2019

Assessment of Access Methods for Mobile Maps for Individuals Who are Blind or Visually Impaired

David Parker
Virginia Commonwealth University

Assessment of Access Methods for Mobile Maps for Individuals Who are Blind or Visually Impaired

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University

by

David Stephen Parker
Bachelor of Science, 2014
Department of Biomedical Engineering
Virginia Commonwealth University

Director: Dr. Dianne Pawluk
Associate Professor
Department of Biomedical Engineering

Virginia Commonwealth University
Richmond, Virginia
August 2019

Acknowledgment

The author withes to thank several people. First, I would like to thank Dr. Pawluk for all her help and guidance along the way in finishing my thesis. I would also like to thank my family not only for their support, but for the patience they have shown me along the way. Last but not least, I would like to say thank you Mom for always being there for me whenever I have needed you.

Table of Contents

List of Figures

List of Tables

Abstract

ASSESSMENT OF AN AUDITORY GUIDE OF LEWIS GINTER BOTANICAL GARDENS FOR INDIVIDUALS WHO ARE BLIND OR VISUALLY IMPAIRED

By David S. Parker, M.S.

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

Virginia Commonwealth University, 2019.

Director: Dr. Dianne Pawluk, Associate Professor, Biomedical Engineering

When people go to a mall, museums, or other such locations they tend to rely on maps to find their way around. However, for people who are blind or visually impaired (BVI) maps are not easily accessible and they depend on other means, such as a guide, to get around. Research has only just begun to investigate providing maps for people who are BVI on touch screen devices. Many different types of feedback have been used: audio (sound), tactile (touch), audio-tactile, and multitouch. Some research has been conducted on the benefit of using multiple fingers (multitouch) and has found conflicting results. Yet, no known research has been conducted on the comparison of using audio feedback to that of tactile feedback.

In this study, we look to try and answer two questions. 1.) Is audio equal to or better than tactile? As well as: 2.) Does multiple fingers help? Participants were asked to use seven different methods (4 audio, 3 tactile) to explore an overview map and an individual map and answer questions about them. Results showed that overall, audio cues

are similar or better than tactile cues which is beneficial since it requires less battery to generate audio cues than tactile cues. It was also shown that the use of multiple fingers was more beneficial in tasks that are spatially demanding. While those who have tactile experience benefited when using two fingers with each finger represented by a different instrument played to separated ears.

1. Introduction

When people go out to places such as shopping malls, museums, gardens, they tend to rely heavily on visual information to gather what is around them and where they want to go. Much of this information is often relayed using a map, either physical or on a phone. With the use of this tool they can determine places that are of interest to them and how they will get there. For many of these places, there are often two types of maps that are often used: an overview map of the general organization (overall map) and more detailed maps of individual areas (individual maps).

The overall map, is used to represent several different areas within a given location. For example, a shopping mall may be broken up into four different wings: north, south, east and west. This map would show people each of the wings and contain a listing of which stores are in each wing. Each wing may be in a different color to help people recognize the different sections, as well as a sign or label. With this, people can gather the necessary information to determine where they are and where they would like to go.

The other type of map that is used is the individual map, which provides detailed information about a specific area. In the case of the mall, it would show the location of the stores, bathrooms, elevators, and other such things pertaining to a given wing. Different stores may be indicated by a number or letter, while other things like bathrooms, stairs, escalators may have their own unique symbol. People can then use this map to help them locate specific stores or other point of interests.

These maps are very helpful in providing people with spatial awareness of what is around them. It allows them to explore the place and be spontaneous as to what and

when they do things. However, the maps are typically only provided in a visually accessible form, either on paper or on a phone. This presents a significant problem for individuals who are blind or visually impaired (BVI). Sometimes physical maps that include braille and textures are available to touch at the information center: however, the BVI user needs to know that such a map is available and where/how to find it. Users would also not be able to take the map with them. Tactile maps can be created on specialized paper, but these are bulky and awkward to carry. Again, someone would also need to create the map for the BVI user. For our mall example, potentially listing the stores in each of the sections of the mall may help but only to a very limited extent.

Currently, people who are BVI must rely on other methods for navigating inside a place. One of the most common ways is using a guide. This is a person that travels with them, guiding them from one place to another. If there is a specific place they want to go, then the guide leads them to that location directly. If they are unsure of where they would like to go, the guide may provide a verbal list of things that are around them as they lead them through the area and then Allow them to make decisions of where they would like to go as they walk. However, this method requires a guide to be available, limiting the independence of people who are BVI.

Another method that may be used by people who are BVI, especially if they were to visit that location multiple times, is the use of step by step navigation commands. The navigation commands would be provided either orally if someone was available at the entrance that could give accurate commands or a guide may make the original trip with the individual who is BVI. In the latter case, the individual who is BVI would count the number of steps before they had to turn, get on an elevator or perform another task. From

this, they would have a list of navigational commands which they could follow to get from one place from another. However, the use of step by step commands makes it difficult for an individual who is BVI to recover from a wrong turn or deviate from the original intention/path.

In addition, neither of the above methods for guiding people who are BVI help them in developing situational awareness of where they are. Situational awareness would allow an individual who is BVI to recover on their own from a wrong turn as they could identify where they ended up. It would also allow them freedom to explore the interesting sounds and smells around them. So, maps are desirable for people who are BVI but need to be made more portable and easier to access.

1.1. Audio Feedback

One of the main senses used as an alternative to vision, to provide information in maps or diagrams is the use of sound: sound being either a verbal description, using speech, or sonification, using audio tones/notes (Bujacz and Strumillo, 2016). These methods involve the use of digital interactive maps, which are maps that are purely digital and are displayed on a screen or projected onto a surface. The maps are then explored using either an assisting device or through the direct contact of fingers (Ducasse et al., 2018). Devices used to assist in the exploration can include keyboards (Zhao et al., 2008; Delogu, 2010; Weir et al., 2012), joysticks (Picinali et al., 2014), and styli (Milne et al., 2011; Daunys and Laruska, 2009; Brittel et al., 2013). The idea is that as the device or finger explores the digitized region, verbal and/or sonified cues provide information of what is currently under the exploring device/finger.

A variety of methods using assistive devices have been created for exploring maps and diagrams (Delogu, 2010). The iSonic (Zhao et al., 2008) used the 3x3 numerical keys on the keyboard to allow for the navigation of a choropleth map that had been divided into a 3x3 grid. While devices such as the joystick (Picinali et al., 2014) and stylus (Milne et al., 2011; Daunys and Laruska, 2009) allowed for a wider range of motion when exploring maps since it was not broken into a 3x3 grid.

Although these devices helped search digital maps they each have their limitations, their use is very awkward on the go and it is unlikely that users who are BVI would bring them when going to a new place they wish to explore. Also, some device can only explore a 3x3 grid (Zhao et al., 2008; Delogu, 2010), while others have trouble keeping track of the location of the cursor (Ducasse et al., 2018; Golledge et al., 2005; Pietrzak et al, 2009). There is also some suggestion (Rice et al., 2005; Levesque et al., 2012) that assistive device that include cutaneous or haptic feedback are likely to be used.

Devices involving direct contact by the fingers on the digital surface, are devices that typically involve the use of a smartphone, tablet, and large touch screen tv's/monitors (Su et al., 2010; Kane et al., 2011; Carroll et al., 2013; Kaklanis et al, 2013). The first two are very portable. For example, TimbreMap (Su et al. 2010), used a smartphone to provide sonification to help users locate items on the screen. Carroll et al. (2013) also used both verbal and audio cues to present information to the uses but through a table to display weather maps. However, again concern was raised as to how well an individual who is BVI could understand the spatial layout and estimate relationships between map elements (Ducasse et al., 2018; Bujacz and Strumillo, 2016).

1.2 Tactile Feedback

Another sense that is commonly used as a substitution to vision is touch, or tactile feedback. Systems that provide this type of feedback are typically divided into three categories: graspable (joystick, game controller, computer mouse, etc.), wearable (glover, arm sleeve, ring, etc.), and touchable (physical displays, tactile displays) (Culbertson, 2018).

The most common form of touchable systems used are tactile graphics, which are images/maps created by cartographers using raised inks, thermoforming, vacuum forming, etching, or accretion (Salisbury, 1992; Schneider and Strothotte, 1999). Although tactile graphics are still the most effective method for presenting graphical information to people who are BVI, they are cumbersome and expensive to make. They are certainly not easy to take with one to a new place.

Figure 1.1 Created Tactile Diagrams

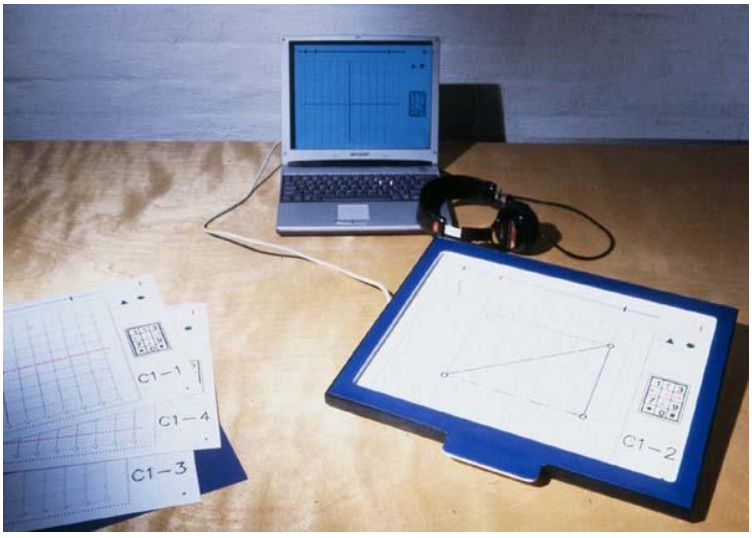
a) b)

Left is a) (Edman, 1992); right is b) (Ferro, 2018)

Another aspect of tactile graphics, especially ones with multiple patterns, is the use of braille to help understand what they are feeling (Vozenilek, 2009). However, this has a limiting effect since it is estimated that 9% of BVI individuals can read braille

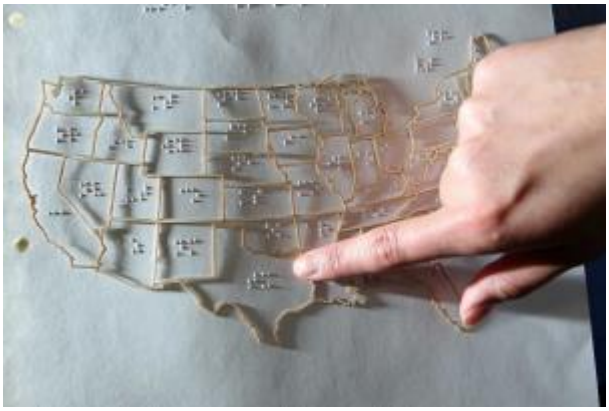
(NFB, 2014). This problem has been addressed by the development of the Talking Tactile Tablet, which provides speech as labels and descriptors for graphics. This consists of a graphic tablet with a touch surface and mounted swell paper (Landau and Gourgey, 2001). However, this provides additional bulk to the system and the need for a computer to attach to. More recently, several different groups have used mobile touch screen devices with vibration feedback (such as some phones and tablets) to provide a more portable method for representing maps, typically with the use of speech labels (see section, 1.3).

Figure 1.2 Talking Tactile Tablet



Showing the swell paper along with the tablet that is hooked up to a computer (Landau and Gourgey, 2001).

Figure 1.3 Tactile Map Containing Braille



A tactile map containing braille from Turner (2012)

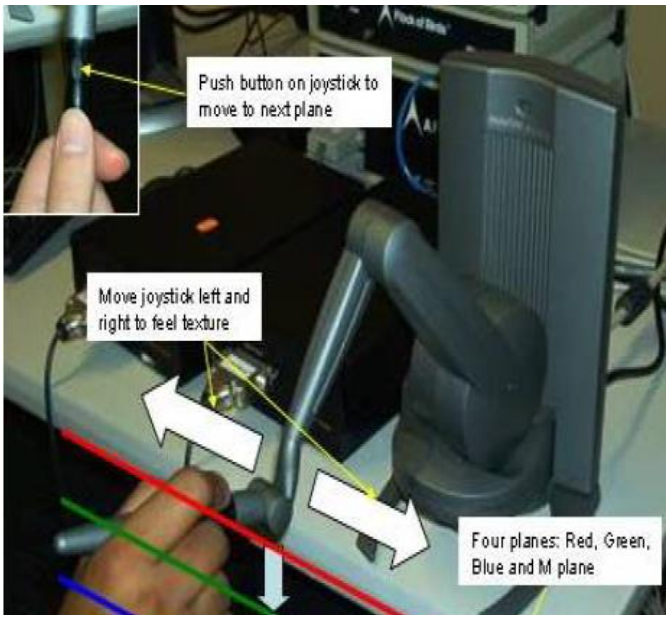
Another example of a touchable system the use of a camera to create a dynamic tactile map (Maingreaud et al., 2005). For this, a person wears a camera implanted in a pair of glasses the image is then processed and presented on device containing 8x8 micro coils. The micro coils are either raised or lowered based on the image processed from the camera. However, due to the small size of the device only simple images can be displayed with real understanding (Maingreaud et al., 2005; Delogu, 2010; Minatani et al., 2010).

Figure 1.4 Dynamic Braille Display

Image from Maingreaud et al. (2005)

For the graspable systems one of the most commonly used to search virtual tactile maps/images is The Phantom™ Haptic Joystick (SensAble Technologies) (Kahol et al., 2006; Moustakas et al., 2007). This is a three degree of freedom for reflecting joystick. Tactile maps are rendered in the 3D virtual workspace (Zeng and Weber, 2011).

Figure 1.5 Phantom Haptic Joystick



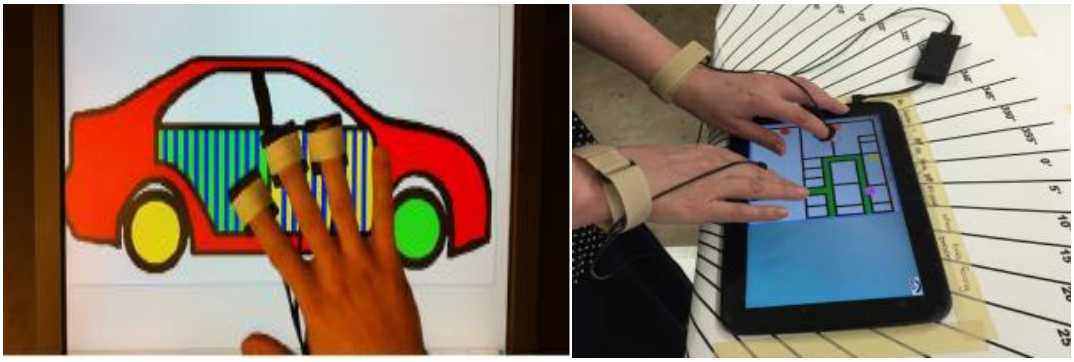
A joystick that provides tactile force feedback to users as the virtual map is searched. The device is created by SensAble Technologies (Kahol et al, 2006)

Results have shown that people could identify objects with 100% accuracy (Kahol et al., 2006), although it is unclear how well they would be able to interpret tactile maps. However, cost and lack of portability of these graspable haptic devices is a significant limitation.

