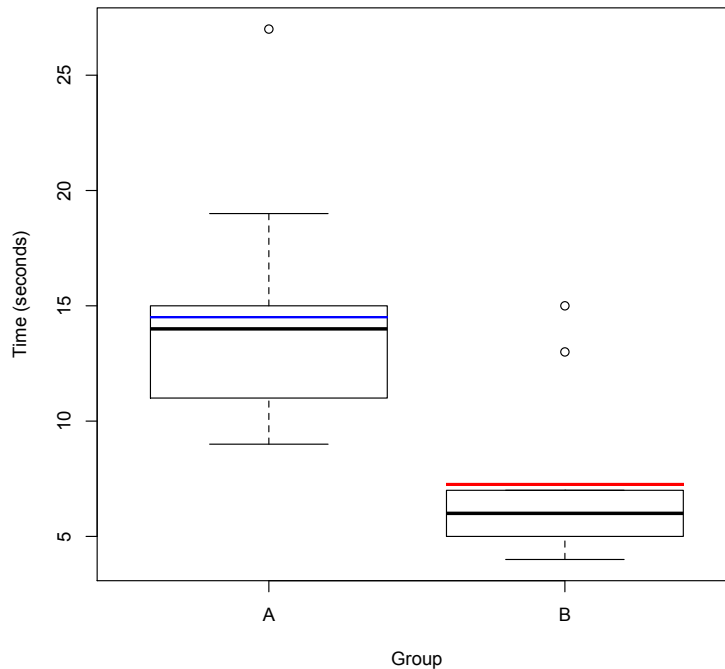


Figure 8.8: Test 2.5 – box plot of Group A (audio and visual) and Group B (visual alone) on a screen 25 times the iPad screen

Discussion

In Tests 2.3 and 2.4 those with auditory assistance appear to outperform those who use their eyes alone – the mean time taken to find the dot was lower than that of the group who just used their eyes alone. Additionally, the lower standard deviation suggests a more consistent performance. From observing the video recordings it is possible to say that Group A seemed to adhere to a highly logical searching pattern – first determining the direction they need to travel in by listening to the sound, and then making minor adjustments toward the feature as they edged closer. Group B, however, opted

for a more sporadic searching method – meaning that the times were highly dispersed, and on average less reliable. Often the participants in Group B would opt for a ‘rastering’ approach, where they would begin in a corner and raster around the image until they found the dot. This approach generally resulted in low times.

Test 2.5 appeared to exhibit anomalous results – the extreme inverse of what had been happening in the previous two tests occurred – Group B (without auditory feedback) outperformed Group A (with the auditory feedback) significantly. A potential reason for this is an experimental error – when creating the image for the application, an error was made in which the dot was mistakenly made larger than expected, and the starting point for the user was very close to the dot. This meant that Group B’s more sporadic searching approach was more effective as the dot was large, and therefore very easy to detect from visual inspection, and if they simply scrolled a little to the left they found it instantly. Hence, it is possible to conclude that the results from Test 2.5 may be erroneous, however further work should be done to assess this. The results for tests 2.3, 2.4 and 2.5 are summarised in Table 8.6.

Test	Group A (sound) time	Group B (no sound) time
2.3 (9x)	10.55 (SD = 10.56)	14.33 (SD = 10.26)
2.4 (16x)	12.44 (SD = 5.98)	15.22 (SD = 11.13)
2.5 (25x)	15.56 (SD = 5.6)	7.22 (SD = 3.99)

Table 8.6: Results for Test 2.3, 2.4 and 2.5

8.3 Results Conclusions

In this section the key findings from the results are discussed as a whole. The key themes throughout the results section are explored – the use of binaural

audio, the users' ability to match colours to sounds, and the effect of auditory feedback when used in tandem with visual cues.

8.3.1 Comparing Binaural and Monaural

The results indicate that the use of binaural audio aided the users in locating image features. Moreover, additional patterns emerged – as the tasks became more complicated, the percentage difference in time between the two groups increased, as shown in Figure 8.9.

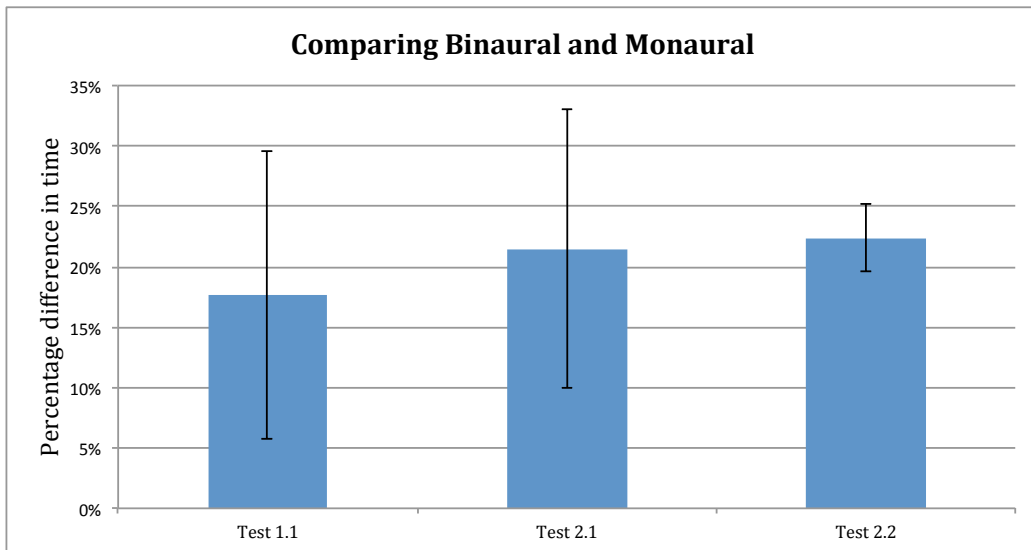


Figure 8.9: Percentage difference in binaural and monaural location times for different tests

In Test 1.1, Group A were able to locate the image feature 17.7% faster than Group B, and in Test 2.1, in which the screen is 9 times larger, there was a 21.5% difference between the groups – those with binaural audio appear to perform even better in comparison to the non-binaural group. The difference between the groups in Test 2.2 becomes larger – an increase of 0.92% between the two groups when an additional difficulty is added – implying that as the

tasks become more complex, the binaural audio helps more. However, as aforementioned the poor number of trials and participants shows resulted in relatively poor confidence, therefore it is not possible to say this definitively.

8.3.2 Comparing Audio and Visual to Visual Alone

Due to the fact that in Test 2.3 the searching pattern was far more logical (as shown in the videos and by the lower standard deviations), it was expected that there would be a positive correlation between the size of the screen and the effectiveness of auditory feedback.

It is evident that in Test 2.3 those with binaural audio were able to find the dots faster with the auditory assistance – Group A were 35.8% faster than Group B. However, in Test 2.4 this difference dropped to 22.23%. These are both noteworthy results, however, additional tests would need to be done to determine why the group without the audio in Test 2.4 seem to perform better, relative to Test 2.3.

9 Conclusions and Further Work

This section provides an overview of the knowledge gained throughout this thesis. The primary hypothesis is re-examined, and the major conclusions and noteworthy results are outlined. Potential further work is then discussed to suggest potential future avenues of exploration.

9.1 Review of Hypothesis

The original hypothesis of this project was as follows:

“It is possible to improve a user’s understanding of graphical data by using spatial audio to provide interactive auditory feedback.”

However, after reviewing the current technological possibilities in Section 2, and reviewing the work done in the area in Section 3, the hypothesis was refined in Section 4, to:

“It is possible to detect graphical features on a tablet display by means of real-time, interactive, binaurally spatialized audio.”

This hypothesis was then challenged through a series of user tests. The major conclusions from the tests are discussed in the next section.

9.2 Major Conclusions from Studies

In the Research Agenda (Section 4) several conclusions were drawn with regards to sonification strategies, interaction techniques, and test design methodologies. These conclusions helped to drive the development of the

auditory display techniques (Section 6) and their corresponding user tests (Section 7). From the user tests, results were attained and analysed (Section 8).

The results acquired suggest that a binaural auditory display can aid users to find image features faster, on a variety of display sizes, without visual cues. On average, those with the assistance of binaural audio were able to locate graphical features more accurately, and speedily, than those without the parameter. Additionally, the searching technique used by those with binaural audio relied on the auditory feedback more – whereas the group without the binaural parameter often relied on a more sporadic, non-auditory-based, searching approach.

As outlined in Section 8.3.1, the participants with the binaural audio performed increasingly better than those without, as the images became larger and more complex. This suggests that the effect of binaural auditory feedback becomes more evident when the tasks are more complicated. As stated in the conclusion of the results section it was clear that the difference between the two groups appeared to be most significant when the size of the iPad display was increased.

As described in Section 8.2.6 it was evident that the participants with both auditory display and visual cues were able to find graphical features faster than those with visual cues alone. Those with auditory cues tended to logically edge closer to the image feature, whereas those without the auditory feedback often adopted a more sporadic searching approach. This suggests that binaural auditory display, when used in tandem with visual cues, can help us find graphical features faster than visual inspection alone. Additionally, it was shown that the participants were able to identify simple pictures from binaural auditory feedback with an insignificantly small possibility of chance. This implies that using the techniques described in this thesis, users

were able to identify simple pictures with a high success rate.

In the work that followed on from the Preliminary Test it was found that the participants who were not told the colour-sound mappings were able to find, and guess correctly, the colour of three dots on a screen better than chance. This implies that we have preconceptions with regards to colour to sound mappings in the context of sonification. This experiment and the preliminary work (Section 5) suggests that we associate high-pitched sounds with high frequency colours in the spectrum (red, orange, etc.), and low-pitched sounds with the low frequency colours (purple, blue, etc.).

9.3 Recommendations for Further Work

This section discusses the potential avenues for further exploration by suggesting areas of work that extend or refine the work in this project.

9.3.1 Additional Testing of Visual vs. AudioVisual Tests

It was apparent that the tests comparing the participant's abilities to detect graphical features by means of auditory and visual cues was partially marred by the experimental error in the last test (see Section 8.2.6). It is therefore suggested that this test should be revisited to some extent. Moreover, additional tests can be devised to determine whether auditory feedback helps visual searching when a more complex visual field is used – for example, asking the user to locate three colours instead of a black dot on an extended display. It would be of interest to see if the complexity of the interface causes the difference between the groups to become larger, much like in the visually restricted tests. It must be noted that if the tests were run again the number of participants should be doubled. Two groups of nine did not allow for sufficient statistical backing in the results.

9.3.2 Investigate the Effect of Binaural Audio Using Other Techniques

The effect of using a binaural parameter was tested by using it with other techniques, then comparing it with another group who did not have binaural parameter, but still had the other techniques. If, for example, a pulse train was not used, but another mapping method, would the results be any different? This is something worth considering when making recommendations for further experimentation.

9.4 Potential Applications

The potential applications of the interactive sonification techniques developed in this project extend that of interacting with images. This section describes some viable applications of the techniques that could be implemented with minimal adaptation to the auditory mapping techniques themselves.

9.4.1 Adapting Computer Interfaces for the Visually Impaired or Visually Engaged

Interfaces such as Apple's 'VoiceOver' utility (more information at <http://www.apple.com/accessibility/osx/voiceover/>) could be improved using techniques similar to those documented in this thesis. Currently, when using VoiceOver, a blind user who wishes to interact with a Microsoft Word document on the desktop of their computer. If they wish to find an icon on the desktop they need to either use the 'tab' key to tab through every item on the desktop, or search with the mouse until the mouse intercepts the image by chance. The utility then announces in synthesized speech "You are currently on a Microsoft Word document".

The following hypothetical scenario provides an example of applying the techniques developed in this thesis to this problem. It is thought that by allowing the user to search themselves, as opposed to tabbing through every single item on the desktop, they would not only have more control, but in time they could potentially become a more effective computer user.

The user places their mouse on the desktop and presses a keyboard shortcut. This keyboard shortcut activates a mode where all files on the desktop are read in the form of spearcons. The spatial audio then allows the user to differentiate between the sources and move their mouse over to the desired file – the pulse train and volume mapping parameter aiding them in determining how far they are away, and the binaural parameter helping them decipher its direction. Once the user hovers their mouse over the file the utility announces “You are currently on a Microsoft Word document”.

Techniques similar to this could also be used to assist those with their eyes elsewhere on a specific task – for example a runner who wishes to access a specific track on their music player while out jogging. Pseudo-haptic interfaces, as discussed in [Fernström et al., 2004], could be enhanced with binaural parameter mapping such that users can locate specific features better, or perhaps get a better gist of the interface from sound alone.

9.4.2 Allowing Users to Search Vast Images Such as Cervical Cancer Data

As discussed in Section 3.2 it was found that auditory display can be used to find specific colour ranges in a section of a cervical cancer slide image. The techniques described in this thesis could be applied to the same problem. Specific areas of interest could be spatialized around the listener – providing them enough information to seek, with their eyes and ears, potentially dangerous cells. The interaction techniques to extend the display would also

allow the clinician to search vast images of cells with ease, using the auditory display to alert them to off-screen cells and find potential areas of interest.

9.4.3 Using Spatial Audio to Improve User-Experience

Most commonplace modern-day operating systems such as OS X, Android, iOS and Windows use multiple desktops to some degree – an example of this being Apple’s ‘Spaces’ paradigm, as shown in *Video Example 9.1*. When an application on another screen requires the user’s attention, it often provides a visual cue – in the case of a Windows machine its icon in the toolbar flashes orange, and in the case of Macintosh computers, the icon bounces up and down and occasionally a mono auditory icon/earcon is given. None of this offers directionality – the user simply does not know what direction to scroll to access the application.

The techniques developed during this project could be applied to this problem by spatialising the auditory icon in the direction the user must scroll to find the application that requires attention. A pulse train parameter could then get faster as the user moves closer to the required desktop – getting faster desktop by desktop. Once the user has reached the target desktop the sound could resolve spatially (become central) and stop making sound once the user had interacted. This simple feature could be used to increase interface immersion and user satisfaction. Additionally it would save valuable screen space by not presenting a visual cue – therefore making this paradigm ideal for smaller devices.

9.5 Summary of Findings

From the information described in the previous 8 Sections it is possible to say that the hypothesis has been supported by the findings in this thesis. The

9.5 Summary of Findings

results from the user experiments that tested the auditory display techniques allow us to make the following five conclusions:

- Using binaural audio as an interactive sonification mapping parameter can aid users in the location of graphical features.
- The larger and the more complex the display, the more the users rely on the auditory mapping parameters.
- Binaural auditory displays can be shown to improve a user's performance when used in tandem with visual cues to search a graphical display, compared to the use of visual cues alone.
- Users can identify simple pictures by means of spatial auditory display using the techniques developed.
- Users have preconceptions with regards to colour-to-sound mappings in the context of sonification.

The work done in this project expands the current ideas used to represent graphical information by means of auditory display. It is evident that using binaural audio as a mapping parameter can not only improve our perception of graphical data when we are unable to see it, but it can also increase our ability to locate graphical features when used in combination with visual cues. Therefore it is possible to conclude that binaural audio is indeed a meaningful interactive sonification mapping parameter to extend the visual domain.

Appendix A *ISon 2013 Paper*

INTERACTIVE SPATIAL AUDITORY DISPLAY OF GRAPHICAL DATA

Timothy Neate & Andy Hunt

Audio Lab
University of York
York
UK

tdjneate@gmail.com
andy.hunt@york.ac.uk

ABSTRACT

This paper describes a series of experiments designed to test auditory display techniques for the interactive sonification of graphical features on a tablet computer. The aim of these experiments was to evaluate new sonic methods for finding graphical features on a tablet computer screen for both regular size, and extended (larger than physical screen size) displays. A series of tests was designed to evaluate the techniques, and determine the effectiveness of binaural information in a series of goal-oriented searching tasks. The results show that the use of binaural audio gives faster location times, allows better searching techniques, and yields an improvement in locational ability when using auditory and visual information in tandem.

1. INTRODUCTION

In an era where displays are smaller, and screen real-estate is limited, the Human-Computer Interaction community is continuously exploring new approaches to tackle the challenge of fitting more content on less screen space. In the field of sonification, several approaches have been tested in an effort to expand screen displays into the auditory domain. With increased audio processing capabilities and interaction modes on smaller and more portable devices, the field of auditory display is becoming a forerunner in presenting additional information by means of multimodality.

One approach used to extend the visual domain is to place all non-essential information into a spatialized auditory field, with different zones of priority [1]. Approaches such as these allow the user

to concentrate on information that is of high importance first, and then deal with information that is of less importance.

Other methodologies simply present all the visual information as it is, with a direct mapping into a raw auditory form [2]. Techniques such as this have been found to be highly successful in enabling those with visual impairments to gain a better idea of their surroundings, but tend to require extensive training on behalf of the user [3]. An alternate approach to this is to filter the data that we seek in the visual domain, before transforming it into the auditory domain [4]. This approach favours a goal-oriented searching task, where the user already knows what they are looking for, but does not fare well when representing raw image data.

This paper describes a goal-oriented approach to enhancing visual representation by means of interactive spatial auditory display. It is found that the approach described can aid users in locating graphical features on displays of different sizes with minimal, or even no visual cues. The work's novelty is derived from binaurally sonifying the relationship between the interaction of the user and the graphical features on a tablet computer, as well as the techniques developed to handle this interaction. As far as the authors know, this type of interaction to feature mapping has not been implemented with spatialized audio before.

This paper first covers the relevant background work in each of the relevant topics of this paper. It then goes on to discuss the implementation of the auditory display and interaction techniques that were developed. The next three sections then discuss the test's setup, the methods of user testing, and the results of the tests. This is done such that

the testing procedure can be reproduced or scrutinised. A discussion section then outlines the results' implications, and then to finalise the paper a conclusions and further work section summarises the paper and discusses potential further work in the area.

2. BACKGROUND

This study brings together three main areas of research in the auditory display community – the sonification of graphical data, spatial audio sonification, and new interfaces for auditory display. The following three subsections give a background on each of these areas.

2.1. The Sonification of Graphical Data

In recent years there have been several methods devised for the transformation of graphical data into the auditory domain. Generally, these take two approaches: transforming all of the graphical data into a complex auditory field that envelops the listener holistically [2] [5], and a goal-driven exploratory methodology where the graphical data is first filtered – the user being left only with the specific features required [4] [6] [7].

Meijer's approach [2] involves scanning a video feed, mapping frequency to the height of a pixel in the display, and mapping the brightness of the pixel to the amplitude of a sinusoidal oscillator. This results in a highly reactive, complex auditory field that is best used to describe complex images or videos. Approaches such as this require the user to learn the mappings over extended periods of time. On the other hand, Bologna's work [8] endeavours to filter specific colours and only transform them into the auditory domain, resulting in a system that is easier to use than Meijer's, but can only provide limited goal-oriented information.

2.2. Spatial Audio Sonification

Spatialized audio has been used numerous times by those wishing to transform information meaningfully into the auditory domain [6] [9] [7] [1]. It is a highly suitable method to use for representing a physical direction in Cartesian space because of our innate ability to determine the location of a spatialized source within 11.8 degrees [10, pg. 39].

Binaural audio, or the notion of portraying 3D sound over headphones, has become an invaluable tool in the auditory display community for representing spatiality [7] [6]. It allows for effective,

cheap, and portable 3D audio – meaning that we can present complex auditory fields outside of the lab environment.

2.3. Interfaces for Auditory Display

As technology has become more powerful, its design has become more suited to our interactions. In the area of interactive sonification, we always try to develop for the best platforms we can at the time. From the first modern personal computers, up until a few years ago, this has almost exclusively involved interaction by mouse or similar PC peripheral. Touch interaction has not been sophisticated enough to become a viable portable platform for development until a little over 3 years ago, with the rebirth of tablet computer – Apple's iPad.

Now that we can use an extensive array of different interaction techniques to explore data, there are fewer limits in the world of interactive sonification – the human-computer interaction loop has become stronger, and there is less cognitive strain on behalf of the user – as they are free to think about what they are interacting with, and concern themselves less with how to operate the system.

3. PROPOSED SYSTEM

The system developed allows the user to experience an image in the auditory domain. When they interact with the image on an iPad by touching the screen they experience auditory feedback that indicates features around their point of touch – their colour represented by different sounds. They are then able to locate these features by moving their touch location around the screen and using the various auditory parameters:

- **Binaural panning** to describe its direction
- a **pulse train** to describe its distance; and
- an **alert** when they find it.

Additionally, other parameters have been developed to assist users when locating multiple features, or when searching on extended displays. The implementation of the experimental system can be broken down into three main parts – the location of the image feature, the user interaction, and the auditory feedback. This system is depicted in Figure 1.

3.1. Touch to Image Feature Calculation

The image-processing algorithm used in this implementation finds the average Cartesian point of a

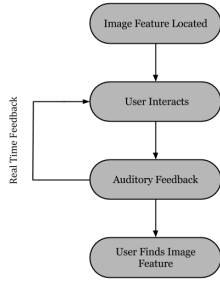


Figure 1: Interactive Graphical sonification system

specific colour by summing the positions of pixels’ ‘x’ and ‘y’ coordinates within a specific threshold set by the user. Using iOS touch delegate methods it is then possible to track the user’s touch. Once this has been tracked, it is possible to find the vector between the user’s touch, and the image feature, as outlined in Figure 2 and Equation 1, where ‘N’ is the number of pixels within the filter in the image, ‘P’ is the filtered pixel’s coordinate in the respective direction, and ‘t’ signifies a touch point by the user. The letter ‘d’ denotes the dimension this algorithm travels through – this will be either ‘x’ (left to right), or ‘y’ (top to bottom). The end result of this algorithm is an integer for both dimensions that represents the average Cartesian coordinate of the pixels filtered.

$$\Delta_d = \frac{\sum_{P_d=0}^{N_d} P_d}{N_d} - t_d \quad (1)$$

The angle (Θ) of the vector is then determined using Equation 2, and the magnitude (M) with Equation 3.

$$\Theta = \frac{180}{\pi} \arctan\left(\frac{\Delta x}{\Delta y}\right) - \frac{\pi}{2} \quad (2)$$

$$M = \sqrt{\Delta x^2 + \Delta y^2} \quad (3)$$

As Equation 3 calculates the angle of the vector from ‘12 o’clock’ for the user’s touch, this allows for the audio processing system to project sources around the listener, with their finger as the ‘central point’. Equation 3 is used to determine the magnitude (M) of the vector. This system is described in Figure 2.

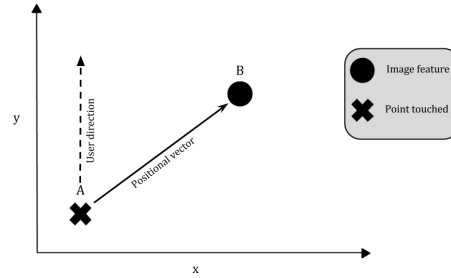


Figure 2: Vector between touch-point and image feature

3.2. Multi-touch Interaction

There needs to be a clear differentiation between when users want to activate the sound mapping, and when they need to move the virtual image around the screen display. After some tests with a small group of individuals it was established that one-finger touches should be used to activate and update the sound engine, and two fingers should be used to move the virtual image around the display. By doing this, it not only offers a clear differentiation between the two techniques, but also allows for simultaneous use of both techniques.

Additionally, a sound-mapping parameter was used for when a user extends the bounds of an image. It was decided that a ‘boing’ sound should be used for this. A simple oscillator with harmonics was panned binaurally, dependent on which side the user had scrolled too much on – its frequency used to represent how much they had scrolled in excess of the scrollable screen.

3.3. Auditory Display Mappings

The vectors calculated are used to drive the audio engine, which was written in Csound, and developed for iOS using the Csound-iOS API developed by Steven Li and Victor Lazzarini [11]. This allows the flexible audio processing capabilities of Csound to be combined with the touch interaction of the iOS platform. Several parameter mapping sonification methods were developed in Csound to provide interactive auditory feedback to the user. These are described below:

Pulse train – Pulse trains have been used to represent distance through sound with great success [12]. The decision was made to increase the pulsing as the users touch got closer to the image feature as it complements the instant real-time

feedback we are familiar with when interacting with real-world systems – the closer the touch, the more frequent the pulse, therefore the faster the feedback to the user as they get closer. From this, it is possible for the user to make fine adjustments of position towards the shape faster, without having to wait for the next pulse. Several proximity zones were devised, as shown in Figure 3. Each proximity zone used a different speed of pulse train – the higher the proximity zone, the faster the pulse train. On an image the same size as the iPad screen, typically 9 proximity zones were used. For larger images, the number of proximity zones was scaled up accordingly so that the mappings remained consistent.

The sound used for the pulse depends on the colour of the visual object being represented. For the purposes of the experiment, four main synthesizers were built – a noise-based synth for BLACK, and three subtractive synth tones differing in pitch to represent the colours RED, BLUE, and GREEN. It would be possible to scale this up to more synthesizers for additional colour mappings.

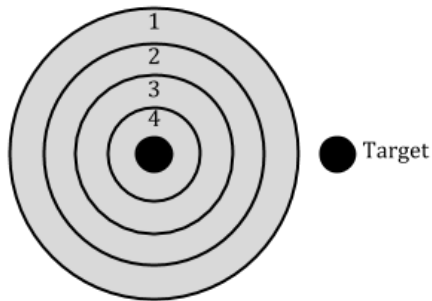


Figure 3: Example of four proximity zones

Binaural Panning – The user was placed in the auditory field by touching the interface. This allowed them to then move through the auditory field with the various image features appearing around them, panned binaurally. The HRTFs, made by Bill Gardner and Keith Martin at the MIT Media Lab [13], were used as they are high quality measurements made using a KEMAR (binaural dummy head) designed according to the mean anatomical size of the population, therefore resulting in a good average head [14]. The Csound opcode ‘hrtfmove2’ [15] was used to interpolate the source (pulse train) around the listener as it offers good quality imaging, with minimal processing.

Alert Sound – A simple alert sound was used

when the user ran their finger over specific areas to indicate that they have found something. This alert sound was ideal when the participants of the experiment were undertaking goal-oriented tasks as it provided them with some element of closure.

Volume – it was noted in the initial designs that when there were multiple sources of sound i.e., multiple image features detected, that the sounds often clashed. Therefore, a volume parameter was developed. This used the proximity zones described previously, but instead of controlling the speed of the pulse train, the volume parameter was used to control the perceived amplitude of sound emanating from the image source, increasing as the user travels closer to it.

4. EXPERIMENTAL SETUP

To evaluate the effectiveness of the techniques developed, the participants were divided into two groups; a ‘control’ group – B, and a ‘treatment’ group – A. Group B were provided with some auditory display parameters to find a specific feature in the image. Group A, however, had additional parameters – with the aim of testing whether the techniques developed affected the performance (determined by speed of location) of the participants undertaking the tasks.

For some of the experiments, the participants needed to be visually restricted so that they could not see the iPad screen, and instead only operated by touch and sonic feedback. Typically, blindfolds are used for this purpose. However, blindfolds can often be considered unethical, and may cause distress in the user. Therefore a visual restrictive device, similar to the device used by Fernström [16], was used. A simple visual restrictor (shown in Figure 4) was created out of a cardboard box and some cloth to ensure that the user could not see through it. The iPad could then be placed in the box and the user could freely interact with it.

Each participant was given a small training session at the beginning of Test 1 and Test 2. The training allowed them to see an example of the tests they were about to undertake such that they knew the relevant auditory and interaction parameters to complete the remaining tests. They were encouraged to practice until they felt that they could find the required image features without visual cues.



Figure 4: Makeshift visual restriction device

5. USER TESTING

Willing participants were asked to undertake a series of tests to examine the techniques developed. There were two main types of test – Test 1 and Test 2:

- **Test 1** focused on using auditory display techniques to portray images the size of a standard iPad screen; and
- **Test 2** involved extending the size of the image – using large scrollable images displayed on an iPad.

The demographics are first discussed such that an impression of the sample can be gained, then the rest of this section describes how each individual test was run, states its results, and discusses any significant findings.

In total 18 participants undertook the experiment – nine in each group, with an average age of 25.8 (standard deviation = 4.3). The group included people of British, Chinese, Dutch, Greek, American, Belgian, and Russian nationalities. In all, 12 were male, and six were female. The majority of the participants (13 out of 18) were from the Department of Electronics (University of York), predominantly in the Audio Lab, and the remaining five were from the Department of Computer Science. Due to the large number of people from the Audio Lab, the subject set included a relatively large number of musicians – 13 out of 18.

Some questions were asked specific to sound perception, binaural audio, and familiarity with tablet devices. It was found that three out of 18 participants claimed to have some form of sound-to-colour synaesthesia (the perception of one sense in the form of another), and all but one of the participants had experienced binaural audio before. When

played a short binaural sample and asked to identify where they believed the source to be coming from, 15 participants said they knew exactly where it was at all times, and the remaining three said that they knew where it was most of the time. With regards to tablet computer/smartphone experience, 16 people owned devices, and the other two had some experience with them.

5.1. Results

This section will describe the details of each test and then go on to state the results.

5.1.1. Test 1.1: finding a black dot [with/without binaural]

In this test the user was tasked with finding a black dot on a screen by means of sound alone. Group A was given the pulse train, alert sound, and the binaural panning parameters. Group B were stripped of the binaural panning parameter and were provided with only mono audio, and thus acted as a control group so that the effect of the binaural parameter could be evaluated.

In this test all participants were able to find the black dot using the auditory feedback. Group A (who used the binaural mapping parameter) succeeded in finding the dot with a mean time of 15.6 seconds (blue line in Figure 5), with a standard deviation of 14.7. Group B (who did not use the binaural mapping parameter) were able to find the dot with a mean time of 18.4 seconds (red line in Figure 5), and a standard deviation of 10.7. The null hypothesis for Test 1.1 was that there would be no difference in times between Group A and B when searching for the dot – the binaural audio would make no difference. The alternative hypothesis was that using binaural audio would speed up the time they took to find the dot. The results were tested using a t-test for two independent samples, attaining a p value of 0.675 – suggesting low levels of confidence in the results.

5.1.2. Test 1.2: three coloured dots [Group B not told colour mappings]

For this test both Group A and Group B were asked to locate three coloured dots on a screen using the pulse train, alert sound, and binaural panning parameters. Group B, however, were not told the colour mappings, which were as follows: Red = high-pitched sound, Green = middle-pitched sound, Blue = low-pitched sound. The main aim of this

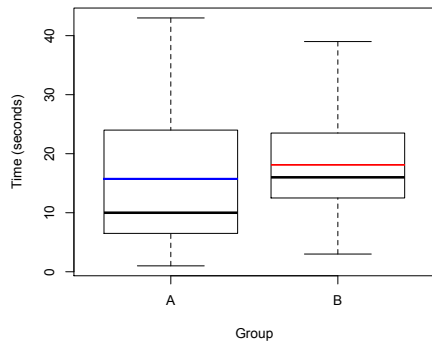


Figure 5: Boxplot for Test 1.1

test was to determine whether we have some pre-conceptions about how colour relates to sound. In addition we hoped to get qualitative information about how subjects coped with more than one acoustic target.

In the test, Group A, who were told the colour-sound mapping parameters, were able to get, on average, 2.45 out of the 3 dots correct. Group B, who were not told the colour-sound mapping parameters were able to get, on average, 2.34 dots correct. The null hypothesis was that by guessing at random, Group B would typically only be expected to get 1 out of 3 dots correct, and the alternative hypothesis was that they would guess more than 1 out of 3 of the dots correct. A Chi-Squared test was run to determine the odds of Group B getting this score by chance, and a confidence interval of $p = 0.097$ was attained – therefore showing relatively high confidence in the results.

5.1.3. Test 1.3: picture identification [both groups with same mappings]

For this test, users were asked to identify a simple picture (picture 3 in Figure 6) by interacting with it, and listening to its auditory response. They were asked to choose from four pictures (shown in Figure 6) the image they believed they had been interacting with. This test was designed to judge the success of the auditory display mappings when representing simple images.

For this test the null hypothesis was that both groups would score the same as they would by guessing – each picture getting, on average, 4.5 users pick it. The alternative hypothesis was that the user

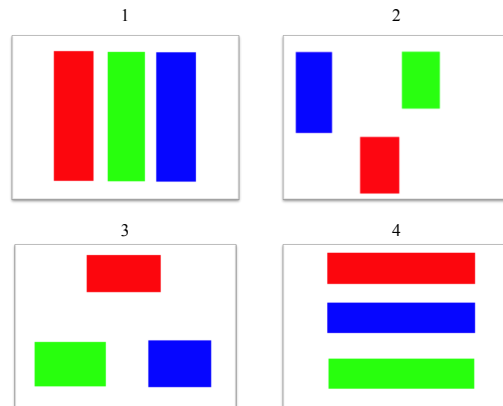


Figure 6: The four pictures the users were presented with (number 3 being the one they were actually interacting with)

would pick Picture 3, the correct answer, more than 4.5 times. In the test the participants chose Pictures 1 and 4 zero times, Picture 2 once, and Picture 3 (the correct picture) 17 times, in an average time of 27.34 seconds, and a standard deviation of 12.86. A Chi-Squared test was run to test the odds of this happening by chance – a confidence interval of $p = 4.562e^{-10}$ was attained – therefore suggesting very high confidence in the results.

5.1.4. Test 2.1: black dot in a large image [with/without binaural]

This following tests challenged the participants in tasking them with navigating a larger-than-display image. Test 2.1 was similar to Test 1.1 – the test where the users were tasked with finding a black dot using sound alone. However, the image used in this test was nine times larger than the iPad screen. The aim of this test was to evaluate the auditory display, and interaction techniques, when navigation around a larger image was involved.

For this test all but one (17/18) of the participants were able to find the black dot. The null hypothesis was that the additional binaural audio feedback provided to Group A would not speed up their performance, and that the mean times of the two groups would be the same. The alternative hypothesis was that Group A would be able to find the dot faster than Group B. In the experiment, Group A found the dot with an average time of 108.25 seconds (blue line in Figure 7) and a standard deviation of 68.16 and Group B 131.56 seconds (red

line in Figure 7) and a standard deviation of 71.55. A t-test for two independent samples was used, attaining a p value of 0.5036, therefore suggesting relatively low confidence in the results.

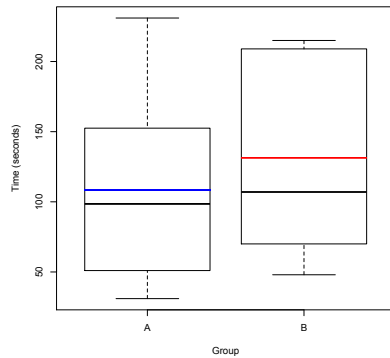


Figure 7: Boxplot for all participants in Test 2.1

5.1.5. Test 2.2: three coloured dots in a large image [with/without binaural]

In this test users were asked to navigate a large display and were tasked with locating three coloured dots. Again, the focus was on a comparison of binaural audio verses mono audio.