Many researchers have considered glove like wearable tactile systems with vibration feedback, although not directly for displaying map information. In applying the use of wearable tactile systems to provide more effective access to tactile diagrams by people who are BVI, the focus was on the ability to provide feedback to multiple fingers.

This is something that only (Bulky) physical tactile diagrams and (very expensive) pin displays could provide. Burch and Pawluk (2011) showed that the use of feedback to multiple fingers improved the ability of BVI users to identify common objects significantly. Some examples of these devices are wearable optical color sensor-vibrator pairs (Burch and Pawluk, 2009, 2011), and tactile rings holding linear resonant actuators from an amplifier circuit created by Barron Associates (Adams et al., 2015). In the case of these two devices, they are designed to be placed on a finger and provide feedback directly to that finger based on the color that is detected (Burch and Pawluk, 2009, 2011; Adams et al., 2015). Color is used to render a part or feature of a diagram.

Figure 1.6 Wearable Tactile Devices



Examples of using wearable tactile systems to search virtual tactile maps, the left image shows a person wearing optical sensors to detect color to provide tactile feedback (Burch and Pawluk, 2011). While the right image uses an app to detect the RGB value the finger is over and provide tactile feedback to the finger through tactile rings worn by the user (Adams et al., 2015).

1.3 Audio-Tactile Feedback

Although most sensory substitution devices for BVIs use either just tactile or audio feedback, some researchers think that it would be better to present information multimodally, using both tactile and audio feedback (Loomis, 2003; Golledge, 2003; Guidice et al., 2012). This is because it would allow for information to be presented, or

integrated, in a redundant way (Jacobson, 2002), making it easier for the user to gather important information from the display they are exploring.

One of the main methods that has been used to provide both audio and tactile feedback is by creating tactile maps and placing them on a touchscreen device (Campin 2003; TMAP, Miele and Martson, 2005; Brock et al. 2010, 2012). The tactile graphic allows for tactile feedback as they explore, while the touchscreen provides the text-to-speech output of what their finger is over. This helps eliminate the need to be able to read braille from using tactile graphics (Miele et al., 2006).

Another approach to systems that provide both audio and tactile feedback is by creating virtual, or digital, maps on a computer or tablet (Klatzky et al., 2014; BATS, Parente and Bishop, 2003; VAVETaM, Habel et al, 2010; Open Touch/Sound Maps, Kaklanis et al., 2013; MAPS, Adams et al., 2015). Most of these methods use a device to provide haptic feedback while the map is searched and audio/speech cues to identify items that have been found or are near. In the case of Adams and his colleagues, haptic and sonified audio cues were used redundantly for some or all features, with speech labels.

Figure 1.7 Audio Tactile Devices

The top left image (Habel et al., 2006) and top right image (Parente and Bishop, 2003) use a joystick and a computer mouse to provide tactile feedback while audio feedback is provided as they cross over important features such as streets or cities. While Adams et al. (2015) provides audio feedback for things such as the edge of the map or wall.

Guidice et al. (2012) focus has been showing whether the us of audio-tactile feedback helps people who are BVI develop spatial awareness. They created a vibro-audio device to test how well graph information and letter recognition could be determined. Results showed that the vibro-audio interface allowed for the building of accurate mental representations, which support various spatial operations. Klatzky et al. (2014), however, used to same interface to compare it to audio and tactile feedback interfaces and found no significant improvement in the development of spatial representations.

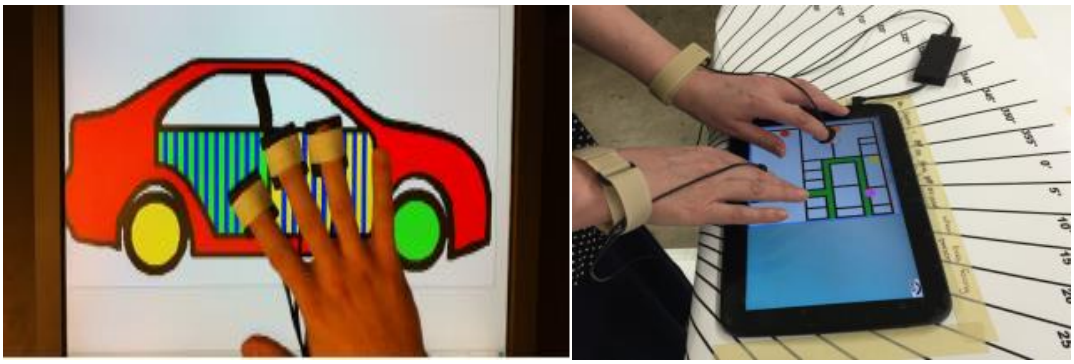
1.4 Single versus Multi-touch

An important consideration when using senses other than vision for graphical access is the significantly more limited information bandwidth of touch and audition compared to vision. Accessing graphical information using either touch or audition is a very difficult task. As the fields of view of the exploring fingers are limited, a lot of the information processing requires serial integration of information over time: a slow and cognitively demanding process. However, this is made significantly worse by the fact that most of the methods described in the previous sections (1.1-1.3) only allow for a single finger to explore a display: the exceptions being (bulky, expensive) physical tactile diagrams, (expensive, non-portable) large pin displays, and wearables designed purposefully to be multi-fingered, cheap and portable.

The benefit of multiple fingers is not a given though. It is known that in searching for a target amongst distractors on all fingers, only some types of tactile properties can be processed in parallel across fingers (Klatzky, Lederman and Metzger, 1985). Teachers for the visually impaired do instruct their students to use all their fingers to explore the overall image of a physical tactile map before a more detailed exploration. So there seems to be some benefit, at least in searching for where the information is located. Students who are BVI also often use one finger as a reference to help gauge distances from a set location for graphs and maps. Burch and Pawluk (2009, 2011) when asking BVI users to identify objects in virtual diagrams using multi-fingered vibration feedback found that not only the speed at which the BVI users completed the task, but also the correctness of their answers, improved significantly. This suggested that multiple fingers helped improve the integration of information on a diagram as well.

In contrast to the above positive results concerning multiple fingers, Adams et al. (2015) did not find that multi-fingered vibration feedback was beneficial in exploring maps. However, the task was only to find a point of interest on a map rather than provide information about the surrounding area or how to navigate from point to point. Many users focused solely on detecting the point of interest (with a total of 5 per map) rather than interpreting any of the other cues about other features in the map provided. What it does suggest is that the advantage of using multiple fingers may depend on the task the user is given.

Figure 1.8 Multi-touch Devices



Burch and Pawluk (2011) allows for the use of multiple optic color sensors to search images to provide feedback to the fingers. While Adams et al. (2015) provides two tactile rings to allow the searching using multiple fingers.

In another study, Brock et al. (2010, 2012) used physical tactile maps placed on a touch screen which had the ability to track multiple fingers touching the surface. A survey of users found that the usability of the method was very high, with many users liking the use of multiple fingers. However, their strategy of using a tap to generate text-to-speech output was often problematic due to users' inability to determine which finger generated a tap: this was fixed by using a double tap (Brock, 2012).

1.5 Motivation

One of the primary motivations for this work is that previous research (Burch and Pawluk, 2011 versus Adams et al., 2015) obtained conflicting results as to the benefit of the use of providing tactile feedback to multiple exploring fingers. For identifying common objects with their parts indicated by different textures, Burch and Pawluk found a large improvement in performance with the use of multiple fingers. (It should be noted that this was not true for raised line drawings.) Adams and his colleagues, despite using texture to indicate features such as pathways, stairs, elevators and points of interest, found for a search task to identify a point of interest that there was no performance improvement was multiple fingers. Because of the conflicting outcomes, further research needed to be conducted to show whether using multiple fingers to explore a map/image helps those who BVI.

The questions that arise from these results are:

1. Does the use of multiple fingers improve a BVI user's performance over the use of a single finger to explore a map with tactile cues?
    1.1. Does this depend on the type of map (overall map versus individual map)?
    1.2. Does this depend on the type of question (varying in the degree it involves spatial understanding)?

However, there is also another set of questions that arise from the fact that vibration feedback involves a lot of battery power in mobile devices. In contrast, audio feedback consumes much less power. Although concerns have been raised that solely audio feedback would result in poorer performance than tactile feedback, the two methods have not been compared directly.

2. Is user performance with one fingered audio feedback equal (or, even better) to one fingered tactile feedback?
    2.1. Again, does this depend on the type of map?
    2.2. The type of question?

Of course, the natural question to arise is: can the use of audio feedback for multiple exploring fingers improve performance? We are unaware of any studies who have investigated how audio cues that are presented in parallel can process spatial information. However, it is known that audio is segmented into separate streams by audition. Two well-known methods that naturally occur are separation based on timbre (the sound of a musical instrument) and separation based on spatial location. This gives rise to the following questions:

3. Does the use of multiple fingers to explore a map with audio cues improve performance?
    3.1. Does this differ for different types of maps?
    3.2. Does this differ for different questions?

Then, naturally, not only for battery conservation reasons but usability (users do not like wearable devices on the hand), it would be good to ask the question:

4. Is user performance with multi-fingered audio feedback equal(or, even better) to multi-fingered tactile feedback?
    4.1. Again, does this depend on the type of map?
    4.2. The type of question?

1.6 Thesis Outline

The research questions asked in this thesis, in regards to providing spatial map information to BVIs, were:

1. Can the use of simple sonified (non-speech audio) cues with kinesthetic information (finger location) work as effectively as using tactile cues with kinesthetic information?

2. Can the use of sonified cues providing information from two exploring fingers be more effective than sonified cues providing information from only one finger? Also, as there is not a natural mapping between the audio feedback and the finger (for tactile feedback, the feedback occurs on the same fingertip as the exploring finger), does the method of mapping onto the audio domain make a difference?

3. Can perceptual principles of audio stream segregation be used to effectively relay information about two exploring fingers? (more effective than a single finger)

4. For both sonified and tactile cues, does the type of map to be explored and the type of question asked effect whether the use of two exploring fingers is more beneficial than a single finger?

5. Do any of these depend on when an individual became blind or their experience using tactile graphics?

The first chapter of this thesis describes the different types of maps chosen and their construction, as well as the systems and methods used for presenting the nonvisual cues. The next two chapters described pilot experiments which were performed to select parameter values for the cues that could be used well. This was particularly important to do in comparing performance with audio versus tactile cues, as variations in difficulty of perceiving the parameters could produce erroneous conclusions. The fourth chapter describes the main experiment, including the questions that were developed. The results of the experiment and statistical analysis are given in chapter 5. Finally, a discussion and conclusion is given in chapter 6.

2. Materials and Methods

The maps that we are interested in considering are ones representing functional spaces such as museums, botanical gardens and shopping centers. These are maps which may represent features such as paths/hallways, stairs, or restrooms, and points of interest (POI), which may have some sort of descriptive content associated with them. Both auditory feedback and tactile feedback will be considered for representing spatial features, with speech being used to provide labels for the features upon request. This chapter will describe the development of the content and format of the diagrams used for device assessment, as well as the systems used for "reading" these maps and providing either auditory or tactile feedback.

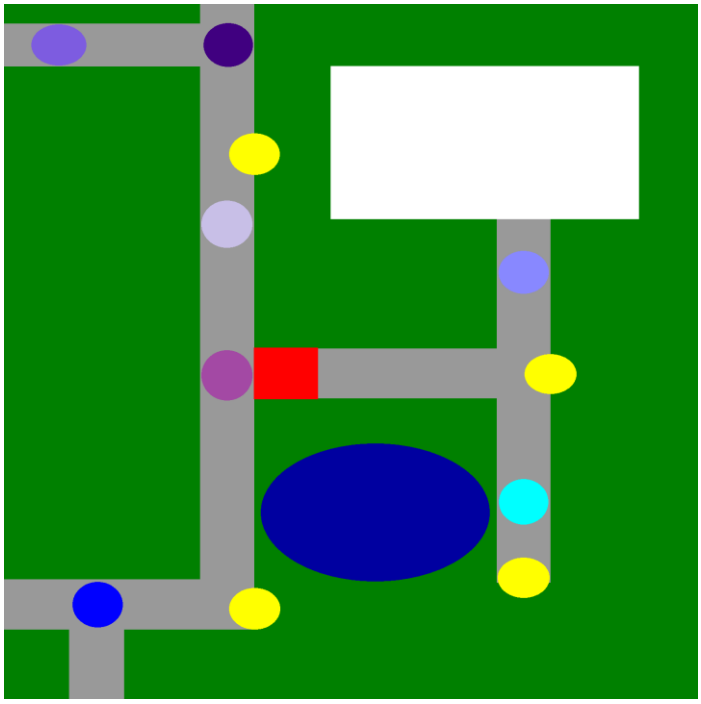
2.1. Diagram Development

The original maps that were constructed were based on the area of the Lewis Ginter Botanical Gardens between the main entrance and the conservatory. This area was chosen because it is the primary area where gardens were modified to provide more accessible features for individuals who are blind (e.g. smell, touch). The area is divided into what are referred to as garden rooms, which are relatively square and separated by walls and/or pathways. Maps were created for the six garden rooms and one overall map (Figure 2.1 shows a garden room map and Figure 1.2 shows an overall map).

In the maps, color was used to indicate the different features. Only six colors were used as previous research in our laboratory (Burch and Pawluk, 2011) suggests it is hard for users to identify more than this number of feedback values. For the individual maps: gray represent pathways, green greenery, red stairs, yellow benches, white buildings, and

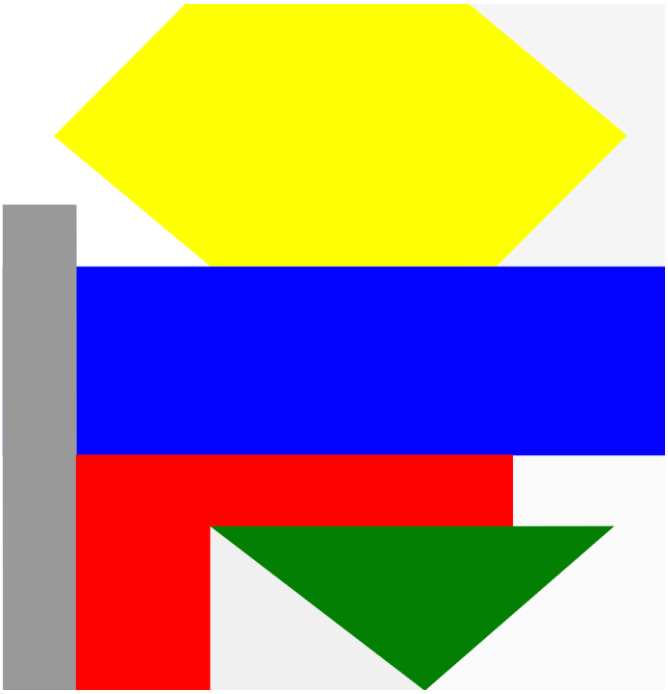
blue POIs. For the overall map: the different colors represent the different gardens with white reserved for buildings. Note that all the shades of blue (including purple) for the individual maps are representing POIs (Figure 2.1). This is done to help the tester find a specific POI when they are asked to place the finger of a test subject on a specific point and ask questions about it, such as: “What is around the Turquoise POI?”

Figure 2.1: Individual Garden (Storage Garden)



As we are interested in comparing different methods of providing auditory or tactile feedback about the map and to avoid learning effect during participant testing, one additional garden room map was created and six additional overall maps. To ensure consistency of difficulty, all overall maps had 5 garden rooms and 1-6 buildings. With each garden room consisting of simple geometric shapes of different sizes. All individual garden maps had a similar density of pathways, stairs and POIs, and 3-10 benches. Appendix 6.1 contains all the maps created for the study.

Figure 2.2: Overall Map C



2.2. Presentation Systems

The idea to present the map information non-visually to individuals who are blind or visually impaired was to allow the user to explore the diagrams with one or more fingers. The system then determines the feature under each of the fingers by the color at the center of the finger. Based on the color, different auditory or tactile feedback signals will be used to indicate the feature.

The platforms used for presenting the maps were multi-touch mobile tablets as they can easily be carried by individuals who are blind or visually impaired when exploring a new space. File of the maps, in png format, were presented on the touchscreen with the different features indicated as different colors (Table 1). The location of finger contact was determined using the built in tablet functions. The location was used to determine the color at that point based on a lookup table. The color was then

translated into the appropriate sonified or tactile feedback. If a user lifted their right finger from the screen, the name of the feature under that finger is spoken.

Table 1: What Each Color Represented For Each Map Type

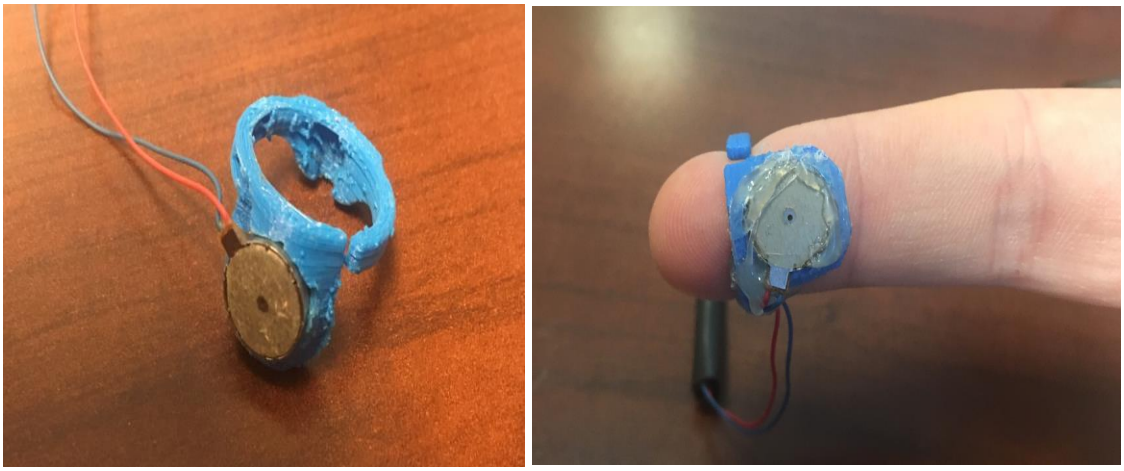
Color	**Overall Maps**	**Individual Maps**
White	Building	Building
Blue	Blue Garden	Point of Interest/Fountain
Green	Green Garden	Garden/Grass
Gray	Gray Garden	Pathway
Yellow	Yellow Garden	Bench
Red	Red Garden	Stairs

The platform for providing the sonification of the map on the iPad was using Auvio headphones (Auvio 3300089, Auvio Electronics Co. Ltd., China) for audio feedback. The code developed for the corresponding App could control the audio feedback to each ear based on the feature in contact with one or more of the fingers. The code is provided in the Appendix (section 6.2). The auditory sounds to be played were initially created in MATLAB with Wind Instrument Toolbox V 0.2 by Martin Rocamora. Due to the limitations in the quality of the timbres produced, the creation method was switched to using MuseScore 2 App. For user testing, the developed App could also select the type of sonification feedback provided, in addition to the map to be used.

The platform for providing the tactile feedback was Nexus 10 Android tablet. To enable tactile feedback to more than one finger, tactile feedback was provided through ring like devices (Figure 10) each containing a linear resonant actuator (C10, Precision Microdrives, Inc., London, U.K.). Each ring like device had a wire connection to the tablet through an amplifier circuit (Barron Associates, Inc.). Different ring sizes were 3D

printed for different hand sizes. The position of the ring like device on the finger in the latter half of the distal pad of the finger (Figure 2.3); this allowed the tablet to still detect the location of a finger while it explored the touchscreen. The App to present map information using tactile feedback to either one or two fingers was developed by Barron Associates, Inc. (Charlottesville, VA). The underlying signal was set to drive the linear resonant actuator at its resonant frequency (175 Hz). Different tactile vibrations were created by changing the amplitude and the on-off duration of the square wave modulation of the underlying signal.

Figure 2.3: 3D Printed Ring Holding Linear Actuator



In the systems used for the different feedback modalities (auditory versus tactile) areas of the maps were slightly different (28.55 square inches on the Android tablet) as compared to (28.6 square inches on the iPad). This was due to differences in how the screen was represented by the overlying software. The difference in size is not expected to impact performance.

2.3. Sonification Presentation Methods

Consideration of the different sounds to provide feedback about the presented map being explored by the fingers took into account indicating the different features and the different fingers that could be in contact with the multi-touch screen in different locations. Distinguishing different features by sonification facilitates exploration of the map without always having time to stop and listen to the speech description of the feature. Providing feedback about features under different fingers exploring the multi-touch screen could increase performance by allowing parallel exploration. However, this is dependent on whether individuals can integrate the information from multi fingers exploring at the same time, which is currently unknown. This is the primary research question when considering different sonification presentation methods.

Feedback could be in the form of different notes, natural sounds (e.g., rain, glass filling) or melodies. Single notes were chosen as the total information about the sound is available almost immediately in contrast to a melody which would require the full time duration of the tune. Variables of single notes that could be manipulated are: frequency, amplitude and timbre. In addition, for spatialized sound, the location of the finger in space could be used. Two dimensions are needed: one to indicate different features and one to indicate different fingers. The combination of dimensions considered were based on what is known about auditory scene segmentation. Methods of pre-attentional grouping of sounds include: spatial location, timbre and high versus low frequencies (Brown et al., 2005).

Tone frequency was chosen to depict the dimension of the different frequencies as it is the most likely auditory dimension to easily create 6 different identifiable signals.

The pre-attentional grouping parameters of spatial location and timbre-were chosen to represent the different fingers. Both parameters were explored as there was no prior knowledge whether either will allow integration of information from different fingers.

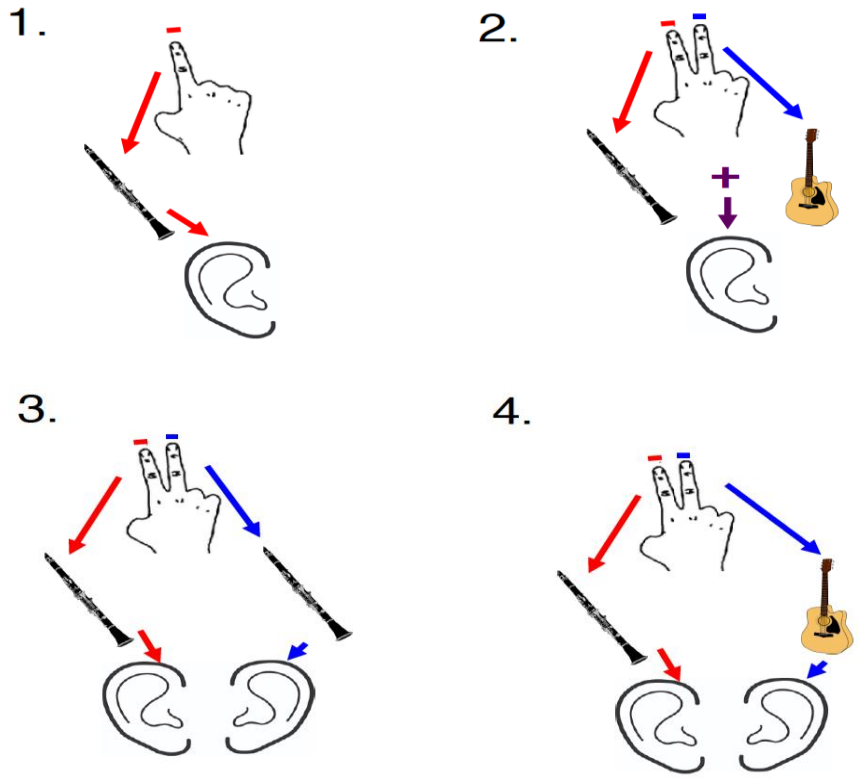
To determine whether pre-attentive grouping parameters can be used to represent individual fingers exploring the map to improve user performance, we have created four different conditions (Table 2, Figure 2.4). The first three conditions examine the use of the two pre-attentive grouping parameters (spation, location, and timbre) being used for separate fingers both individuals and together as redundant dimensions. To simplify the implementation, sonification was only provided for two fingers. This allowed spatial location to be implemented terms of the stere-positioning of the audio feedback in terms of left ear (left finger) and right ear (right finger). The fourth condition was a control condition allowing the typical single finger exploration (only with sonification not the typical tactile feedback). This condition also explores the use of sonification in place of tactile feedback. This is the second important research question of the study. There is an advantage to using sonification as sound takes less power consumption than providing tactile feedback; this is important for mobile applications.

Table 2: Methods Used to Provide Audio Feedback

Method Name	**Number of Fingers**	**Timbre**	**Ear(s)**
One Finger One Ear One Timbre	1	-	-
Two Fingers One Ear Two Timbre	2	Different	Same
Two Fingers Two Ears Same Timbre	2	Same	Different

Two Finger Two Ears Different Timbre	2	Different	Different

Figure 2.4: Audio Feedback Methods Used



2.4. Tactile Presentation Methods

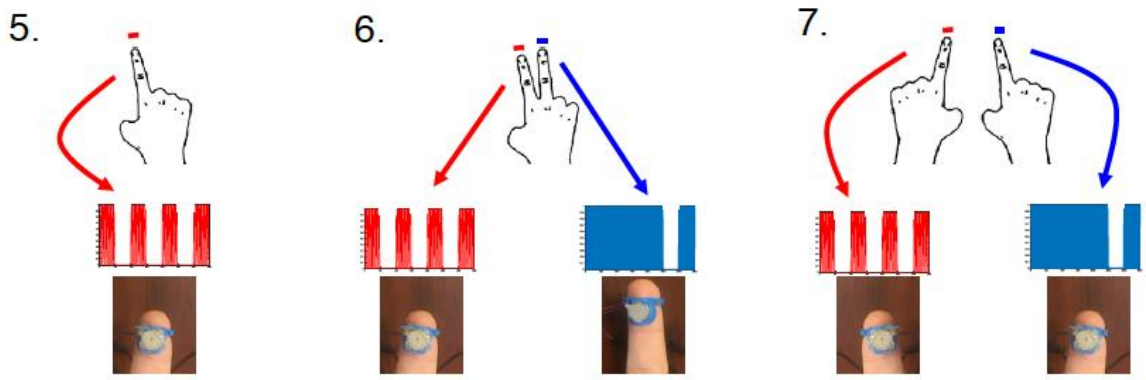
Three different vibration feedback methods to present information to the user were developed (Table 3, Figure 2.5). Again, two dimensions of the map exploration needed to be represented by the vibration signals: the features and the finger that is over the feature. Also again, there was the choice to use single tactile "notes or "melodies." Single tactile "notes" were chosen as the total information about the sound is, again, available almost immediately in contrast to a melody which would require the full time duration of the tune. For all of these methods, each of 6 amplitude/frequency pairs specifying a "pure tone" was used to represent one of 6 features.

As for the sonification methods, the other dimension for the feedback was needed to separate the fingers. The most natural mapping, and one that has been examined previously (i.e., Burch and Pawluk, 2011; Adams et al, 2105) is to provide tactile feedback separately on each of the fingers based on the finger's location . However, very different results were obtained based on the type of graphic. It is also likely that type of question matters as well. The goal here is to explore these issues more for maps of functional spaces. As in Burch (2012), we also wanted to explore whether using two adjacent fingers (kept side by side) on a single hand or one finger on each hand (free to explore independently) makes a difference in performance. It would have been desirable to consider timbre as a method of grouping features by exploring as then only a single vibration motor would be needed. This would allow a mobile tablet, without any external hardware, to be used to provide information about two fingers. However, timbres are not differentiable for vibratory feedback. Therefore, only spatial location is considered.

Table 3: Methods Used to Provide Tactile Feedback

Method Name	**Number of Fingers**	**Exploration Strategy of Fingers**
Single Finger	1	-
Two Fingers Single Hand	2	Same hand, side by side
Two Fingers Two Hands	2	Different hand, free exploration

Figure 2.5: Tactile Feedback Methods Used



3. Selection of Audio Parameters

Parameter values needed to be chosen for the different dimensions to be used: spatial location and/or timbre for the different exploring fingers and tone frequency for the different map features. An assessment of the potential benefit of the use of multiple exploring fingers was to use only 2 fingers, only 2 spatial locations and 2 timbres needed to be chosen. The two furthest apart and easily created spatial locations were: to the right of the user (created by playing a sound exclusively to the right ear) and to the left of the user (created by playing a sound exclusively to the left ear). As the initial method to create timbres was through the open source Wind Instrument Toolbox, the two timbres were restricted to wind instruments. However, as the timbres of different wind instruments were difficult for individuals to distinguish the creation method was switched to using the MuseScore 2 App.

Six different tone frequencies needed to be selected to differentiate the different map features. Frequencies were limited to those below a value at which they become irritating to the user. Preliminary testing was performed comparing the set of sounds created (6 frequencies at one timbre, 6 frequencies at a second timbre) to ensure that all sounds created could be uniquely identifiable. This section describes the set of experiments toward this determination.

3.1. Selection of Six Tonal Values for First Instrument (Timbre)

The tonal values were selected first as it was expected that then comparison between tones of different timbres would then be able to be limited in comparison to each other to validate their difference. The clarinet was chosen as the first instrument to be

tested since it is considered the most aesthetically pleasing instrument. The tonal values selected were: 160, 260, 360, 460, 560, and 660 Hz. A small study was used to verify that the six tonal values selected could be identified uniquely.

3.1.1. Participants

There was a total of three sighted participants

3.1.2. Signals

Tones were generated using the open source Wind Instruments Toolbox in MatLab and were: 160 Hz, 260 Hz, 360 Hz, 460 Hz, 560 Hz, and 660 Hz. Amplitude was held constant at 40 decibels.

3.1.3. Experimental Design

Four repetitions of each tone were played to the participants through the headphones to both ears, blocked on repetition. Participants were required to identify each of the tones played.