In this test, seven out of nine participants in Group A found all three dots, compared with four out of nine for Group B. The null hypothesis was that Group A, with the binaural audio, would perform the same as Group B, without the binaural audio. The alternative hypothesis was that Group A would be able to find the dots quicker, on average, than Group B. The average time to finish the test for all members of Group A was 242.8 seconds (blue line in Figure 8) with a standard deviation of 65.7, and for Group B it was 391.3 seconds (red line in Figure 8) with a standard deviation of 249.8. A t-test for two independent samples was used, attaining a p-value of 0.124, therefore suggesting some confidence in the results.

The average time for those who found all three dots in Group A was 254.85 seconds, with a standard deviation of 60.58. In Group B this was higher at 312 seconds, with a standard deviation of 255.16. A one tailed t-test for two independent samples produced a p value of 0.587 – suggesting relatively low confidence in the results.

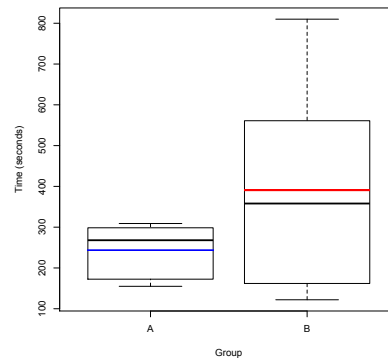


Figure 8: Boxplot for those who finished Test 2.2

5.1.6. Test 2.3 & Test 2.4: black dot in a large image [no visual restriction]

These tests involved a multimodal task using larger scrollable images – Test 2.3 featuring an image nine times the iPad screen, and Test 2.4 16 times the size. For these tests Group A were tasked with finding a black dot by means of using an auditory display, and Group B were tasked with finding the dot by means of visual cues alone. The aim of this test was to judge if the sonification techniques affected the performance of the user when searching.

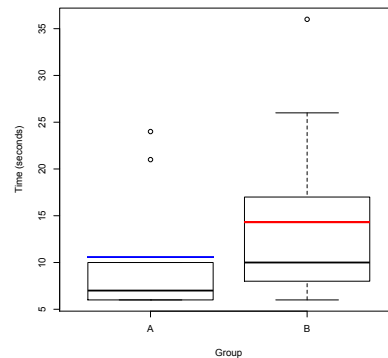


Figure 9: Boxplot for results of Test 2.3

In Test 2.3 the null hypothesis was that the group who were able to see the iPad screen and had auditory feedback (Group A) would perform the same as the group with visual cues alone (Group B). The alternative hypothesis was that Group A

would be able to find the dot quicker. In the test, all participants found the dot – Group A attaining an average time of 10.6 seconds (blue line in Figure 9) and a standard deviation of 7, Group B attaining an average time of 14.3 seconds (red line in Figure 9) and a standard deviation of 10.3. A t-test for two independent samples was used, attaining a p value of 0.373 – showing relatively low levels of confidence.

In Test 2.4 the null hypothesis was the same as in the previous test – that Group A, with the additional auditory feedback, would perform the same as Group B, who had no auditory feedback. The alternative hypothesis was that Group A would find the dot in less time than Group B. The results show that Group A were able to find the dot with an average time of 12.4 seconds (blue line in Figure 10) and a standard deviation of 6, and Group B an average time of 15.2 seconds (red line in Figure 10) with a standard deviation of 11.1. Upon running a t-test for two independent samples, a p value of 0.519 was found – suggesting relatively low levels of confidence in the results.

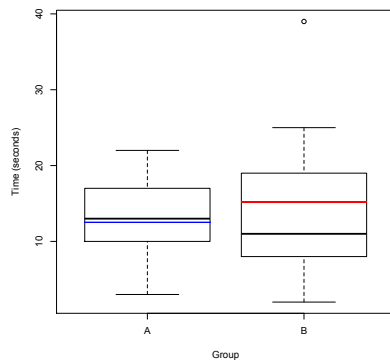


Figure 10: Boxplot for results of Test 2.4

6. DISCUSSION

It was evident from the results of Tests 1.1 (locating a black dot), 2.1 (locating a black dot on large screen), and 2.2 (locating three coloured dots on large screen) that the binaural audio allowed the participants to, on average, find the image features faster than with the other mappings alone – the pulse train, volume, ‘boing’ sound, and alert sound.

The results suggest that as the images became larger, and the tasks more complex Group A outperformed Group B more and more. In Test 1.1 (locating the black dot) Group A were able to locate the image feature 17.7% faster than Group B, and in Test 2.1 (locating a black dot within a large scrollable image), in which the image is 9 times larger, there was a 21.5% difference between the groups – those with binaural audio appear to perform even better in comparison to the non-binaural group. The difference between the groups in Test 2.2 becomes larger – an (albeit small) increase of 0.92% between the two groups when an additional difficulty is added – this may suggest that as the tasks become more complex, the binaural audio helps more. However, the significance levels were not high enough for us to state this categorically.

Upon observation of the videos (all available on the link provided in the supporting material folder with this paper) it is possible to say, anecdotally, that the participants with the binaural audio (Group A) undertook more logical searching strategies, whereas the group without the binaural audio (Group B) normally undertook a more sporadic ‘brute force’ searching method, which would account for the larger spread in times, and therefore higher standard deviation, for Group B. However, it must be noted that even in Group A there was a large spread of times. We believe that this large variation between the participants, as well as the relatively low number of participants, has led to the lower levels of significance found.

Test 1.3 (image identification using binaural audio) indicated strongly that participants were able to detect a picture from sound alone using the techniques. The techniques excelled at allowing the participants to gain a quick overview of the graphical features on display with a very high success rate, and an insignificantly low probability of attaining the same results through guessing. The large variation in times can be attributed to the searching techniques of the individuals. From discussing the test with the participants after the experiment it became evident that some participants looked at the pictures beforehand and made a mental model of what they believed they were looking for. Meanwhile, others investigated the auditory response, and then by process of elimination chose their answer. The subjects with the faster times generally adhered to the first approach.

When comparing visual and audio cues, against visual cues alone in Tests 2.3 and 2.4 (black dot within progressively larger images with and without visual cues) the difference in searching tech-

niques became even more evident. The relatively large standard deviations of Group B can be attributed to the sporadic searching patterns the group used. Sometimes a participant would randomly scroll around very quickly and get lucky, whilst others spent quite a while ‘raster scanning’ the image in a logical search. Most participants in Group A took a while to assess the local area, then gradually moved in a straight line towards the image feature, resulting in the higher mean times and smaller standard deviations.

7. CONCLUSIONS AND FURTHER WORK

Several auditory display techniques were developed to allow for images of varying sizes to be explored by means of binaural interactive sonification. Gesture-based interaction modes were created to facilitate the exploration of these images on an iPad, whilst listening to an auditory representation. Seven tests were then carried out to deduce whether the techniques were improved by use of binaural audio.

The results from the tests showed that binaural audio could be used to improve our understanding of simple images of varying sizes. It is evident that the experiments could benefit from additional participants – the number used (eight in each group) was not enough to produce very significant results. It is recommended that if this experiment is replicated, additional participants should be recruited. To reproduce the tests described in this paper a folder including the test scripts, a document describing the technical setup, videos of the participant’s performances (provided pending acceptance), and the code needed to reproduce the tests has been provided at the following Dropbox link:

<https://db.tt/eyVcfAFf>

Further work in this area should involve increasing the complexity of the images and the image processing algorithm. Similar methods could be used to explore more complex images where a user wishes to search for a specific colour, such as when scanning cervical cancer slides, or looking for objects in Deep Field space photography. Additionally, the binaural auditory display techniques could be used for numerous applications, not just the exploration of images. For example – improving immersion in computer interfaces or assisting those who are visually impaired, or have their eyes on other tasks, by extending the visual domain with

spatial audio.

This paper has demonstrated the potential of binaural audio to provide real-time feedback to visually restricted or distracted users to improve the location of objects in the data being represented both on and off-screen.

8. REFERENCES

- [1] David McGookin and Stephen Brewster, “Fishears - the design of a multimodal focus and context system,” Tech. Rep., Glasgow, 2001.
- [2] Peter Meijer, “An experimental system for auditory image representations,” *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 2, pp. 112–121, 1992.
- [3] Amir Amedi, William Stern, Joan Camprodon, Felix Bempohl, Lofti Merabet, Stephen Rotman, Christopher Hemond, Peter Meijer, and Alvaro Pascual-Leone, “Shape conveyed by visual-to-auditory substitution activates the lateral occipital complex,” *Nature Neuroscience*, vol. 10, no. 6, pp. 687–689, 2007.
- [4] Guido Bologna, Benoît Deville, Pun Thierry, and M Vinckenbosch, “A perceptual interface for vision substitution in a color matching experiment,” *Joint Conference for Neural Networks - IEE World Congress on Computational Intelligence*, pp. 1–6, 2008.
- [5] Ryan McGee, Joshua Dickinson, and George Legrady, “Voice of sisyphus: An image sonification multimedia installation,” *ICAD*, vol. 18, 2012.
- [6] Wilko Heuten, Daniel Wichmann, and Susanne Boll, “Interactive 3d sonification for the exploration of city maps,” Oslo, 2006, NordiCHI.
- [7] Piotr Skulimowski Michal Bujacz and Pawel Strumillo, “Naviton: A prototype mobility aid for auditory presentation of three-dimensional scenes to the visually impaired,” *AES*, vol. 60, pp. 696 – 708, 2012.
- [8] Guido Bologna, Benoît Deville, and Thierry Pun, “Sonification of color and depth in a mobility aid for blind people,” *International Conference for Auditory Display*, vol. 16, pp. 9–13, 2010.
- [9] Brian Katz, Emmanuel Rio, Lorenzo Picnali, and Olivier Warusfel, “The effect of spa-

- tialization in a data sonification exploration task,” *ICAD*, vol. 14, pp. 1 – 7, 2008.
- [10] Jens Blauert, *Spatial Hearing - Revised Edition: The Psychophysics of Human Sound Localization*, Hirzer Verlag, Massachusetts, 1997.
 - [11] Lazzarini, “The audio programming blog,” 2012.
 - [12] Tsubasa Yoshida, Kris Kitani, Hideki Koike, Serge Belongie, and Kevin Scheli, “Edgesonic: Image feature sonification for the visually impaired,” *Augmented Human Conference*, vol. 2, pp. Article 11, 2011.
 - [13] Bill Gardner and Keith Martin, “Hrtf measurements of a kemar dummy-head microphone,” Tech. Rep., MIT Media Lab, 1994.
 - [14] Wen Zhang, Thushara Abhayapala, and Rodney Kennedy, “Insights into head-related transfer function: Spatial dimensionality and continuous representation,” *Acoustical Society of America*, vol. 127, pp. 23472357, 2010.
 - [15] Brian Carty, “hrftmove, hrtfstat, hrtfmove2: Using the new hrtf opcodes,” *Csound Journal*, vol. 9, 2008.
 - [16] Mikael Fernström, Liam Bannon, and Eoin Brazil, “An investigation of soft-button widgets using sound,” *Le Journe de Design Sonore*, pp. 1–11, 2004.

Appendix B *Pilot Study Script*

Pilot Study: An Investigation of Sound and Colour

Timothy Neate

Background

The author is currently designing a series of auditory displays for 3D graphical sonification – turning visual information into spatial sound. The main aim of the project is to find out the best image processing methods, sonification algorithms, and interaction techniques, for allowing a user to find specific features on a display with minimal visual cues.

Study Outline

The aim of this study is to explore our pre-determined disposition to colour when using specific sonification techniques. Since 18th century, scientists such as Isaac Newton and Louis Bertrand Castel have tried to find relationships between colour and frequency, this study aims to explore their hypotheses, and explore some new ideas with regards to how best to represent colour, in the context of sonification.

The tests consist of three main sections:

- Demographics Questionnaire
- **Test 1** – a test to determine the pleasantness and meaning of synthetic sounds
- **Test 2** – a test to explore how pitches in a scale related to colour

Note: The aim of this experiment is to try and gather some subjective data. It must be expressed that there are **no right answers**, but all answers will provide insight into the sound design of auditory displays.

Demographics questionnaire

This is a small questionnaire to gauge the participant's background to ensure a well encompassing data set. Your information will be anonymous, and any personal information will be destroyed after the overall results are analysed.

1. What is your age? _____
2. What is your gender? (Male / Female)
3. What is your nationality? _____
4. Are you a student? (Yes / No)

If answered 'Yes', please state the department are you based in?

5. Do you have musical training or play an instrument? (Yes / No)

If so, elaborate:

6. Do you associate certain timbres or pitches with colour (synaesthesia)? (Yes/ No)

If so, elaborate:















7. Would you consider yourself to have perfect pitch (the ability to recognize the pitch of a note)? (Yes / No)

8. Would you consider yourself to have relative pitch (the ability to determine a pitch from some reference pitch)? (Yes / No)

Test 1
































































Listen to all the sound clips to take in all the information so that you can make an unbiased decision. Each sound will repeat three times before you can move on to the next sound. On the second listen you may mark down **one** colour, of the two shown, you believe the sound represents best. You may skip questions, and go back, compare sounds, and listen as much as you like to the sounds.

Use the 'Associated Colour' column to choose from the options. The 'Sound Rating' column shows the 'pleasantness' of a sound. Please circle the colour that that best represents the sound you hear for each sound clip.

Sound Clip	Associated Colour		Sound Rating
1.1			Bad [1] – [2] – [3] – [4] – [5] Good
1.2			Bad [1] – [2] – [3] – [4] – [5] Good
1.3			Bad [1] – [2] – [3] – [4] – [5] Good
1.4			Bad [1] – [2] – [3] – [4] – [5] Good
1.5			Bad [1] – [2] – [3] – [4] – [5] Good
1.6			Bad [1] – [2] – [3] – [4] – [5] Good
1.7			Bad [1] – [2] – [3] – [4] – [5] Good

Test 2

Listen to all the sound clips to take in all the information so that you can make an unbiased decision. Each sound will repeat three times before you can move on to the next sound. On the second listen you may begin to mark down **one** colour you believe the sound represents. You may skip questions, and go back, compare sounds, and listen as much as you like to the sounds. Use the 'Associated Colour' column to choose from the options. Please circle the colour that that best represents the sound you hear for each sound clip.

Sound Clip	Associated Colour						
2.1							
2.2							
2.3							
2.4							
2.5							
2.6							
2.7							
2.8							
2.9							

Appendix C *Csound for iOS API – A Beginner’s
Guide*

THE UNIVERSITY *of York*

Department of Electronics

Audio Lab

Csound for iOS API

A Beginner's Guide 1.1

17/2/2013

**Timothy Neate, Nicholas Arner & Abigail
Richardson**

Abstract

This tutorial aims to help iOS developers with the implementation of the Mobile Csound Platform for iOS. Developers who are looking to incorporate audio into their apps, but do not want to deal with the complexities of Core Audio, will find this particularly useful.

It provides some background information on the API and outlines how to integrate Csound and iOS, and allow them to communicate. The provided example project is then described - outlining the key features of the API. Some common problems that users are likely to encounter are then discussed to troubleshoot potential issues

Acknowledgements

We would like to thank our supervisor, Dr. Andy Hunt, for his support and guidance while working through the Csound for iOS API, as well as working on this tutorial.

We would also like to thank Dr. Victor Lazzarini and Steven Yi, the authors of the Csound for iOS API. We would especially like to thank Steven for his extremely helpful responses to our questions regarding the API.

1. Introduction

The traditional way of working with audio on both Apple computers and mobile devices is through the use of Core Audio. Core Audio is a low-level API which Apple provides to developers for writing applications utilizing digital audio. The downside of Core Audio being low-level is that it is often considered to be rather cryptic and difficult to implement, making audio one of the more difficult aspects of writing an iOS app.

In an apparent response to the difficulties of implementing Core Audio, there have been a number of tools released to make audio development on the iOS platform easier to work with. One of these is *libpd*, an open-source library released in 2010. *libpd* allows developers to run Pure Data on both iOS and Android mobile devices. Pure Data is a visual programming language whose primary application is sound processing.

The recent release of the Mobile Csound Platform provides an alternative to the use of PD for mobile audio applications. Csound is a synthesis program which utilizes a toolkit of over 1200 signal processing modules, called opcodes. The release of the Mobile Csound Platform now allows Csound to run on mobile devices, providing new opportunities in audio programming for developers. Developers unfamiliar with Pure Data's visual language paradigm may be more comfortable with Csound's 'C'-programming based environment.

For those who are unfamiliar with Csound, or want to learn more, the FLOSS manuals are an excellent resource, and can be found here:

<http://flossmanuals.net/csound/>

For more advanced topics in Csound programming, the Csound Book (Boulangier ed., 2000) will provide an in-depth coverage.

In order to make use of the material in this tutorial, the reader is assumed to have basic knowledge of Objective-C and iOS development. Apple's Xcode

4.6.1 IDE (integrated development environment) will be used for the provided example project.

Although the Mobile Csound API is provided with an excellent example project, it was felt that this tutorial will be a helpful supplement in setting up a basic Csound for iOS project for the first time, by including screenshots from the project set-up, and a section on common errors the user may encounter when working with the API.

The example project provided by the authors of the API includes a number of files illustrating various aspects of the API, including audio input/output, recording, interaction with GUI widgets, and multi-touch. More information on the example project can be found in the API manual, which is included in the example projects folder.

1.1. The Csound for iOS API

The Mobile Csound Platform allows programmers to embed the Csound audio engine inside of their iOS project. The API provides methods for sending static program information from iOS to the instance of Csound, as well as sending dynamic value changes based on user interaction with standard UI interface elements, including multi-touch interaction.

1.2. Document Structure

This document begins, in Section 2, by describing the example provided by the authors. Section 2 is divided into two further sections: Section 2.1 which describes the functionality of the example application and Section 2.2 which details line by line through the example code how this application works. Section 3 provides a step by step guide to setting up an Xcode project for use with the Mobile Csound API. This section describes how to download the API and include it into the project (Section 3.1) as well as the necessary components of the view controller (Section 3.2) and Csound file (Section 3.3).

Section 4 outlines some common problems, which have been found through the creation of this tutorial, and their solutions. Section 5 is a reference of the methods which are available for use in the Mobile Csound API. This section briefly details the functionality of these methods and their method calls. Section 6 provides the authors' conclusions about this tutorial.

NOTE: This tutorial uses Csound 5, and has not been tested with Csound6.

2 Example Walkthrough

This section discusses why the example was made, and what can be learned from it; giving an overview of its functionality, then going into a more detailed description of its code. A copy of the example project can be found at the following link.

<https://sourceforge.net/projects/csoundiosguide/>

2.1 Running the Example Project

Run the provided Xcode project, CsoundTutorial.xcodeproj, and the example app should launch (either on a simulator or a hardware device). A screenshot of the app is shown in Figure 2.1 below. The app consists of two sliders, each controlling a parameter of a Csound oscillator. The top slider controls the amplitude, and the bottom slider controls the frequency.

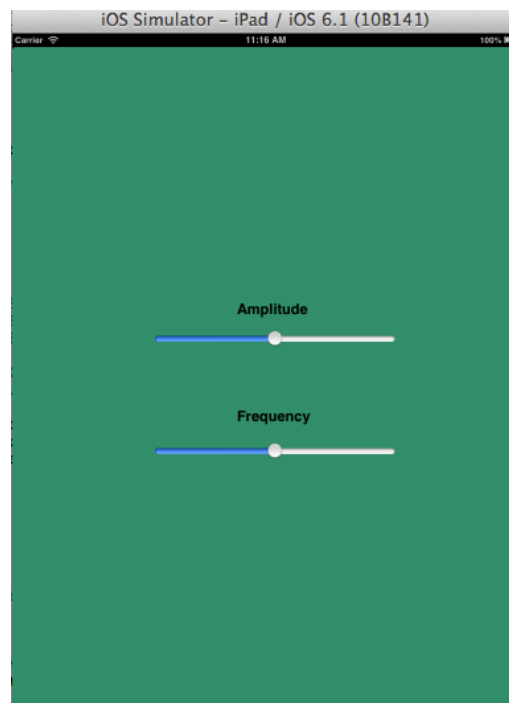


Figure 2.1-App running on iPad simulator

2.2 Oscillator Example Walkthrough

This example outlines how to use the methods in the Csound-iOS API to send values from iOS into Csound. This example was made purposefully simple, with the intent of making its functionality as obvious as possible to the reader. This section begins by giving an overview of both the iOS and Csound implementation, and then describes how this achieved by breaking down the example code. The code to create this oscillator example was done in the *ViewController.h* and the *ViewController.m* files, which are discussed below in sections 2.2.3.1 and 2.2.3.2. The project is split into Objective-C code, Storyboards for the user interface elements, and a Csound file for the audio engine.

2.2.1 iOS Example Outline

In the Xcode project user interface sliders are used to allow a user to control the Csound audio engine through iOS. Communication begins with iOS requesting some memory within Csound; setting a pointer to this location. It updates this pointer with values from the user interface sliders. Csound references the same memory location by naming it with a string, this named communication link is called a channel. When sending this information, iOS uses methods within the iOS-Csound API to setup this channel name, and update it dependant on the control rate.

2.2.2. Csound Example Outline

In this example, Csound is not aware of iOS. All it knows is that there is a piece of memory assigned for it, and it must retrieve information from here dependent on its control rate. Csound uses the *chnget* opcode to do this. *chnget* searches for some channel with a specific name and retrieves values from it.

2.2.3. The iOS File

This example is implemented across two main files:

The **.h file** is used to include all the necessary classes, declare properties, and allow for user interaction by connecting the interface to the implementation.

The **.m file** is used to implement communication between the interface methods declared in the .h file, and the Csound file. These will now be discussed in more depth, with code examples.

2.2.3.1. The .h File

The imports (discussed in detail in section 3.2.1) are declared:

```
#import <UIKit/UIKit.h>
#import "CsoundObj.h"
#import "CsoundValueCacheable.h"
```

Apart from the standard UIKit.h (which gives access to iOS interface widgets) these ensure that the code written can access the information in the other files in the Csound API.

Next comes the class definition:

```
@interface ViewController : UIViewController
<CsoundObjCompletionListener, CsoundValueCacheable>
```

Every iOS class definition begins with the **@interface** keyword, followed by the name of the class. So our class is called *ViewController*, and the colon indicates that our class inherits all the functionality of the *UIViewController*.

Following this are two Protocol definitions, which are listed between the triangular brackets `< >`. In Objective-C a Protocol is a list of **required** functionality (i.e., methods) that a class needs to implement. In this case there are two Protocols that are defined by the Csound API, that we want our class to conform to: *CsoundObjCompletionListener* and *CsoundValueCacheable*. This will allow us to send data between iOS and Csound, and so is essential for what we are about to do. The required functions that we have to implement are described in the section following this one (2.2.3.2).

The Csound object needs to be declared as a property in the .h file, which is what this next line of code does:

```
//Declare a Csound object
@property (nonatomic, retain) CsoundObj* csound;
```

The next section of code allows for the interface objects (sliders) to communicate with the .m file:

```
- (IBAction)amplitudeSlider:(UISlider *)sender;
- (IBAction)frequencySlider:(UISlider *)sender;
```

Just to the left of each of these IBAction methods, you should see a little circle. If the storyboard is open (MainStoryboard.storyboard) you will see the appropriate slider being highlighted if you hover over one of the little circles.

2.2.3.2. The .m File

The .m file imports the .h file so that it can access the information within it, and the information that it accesses.

At the beginning of the implementation of the ViewController, the *csound* variable which was declared in the .h file is instantiated with *@synthesize* thus:

```
@implementation ViewController
@synthesize csound = mCsound;
```

Note that the Csound object must be released later in the *dealloc* method as shown below:

```
- (void)dealloc
{
    [mCsound release];
    [super dealloc];
}
```


For each parameter you have in iOS that you wish to send to Csound, you need to do the things outlined in this tutorial. In our simple example we have an iOS slider for Frequency, and one for Amplitude, both of which are values we want to send to Csound.

Some global variables are then declared, as shown in Table 2.1, a holder for each iOS parameter's current value, and a pointer for each which is going to point to a memory location within Csound.

Variable	Description
<code>float myFrequency;</code>	This value comes from the frequency slider in the interface. It is a float, as the value to send from iOS to Csound needs to be a floating point number. Its range is 0 – 500.
<code>float myAmplitude;</code>	This value comes from the amplitude slider in the interface. Its range is 0 – 1 because of the way the gain is controlled in the .csd file.
<code>float* freqChannelPtr;</code>	These variables are used in conjunction with the method <i>getInputChannelPtr</i> (described towards the end of this section) to send frequency and amplitude values to Csound.
<code>float* ampChannelPtr;</code>	

Table 2.1-Variables for the .m File

The next significant part of the .m file is the *viewDidAppear* method. When the view loads, and appears in iOS, this iOS SDK method is called. In the example, the following code is used to locate the Csound file:

```
NSString *tempFile = [[NSBundle mainBundle] pathForResource:@"aSimpleOscillator" ofType:@"csd"];
NSLog(@"FILE PATH: %@", tempFile);
```

This code searches the main bundle for a file called *aSimpleOscillator* of the type *csd* (which you will be able to see in Xcode's left-hand File List, under the folder Supporting Files). It then assigns it to an *NSString* named *tempFile*. The name of the string *tempFile* is then printed out to confirm which file is running.

The methods shown in Table 2.2 are then called:

Method Call	Description
<code>self.csound = [[CsoundObj alloc] init];</code>	This instantiates the <i>csound</i> object, which will be our main contact between iOS and Csound. It allocates and initialises some memory to make an instance of the <i>CsoundObj</i> class.
<code>[self.csound addCompletionListener:self];</code>	Sets our code (self – i.e. ViewController) to be a listener for the <i>Csound</i> object.
<code>[self.csound addValueCacheable:self];</code>	Sets our code (self) to be able to send real-time values to the <i>Csound</i> object.
<code>[self.csound startCsound:tempFile];</code>	The <i>Csound</i> object uses the method <i>startCsound</i> to run the file at the string <i>tempFile</i> . Remember how <i>tempFile</i> was set up to point to the <i>Csound</i> <i>csd</i> file (in our case <i>aSimpleOscillator.csd</i>). So, in other words, this line launches <i>Csound</i> with the <i>csd</i> file you have provided.

Table 2.2-Csound API Methods

The methods that allow the value of the slider to be assigned to a variable are then implemented. This is done with both frequency, and amplitude. As shown below for the amplitude slider:

```
- (IBAction)amplitudeSlider:(UISlider *)sender
{
    UISlider *ampSlider = (UISlider *)sender;
    myAmplitude = ampSlider.value;
}
```

This method is called by iOS every time the slider is moved (because it is denoted as an *IBAction*, i.e. an Interface Builder Action call). The code shows that the *ampSlider* variable is of type *UISlider*, and because of that the current (new) value of the slider is held in *ampSlider.value*. This is allocated to the variable *myAmplitude*. Similar code exists for the frequency slider.

The protocol methods are then implemented. The previous section showed how we set up our class (*ViewController*) to conform to two Protocols that the Csound API provides: *CsoundObjCompletionListener* and *CsoundValueCacheable*.

Take a look at the place where these Protocols are defined, because a Protocol definition lists clearly what methods are required to be implemented to use their functionality.

For *CsoundValueCacheable* you need to look in the file *CsoundValueCacheable.h* (in the folder *valueCacheable*). In that file it's possible to see the protocol definition, as shown below, and its four required methods.

```
#import <Foundation/Foundation.h>

@class CsoundObj;

@protocol CsoundValueCacheable <NSObject>

-(void)setup:(CsoundObj*)csoundObj;
-(void)updateValuesToCsound;
-(void)updateValuesFromCsound;
-(void)cleanup;

@end
```

Every method needs at least an empty function shell. Some methods, such as *updateValuesFromCsound* are left empty, because – for the tutorial example – there is no need to get values from Csound. Other protocol methods have functionality added. These are discussed below.

The *setup* method is used to prepare the *updateValuesToCsound* method for communication with Csound:

```
-(void)setup:(CsoundObj* )csoundObj
{
    NSString *freqString = @"freqVal";
    freqChannelPtr = [csoundObj getInputChannelPtr:freqString];

    NSString *ampString = @"ampVal";
    ampChannelPtr = [csoundObj getInputChannelPtr:ampString];
}
```

The first line of the method body creates a string; *freqString*, to name the communication channel that Csound will be sending values to. The next line uses the *getInputChannelPtr* method to create the channel pointer for Csound to transfer information to. Effectively, iOS has sent a message to Csound, asking it to open a communication channel with the name “*freqVal*”. The Csound object allocates memory that iOS can write to, and returns a pointer to that memory address. From this point onwards iOS could send data values to this address, and Csound can retrieve that data by quoting the channel name “*freqVal*”. This is described in more detail in the next section (2.2.4).

The next two lines of the code do the same thing, for amplitude parameter. This process creates two named channels for Csound to communicate through.

The protocol method *updateValuesToCsound* uses variables in the .m file and assigns them to the newly allocated memory address used for communication. This ensures that when Csound looks at this specific memory location, it will find the most up to date value of the variable. This is shown below:

```
-(void)updateValuesToCsound
{
    *freqChannelPtr = myFrequency;
    *ampChannelPtr = myAmplitude;
}
```

The first line assigns the variable *myFrequency* (the value coming from the iOS slider for Frequency) to the channel *freqChannelPtr* which, as discussed earlier, is of type `float*`. The second line does a similar thing, but for amplitude.

For the other Protocol `CsoundObjCompletionListener` it is possible to look for the file `CsoundObj.h` (which is found in Xcode's left-hand file list, in the folder called `classes`). In there is definition of the protocol.

```
@protocol CsoundObjCompletionListener
-(void)csoundObjDidStart:(CsoundObj*)csoundObj;
-(void)csoundObjComplete:(CsoundObj*)csoundObj;
```

In this example there is nothing special that needs to be done when Csound starts running, or when it completes, so the two methods (`csoundObjDidStart:` and `csoundObjComplete:`) are left as empty function shells. In the example, the protocol is left included, along with the empty methods, in case you wish to use them in your App.

2.2.4 The Csound File

This Csound file contains all the code to allow iOS to control its values and output a sinusoid at some frequency and amplitude taken from the on-screen sliders. There are three main sections: The Options, the Instruments, and the Score. These are all discussed in more detail in section 4. Each of these constituent parts of the `.csd` file will now be broken down to determine how iOS and Csound work together.

2.2.4.1 The Options

There's only one feature in the options section of the .csd that needs to be considered here; the flags. Each flag and its properties are summarised in Table 2.3.

Flag	Description
-o dac	Enables audio output to default device
-+rtmidi=null	Disables real-time MIDI Control
-d	Suppress all displays

Table 2.3-Csound Flags

2.2.4.2 The Instrument

The first lines of code in the instrument set up some important values for the .csd to use when processing audio. These are described in Table 2.4, and are discussed in more detail in the Reference section of the Csound Manual

Line	Description
sr = 44100	This sets the sample rate of Csound to 44100 Hz. It is imperative that the sample rate of the Csound file corresponds with the sample rate of the sound card the code is running on.
ksmps = 64	This defines the control rate. In the example this will determine the speed that the variables in Csound are read. ksmps is actually the number of audio samples that are processed before another control update occurs. The actual control rate equates to sample rate / ksmps (i.e. $44100 / 64 = 689.0625$ Hz).
nchnls = 2	This is the number of audio channels. 2 = standard stereo.
0dbfs = 1	This is used to ensure that audio samples are within the appropriate range, between zero and one. Anything greater than

one will induce clipping to the waveform.

Table 2.4-Csound .csd Options

The instrument then takes values from Csound using the *chnget* opcode:

```
kfreq chnget "freqVal"  
kamp chnget "ampVal"
```

Here, the *chnget* command uses the “*freqVal*” and “*ampVal*” channels previously created in iOS to assign a new control variable. The variables *kfreq* and *kamp* are control-rate variables because they begin with the letter ‘k’. They will be updated 689.0625 times per second. This may be faster or slower than iOS updates the agreed memory addresses, but it doesn’t matter. Csound will just take the value that is there when it accesses the address via the named channel.

These control-rate variables are used to control the amplitude and frequency fields of the opcode *oscil*; the Csound opcode for generating sinusoidal waves. This is then output in stereo using the next line.

```
asig poscil kamp,kfreq,1  
outs asig,asig  
endin
```

The third parameter of the *oscil* opcode in this case is 1. This means ‘use f-table 1’. Section 3.3 explains f-tables in more depth.

2.2.4.3 *The Score*

The score is used to store the f-tables the instrument is using to generate sounds, and it allows for the playing of an instrument. This instrument is then played, as shown below:

```
i1 0 10000
```

This line plays instrument 1 from 0 seconds, to 10000 seconds. This means that the instrument continues to play until it is stopped, or a great amount of time passes.

It is possible to send score events from iOS using the method *sendScore*. This is discussed in more depth in section. 6.1

3 Using the Mobile Csound API in an Xcode Project

Section 3 provides an overview of how to set up your Xcode project to utilize the Mobile Csound API, as well as how to download the API and include it into your project.

3.1 Setting up an Xcode Project with the Mobile Csound API

This section describes the steps required to set up an Xcode project for use with the Mobile Csound API. Explanations include where to find the Mobile Csound API, how to include it into an Xcode project and what settings are needed.

3.1.2 Creating an Xcode Project

This section briefly describes the settings which are needed to set up an Xcode project for use with the Mobile Csound API. Choose the appropriate template to suit the needs of the project being created. When choosing the options for the project, it is important that *Use Automatic Reference Counting* is not checked (Figure. 3.1). It is also unnecessary to include unit tests.

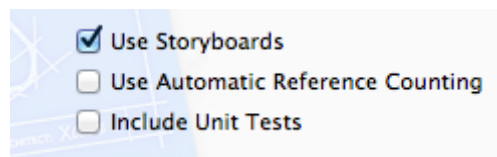


Figure 3.1-Project Set Up

Note: When including this API into a pre-existing project, it is possible to turn off ARC on specific files by entering the compiler sources, and changing the compiler flag to: '-fno-objc-arc'

3.1.3 Adding the Mobile Csound API to an Xcode Project

Once an Xcode project has been created, the API needs to be added to the Xcode project. To add the Mobile Csound API to the project, right click on the Xcode project and select *Add files to <myProject>*. This will bring up a navigation window to search for the files to be added to the project. Navigate to the *Csound-iOS* folder, which is located as shown in Figure 3.2 below.

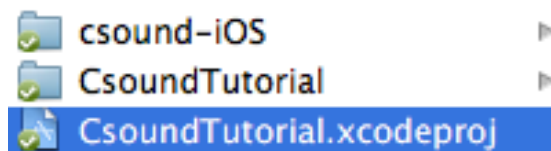


Figure 3.2-Navigating to the API Folder

Select the whole folder as shown and click *add*. Once this has been done, Xcode will provide an options box as shown in Figure 3.3. Check *Copy items into destination group's folder (if needed)*.