3.1.4 Experimental Procedure

3.1.4.1. Training

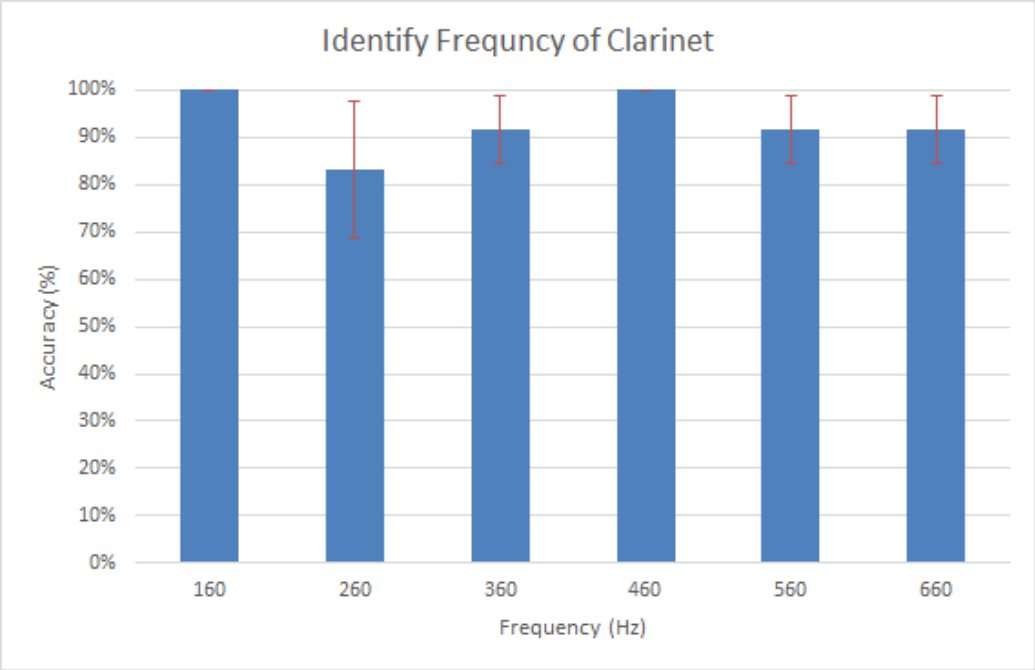
A participant was played each tone in order, after each tone was played, the frequency was spoken to the participant by the experimenter. This was repeated twice. If a participant wanted to listen to a specific frequency again, they were allowed to ask the experimenter to play that specific frequency again. Once the participant felt they could identify each of the six frequencies, testing was begun.

3.1.4.2. Testing

For the testing phase, participants were played four repetitions of each of the six sound blocked on repetition. The order within blocks was random. Participants were asked to identify the frequency of each sound immediately after it was played.

3.1.5. Results

Graph 1: Identification of Each Frequency Played



Performance for identifying each of the different tones was above 90% for five of the frequencies (Graph 1). The hardest note to identify was the 260 Hz signal as it was identified only 83⅓% of the time.

3.1.6. Discussion

The results suggest that the set of 6 values chosen can be easily identified: rate of correctness was high with relatively little training. Therefore, the next step was to determine whether 6 tone values for each of two timbres could be identified with respect to each other.

3.2. Selection of Two Timbres for the Six Tonal Values

In exploring the wind instruments created through the Wind Instruments Toolbox, the most distinctly different instrument from the clarinet for the six tonal values was the trumpet. Verification that the twelve unique tones created were identifiable was needed before their use to represent map information, this was in terms of: (a) the instrument being identified correctly (which would map onto the exploring finger) and (b) the tonal value being identified correctly (which would map onto the map feature). The next study simply asked whether one or two instruments were played, as this was a precursor to the above concerns.

3.2.1. Participants

For this test a total of nine sighted participants were selected.

3.2.2 Signals

Tonal sounds were created using the Wind Instruments Toolbox for MatLab for the clarinet and trumpet. The tonal frequencies created were, again, 160Hz, 260Hz, 360Hz, 460Hz, 560Hz, and 660Hz. Again, the amplitude was held constant at 40 decibels.

3.2.3 Experimental Design

Both the trumpet and the clarinet played a tone either individually or simultaneously through headphones to both ears. All possible combinations were given to the user: 6 tones of the trumpet individually, 6 tones of the clarinet individually, and 6x6 combinations of simultaneous tones. For this initial study, participants were simply asked to identify the instrument(s) played.

3.2.4 Experimental Procedure

3.2.4.1 Training

Each participant was played each of the 6 tones for the clarinet individually and the 6 tones for the trumpet individually. They were told that each set of 6 tones were from different instruments. If the participant wanted to listen to an individual tone played by the clarinet or the trumpet again, they were allowed to ask the experimenter to play the tone again.

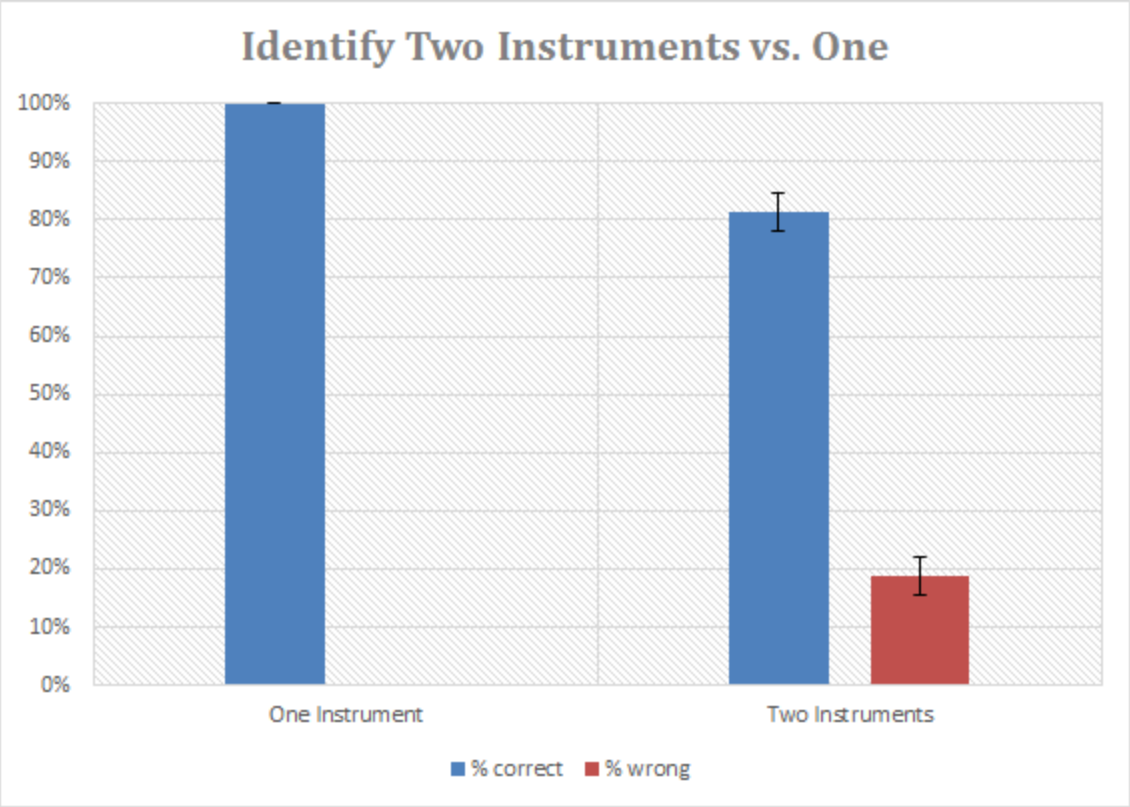
Next, the participant was played all combinations of the clarinet and trumpet tones simultaneously (e.g., clarinet at 160 Hz with the trumpet at 560 Hz). They were told that all sounds played from two separate instruments. After listening to all the combinations, the participant was allowed to ask the experimenter to replay any combination of frequencies they were unsure of again.

3.2.4.2 Testing

During the testing phase, the participant was played all combination of single tones for the clarinet and trumpet, and all combinations of tones, in random order. After each sound was played, they were then asked to identify whether there were two instruments or one instrument. Participants were also allowed to ask for the sound to be played again before answering.

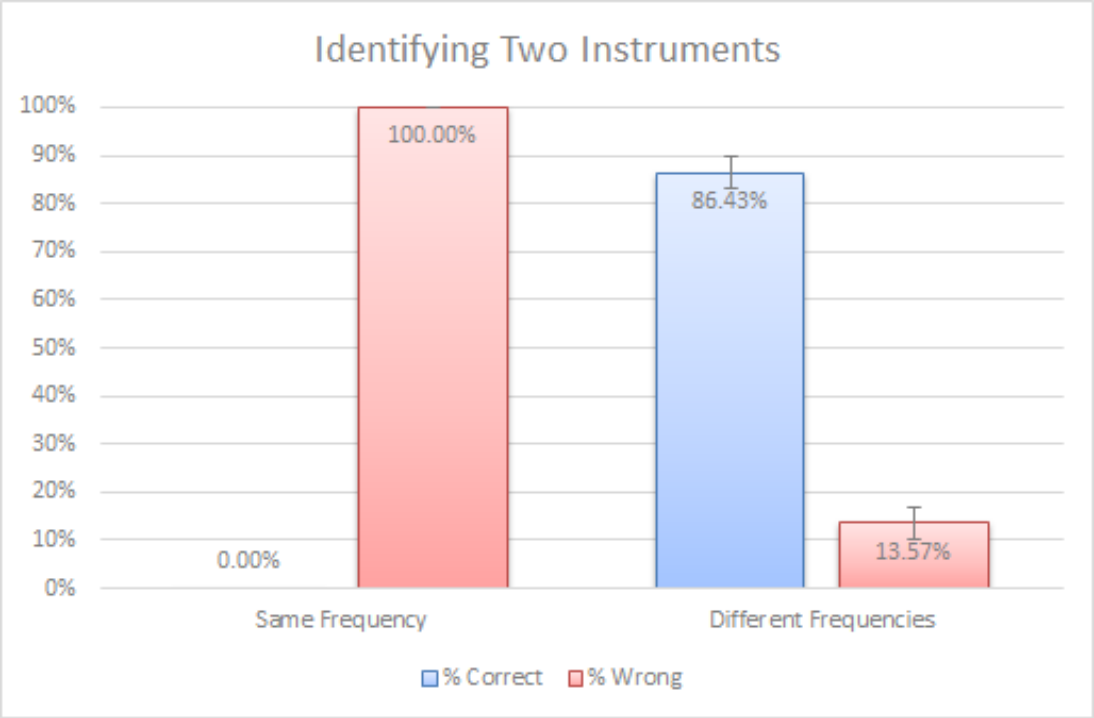
3.2.5 Results

Graph 2: Identifying Single Instrument vs. Multiple Instruments

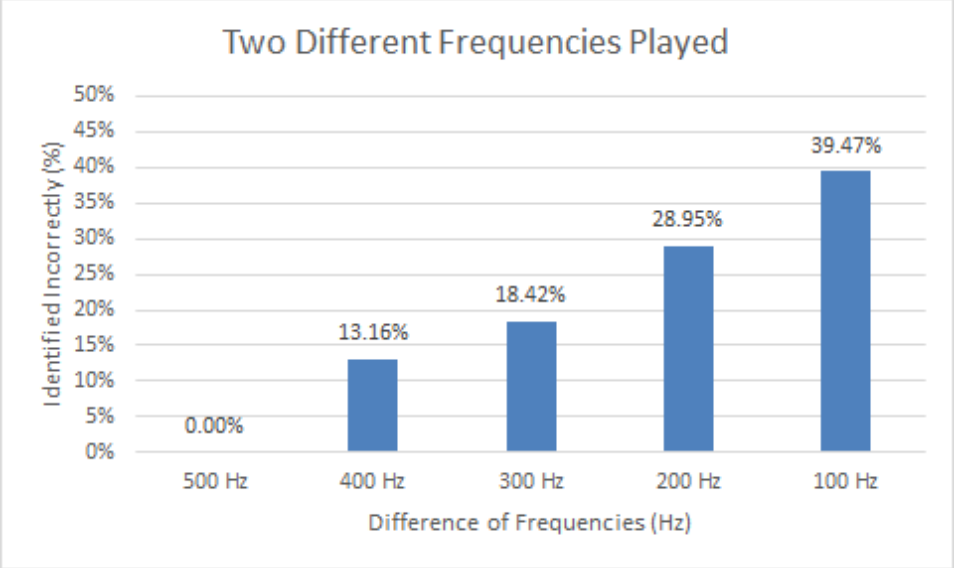


Results showed that participants were 100% accurate in determining when one instrument was played but only 80% accurate when two instruments were played (Graph 2). To more clearly determine the reason for why participants were less accurate in determining if two instruments were played, the data was further divided into trials where the same frequency was used for the two instruments versus different frequencies (Graph 3) and the effect of the difference in frequencies on performance (Graph 4).

Graph 3: Same Frequency Played vs. Multiple Frequencies Played Using Matlab



Graph 4: Difficulty of Identifying Multiple Instruments with Close Frequencies



3.2.6 Discussion

As can be seen in Graph 2, the results showed that in most cases people were able to identify whether two instruments, or one instrument, was being played. However, in

the case when two instruments were played, as the differences between the frequencies of the tones decreased, the error in the answer significantly increased (Graph 4) until it was 100% (Graph 3) when the frequencies were the same. One potential reason for this result may be because the range of frequencies used needed to be more spread out for easier identification. However, this does not deal with the case when the two instruments use the same frequency: which is inherent in the design of the feedback as frequency is equal to a map feature. Using two frequencies (one for each instrument) for each map feature would require more demands on a participant's memory. Therefore, there was a need to improve the distinction between the timbres.

3.3 Use of MuseScore 2 to Represent 6 Tonal Values for Clarinet and Trumpet

With further exploration, it was observed that the open source Wind Instrument Toolbox was not particularly good at providing different sounds for the clarinet and trumpet. Therefore, the next step in the design of the sonification parameters was to find a better method for generating the timbres: the MuseScore 2 App was used for this purpose. This step also investigated the use of a logarithmic separation of frequency for the lower tonal values versus a linear separation of frequency for the higher tonal values: perceptual evidence suggests that humans distinguish a difference in a frequency on a logarithmic scale (Wolfe et al., 2015).

3.3.1. Participants

There was a total of 4 sighted participants.

3.3.2. Signals

Tones for a clarinet and trumpet were generated with the MuseScore 2 App. For the first part of this study tones were generated for the clarinet only with six frequency values of: 150 Hz, 300 Hz, 600 Hz, 1,100 Hz, 1,600 Hz, and 2,100 Hz. For the second part of this study, tones were generated for both the clarinet and trumpet with four frequency values of: 160 Hz, 340 Hz, 520 Hz, and 700 Hz. All tone amplitudes are constant at 40 dBA.

3.3.3. Experimental Design

The study had 2 parts. The first part focused on examining whether a logarithmic separation between frequencies made the tones more easily identifiable than a linear separation by repeating the design of study in 3.1. Three repetitions of the 6 tones were played for the participant, blocked on repetition and frequencies randomized within blocks. The second part examined whether participants could determine both the timbre and frequency of an isolated tone. One repetition of the 8 notes (4 notes per timbre, with two timbres) were played for the participant, randomized within the repetition.

3.3.4. Experimental Procedure

Part 1: Individual Instrument Tone Identification

3.3.4.1. Individual Instrument Frequency Training

Training was the same as in study 3.1.

3.3.4.2. Individual Instrument Frequency Testing

The testing procedure was the same as in 3.1, but with 3 repetitions only

Part II: Tone and Timbre Identification for Two Instruments

3.3.4.3. Two Instruments Played Individually Frequency Training

Participants were played four notes from the clarinet and four notes from the trumpet. The order of which instrument was played first alternated from person to person. For each instrument, the four notes were played in sequence from lowest to highest frequency. For each note played, the experimenter told the participant the frequency of the note. After all the notes of an instrument had been played, participants were allowed to ask the experimenter to replay any of the four notes. The process was then repeated with the second instrument.