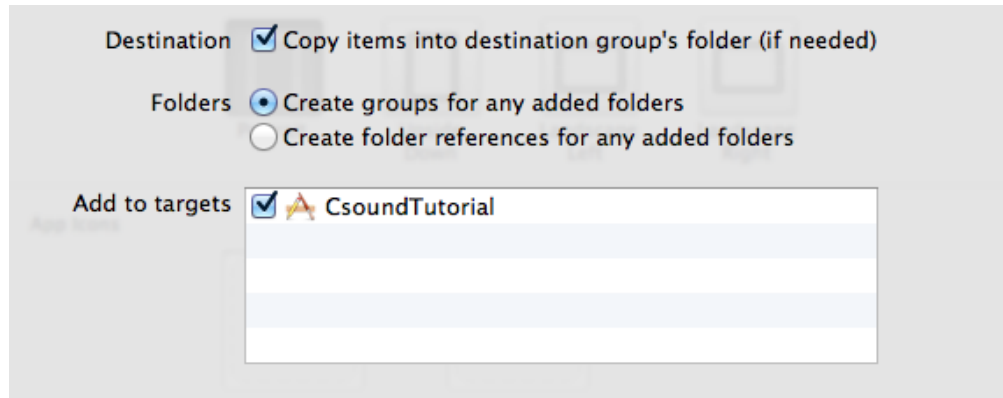


Figure 3.3-Adding the API Folder

The options in Figure 3.3 are selected so that the files which are necessary to run the project are copied into the project folder. This is done to make sure that there are no problems when the project folder is moved to another location - ensuring all the file-paths for the project files remain the same.

Once this addition from this section has been made, the project structure displayed in Xcode should look similar to that in Figure 3.4.

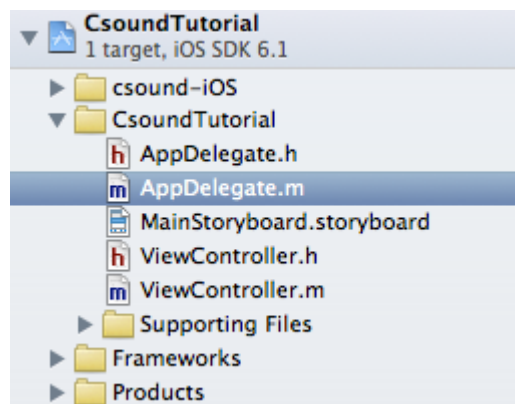


Figure 3.4 - The Main Bundle for the Project

3.1.4 Compiling Sources

A list of compile sources is found by clicking on the blue project file in Xcode, navigating to the *Build Phases* tab and opening *Compile Sources*. Check that the required sources for the project are present in the *Compile Sources* in Xcode. All the files displayed in Figure 3.5 should be present, but not necessarily in the same order as shown.

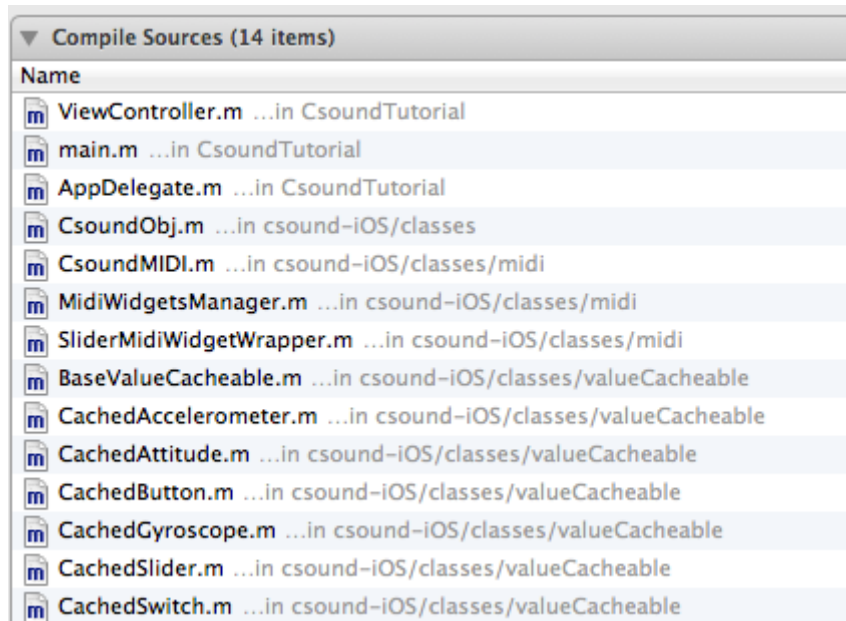


Figure 3.5-View of 'Compile Sources' Window

3.1.5 Including the Necessary Frameworks

There are some additional frameworks which are required to allow the project to run. These frameworks are:

- AudioToolbox.framework
- CoreGraphics.framework
- CoreMotion.framework
- CoreMIDI.framework

To add these frameworks to the project, navigate to the 'Link Binary With Libraries' section of Xcode. This is found by clicking on the blue project folder and navigating to the 'Build Phases' tab, followed by opening 'Link Binary With Libraries'. To add a framework, click on the plus sign and search for the framework required. Once all the necessary frameworks are added, the 'Link Binary With Libraries' should look similar to Figure 3.6 below.

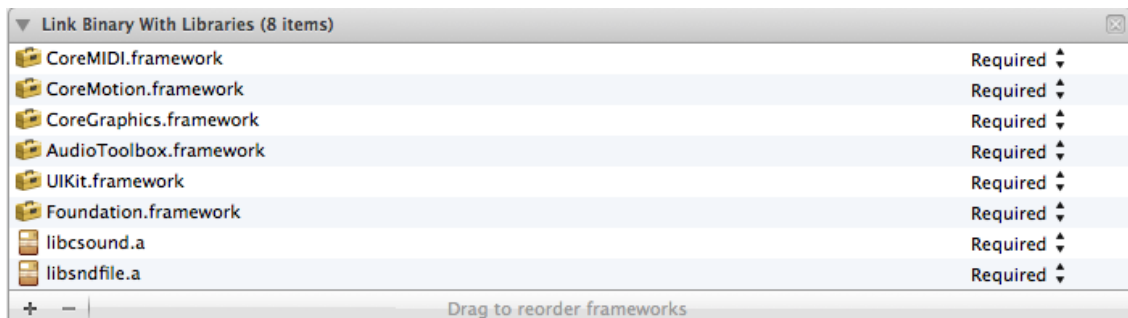


Figure 3.6-Adding Necessary Frameworks

3.1.6 The .csd File

The project is now set up for use with the Mobile Csound API. The final file which will be required by the project is a .csd file which will describe the Csound instruments to be used by the application. A description of what the .csd file is and how to include one into the project is found in *Section 3.3*. This file will additionally need to be referenced appropriately in the Xcode project.

A description of where and how this reference is made is available in *Section 2.2.3.2*

3.2 Setting up the View Controller

This section describes how the *ViewController.h* and the *ViewController.m* should be set up to ensure that they are able to use the API. It will discuss what imports are needed; conforming to the protocols defined by the API; giving a brief overview. This section can be viewed in conjunction with the example project provided.

3.2.1 Importing

So that the code is able to access other code in the API, it is important to include the following imports, along with imports for any additional files required. The three imports shown in Table 3.1 are used in the header file of the view controller to access the necessary files to get Csound-iOS working:

Import	Description
<code>#import "CsoundObj.h"</code>	This is used so that the code is able to access all the key methods of the API.
<code>#import "CsoundValueCacheable.h"</code>	This must be used to access the methods 'updateValuesFromCsound' and 'updateValuesToCsound'. These methods are used to communicate between Csound and iOS.

Table 3.1-Header File Imports

In our example you can see these at the top of *ViewController.h*

3.2.2 Conforming to Protocols

It is imperative that the view controller conforms to the protocols outlined the `CsoundObj.h` file; the file in the API that allows for communication between iOS and Csound. This must then be declared in the `ViewController.h` file:

```
@interface ViewController : UIViewController  
<CsoundObjCompletionListener, CsoundValueCacheable>
```

The API authors chose to use protocols so that there is a defined set of methods that must be used in the code. This ensures that a consistent design is adhered to. They are defined in the `CsoundValueCacheable.h` file thus:

```
@class CsoundObj;  
  
@protocol CsoundValueCacheable <NSObject>  
  
-(void)setup:(CsoundObj*)csoundObj;  
-(void)updateValuesToCsound;  
-(void)updateValuesFromCsound;  
-(void)cleanup;
```

Each of these must then be implemented in the `ViewController.m` file. If it is unnecessary to implement one of these methods, it still *must* appear but the method body can be left blank, thus:

```
-(void)updateValuesFromCsound  
{  
  
    //No values coming from Csound to iOS  
  
}
```


3.2.3 Overview of Protocols

When writing the code which allows us to send values from iOS to Csound, it is important that the code conforms to the following protocol methods (Table 3.2):

Protocol methods	Action
-(void)setup:(CsoundObj*)CsoundObj	Set up the necessary channels and pointers to communicate with Csound.
-(void)updateValuesToCsound	Update the values being sent from iOS to Csound.
-(void)updateValuesFromCsound	Collect any values from Csound.
-(void)cleanup	Reset any values used in communication and de-allocate any memory used.
-(void)csoundObjDidStart:(CsoundObj*)csoundObj	This method is called when a Csound object is created. This allows developers to notify the user that Csound is running on iOS.
-(void)csoundObjComplete:(CsoundObj*)csoundObj	Much like the way the 'csoundObjDidStart' method works, this allows developers to notify the user that Csound has stopped running in iOS.

Table 3.2-Protocol methods which must be implemented in your
ViewController

3.3 Looking at the Csound '.csd' File

The following section provides an overview of the Csound editing environment, the structure of the .csd file, and how to include the .csd file into your Xcode project.

3.3.1 Downloading Csound

A Csound front-end editor, CsoundQt, can be used for editing the .csd file in the provided example project. It is advised to use CsoundQt with iOS because it is an ideal environment for developing and testing the Csound audio engine – error reports for debugging, the ability to run the Csound audio code on its own, and listen to its results. However, using CsoundQt is not essential to use Csound as an audio engine as Csound is a standalone language. CsoundQt is included in the Csound package download.

In order to use Csound in iOS, the latest version of Csound (*Version 5.19*) will need to be installed.

Csound 5.19 can be downloaded from the following link:

<http://sourceforge.net/projects/Csound/files/Csound5/Csound5.19/>

In order for Xcode to see the .csd file, it must be imported it into the Xcode project. This is done by right-clicking on the 'Supporting Files' folder in the project, and clicking on 'Add files to (*project name*)' (Figure 3.7).

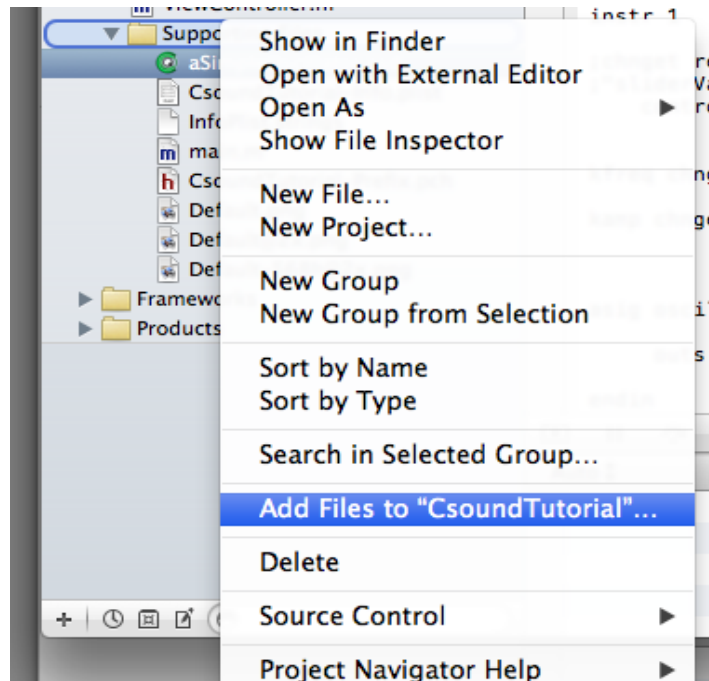


Figure 3.7-Adding the .csd to iOS Project

It is possible to edit the .csd file while also working in Xcode. This is done by right-clicking on the .csd file in Xcode, and clicking on 'Open With External Editor' (Figure 3.8).

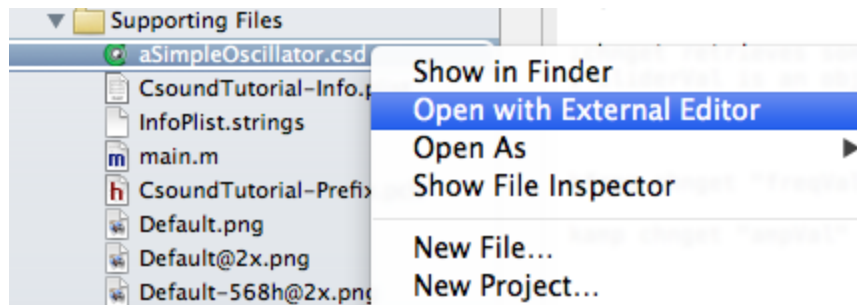


Figure 3.8-Opening the .csd file with an external editor

However, it is important to remember to save any changes to the .csd file before the Xcode project is recompiled.

3.3.2 The .csd File

When setting up a Csound project, it is important that various audio and performance settings configured correctly in the header section of the .csd file. These settings are described in Table 3.3, and are discussed in more detail in the Csound Manual.

Setting	Description
sr	Sample rate
kr	Control rate
ksmps	Number of samples in control period (sr/kr)
nchnls	Number of channels of audio output
0dbfs	Sets value of 0 decibels using full scale amplitude

Table 3.3-Csound .csd Settings

It is important that the sample rate for the Csound project be set to the same sample rate as the hardware it will be run on. For this project, make sure the sample rate set to 44100, as depicted in Figure 3.9. This is done by opening the Audio MIDI Setup, which is easily found on all *Mac* computers by searching in *Spotlight*.

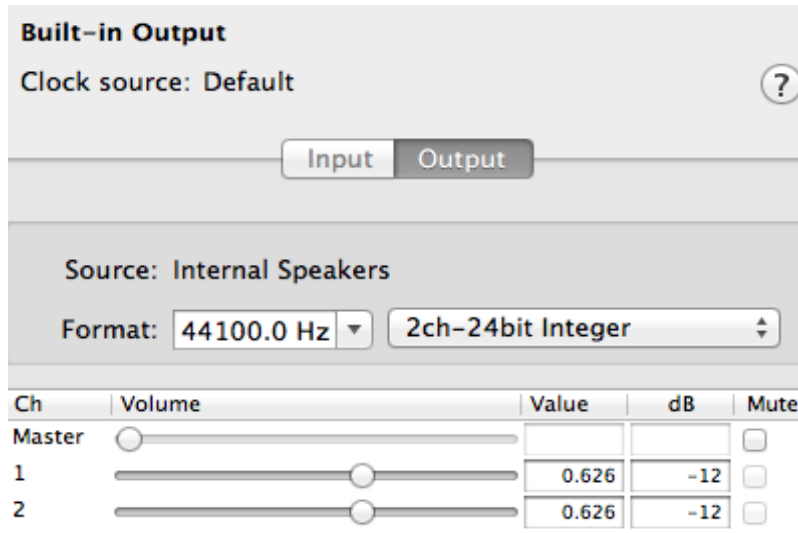


Figure 3.9-Configuring Audio Hardware Settings

3.3.3 Instruments

As mentioned previously, Csound instruments are defined in the orchestra section of the .csd file. The example project provided by the authors uses a simple oscillator that has two parameters: amplitude and frequency, both of which are controlled by UI sliders.

Figure 3.10 on the following page shows a block diagram of the synthesizer we are using in the example project.

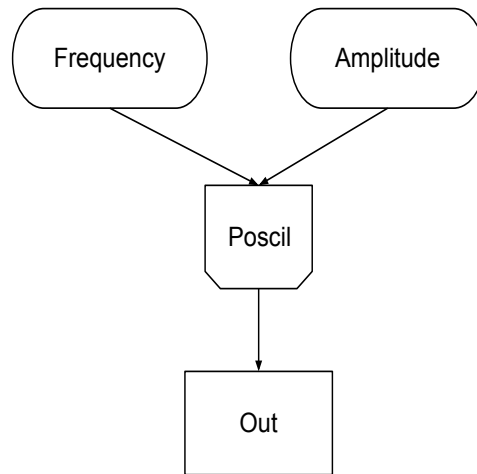


Figure 3.10-Block Diagram for .csd Instrument

3.3.4 Score

The score is the section of the .csd file which provides instruments with control instruction, for example pitch, volume, and duration. However, as the goal here is for users to be able to interact with the Csound audio engine in real-time, developers will most likely opt instead to send score information to Csound that is generated by UI elements in the Xcode project. Details of the instrument and score can be found in the comments of the *aSimpleOscillator.csd* file

Csound uses GEN (f-table generator) routines for a variety of functions. This project uses GEN10, which create composite waveforms by adding partials. At the start of the score section, a GEN routine is specified by function statements (also known as *f-statements*). The parameters are shown below in Table 3.4:

Parameter	Description
f 1	Unique f-table identification number
0	f-statement initialization time expressed in score beats
16384	f-table size
10	GEN routine called to create the f-table
1	strength of ascending partials

Table 3.4-Csound .csd F-Table Parameters

In a Csound score, the first three parameter fields (also known as p-fields) are reserved for the instrument number, the start time, and duration amount. P-fields 4 and 5 are conventionally reserved for amplitude and frequency, however, P-fields beyond 3 can be programmed as desired.

The p-fields used in the example project are shown in Table 3.5.

p-field	1	2	3	4	5
Parameter	Instrument Number	Start	Duration	Amplitude	Frequency

Table 3.5-Csound .csd P-field Parameters

In this project, the first three p-fields are used: the instrument number (i1), the start time (time = 0 seconds), and the duration (time = 1000 seconds). Amplitude and frequency are controlled by UI sliders in iOS.

4 Common Problems

This section is designed to document some common problems faced during the creation of this tutorial. It is hoped that by outlining these common errors, readers can debug some common errors they are likely to come across when creating applications using this API. It discusses each error, describes the cause and outlines a possible solution.

4.1 UIKnob.h is Not Found

This is a problem related to the API. The older versions of the API import a file in the examples that sketches a UIKnob in Core Graphics. This is not a part of the API, and should not be included in the project.

The file in question is a part of the examples library provided with the SDK. It is used in the file 'AudioIn test' and is used to sketch a radial knob on the screen. It gives a good insight into how the user can generate an interface to interact with the API.

Solution: Comment the line out, or download the latest version of the API.

4.2 Feedback from Microphone

This is generally caused by the sample rate of a .csd file being wrong.

Solution: Ensure that the system's sample rate is the same as in the .csd file. Going to the audio and MIDI set-up and checking the current output can find the computer's sample rate. See section 3.3.2 for more information.

4.3 Crackling Audio

There are numerous possible issues here, but the main cause of this happening is a CPU overload.

Solution: The best way to debug this problem is to look through the code and ensure that there are no memory intensive processes, especially in code that is getting used a lot. Problem areas include fast iterations (loops), and code where Csound is calling a variable. Functions such as *updateValuesToCsound* and *updateValuesFromCsound* are examples of frequently called functions.

An example: an NSLog in the *updateValuesToCsound* method can cause a problem. Say, the *ksmps* in the .csd is set to 64. This means that the Csound is calling for iOS to run the method *updateValuesToCsound* every 64 samples. Assuming the sample rate is 44.1k this means that this CPU intensive NSLog is being called ~689 times a second; very computationally expensive.

4.4 Crackling from amplitude slider

When manipulating the amplitude slider in iOS, a small amount of clicking is noticeable. This is due to the fact that there is no envelope-smoothing function applied to the amplitude changes. While this would be an improvement on the current implementation, however; it was felt that the current implementation would be more conducive to learning for the novice Csound user. This would be implemented by using a *port* opcode.

5 Csound Library Methods

This section will present and briefly describe the methods which are available in the Manual.

Name	Method Call	Description
startCsound	<code>-(void) startCsound: (NSString*) csdFilePath;</code>	Provides the location of the .csd file which is to be used with the Csound object.
	<code>-(void) startCsound: (NSString *) csdFilePath recordToURL: (NSURL *) outputURL;</code>	Provides the location of the .csd file which is to be used with the Csound object and specifies a URL to which it will record.
startCsoundToDisk	<code>-(void) startCsoundToDisk: (NSString*) csdFilePath outputFile: (NSString*) outputFile;</code>	Provides the location of the .csd file which is to be used with the Csound object and specifies a file to which it will record. This does not occur in realtime, but as fast as possible to the disk. This method is useful for batch rendering.
stopCsound	<code>-(void) stopCsound;</code>	This uses the Csound object's method 'stopCsound' to stop the instance of CsoundObj that it is called on.
muteCsound	<code>-(void) muteCsound;</code>	Mutes all instances of Csound
unmuteCsound	<code>-(void) unmuteCsound;</code>	Unmutes all instances of Csound

recordToURL	<code>-(void) recordToURL: (NSURL *)outputURL;</code>	Begins recording to a specified URL. This can be defined at a later point in the code, even after Csound has been started.
stopRecording	<code>-(void) stopRecording;</code>	Stops recording to URL

Table 5.1-Basic API Methods

5.2 UI and Hardware Methods

Table 5.2-UI and Hardware Methods

Name	Method Call	Description
addSwitch	<code>(id<CsoundValueCacheable>) addSwitch: (UISwitch*) uiSwitch forChannelName: (NSString*) channelName;</code>	Adds a switch to the Csound object. The method requires a switch which already exists as part of the user interface and a name for the channel which will provide information about this switch to the .csd file. For more information about channels of information between Xcode and Csound see section 5.
addSlider	<code>(id<CsoundValueCacheable>) addSlider: (UISlider*) uiSlider forChannelName: (NSString*) channelName;</code>	Adds a slider to the Csound Object. The method requires a slider and a channel name.
addButton	<code>(id<CsoundValueCacheable>) addButton: (UIButton*) uiButton forChannelName: (NSString*) channelName;</code>	Adds a button to the Csound Object. The method requires a button and a channel name.
enableAccelerometer	<code>(id<CsoundValueCacheable>) enableAccelerometer;</code>	Enables the accelerometer for use with the Csound object.
enableGyroscope	<code>(id<CsoundValueCacheable>) enableGyroscope;</code>	Enables the gyroscope for use with the Csound object.
enableAttitude	<code>(id<CsoundValueCacheable>) enableAttitude;</code>	Enables attitude to allow device motion to be usable with the Csound object.

5.3 Communicating between Xcode and Csound

Name	Method Call	Description
addValueCacheable	<code>-(void)addValueCacheable:(id<CsoundValueCacheable>)valueCacheable;</code>	Adds to a list of watched objects so that they can update every cycle of ksmps.
removeValueCacheable	<code>-(void)removeValueCacheable:(id<CsoundValueCacheable>)valueCacheable;</code>	Removes a cacheable value from the Csound Object.
sendScore	<code>-(void)sendScore:(NSString*)score;</code> Eg: <pre>[self.csound sendScore:[NSString stringWithFormat:@"i1 0 10 0.5 %d", myPitch,]];</pre> <p>(sends a score to instrument 1 that begins at 0 seconds, stops at 10 seconds, with amplitude 0.5 and a pitch of the objective-C variable 'myPitch').</p>	Sends a score as a string to the .csd file. See section 4 for formatting a Csound score line.

Table 5.3-API Communication Methods

5.4 Retrieve Csound-iOS Information

Name	Method Call	Description
getCsound	<code>-(CSOUND*) getCsound;</code>	Returns the C structure that the CsoundObj uses. This allows developers to use the Csound C API in conjunction with the Objective-C CsoundObj API.
getInputChannelPtr	<code>(float*) getInputChannelPtr: (NSString*) channelName;</code>	Returns the float of an input channel pointer.
getOutputChannelPtr	<code>(float*) getOutputChannelPtr: (NSString*) channelName;</code>	Returns the float of an output channel pointer.
getOutSamples	<code>-(NSData*) getOutSamples;</code>	Gets audio samples from Csound.
getNumChannels	<code>-(int) getNumChannels;</code>	Returns the number of channels in operation.
getKsmps	<code>-(int) getKsmps;</code>	Returns ksmps as defined in the .csd file.
setMessageCallback	<code>-(void) setMessageCallback: (SEL) method withListener: (id) listener;</code>	Sets up a method to be the callback method and a listener id.
performMessageCallback	<code>(void) performMessageCallback: (NSValue *) infoObj;</code>	Performs the message callback.

Table 5.4-Retrieve Csound-iOS Information Methods

6 Conclusions

This tutorial provided an overview of the Csound-iOS API, outlining its benefits, and describing its functionality by means of an example project. It provided the basic tools for using the API, equipping iOS developers to explore the potential of this API in their own time.

APIs such as this one, as well as others including *libpd* and *The Amazing Audio Engine* provide developers with the ability to integrate interactive audio into their apps, without having to deal with the low-level complexities of Core Audio.

6.1 Additional Resources

Upon completion of this tutorial, the authors suggest that the reader look at the original Csound for iOS example project, written by Steven Yi and Victor Lazzarini.

This is available for download from:

<http://sourceforge.net/projects/csound/files/csound5/iOS/>

About the Authors

The authors are Masters students at the University of York Audio Lab. Each one is working on a separate interactive audio app for the iPad, and has each been incorporating the Mobile Csound API for that purpose. They came together to write this tutorial to make other developers aware of the Mobile Csound API, and how to utilize it.

The motivation behind this tutorial was to create a step by step guide to using the Mobile Csound API. When the authors originally started to develop with the API, they found it difficult to emulate the results of the examples that were provided with the API download. As a result, the authors created a simple example using the API, and wanted others to learn from our methods and mistakes. The authors hope that this tutorial provides clarity in the use of the Mobile Csound API.

Appendix D *Test Script (Group A)*

Study: An Investigation into the Spatial Auditory Display of Graphical Data

Timothy Neate

GROUP A

Background

The author has designed a series of tests to examine newly developed methods of turning graphical data into spatial audio. Some applications have been developed for the iPad that allow a user to interact with an image, and get its sound representation as an output. The main aim of the project is to find the best image processing methods, sonification algorithms, and interaction techniques for allowing a user to find specific features on a display, with minimal visual cues.

Potential applications of the information gathered are as follows:

- Developing more immersive auditory experiences for users. For example, an alerting sound on a tablet computer emanating from a specific location on, or off, and the screen - complementing extended displays.
- Allowing for 'eyes free' technology, for those with their eyes on other tasks. For example, creating an interface for a tablet computer that provides auditory information to a construction worker who needs his eyes on the work at hand.
- Improving computer-based experience for the visually impaired. For example, allowing for the blind to locate specific features on the screen - with the binaural audio providing them with important location-based cues.

Study Outline

The aim of this study is to determine the effectiveness of the auditory display techniques that have been developed to transform graphical data into sound. A series of challenges will be set involving the detection of specific

features on the display of an iPad. You will be asked to interact with the iPad by touching the screen and trying to make sense of the sound.

The study will consist of three main sections:

- Demographics questionnaire
- Test 1 – a test that focuses on determining the effectiveness of auditory display methods when interacting with an image the size of the iPad screen.
- Test 2 – a test that focuses on determining how successful auditory display methods are when interacting with a bigger-than-display image that requires extra interaction to scroll around and navigate.

Consent Form – Informed Consent for Auditory Display Study

Who is running this study? – This study is being run by Timothy Neate who is an MSc by research student in the Audio Lab, at the Department of Electronics, University of York.

What will I have to do? – You will first have to complete a simple demographics form, then you will have to undertake a series of tests involving interacting with pictures on an iPad that produce an auditory output. The tests are not designed to assess you, but to test the system. The tests will be recorded, as they need to be evaluated at a later date.

Who will see this data? Though these tests are recorded, only your hands will be visible in the recording, by signing below you give permission for this footage to be used for analysis by the test co-ordinator, and by students who wish to continue the work in this project. Note that: your information, video, and performance will be anonymised to ensure your confidentiality. The forms used will not be distributed, and only the raw anonymised data will be accessible.

Do I have to do this? Your participation is completely voluntary. You can therefore withdraw from the study, and if requested, your data can be destroyed.

Can I ask a question? Do ask the test administrator if you have any questions, however, at some points there may be no answer to ensure there is no bias in the tests.

Please sign below if you agree to take part in the study under the conditions laid out above. This will indicate that you have read and understood the above and that we will be obliged to treat your data as described.

Name:

Signature:

Date:

Demographics questionnaire

This is a small questionnaire to gauge the participants' background to ensure a well encompassing data set. The information you provide here will be used to find further correlation in the data. All information gathered is anonymous and confidential.

1. What is your age? _____

2. What is your gender? (Male / Female)

3. What is your nationality? _____

4. Are you a student? (Yes/No)

If so, indicate your department or area of study:

5. How would you describe your experience with portable touch screen devices (E.g. an iPad/iPhone, Android Device)?

Tick your answer:

I own one

I have some experience using them but do not own one

I have a little experience using touch screen devices

I have never used a touch screen device

6. Are you a musician? (Yes/No)

If 'Yes', elaborate:

7. Do you associate certain timbres or pitches with colour (synaesthesia)?

Circle your answer: (Yes/ No)

If 'Yes', elaborate:

8. How would you describe your experience of binaural audio (sound created with the intention of creating 3D audio over headphones)?

Tick your answer:

I have listened to lots of binaural audio and/or have good knowledge about it

I have listened to binaural audio

I have never listened to binaural audio

9. You may now contact the **test administrator**, who will play you a short binaural extract over headphones. How effectively can you determine the actor's position?

Tick your answer:

I can tell where he is in the room at all times

I am unsure at some points, but for the most part I know where he is

I have no idea where he is – the binaural effects do not work.

Test 1

Test 1 will focus on determining how well the developed auditory display methods work – this will be done by seeing how participants perform when interacting with the display to find the location of simple shapes and colours. You will be asked to find features on the screen of the iPad, and you will be tasked with locating the features by means of feel and sound alone.

As soon as you touch the screen you will have real-time auditory feedback, you must then use this auditory feedback to try and locate the specific features on the screen. The sound produced will differ from test to test; however, each test will use at least two of the following sound mappings:

- **Pulse Train** – a pulsing sound is used to allow you to determine how far away from the image feature you are – the faster the pulsing of the sound, the closer to the image feature you are. The sound that pulses depends on the feature that you're detecting.
- **Binaural Panning** – the sound will be panned binaurally, this means that the sound you are listening for will appear to be coming from a particular direction, move your finger towards this point to locate the feature.
- **Volume** – as you get closer to the sound, the sound will become louder.
- **Alert** – when you have found the image feature, i.e. your finger is touching it; you will hear a high-pitched beeping sound. This means you have found the dot and you may indicate that you have found it to the test organizer.

An **example training application** (Example 1) has been provided so that you may familiarize yourself with the mappings. For this, you will not be blindfolded and will be allowed to look at the screen to gauge an idea of the mappings. *The experiment coordinator* will now talk you through each of the parameters, and

Test 1.1

[Visual Restriction]

For this test, the aim is to find a black dot on the screen by sound and feel alone. The algorithm is currently only looking for the colour black, and as there is a purely white background, you will only hear the black dot. Your aim is to try and find its location. When you believe you have found its position, indicate so by saying ‘found’, ‘got it’, or similar. The sound used for the pulse train is a burst of white noise; the same as in the training example.

Test 1.2

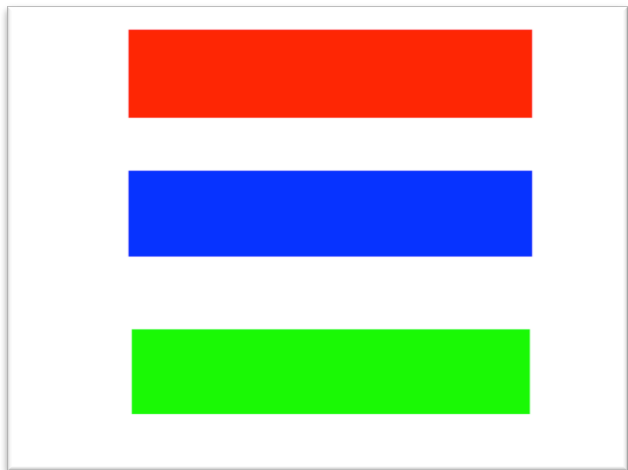
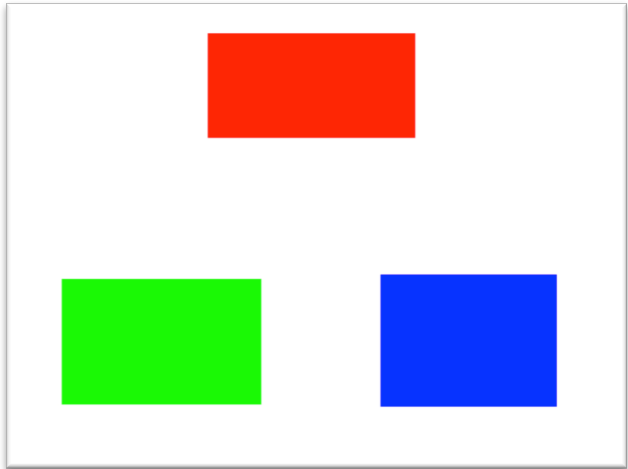
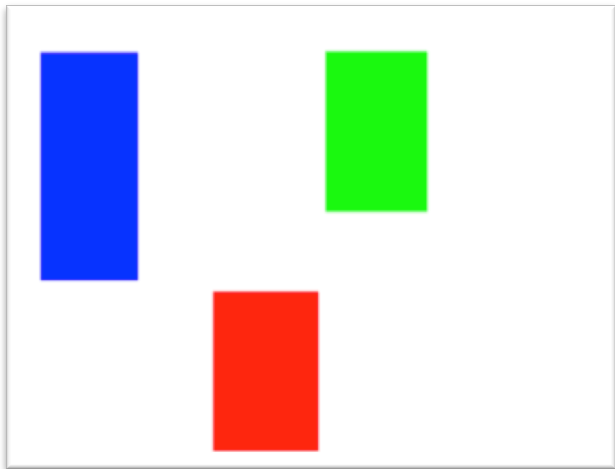
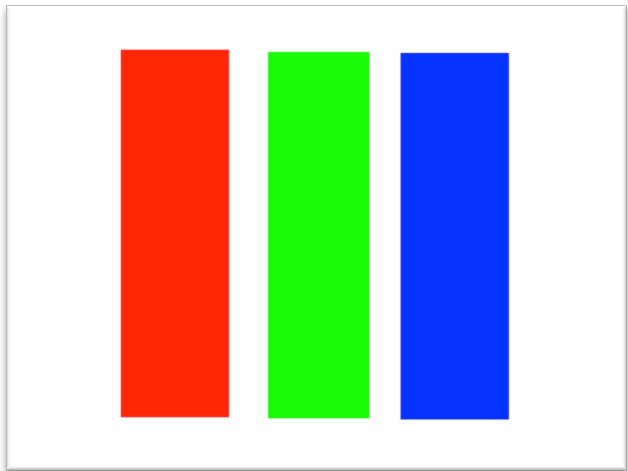
[Visual Restriction]

This experiment will be similar to Test 1.1; however, there will be three colours – red, green, and blue. The sound mappings are the same as before; however, there will be three sounds, each to signify a colour; a high pitched sound to indicate red, a medium pitched sound to indicate green, and a low pitched sound to indicate blue. The **test administrator** will now play an example of the **red sound**, the **green sound**, and the **blue sound**. Your task is to navigate towards these sounds using the various cues, and once you have found a colour (the beeping alert sound triggers), you must indicate to the test co-ordinator what colour you think you are touching – red, green, or blue? Once you have found one colour, go on to try and find the remaining colours in a similar fashion.