3.3.4.4. Two Instruments Played Individually Frequency Testing

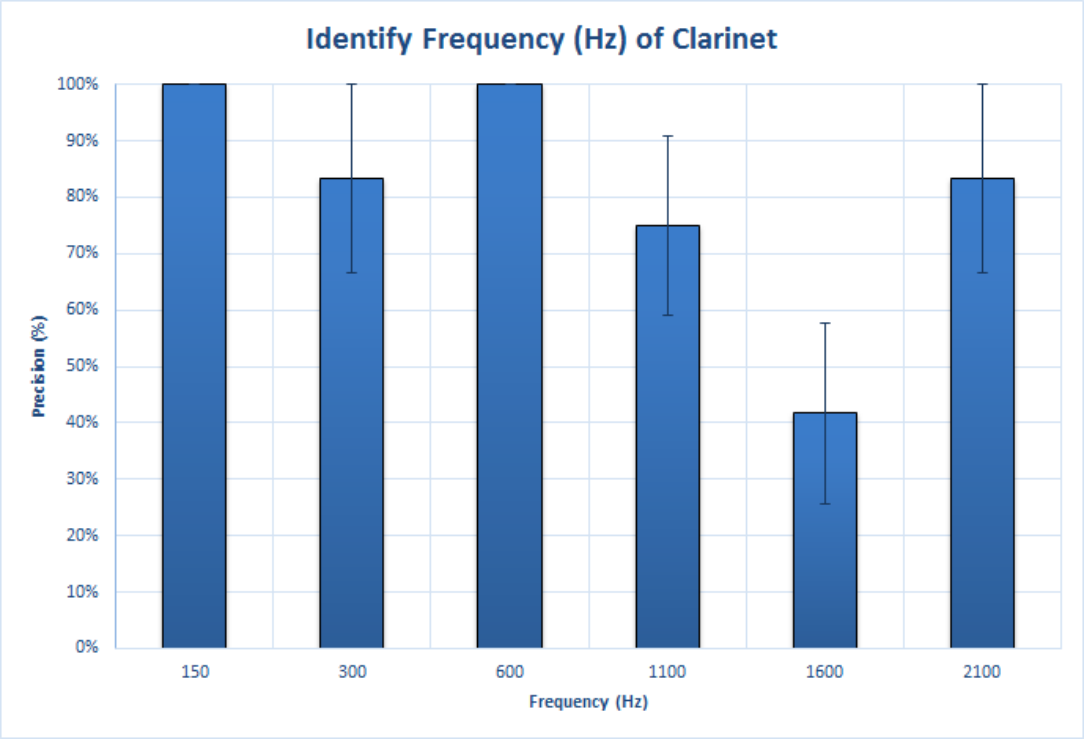
The eight notes (4 notes for each instrument) were played in random order for the participant. After each note was played, the participant was asked to identify which instrument was being played, as well as which frequency was being played. Participants were allowed to have the note replayed before they responded.

3.3.5. Results

Part 1: Individual Instrument Tone Identification

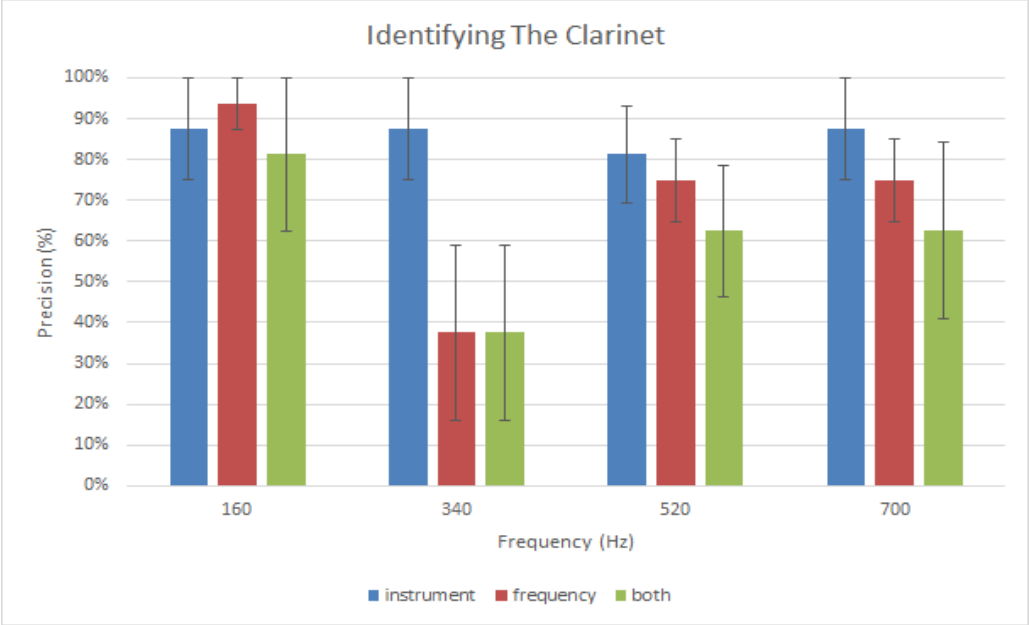
The lower frequencies which were logarithmically separated were much easier to identify than the higher frequencies, which were linearly separated (Graph 5).

Graph 5: Identification of Clarinet Frequency



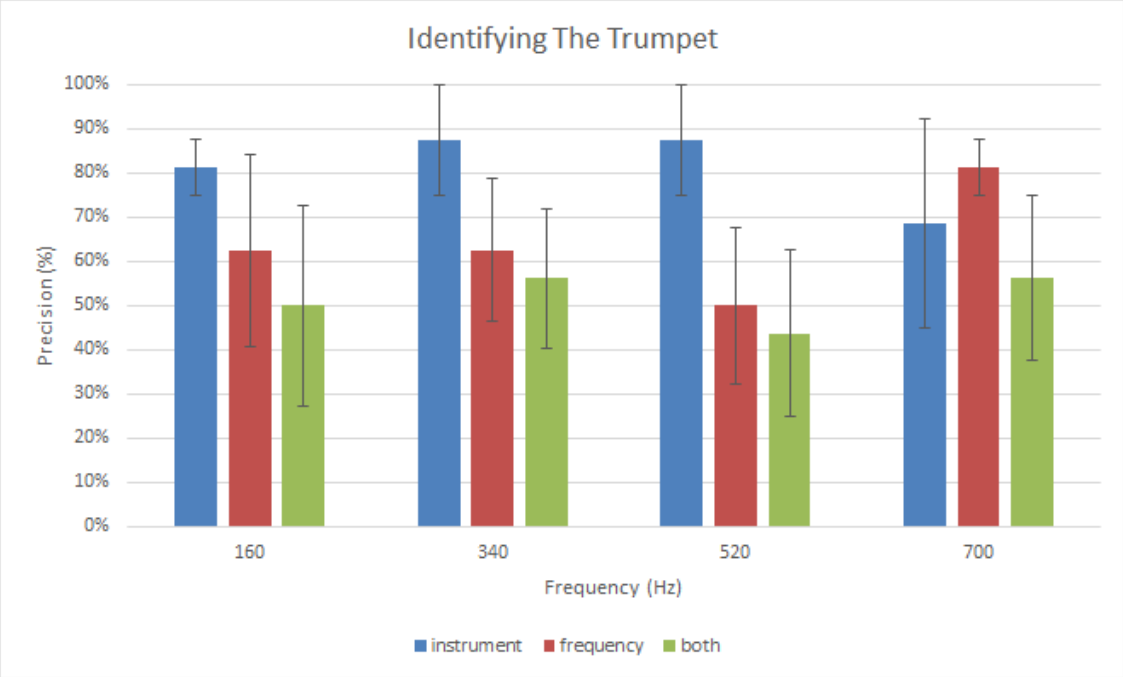
Part II: Tone and Timbre Identification for Two Instruments

Graph 6: Identifying The Clarinet and The Frequency Played



While naming the instrument clarinet (Graph 6) or trumpet (Graph 7) was pretty consistent people mostly struggled to identify the frequency the instrument was being played at.

Graph 7: Identifying the Trumpet and The Frequency Played



3.3.6. Discussion

Part 1: Individual Instrument Tone Identification

The results suggest that it is easier to identify notes on a logarithmic scale than a linear scale. The other item of note, was that some people described the higher frequency notes (1100 Hz, 1600 Hz, and 2100 Hz) as being more annoying than the lower frequency notes. Although this may have impacted performance, it is not believed to have affected performance greatly.

Part II: Tone and Timbre Identification for Two Instruments

The results show that not only did people struggle in identifying which instrument was being played but also which frequency was being played. Although, the linear scale used in this test was completely different than the one used in the previous test, it supports the theory that the reason people did so poorly in identifying the high notes in the previous test was because they were on a linear scale, opposed to the notes being annoying.

As a result of these findings it has been determined that a log scale will be used in further testing. It was also determined that the log scale used would contain either one or no frequencies over a 1000 Hz in an attempt to eliminate the complaint of annoying high tones.

3.4 Selection of More Distinct Timbres Using Logarithmic Scaling Frequency

In this test the MuseScore 2 software was used to generate the notes that would be used for testing. To increase the distinction between timbres, the clarinet was used in comparison to three other instruments. The instruments chosen were qualitatively determined by the experimenter to be some of the most different from the clarinet: the tuba, guitar, and bassoon. The trumpet was not selected as its timbre appeared more similar than the above mentioned instruments to the clarinet. Each of these instruments were experimentally assessed to determine: a) how well they could be differentiated from the clarinet, and b) how easily each note could be identified.

This step also investigated the use of a logarithmic separation of frequency for the tonal values as perceptual evidence suggests that humans distinguish approximately constant difference in logarithmic frequency as equal.

3.4.1. Participants

There were a total of three blind participants.

3.4.2. Signals

The six frequencies selected for the map features followed the formula $□(□) = 55 \times 2^{□}$. With x being an integer within 0 and 5. The frequencies used were 55 Hz, 110 Hz, 220 Hz, 440 Hz, 880 Hz, and 1760 Hz. Each sound was generated for the clarinet, tuba, guitar, and bassoon with amplitudes of 40 dBA; using the MuseScore 2 App.

3.4.3. Experimental Design

This experiment consisted of 5 small studies. In the first study, three repetitions of each tone of the clarinet (only) were played to the participants through the headphones to both ears, blocked on repetition. Participants were required to identify each of the tones played. Studies 2-4 were similar to each other, with the clarinet paired with one of the tuba, guitar or bassoon in each study. The order of instruments being paired was chosen at random. For each pairing, four repetitions of each tone of both instruments (clarinet and paired instrument) were played individually to the participants, blocked on repetition. Within repetition, the order of the tones was random. Participants were required to: 1) determine the instrument being played, and 2) determine the frequency of the note.

3.4.4. Experimental Procedure

Study 1: Clarinet Alone

3.4.4.1 Training Clarinet

Training was similar to that in study 3.1. The six selected notes of the clarinet were played to the participant through headphones. Before playing each sound, the experimenter told the participant which frequency would be played. Once each sound had

been played, the participant could ask for any sound to be repeated. Each sound was allowed to be repeated as many times as the participant liked. Training continued until the participant felt confident they could identify each sound.

3.4.4.2. Identifying Clarinet Frequency

For the testing phase each of the six frequencies were played a total of three times, blocked on repetition with the tones presented randomly within repetition. The participant was then asked to identify the frequency of the sound. If they had trouble identifying which frequency was being played, they were allowed to ask for the sound to be played again. However, the sound could only be repeated once before a participant had to identify the frequency.

Studies 2-4: Clarinet with Paired Instrument, Individually Played

3.4.4.3. Training Clarinet/Second Instrument

For each of studies 2-4, after a short break, the participant was told which of the three instruments would be played with the clarinet. They were then given the headphones to put on their ears. Participants were then told which of the paired instruments they would listen to first, clarinet or other instrument: order of the two instruments was random. They were then played each of the six notes for the given instrument, before the notes were played, the experimenter would state the frequency of the note. Once all the frequencies had been played for the instrument, participants were allowed to ask the experimenter to repeat any notes until they were comfortable with the association between the sounds and frequencies. Once participants felt confident they could identify each of the six frequencies played by that instrument, the process was repeated using the second instrument.

After participants felt confident they could identify each of the frequencies of the second instrument, they were asked if they would like to have any of the sounds from either instrument repeated. If they did, then those particular sounds were played. Training continued until the participant felt confident they could identify each sound and instrument.

3.4.4.4 Identifying Clarinet and Paired Instrument, Individually Played

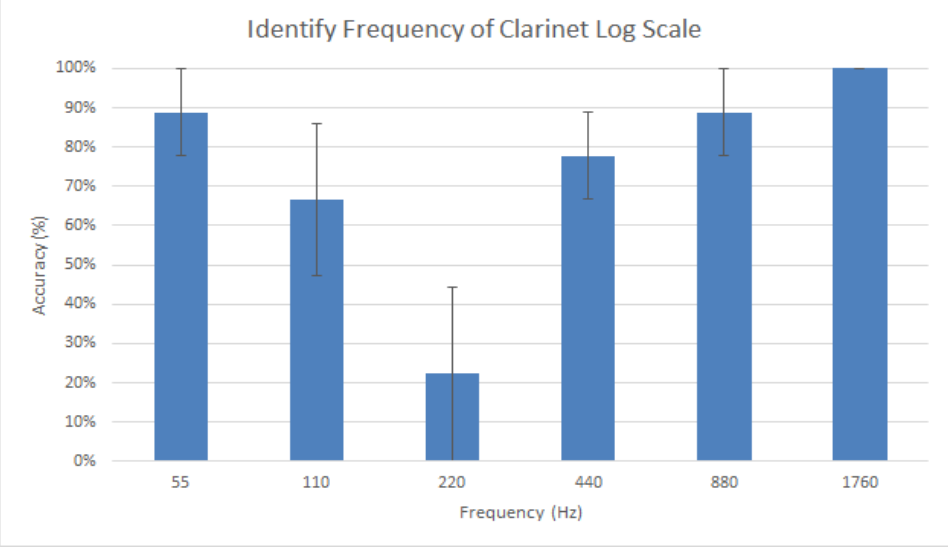
For each pairing, four repetitions of each tone of both instruments (clarinet and paired instrument) were played individually to the participants, blacked on repetition. Within repetition, the order of the tones was random. Participants were required to: 1) determine the instrument being played, and 2) determine the frequency of the note.

3.4.5. Results

3.4.5.1. Study 1: Clarinet Alone

Although a few of the logarithmically spaced notes were able to be accurately determined, participants were quite inaccurate with the remaining notes.

Graph 8: Clarinet Played on a Log Scale

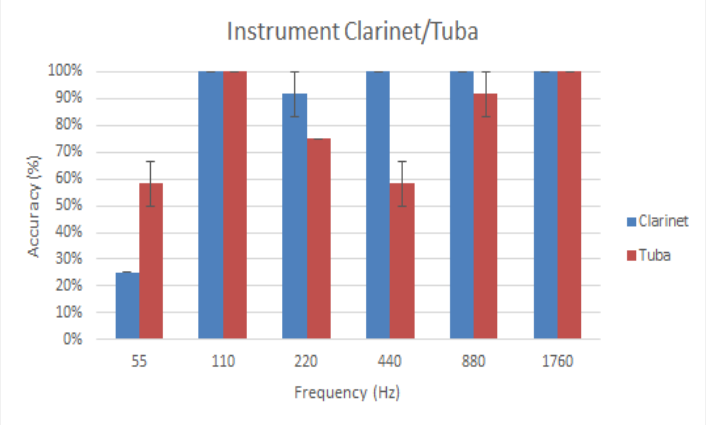


The frequency that people struggled with the most with identifying was the 220 Hz frequency (only slightly above 20%).