Test 1.3

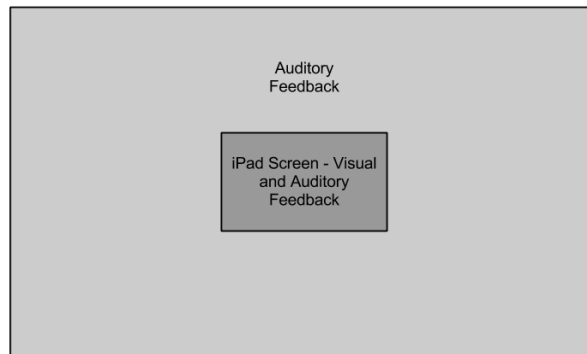
[Visual Restriction]

This test will examine the system’s ability to help users identify some minimalist pictures. You will be presented with an auditory representation of a picture and then choose the picture you believe to be its visual representation from a series of four pictures. You will have the opportunity to view the pictures beforehand, and afterwards to ensure you pick the one you believe it to be. The colours will only consist of the colours in Test 1.2, and will adhere to the same mappings; a high pitched sound to indicate red, a medium pitched sound to indicate green, and a low pitched sound to indicate blue. The pictures are on the next page. Please put a tick next to the one you believe to be the picture on the iPad screen.



Test 2

Test 2 will focus on determining how well the auditory display methods work when the image extends the iPad's display. You will be tasked with finding image features that are outside of the iPad's physical display, as shown below.



To find the image features, place your finger down on the screen to hear the auditory feedback, then use two or more fingers to scroll in the direction of the sound you hear. By doing this it is possible to search through a large image and stop scrolling when you believe you are close to the source of the sound. Then, it is possible to search the local area with the conventional one finger searching method, in a similar manner to Test 1.

If you become disoriented in the tests where you are allowed visual cues, it is possible to determine your location on the screen by looking at the scrollbars on the sides of the screen. Alternatively, if you have no visual cues an additional parameter has been added to determine when a user has scrolled beyond the bounds of the image:

Edge Hit 'Boing' – when the bounds of an image have been exceeded an oscillator will feedback that the user has exceeded the allocated scrollable size.

An **example training application** (Example 2) has been provided so that you may familiarize yourself with the mappings. For this, you will not be blindfolded and will be allowed to look at the screen to gauge an idea of the mappings. *The experiment co-ordinator* will now talk you through each of the parameters, and how they relate to the features

Test 2.1

[Visual Restriction]

Test 2.1 will test the effectiveness of the system's ability to portray large images with binaural feedback. The image you are tasked with exploring is three times the size of the iPad screen, and you must explore it by scrolling around it. Your task, much like in the example application, is to find the black dot. However, during this test you will not be able to see the display of the iPad. The image you are searching is three times the size of the iPad screen.

Test 2.2

[Visual Restriction]

Test 2.2 is similar to Test 2.1; however, in this task you are searching for three coloured dots – red, green, and blue. The sound mappings are the same as in Test 1.2 - a high pitched sound to symbolize red, a medium pitched sound to symbolize green, and a low pitched sound to symbolize blue. The image you are searching is three times the size of the iPad's display.

Test 2.3

[No Visual Restriction]

The following tests (2.3, 2.4 and 2.5) involve searching progressively larger images with both your ears and your eyes. In this test you are tasked with searching for a black dot, much like in Test 2.1. However, this time, you will be able to use your eyes – the visual restrictions will be removed. The image that you are searching is three times the size of the iPad display.

Test 2.4

[No Visual Restriction]

Test 2.4 is the same as Test 2.3; you must find the black dot with no visual restrictions. However, in this test, the image you are searching will be four times the size of the iPad display.

Test 2.5

[No Visual Restriction]

Test 2.5 is the same as Test 2.4; you must find the black dot with no visual restrictions. However, in this test, the image you are searching will be five times the size of the iPad's display.

Appendix E *Test Script (Group B)*

Study: An Investigation into the Spatial Auditory Display of Graphical Data

Timothy Neate

GROUP B

Background

The author has designed a series of tests to examine newly developed methods of turning graphical data into spatial audio. Some applications have been developed for the iPad that allow a user to interact with an image, and get its sound representation as an output. The main aim of the project is to find the best image processing methods, sonification algorithms, and interaction techniques for allowing a user to find specific features on a display, with minimal visual cues.

Potential applications of the information gathered are as follows:

- Developing more immersive auditory experiences for users. For example, an alerting sound on a tablet computer emanating from a specific location on, or off, and the screen - complementing extended displays.
- Allowing for 'eyes free' technology, for those with their eyes on other tasks. For example, creating an interface for a tablet computer that provides auditory information to a construction worker who needs his eyes on the work at hand.
- Improving computer-based experience for the visually impaired. For example, allowing for the blind to locate specific features on the screen - with the binaural audio providing them with important location-based cues.

Study Outline

The aim of this study is to determine the effectiveness of the auditory display techniques that have been developed to transform graphical data into sound. A series of challenges will be set involving the detection of specific features on the display of an iPad. You will be asked to interact with the iPad by touching the screen and trying to make sense of the sound.

The study will consist of three main sections:

- Demographics questionnaire
- Test 1 – a test that focuses on determining the effectiveness of auditory display methods when interacting with an image the size of the iPad screen.
- Test 2 – a test that focuses on determining how successful auditory display methods are when interacting with a bigger-than-display image that requires extra interaction to scroll around and navigate.

Consent Form – Informed Consent for Auditory Display Study

Who is running this study? – This study is being run by Timothy Neate, who is an MSc by research student in the Audio Lab, Department of Electronics, University of York.

What will I have to do? – You will first have to complete a simple demographics form, then you will have to undertake a series of tests involving interacting with pictures on an iPad that produce an auditory output. The tests are not designed to assess you, but to test the system. The tests will be recorded, as they need to be evaluated at a later date.

Who will see this data? Though these tests are recorded, only your hands will be visible in the recording, by signing below you give permission for this footage to be used for analysis by the test co-ordinator, and by students who wish to continue the work in this project. Note that: your information, video, and performance will be anonymised to ensure your confidentiality. The forms used will not be distributed, and only the raw anonymised data will be accessible.

Do I have to do this? Your participation is completely voluntary. You can therefore withdraw from the study, and if requested, your data can be destroyed.

Can I ask a question? Do ask the test administrator if you have any questions, however, at some points there may be no answer to ensure there is no bias in the tests.

Please sign below if you agree to take part in the study under the conditions laid out above. This will indicate that you have read and understood the above and that we will be obliged to treat your data as described.

Name:

Signature:

Date:

Demographics questionnaire

This is a small questionnaire to gauge the participants' background to ensure a well encompassing data set. The information you provide here will be used to find further correlation in the data. All information gathered is anonymous and confidential.

1. What is your age? _____

2. What is your gender? (Male / Female)

3. What is your nationality? _____

4. Are you a student? (Yes/No)

If so, indicate your department or area of study:

5. How would you describe your experience with portable touch screen devices (E.g. an iPad/iPhone, Android Device)?

Tick your answer:

I own one

I have some experience using them but do not own one

I have a little experience using touch screen devices

I have never used a touch screen device

6. Are you a musician? (Yes/No)

If 'Yes', elaborate:

7. Do you associate certain timbres or pitches with colour (synaesthesia)?

Circle your answer: (Yes/ No)

If 'Yes', elaborate:

8. How would you describe your experience of binaural audio (sound created with the intention of creating 3D audio over headphones)?

Tick your answer:

I have listened to lots of binaural audio and/or have good knowledge about it

I have listened to binaural audio

I have never listened to binaural audio

9. You may now contact the **test administrator**, who will play you a short binaural extract over headphones. How effectively can you determine the actor's position?

Tick your answer:

- I can tell where he is in the room at all times
- I am unsure at some points, but for the most part I know where he is
- I have no idea where he is – the binaural effects do not work.

Test 1

Test 1 will focus on determining how well the developed auditory display methods work – this will be done by seeing how participants perform when interacting with the display to find the location of simple shapes and colours. You will be asked to find features on the screen on the iPad, and you will be tasked with locating the features by means of feel and sound alone.

As soon as you touch the screen you will have real-time auditory feedback, you must then use this auditory feedback to try and locate the specific features on the screen. The sound produced will differ from test to test; however, each test will use at least two of the following four sound mappings:

- **Pulse Train** – a pulsing sound is used to allow you to determine how far away from the image feature you are – the faster the pulsing of the sound, the closer to the image feature you are. The sound that pulses depends on the feature that you're detecting.
- **Binaural Panning** – the sound will be panned binaurally, this means that the sound you are listening for will appear to be coming from a particular direction, move your finger towards this point to locate the feature.
- **Volume** – as you get closer to the sound, the sound will become louder.
- **Alert** – when you have found the image feature, i.e. your finger is touching it; you will hear a high pitched beeping sound. This means you have found the dot and may indicate that you have found it to the test organizer.

An **example training application** (Example 1) has been provided so that you may familiarize yourself with the mappings. For this, you will not be blindfolded and will be allowed to look at the screen to gauge an idea of the mappings. *The experiment co-ordinator* will now talk you through each of the parameters, and how they relate to the features in

Test 1.1

[Visual Restriction]

For this test, the aim is to find a black dot on the screen by sound and feel alone. The algorithm is currently only looking for the colour black, and as there is a purely white background, therefore you will only hear the black dot. Your aim is to try and find its location. When you believe you have found its position, indicate so by saying ‘found’, ‘got it’, or similar.

The sound used for the pulse train is a burst of white noise; the same as in the training example, and the closer you get to the dot the faster its rate. There is no binaural audio this test, so you will not have an impression of where the dot is, only an impression of how far you are away from it.

Test 1.2

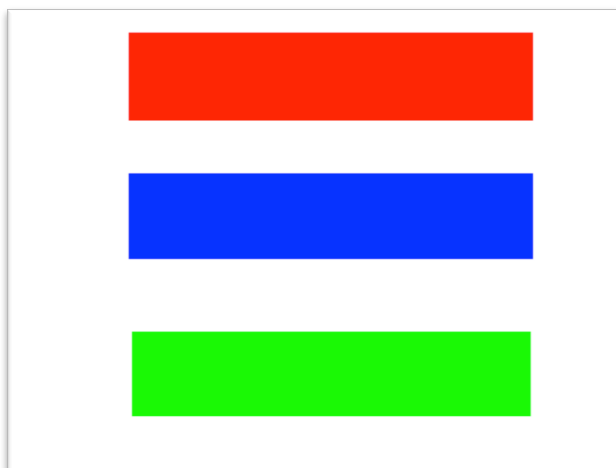
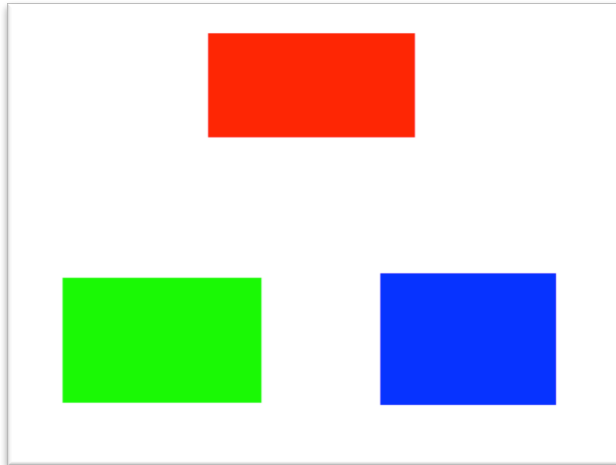
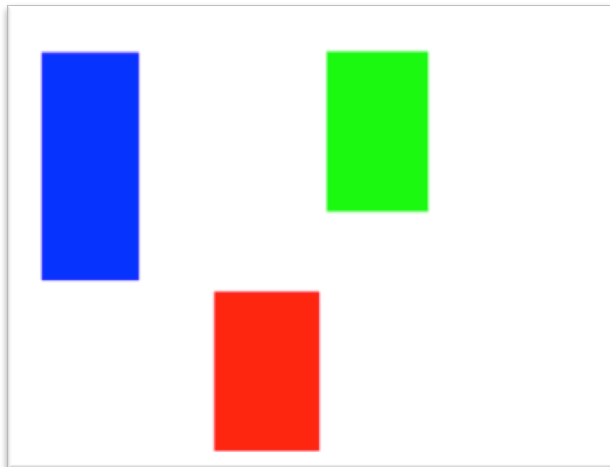
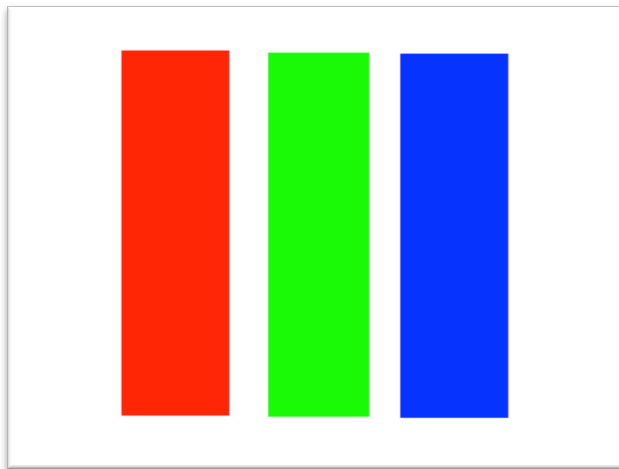
[Visual Restriction]

This experiment will be similar to Test 1.1; however, there will be three colours – red, green, and blue. The sound mappings are the same before; however, there will be three sounds pulsing, each to signify the colours red, green, and blue. Your task is to navigate towards these sounds using the various cues, and once you have found a colour (the beeping alert sound triggers), you must indicate to the test co-ordinator what colour you think you are touching from the sound alone – red, green, or blue. Then move on to identify the remaining colours.

Test 1.3

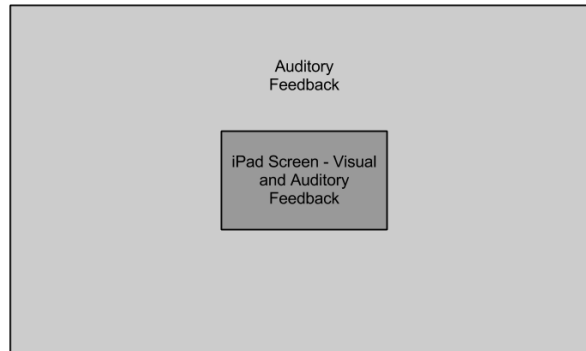
[Visual Restriction]

This test will examine the system’s ability to help users identify some minimalist pictures. You will be presented with an auditory representation of a picture and then choose the picture you believe to be its visual representation from a series of four pictures. You will have the opportunity to view the pictures beforehand, and afterwards to ensure you pick the one you believe it to be. The colours will only consist of the colours in Test 1.2, and will adhere to the same mappings. The test coordinator will now tell you the mappings, and play some **sound examples**. The pictures are on the next page. Please put a tick next to the one you believe to be the picture on the iPad screen.



Test 2

Test 2 will focus on determining how well the auditory display methods work when the image extends the iPad's display. You will be tasked with finding image features that are outside of the iPad's physical display, as shown below.



To find the image features, place your finger down on the screen to hear the auditory feedback, then used two or more fingers to scroll in the direction of the sound you hear. By doing this it is possible to search through a large image and stop scrolling when you believe you are close to the source of the sound. Then, it is possible to search the local area with the conventional one finger searching method, in a similar manner to Test 1.

If you become disoriented in the tests where you are allowed visual cues, it is possible to determine your location on the screen by looking at the scrollbars on the sides of the screen. Alternatively, if you have no visual cues an additional parameter has been added to determine when a user has scrolled beyond the bounds of the image:

Edge Hit 'Boing' – when the bounds of an image have been exceeded an oscillator will feedback that the user has exceeded the allocated scrollable size.

An **example training application** (Example 2) has been provided so that you may familiarize yourself with the mappings. For this, you will not be blindfolded and will be allowed to look at the screen to gauge an idea of the mappings. *The experiment co-ordinator* will now talk you through each of the parameters, and how they relate to the features in the image, as well as

Test 2.1

[Visual Restriction]

Test 2.1 will test the effectiveness of the system's ability to portray large images without binaural feedback. The image you are tasked with exploring is three times the size of the iPad screen, and you must explore it by scrolling around it. Your task, much like in the example application, is to find the black dot. However, during this test you will not be able to see the display of the iPad. The image you are searching is three times the size of the iPad screen.