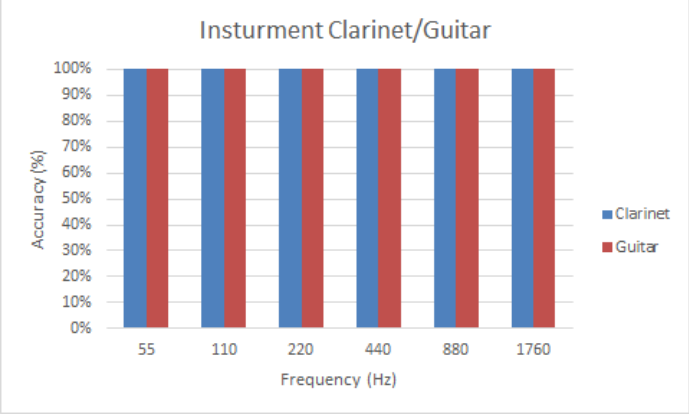
3.4.5.2. Studies 2-4: Clarinet Paired with Second Instrument

For these studies, the clarinet was paired with the tuba, guitar, or bassoon in random order. The first set of graphs (9-11) show the accuracy of identifying the correct instrument of a presented pair for the given notes.

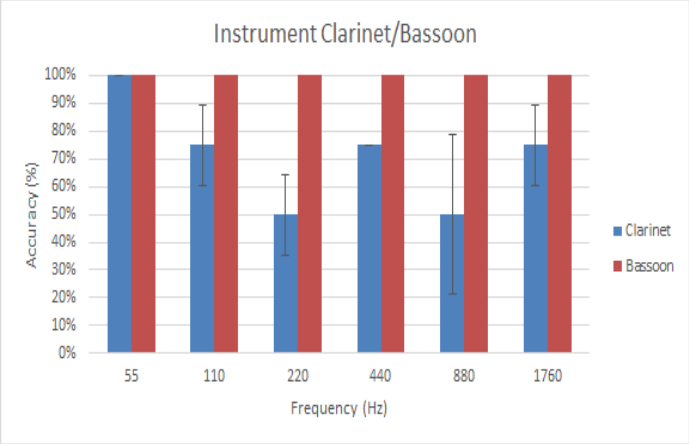
Graph 9: Identifying the Instrument Clarinet/Tuba



Graph 10: Identifying the Instrument Clarinet/Guitar



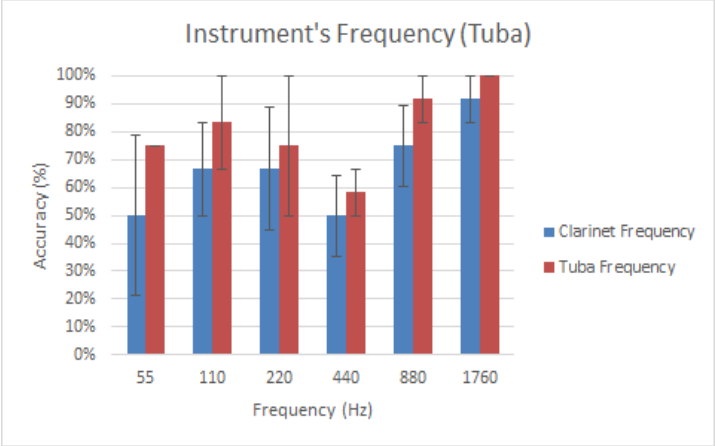
Graph 11: Identifying the Instrument Clarinet/Bassoon



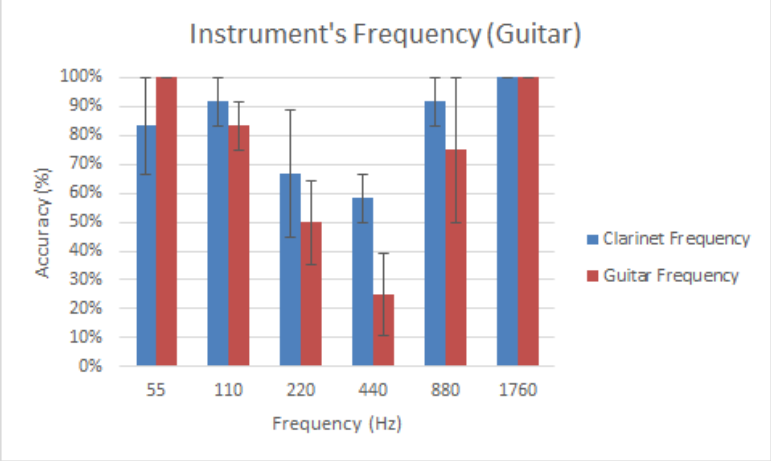
Only the guitar was able to be distinguished from the clarinet greater than 80% of the time for all notes/frequencies. In fact, for all notes/frequencies, the clarinet and guitar were able to be identified with 100% accuracy. Pairing the bassoon with the clarinet decreased performance significantly: although the bassoon was detected with 100% accuracy, participants were only able to distinguish the clarinet with greater than 80% accuracy for a single note/frequency (55 Hz). Pairing the tuba with the clarinet seemed better than with the bassoon, with 3 of the 6 frequencies resulting in an accuracy above 80% (in fact, above 90%) for both instruments.

The second set of graphs show the accuracy in determining the frequencies of the notes presented for both instruments (Graphs 12-14).

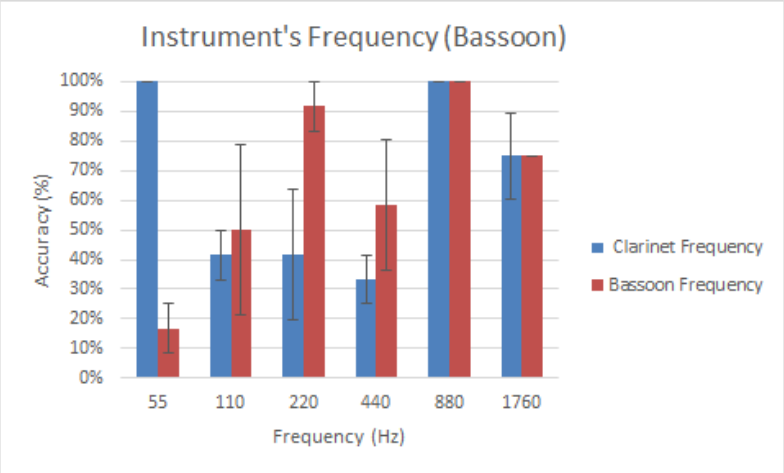
Graph 12: Identifying the Frequency of Clarinet/Tuba



Graph 13: Identifying the Frequency of Clarinet/Guitar



Graph 14: Identifying the Frequency of Clarinet/Bassoon



For all pairs, participants had a hard time identifying the frequency of the instrument being played. For the clarinet-tuba pairing, participants were better at identifying the frequency of a note when the tuba was played compared to the clarinet. Accuracy also appeared to improve, in general, with increasing frequency. For the clarinet-guitar pairing, although people were quickly able to identify which instrument was being played with 100% accuracy, this did not translate to participants having a high accuracy in identifying the frequency. In fact, performance at 440 Hz for the clarinet-guitar pair was worse than any of the pairs for the clarinet-tuba.

3.4.6 Discussion

3.4.6.1. Study 1: Alone

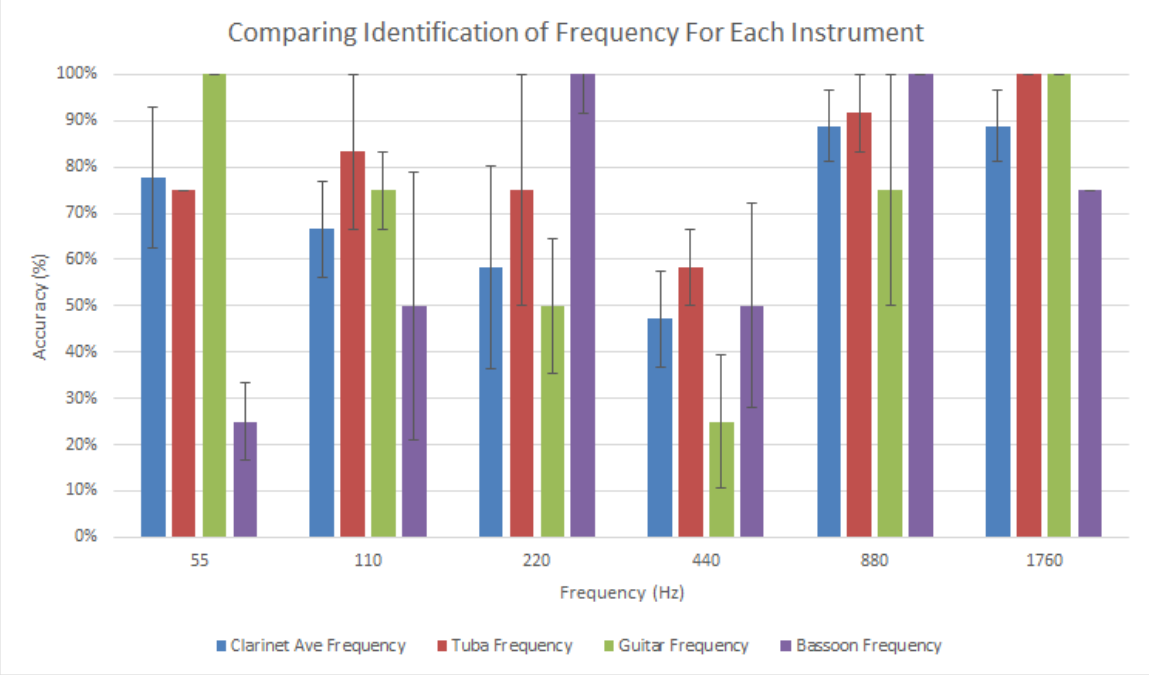
Most puzzling was the fact that accuracy of identifying the mid-frequencies of notes had poor accuracy, with identification being approximately 20% for the note at 220 Hz (Graph 8). This should be contrasted with the accuracy in identifying the 6 notes/frequencies scaled linearly over a smaller range (Graph 1). Although small sizes were used in both studies, the differences at some frequencies are extremely large. This result was unexpected considering the logarithmic form of frequency discrimination for human hearing. One potential problem may have been that the notes/frequencies used corresponded to the harmonic frequencies of the first note (55 Hz). As timbres are made of a weighted sum of the harmonics of a note, this may have produced problems in identification. Better performance may be achievable by selecting notes/frequencies that are spaced apart in an approximate logarithmic fashion but avoiding the harmonics of the other notes.

3.4.6.2. Clarinet/Second Instrument Testing

The results suggest that the pairing of the clarinet with the guitar produced the best performance (100% accuracy for both instruments at all frequencies tested). The fact that performance did not appear to vary with the note/frequency (Graph 10), as with the other pairings (Graphs 9 and 11), suggests that this high performance would remain even when the frequencies to be selected for the map features are further tweaked. This was particularly important as it was clear from Study 1 that the notes/frequencies used needed to be further adjusted to improve performance.

To summarize the accuracy of identifying the different notes/frequencies for all instruments (Graph 15), the average of all the correctly identified clarinet frequencies was taken and then compared to each of the three other instruments played (tuba, guitar, and bassoon). The reason this was done was to help validate the selection of the clarinet-guitar pairing while selecting better notes/frequencies to represent the map features.

Graph 15: Comparing the Average Results of the Clarinet to All Instruments



Comparing the trends across frequency of the different instruments, show that the clarinet and guitar (and only the clarinet and guitar) show a similar pattern. This is supportive of the choice of the clarinet-guitar pairing (in addition to the ability to identify the instruments) as any tweak in the frequency to improve performance with one instrument should also translate to the other instrument as well.

The biggest question that remained was determining how to improve the results in the identification of the notes/frequencies for both instruments. We first reconsidered the number of notes needed to map onto the six map features. Then we considered tweaking the notes/frequencies to avoid harmonics of each other.

When re-examining the mapping for the six map features, it was determined that only five frequencies were essential since one of the features/colors could be represented by no sound. The best feature to be represented with no frequency was determined to be buildings, as the walls of the building prevent looking inside. The only concern was that originally the lack of feedback was to indicate that a map user was off the map. Instead, the map and the border of the map will be indicated by a cut out the size of the map on a screen protector attached to the screen. Detecting the screen protector edge will indicate the edge of the map is reached.

As a result, it was determined that only five frequencies were needed, and not six as originally thought. To determine what frequencies participants confused with each other in studies 2-4 for the clarinet and guitar , a confusion matrix (Graph 16) was created where the results for the clarinet and guitar were weighted equally.

Graph 16: Confusion Matrix for Notes/Frequencies of the Clarinet and Guitar

n=144	55 (Hz)	110	220	440	880	1760
55 (Hz)	21	4	0	0	0	0
110	3	20	6	0	1	0
220	0	0	14	13	2	0
440	0	0	4	10	1	0
880	0	0	0	1	20	0
1760	0	0	0	0	0	24

The confusion matrix shows that participants confused the 110 Hz with the 55 Hz notes and vice versa. Errors in identifying the 220 Hz notes was split between identifying them as the 110 Hz or the 440 Hz notes. However, the 440 Hz notes were mainly confused with the 220 Hz notes. The matrix also shows the 880 Hz note as being the only note being confused with another note other than the one above or below it. This is believed to be a result of a participant rushing through training, since only one participant struggled in identifying the 880 Hz note.

Since the majority of people only confused the 110 Hz note with the 55 Hz note and vice-versa, it was determined that it would be best to eliminate one of the frequencies as only 5 frequencies are needed. The note eliminated was the higher note (at a frequency of 110 Hz) as it was also confused with the 220 Hz notes. For the 220 Hz and 440 Hz notes, it was determined that it would be best to alter their frequencies. To break the harmonic relationship between the 220 Hz notes and 440 Hz notes while keeping an approximate logarithmic spacing, the 200 Hz frequency was changed to 196 Hz and the 440 Hz frequency was changed to 466 Hz. This also broke the harmonic relationship to the 880 Hz notes and the 1760 Hz notes.

3.5. Performance of Two Timbres with Five Notes

In this phase of testing, the instruments used were the clarinet and the guitar. The focus was on examining participant performance with the revised tones and validating their effectiveness for the target population (i.e., individuals who are blind or visually impaired rather than sighted individuals) in being individually distinguishable from the others. Similar to the previous set of studies, this was assessed by the ability of participants to identify the 5 frequencies for the clarinet (Study 1) and to identify the timbre and 5 frequencies when the tones were played individually for the clarinet and guitar (Study 2). A secondary focus was to determine how well participants could determine the frequencies of both instruments under the conditions of the different map methods: (1) one finger to one ear, (2) 2 fingers to one ear, different timbres, (3) 2 fingers to different ears , same timbre and (4) 2 fingers to different ears, different timbres. Results for condition 1 were already being determined by Study 1 above. It was assumed that condition 3, providing the same timbre to 2 separate ears (one for each finger), would have the same performance (although potential issues with this assumption will be considered in the discussion). A further study (Study 3) considered the performance for condition 2 (two different timbres simultaneously played to one ear) and condition 3 (two different timbres simultaneously played separately to each ear).