Test 2.2

[Visual Restriction]

Test 2.2 is similar to Test 2.1; however, in this task you are searching for three coloured dots – red, green, and blue. Its aim is to determine how effective the system is without binaural feedback. The sound mappings are as follows - a high pitched sound to symbolize red, a medium pitched sound to symbolize green, and a low pitched sound to symbolize blue. The image you are searching is three times the size of the iPad's display.

Test 2.3

[No Visual Restriction] [No Auditory Feedback]

The following tests (2.3, 2.4 and 2.5) involve searching progressively larger images with both your ears and your eyes. In this test you are tasked with searching for a black dot, much like in Test 2.1. However, this time, you can only use your eyes – the applications will no longer have auditory feedback. The image that you are searching is three times the size of the iPad display

Test 2.4

[No Visual Restriction] [No Auditory Feedback]

Test 2.4 is the same as Test 2.3; you must find the black dot with no visual restrictions. However, in this test the image, you are searching will be four times the size of the iPad display. This application has no auditory feedback, so you must search with your eyes alone.

Test 2.5

[No Visual Restriction] [No Auditory Feedback]

Test 2.5 is the same as Test 2.4; you must find the black dot with no visual restrictions. However, in this test the image, you are searching will be five times the size of the iPad's display. This application uses no auditory feedback, so you must search with your eyes alone.

Appendix F *Test Coordinator's Script*

Study: An Investigation into the Spatial Auditory Display of Graphical Data

Timothy Neate

TEST CO-ORDINATOR SCRIPT

Overview

This is the test co-ordinator's script. It will be used to ensure that the test procedure flows as smoothly and consistently as possible.

Introducing the Participant

- Sit down participant and ask them to read through the background, the study outline, and the consent form.
- Let them know that if they have any questions during this part, or any other part of the test, feel free to ask. However, that the test co-ordinator will not be able to answer all questions. Do not answer any questions that may give any participant an advantage, or a disadvantage.
- Ask them to sign the consent form if they agree to its contents, and then ask them to fill out the questionnaire. Note that when they get to '9', let the test co-ordinator know so that a **sound clip** may be played over headphones. This will be played on the iPad through the participant's headphones.

Test 1

- Ask them to read 'Test 1' up to the box, and then allow them to look at Example 1 freely – ensuring that they have understood the parameters.
- **Begin recording**, and ask the participant to complete Test 1.1.

- Ask the participant to read through Test 1.2. If they are in **Group A** play them the audio examples off the iPad, and tell them how they relate related to each colour. If they are in **Group B**, do not.
- Ask the participant to complete Test 1.2, and then set up app for next test.
- Ask them to read through and complete Test 1.3.

Test 2

- Ask the participant to read through Test 2, and stop when they reach the box.
- Ask the participant to follow the instructions in the box – allowing them to see the device.
- Then give a short demonstration of the **technique** associated with extended display scrolling. Also, note that they should search **horizontally, then vertically** to improve efficiency.
- Ask them to read Test 2.1, and complete it.
- Ask them to read Test 2.2, and complete it.
- Remove the iPad from the black box such that they can see it fully.
- Ask them to read and complete Test 2.3, 2.4, and 2.5, noting the long loading times.
- Notify the participant that they have finished the test, and if they wish they may make any comments with regards to their performance.
- Turn off recording equipment and distribute chocolate.

References

- [Gra, 2007] (2007). Kemar - manikin type 45ba. Technical report, G.R.A.S Sound and Vibration.
- [Adamson and Avila, 2012] Adamson, C. and Avila, K. (2012). *Learning Core Audio*. Pearson, Upper Saddle River, 1st edition.
- [Amedi et al., 2007] Amedi, A., Stern, W., Camprodon, J., Bermpohl, F., Merabet, L., Rotman, S., Hemond, C., Meijer, P., and Pascual-Leone, A. (2007). Shape conveyed by visual-to-auditory substitution activates the lateral occipital complex. *Nature Neuroscience*, 10(6):687–689.
- [Angus and Howard, 2006] Angus, J. and Howard, D. (2006). *Acoustics and Psychoacoustics*. Focal Press, Oxford.
- [Arnott et al., 2013] Arnott, S., Thaler, L., Milne, J., Kish, D., and Goodale, M. (2013). Shape-specific activation of occipital cortex in an early blind echolocation expert. *Neuropsychologia*.
- [Atkins-Wakefield, 2012] Atkins-Wakefield, J. (2012). Research into novel interfaces for sonification on the ipad. Technical report, York.
- [Begault and Wenzel, 2000] Begault, D. and Wenzel, E. (2000). Direct comparison of the impact of head tracking, reverberation, and individualized head-related transfer functions on the spatial perception of a virtual speech source, Berlin. *AES*, 49(10):904–916.
- [Ben-Tal et al., 2001] Ben-Tal, O., Daniels, M., and Berger, J. (2001). De natura sonoris: Sonification of complex data. In *in Mathematics and Simulation with Biological, Economical, and Musicoacoustical Applications*, C.E. Dattellis, V.V. Kluev, and N.E. Mastorakis, Eds, page 330. WSES Press.
- [Berger and Grond, 2011] Berger and Grond (2011). *Parameter Mapping Sonification*, pages 363–379. Logos Verlag Berlin, Berlin.

REFERENCES

- [Berk et al., 1982] Berk, T., Brownston, L., and Kaufman, A. (1982). A human factors study of color notation systems for computer graphics. *Communications of the ACM*, 25(8):547–550.
- [Blattner et al., 1989] Blattner, M. M., Sumikawa, D. A., and Greenberg, R. M. (1989). Earcons and icons: Their structure and common design principals. *Human-Computer Interaction*, 4:11–44.
- [Bly, 1984] Bly, S. (1984). Communicating with sound. *Proceedings of the CHI conference on Human Factors in Computer Systems*, pages 115–119.
- [Bologna et al., 2009] Bologna, G., Deville, B., and Pun, T. (2009). Blind navigation along a sinuous path by means of the see color interface. *International Work-Conference on the Interplay Between Natural and Artificial Computation*, Berlin.
- [Bologna et al., 2010] Bologna, G., Deville, B., and Pun, T. (2010). Sonification of color and depth in a mobility aid for blind people. *International Conference for Auditory Display*, 16:9–13.
- [Bologna et al., 2007] Bologna, G., Deville, B., and Thierry, P. (2007). Transforming 3d coloured pixels into musical instrument notes for vision substitution. *Journal of Image and Video Processing*, page 14.
- [Bologna et al., 2008] Bologna, G., Deville, B., Thierry, P., and Vinckenbosch, M. (2008). A perceptual interface for vision substitution in a color matching experiment. *Joint Conference for Neural Networks - IEEE World Congress on Computational Intelligence*, pages 1–6.
- [Boulanger, 2001] Boulanger, R. (2001). *The Csound Book*. Massachusetts Institute of Technology, Massachusetts, 2nd edition.
- [Boulanger, 2012] Boulanger, R. (2012). Boulanger labs.
- [Brinkmann, 2012] Brinkmann, P. (2012). Circle of fifths.

REFERENCES

- [Brückner et al., 2012] Brückner, H.-P., Wielage, M., and Blume, H. (2012). Intuitive and interactive movement sonification on a risc/dsp platform. Atlanta, ICAD.
- [Butterfield, 2005] Butterfield, T. (2005). Improving the detection of cancer by using sonification to supplement visual displays. Technical report, The University of York, York.
- [Buxton et al., 1994] Buxton, Glaver, and Bye (1994). *Auditory Icons*, pages 6.1 – 6.11.
- [Collomosse, 2008] Collomosse, J. (2008). *The Fundamentals of Image Processing*. University of Bath, Bath, 1st edition.
- [Dalrymple and Knaster, 2009] Dalrymple, M. and Knaster, S. (2009). *Learn Objective-C on the Mac*. App Press, New York, 1st edition.
- [Darwin, 1859] Darwin, C. (1859). *Chapter IV - Natural Selection*. John Murray, Kent.
- [Dombois and Eckel, 2011] Dombois, F. and Eckel, G. (2011). *Audification*, pages 301–324. Logos Verlag, Berlin.
- [Edwards et al., 2008] Edwards, A., Hines, G., and Hunt, A. (2008). Segmentation of biological cells for sonification. Technical report, The University of York, York.
- [Edwards et al., 2010] Edwards, A., Hunt, A., Stammers, J., Podvoiskis, A., and Roseblade, R. (2010). Sonification strategies for examination of biological cells. *ICAD*, 16:193–200.
- [Eldridge, 2005] Eldridge, A. (2005). Issues in auditory display. Technical report, Brighton.
- [Fernström et al., 2004] Fernström, M., Bannon, L., and Brazil, E. (2004). An investigation of soft-button widgets using sound. *Le Journe de Design Sonore*, pages 1–11.

REFERENCES

- [Fetoni et al., 2011] Fetoni, A. R., Picciotti, P. M., Paludetti, G., and Troiani, D. (2011). Pathogenesis of presbycusis in animal models: A review. *Experimental Gerontology*, 46(6):413 – 425.
- [Gibson, 1961] Gibson, J. (1961). *The Accuracy of Form Discrimination with Three Types of Tactual Input: Passive, Moving and Active*. Cornell University, Ithaca.
- [Gonzlez, 2010] Gonzlez, P. C. (2010). What does badness sound like? Technical report, The University of York, York.
- [Hepper, 2013] Hepper, S. (2013). barefoot-coders.
- [Hermann, 2006] Hermann, T. (2006). Beyond the gui - sound and sonification examples for auditory interfaces. Technical report.
- [Hermann, 2011] Hermann, T. (2011). *Model Based Sonification*, page 403. Logos Verlag Berlin, Berlin.
- [Hermann and Hunt, 2004] Hermann, T. and Hunt, A. (2004). The discipline of interactive sonification. *ICAD*, Sydney.
- [Hermann and Hunt, 2011] Hermann, T. and Hunt, A. (2011). *The Sonification Handbook*. Logos Verlag Berlin GmbH, Berlin.
- [Heuten et al., 2006] Heuten, W., Wichmann, D., and Boll, S. (2006). Interactive 3d sonification for the exploration of city maps. Oslo, NordiCHI.
- [Hines, 2007] Hines, G. (2007). Sonification for cervical cancer screening. Technical report, The University of York, York.
- [Howard, 2005] Howard, D. (2005). *The Penguin Dictionary of Electronics*. Penguin Reference.
- [Howell, 2011] Howell, D. C. (2011). *Fundamental statistics for the behavioural sciences*. Wadsworth Cengage Learning.
- [Hunt et al., 2011] Hunt, A., Neuhoff, J., and Hermann, T. (2011). *Interactive Sonification*, pages 273–280. Logos Verlag Berlin, Berlin.

REFERENCES

- [Hunt and Pauletto, 2004] Hunt, A. and Pauletto, S. (2004). Interactive sonification in two domains: helicopter flight analysis and physiotherapy movement analysis. *Proceedings of the int. workshop on interactive sonification*, 1(1):1–7.
- [Hutchinson, 2012] Hutchinson, N. (2012). Colour music in austrlia: Applied mathematics.
- [iZotope inc., 2012] iZotope inc. (2012). izotope audio programming guide. Technical report.
- [Katz and Parseihian, 2011] Katz, B. and Parseihian, G. (2011). Perceptually based head-related transfer function database optimization. *Acoustical Society of America*, 131:99–105.
- [Kaufman, 2012] Kaufman, J. (2012). Android sound synthesis and latency.
- [Krebbber et al., 1999] Krebber, W., Gierlich, H.-W., and Genuit, K. (1999). Auditory virtual environments: basics and applications for interactive simulations. *Signal Processing*, 80:2308 – 2322.
- [Lazzarini, 2012] Lazzarini (2012). The audio programming blog.
- [Lee, 2006] Lee, P. (2006). Image (pre)-processing in detecting cervical neoplasia (cervical cancer) for sonification with the use of image feature extraction. Technical report, The University of York, York.
- [McGee, 2009] McGee, R. (2009). Auditory displays and sonification: Introduction and overview. Technical report, Santa Barbara.
- [McGookin and Brewster, 2001] McGookin, D. and Brewster, S. (2001). Fishhears - the design of a multimodal focus and context system. Technical report, Glasgow.
- [Meijer, 1992] Meijer, P. (1992). An experimental system for auditory image representations. *IEEE Transactions on Biomedical Engineering*, 39(2):112–121.

REFERENCES

- [Meijer, 2013] Meijer, P. (2013). What blind users say about the voice (seeing with sound website).
- [Merabet et al., 2008] Merabet, L., Poggel, D., Stern, W., Bhatt, E., Hemond, C., Maguire, S., Peter, M., and Pascual-Leone, A. (2008). Activation of visual cortex using crossmodal retinotopic mapping. Melbourne.
- [Michal Bujacz and Strumillo, 2012] Michal Bujacz, P. S. and Strumillo, P. (2012). Naviton: A prototype mobility aid for auditory presentation of three-dimensional scenes to the visually impaired. *AES*, 60:696 – 708.
- [Miller, 2004] Miller, M. (2004). The history of surround sound.
- [Newton, 1730] Newton, I. (1730). *Optiks: or, a treatise of the reflections, refractions, inflections and colours of light*. London.
- [Orr, 2012] Orr, G. (2012). Understanding color.
- [Papetti et al., 2008] Papetti, S., Devallez, D., and Fontana, F. (2008). Depthrow: A physics based audio game. Paris, ICAD.
- [Patel, 2006] Patel, S. B. (2006). *Nuclear Physics, an introduction*. New Age Publishers, New Delhi.
- [Pauletto and Hunt, 2009] Pauletto, S. and Hunt, A. (2009). The sonification of complex data. *Internation Journal for Computer Studies*, 67(11):923–933.
- [Petrolati, 2013] Petrolati, A. (2013). Density pulsaret.
- [Podvoiskis, 2004] Podvoiskis, A. (2004). *MEng Project Report: Improving the Efficiency of Cervical Cell Sample Analysis*. University of York, York.
- [Pulkki, 1997] Pulkki, V. (1997). Virtual sound source positioning using vector base amplitude panning. *AES*, 45:456–466.
- [Pulkki, 1999] Pulkki, V. (1999). Uniform spreading of amplitude panned virtual sources. New York, IEEE.

REFERENCES

- [Rossi et al., 2009] Rossi, J., Perales, F., Varona, J., and Roca, M. (2009). Col.diesis: Transforming colour into melody and implementing the result in a colour sensor device. Technical report, Barcelona.
- [Rumsey, 2001] Rumsey, F. (2001). *Spatial Audio*. Focal Press, Oxford.
- [Sanchez, 2010] Sanchez, J. (2010). Identifying and communicating 2d shapes using auditory display. *ICAD*, 16:89–95.
- [Sandberg and Håkansson, 2006] Sandberg and Håkansson, C. (2006). Using 3d audio guidance to locate indoor static objects. Technical report, Göteborg.
- [Schaffert and Mattes, 2012] Schaffert, N. and Mattes, K. (2012). Acoustic feedback training in adaptive rowing. *International Conference on Auditory Display*, 18:83–88.
- [Shinohara and Tenenber, 2009] Shinohara, K. and Tenenber, J. (2009). A blind person’s interactions with technology. *Communications of the ACM*, 52(8):58–66.
- [Shirley, 2002] Shirley, P. (2002). *Fundamentals of Computer Graphics*. A K Peters, Natick, 10th edition.
- [Spence and Apperley, 2013] Spence, R. and Apperley, M. (2013). *Bifocal Display*. Aarhus, 2 edition.
- [Stammers, 2006] Stammers, J. (2006). Developing synthesis techniques for the sonification of precancerous cells. Technical report, The University of York, York.
- [Trimble, 2009] Trimble, E. (2009). Pap test fact sheet.
- [Tyson, 2012] Tyson, M. (2012). Audio bus.
- [Valentine, 1930] Valentine, C. (1930). The inner bases of fear. *Pedagogical Seminary and Journal of Genetic Psychology*, 37(1):388–389.

REFERENCES

- [van den Doel et al., 2004] van den Doel, A., Smilek, D., Bodnar, A., Chita, C., Corbett, R., Nekrasovski, D., and McGrenere, J. (2004). Geometric shape detection with soundview. Sydney, ICAD.
- [van den Doel, 2003] van den Doel, K. (2003). Soundview: Sensing color images by kinesthetic audio. Boston, ICAD.
- [Vogt et al., 2009] Vogt, K., Pirr, D., Kobenz, I., Hldrich, and Eckel, G. (2009). Physiosonic: Movement sonification as auditory feedback. Copenhagen, ICAD.
- [Walker and Nance, 2006] Walker, B. and Nance, A. (2006). Spearcons: Speech-based earcons improve navigation performance in auditory terms. *International Conference on Auditory Display*, (12):63–68.
- [Walker and Nees, 2011] Walker, B. and Nees, M. (2011). *Theory of Sonification*, page 9. Logos Verlag, Berlin.
- [Walker et al., 2006] Walker, B. N., Nance, A., and rey Lindsay, J. (2006). Spearcons: Speech based earcons improve navigation in auditory menus. *ICAD*, 12(1):64–68.
- [Woodworth and Schlosberg, 1962] Woodworth, R. and Schlosberg, G. (1962). *Experimental Psychology*.
- [Yoshida et al., 2011] Yoshida, T., Kitani, K., Koike, H., Belongie, S., and Scheli, K. (2011). Edgesonic: Image feature sonification for the visually impaired. *Augmented Human Conference*, 2:Article 11.