3.5.1. Participants

There was a total of five participants. All participants were either blind or visually impaired.

3.5.2 Signals

For this experiment, only the clarinet and guitar were used. Each had notes generated with the MuseScore 2 software, with amplitudes of 40 dBA, for frequencies of 55 Hz, 196 Hz, 466 Hz, 880 Hz and 1760 Hz.

3.5.3. Experimental Design

The first test study (Study 1) examined how well the user could identify each of the five notes/frequencies of the clarinet: 3 repetitions of the notes were played, blocked on repetition and randomized within blocks. The second study (Study 2) examined how well the instrument and frequency could be identified when both instruments were played individually at a random frequency: 4 repetitions of the 10 notes were played, blocked on repetition with order of the notes randomized within the blocks. The third study (Study 3) examined how well the frequency of the instruments could be identified when played simultaneously in random pairings to either both ears or one instrument per ear. For the latter condition, the clarinet was always played to the left ear and the guitar to the right ear. Testing was performed for each condition in counterbalanced order across participants. For each condition, 3 repetitions of all combinations of tones (5x5) were played, blocked on repetition with order of frequency pairs randomized with blocks.

3.5.4. Experimental Procedure

Study 1: Identifying Frequencies of the Clarinet

3.5.4.1. Clarinet 5 Frequency Training

Training was similar to that in Experiment 3.4 Study 1. The five selected notes of the clarinet were played to the participant through headphones. Before playing each sound, the experimenter told the participant could ask for any sound to be repeated. Each sound was allowed to be repeated as many times as the participant liked. Training continued until the participant felt confident they could identify each sound.

3.5.4.2 Clarinet 5 Frequency Testing

Testing was similar to that in Experiment 3.4 Study 1. For the testing phase, each of the five notes were played a total of 3 times, blocked on repetition with the tones presented randomly within repetition. The participant was then asked to identify the frequency of the sound. If they had trouble identifying which frequency was being played, they were allowed to ask for the sound to be played again. However, the sound could only be repeated once before a participant had to identify the frequency.

Study 2: Identifying Timbre and Frequency of Clarinet and Guitar

3.5.4.3 Individual Instrument 5 Frequency Training

Training was similar to that for Experiment 3.4 Studies 2-4, although only the clarinet and guitar were used in this study. After a short break, the participant was given the headphones to put on their ears. Participants were then told which of the two instruments they would listen to first, the clarinet or guitar; order of presentation of the two instruments was random. They were then played each of the five notes for the given instrument in random order, before the notes were played the experimenter would state

the frequency of the note. Once all the frequencies had been played for the instrument, participants were allowed to ask the experimenter to repeat any notes until they were comfortable with the association between the sounds and frequencies. Once participants felt confident they could identify each of the five frequencies played by that instrument, the process was repeated using the second instrument.

Once participants felt confident they could identify each of the frequencies of the second instrument, they were asked if they would like to have any of the sounds from either instrument repeated. If they did, then those particular sounds were played. Training continued until the participant felt confident they could identify each sound and instrument.

3.5.4.4 Individual Instrument 5 Frequency Testing

Testing was similar to that in Experiment 3.4 Studies 2-4, although only the clarinet and guitar were used. Four repetitions of each tone of both instruments (clarinet and guitar) were played individually to the participants, blocked on repetition. Within repetition, the order of the tones was random. Participants were required to: 1) determine the instrument being played, and 2) determine the frequency of the note. If they were unsure of the frequency or the instrument being played they were allowed to have the note repeated once before having to provide an answer.

Study 3: Identifying Timbre and Frequency of Clarinet and Guitar When Played Simultaneously

3.5.4.5. Simultaneous Instrument Training

After a break, the condition that was to be used was described to the participants. Then participants were asked to put on the headphones with the labeled left side going to the left ear and the labeled right side going to the right ear. The experimenter then verified that the participant was wearing the headphones correctly.

The twenty-five combinations of simultaneous clarinet/guitar tones were learned in the following manner:

1. Starting with lowest frequency for the clarinet,
    a. It was played with each of the tones of the guitar from lowest to highest.
    b. After each pair of tones was played, the participant was told the frequency of the left and then the right ear.
    c. After all pairs with the lowest frequency for the clarinet were played and stated, the participant was asked if they wanted any of the five combinations replayed. Any requested combinations were replayed.
    d. The participant was asked to continue to request combinations until they were comfortable in identifying all the frequency pairs played.
2. Training was then moved to the next lowest frequency of the clarinet, for which the procedure described in a-d was repeated.
3. When training was completed for all combinations of frequencies, the participant was asked if they wanted to listen to any of all the possible combinations again. If they wanted to listen to any of the combinations, they were then played those combinations again. This continued until they felt they were able to identify all frequency combinations.

3.5.4.6. Simultaneous Instruments Testing

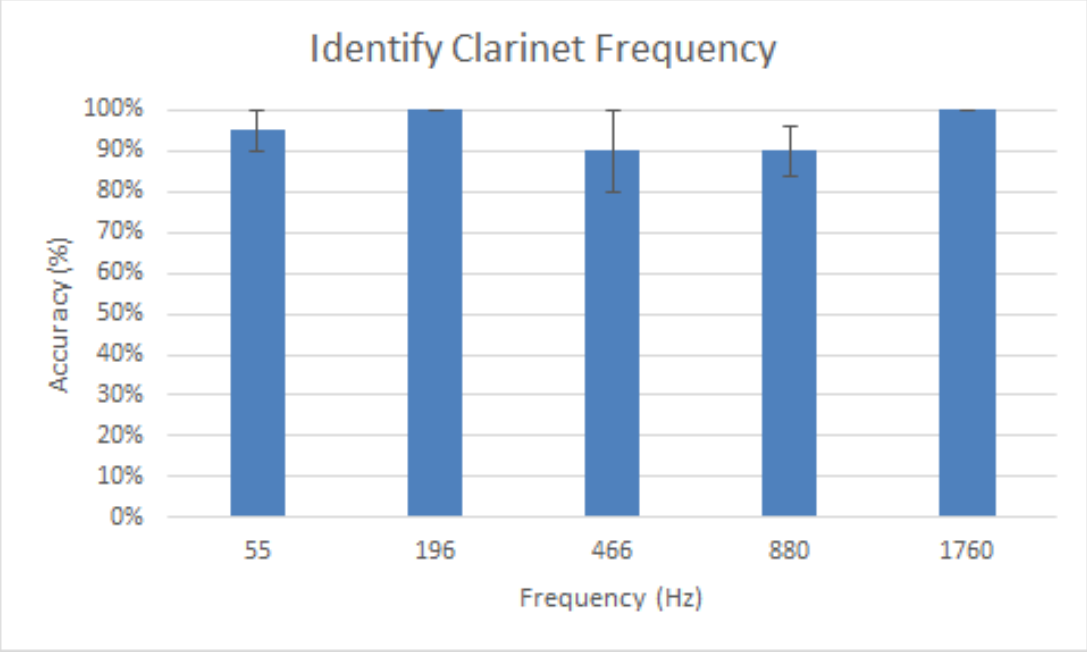
After training for a given condition, testing occurred for that same condition. Testing consisted of 3 repetitions of all combinations of tones (25) blocked on repetitions with order of the combinations random within blocks. After each pair of tones were played simultaneously, the participant was asked to identify each of the frequencies played by both instruments. If they were unsure of the frequency played by either one or both instruments, participants were allowed to ask for the sound to be replayed once before having to identify the frequency of each instrument.

3.5.5. Results

Study 1: Identifying Frequencies of the Clarinet

Participants were able to identify the frequencies of the clarinet with over 90% accuracy (Graph 17).

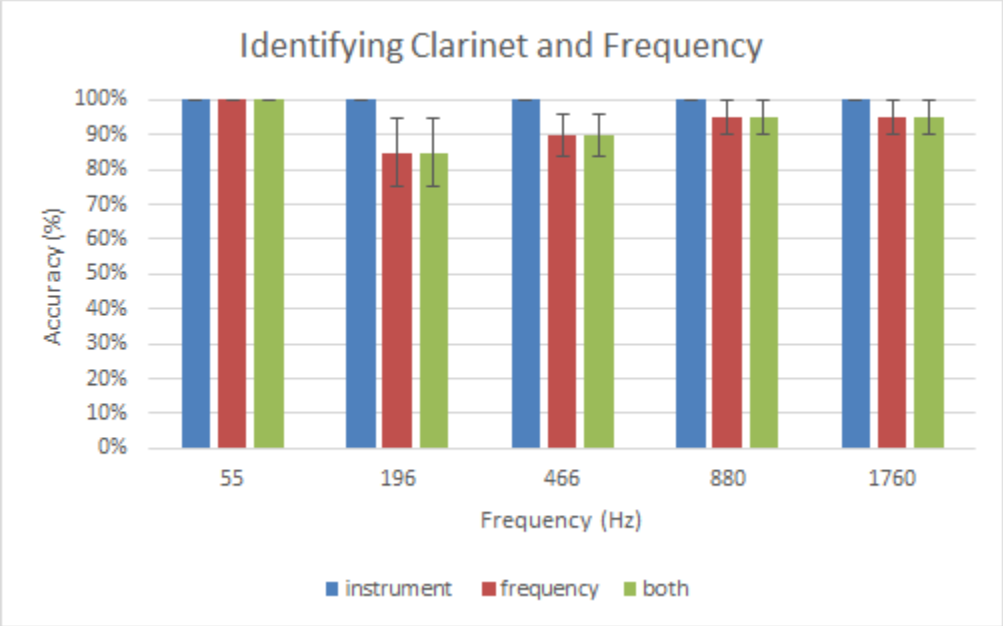
Graph 17: Identified Clarinet Frequency



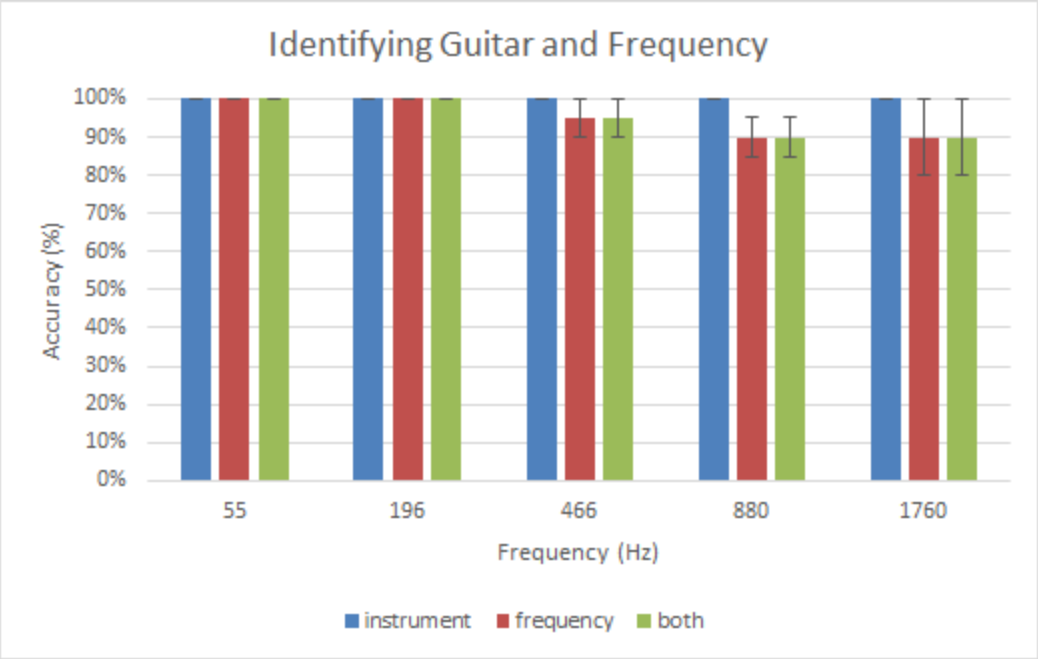
Study 2: Identifying Timbre and Frequency of Clarinet and Guitar

When the clarinet and guitar were played individually everybody was able to correctly identify the instrument that was being played (Graphs 15 and 16). Results for the clarinet also revealed that four of the frequencies were correctly identified at a rate of 90% or above, with only one frequency, 196 Hz, identified at an 85% rate (Graph 18). Results for the guitar revealed that participants could identify all five frequencies with an accuracy above 90% (Graph 19).

Graph 18: Identifying The Clarinet and The Frequency Played



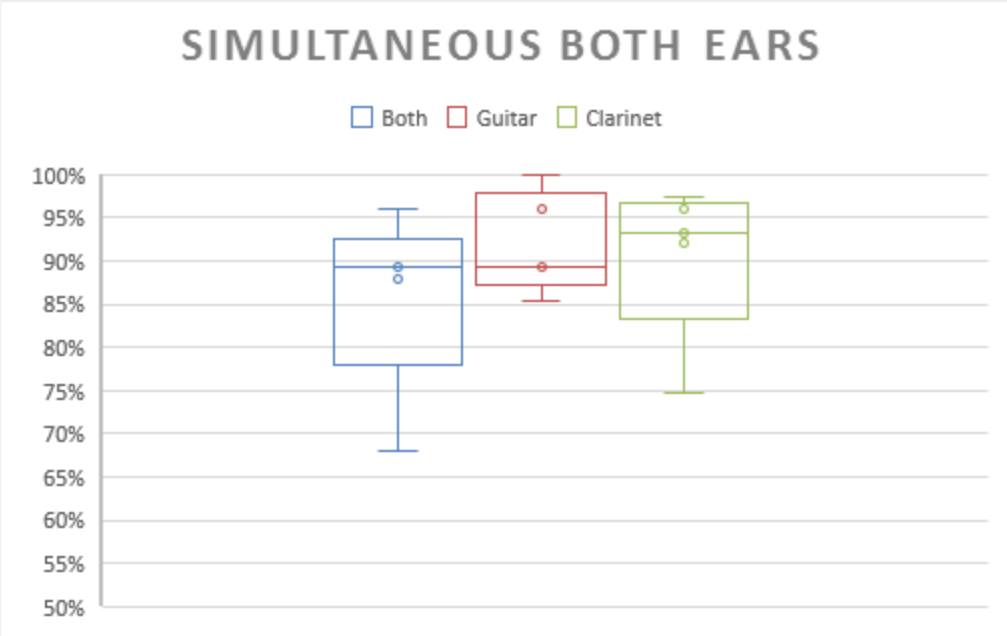
Graph 19: Identify The Guitar and The Frequency Played



Study 3: Identifying Timbre and Frequency of Clarinet and Guitar When Played Simultaneously

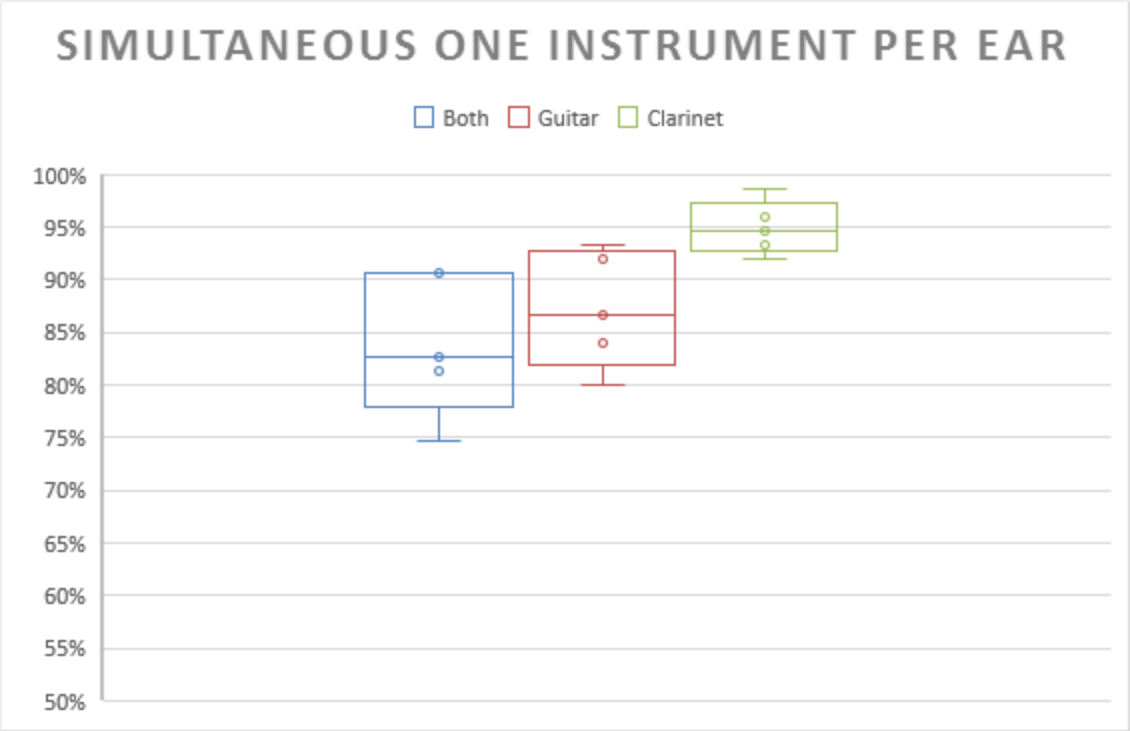
Accuracy in identifying frequencies of the instruments when pairs of notes were played together to both ears had a median higher than 90% (Graph 20). A box plot, which gives median rather than mean, was chosen to display the data because one participant had particular difficulty in identifying the frequencies of the clarinet and was an outlier compared to the other participants.

Graph 20: Simultaneous Both Ears



Accuracy in identifying frequencies of the instruments when pairs of notes were played simultaneously, but each instrument to its own corresponding ear had a median higher than 85% (Graph 21). Median accuracy of determining the frequencies of both notes of a pair was slightly lower in accuracy at approximately 83%. A box plot, which gives median rather than mean, was chosen to display the data in order to enable a comparison to Graph 20.

Graph 21: Simultaneous One Instrument Per Ear



3.5.6. Discussion

Study 1 found performance to be at or above 90% correct for identifying all frequencies. This was substantially higher than for Study 1 of Experiment 4. This suggests that the redesign, taking into consideration the effect of harmonics, was successful. Performance was also higher than for the linearly spaced frequencies in Experiment 1, although this may also be due to the fact that only 5, rather than 6, frequencies were used in this study (as compared to Experiment 1). Study 2, found that the identification of the timbre was 100% accurate and the identification of the frequencies of both instruments was over 90% accurate, except for the clarinet at 196Hz (Which was 85% correct). This was substantially better than for Experiment 4 for the clarinet and guitar (Graph 13). Again, this suggests that the modifications to avoid harmonics of the other frequencies were very beneficial to improving performance.

With how well people performed on both the single and two individual instruments it was determined that these were the individual frequencies that would be used.

Study 3 found that when both instruments were played simultaneously to either both ears together or one to each ear separately, performance did not seem to drop significantly compared to performance when each instrument was played in isolation (Study 2). This suggests that a) the selected parameters are acceptable for the main study and b) any difference in performance with the different auditory finger/ear configurations in using navigation maps in the main study will not likely be due to perceptual differences in determining the frequencies of the tones. Interestingly, performance was lower when the two instruments were played to separate ears rather than the same ear. This was surprising as it was expected that it would be easier to identify the sounds if they were spatially separated. However, participants said that they found themselves paying more attention to one ear over the over. This is consistent with performance being higher and with less variation of the clarinet than the guitar in this condition, resulting in poorer overall performance.

To try and help people who did have some difficulty identifying frequencies, it was decided that spoken feedback, for the feature corresponding to the frequency will be given in the map exploration program when needed: i.e., the frequency will be announced when a participant's finger is lifted off the screen.

4. Selection of Tactile Parameters

The two dimensions of features and exploring fingers also needed to be selected for the tactile parameters. As timbre is not well perceived in the tactile dimension, the only dimension considered for mapping sensations from each exploring finger was spatial location. Custom vibrators, described in section 2, were used on the proximal portion of the distal phalanges. The natural mapping of providing information from the exploring finger to the vibrator on the same finger was used. The dimension chosen to be used to represent the features was the use of tactile "notes". Since it was determined that only five colors needed to be represented with sound, only five "notes" had to be selected for the vibration feedback.

As the vibrators were linear resonant actuators (similar to tablet vibrators), the vibration frequency was set at the resonance frequency. Different tones were considered by the temporal modulation of the vibration using square waves that could vary in frequency and duty cycle: this corresponded to controlling the on (duration) and off (delay) component of the vibration. Previous work in our laboratory (Burch and Pawluk, 2011) found that using a logarithmic scale for vibration frequency was most effective for selecting values that could be easily differentiated. A logarithmic scale was chosen here for the modulation frequency as it appeared to be the dominant tempo perceived.

4.1. Vibration "notes" on a Single Finger

This test considered participant performance in being able to identify the 5 different "notes" selected for the 5 features.

4.1.1. Participants

There were four people who participated.

4.1.2. Signals

The five vibration signals created had square wave modulation frequencies of 2 Hz, 4 Hz, 8 Hz, 16.13 Hz, and 32.26 Hz. The duty cycle was also varied between the different frequencies used as they produced different tempos more easily distinguished than if the duty cycle was kept constant. The parameter values are given in Table 4.

Table 4: Initial 5 Frequencies For Tactile Testing

Frequency (Hz)	Amplitude (μm)	Duration (ms)	Delay (ms)
2	40	499	1
4	40	125	125
8	40	25	100
16.13	40	40	22
32.26	40	10	21

4.1.3. Experimental Design

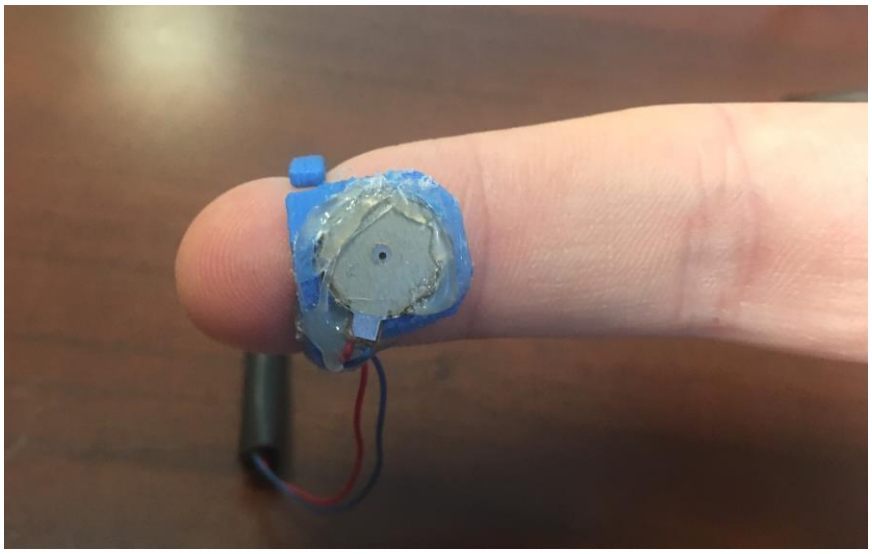
Four repetitions of all vibration signals were played to the participant, blocked on repetition, with the 5 vibrations played in random order within each block. Participants were asked to identify the frequency of each signal.

4.1.4. Experimental Procedure

4.1.4.1. Training

The person was given one of the 3D printed rings which held the linear actuator (Figure 4.1) to place on the distal phalanx of the palm side of their index finger. The experimenter insured that the vibrator was placed on the proximal portion of the distal phalanx so that the distal tip was still able to contact the touch screen.

Figure 4.1: 3D Printed Ring Oriented Properly on the Index Finger



Then the test subject was oriented to face away from the computer controlling the test. Participants were then played each of the 5 vibrations in order. After each vibration was played, the frequency was spoken to the participant by the experimenter. If a participant wanted to listen to a specific frequency again, they were allowed to ask the experimenter to play that specific frequency again. Once the participant felt they could identify each of the five frequencies, testing was begun.

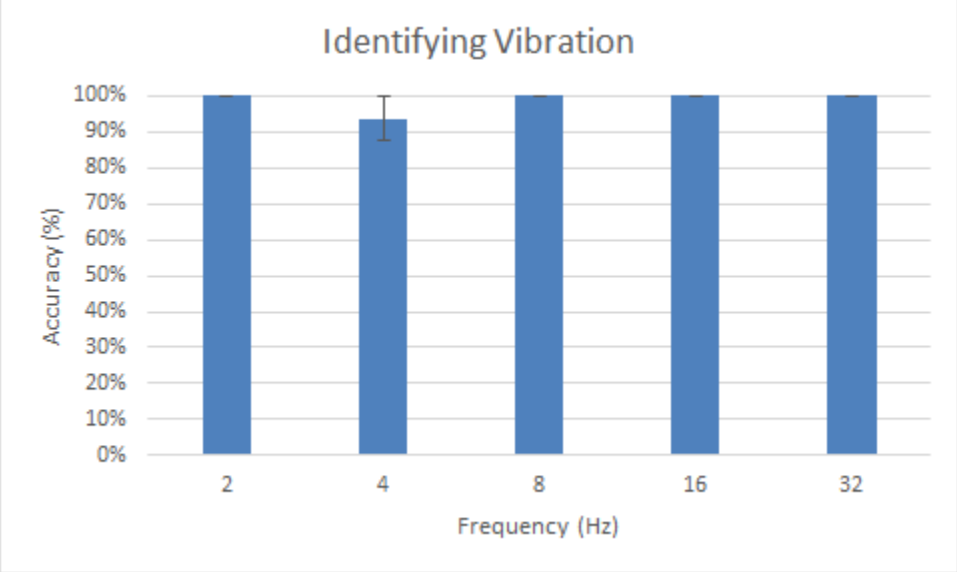
4.1.4.2. Testing

For testing, participants were played four repetitions of the five signals blocked on repetition, with each the order within repetition random. A signal would be played on the finger. Then the participant was asked to identify the frequency of the vibration. Participants could ask for the vibration to be replayed once if they needed it to.

4.1.5. Results

Participants were easily able to identify all of the signals being played (Graph 22). Only the 4 Hz signal was misidentified once (25% of the time) while all the other frequencies identified correctly each time they were played.

Graph 22: Identification of the Five Frequencies Played



4.1.6. Discussion

With participants doing so well with identifying the frequencies played, it was decided that two finger testing could begin with the current frequencies being used.

4.2. Two Finger Testing

This experiment considered performance of the actual targeted user group: individuals who are blind or visually impaired. The experiment had 2 studies: Study 1 was very similar to the single finger testing in 4.1. Participants were required to determine the frequencies being played. Study 2 examined performance when both fingers were simultaneously used. This was done to ensure masking effects from the other finger did not affect perception of a note on a given finger. Participants were required to determine whether the same frequency was played on both fingers.

4.2.1. Participants

There were five people who participated in testing. Each person was either blind or visually impaired.

4.2.2. Signals

The six vibration signals used were no signal, 2 Hz, 4 Hz, 8 Hz, 16.13 Hz, and 32.26 Hz. Vibrations occurred in response to a participant contacting a colored map placed on a tablet (Nexus 10) screen. The vibration "note" depended on the color contacted by the fingertip. Correspondence between the colors and "notes" are given in table 5.

Table 5: Six Frequencies Used For Map Testing

Color	Frequency (Hz)	Amplitude (μm)	Duration (ms)	Delay (ms)
White	0	0	0	0
Green	2	40	499	1
Yellow	4	40	125	125
Gray	8	40	25	100
Red	16.13	40	40	22
Blue	32.26	40	10	21

4.2.3. Experimental Design

Study 1 consisted of four repetitions of having the participant access the 6 colors/“notes” with their index finger, where the order of “notes” was randomized within repetition. Participants were required to identify the signals. Study 2 required participants to access the 6 colors/”notes” with both fingers “simultaneously”. Two repetitions of all combinations of notes (6x6) were played to the fingers, where the order of pairs played was randomized within repetition. Participants were asked if the notes played to the two fingers were the same or different.

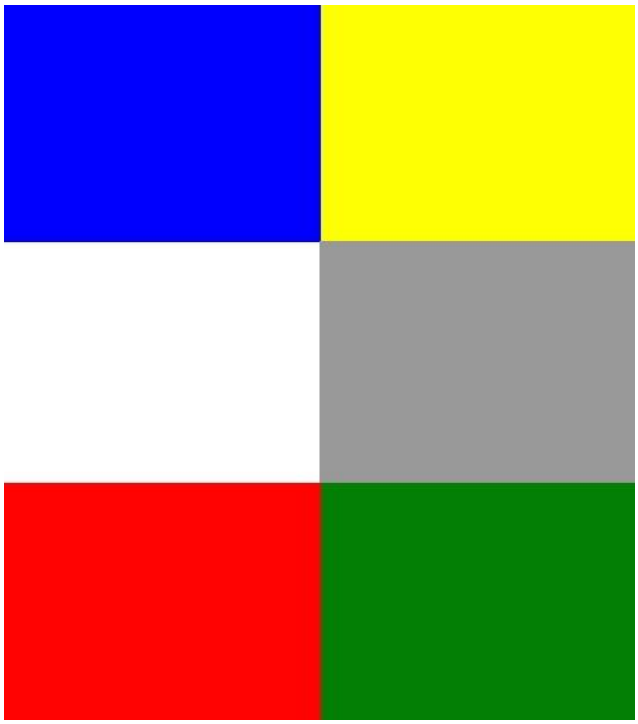
4.2.4. Experimental Procedure

4.2.4.1. Study 1: Sinle Finger Training

The person was presented with one of the vibration rings to place on the distal phalanx of either index finger (Figure 4.1). The experimenter made sure that the vibrator was placed properly. If a participant was visually impaired but not blind, a blindfold was then given to them to be placed over their eyes.

Once the blindfold was on the participant, the training map (Figure 4.2) was presented to them on the tablet. The experimenter then guided the participant's finger to contact each of the six colors on the map in turn. The participant was allowed to feel the vibration and then was told the color their finger was over. Participants were allowed to feel any vibration "note" again by naming the color associated with the note. After a participant was comfortable that they had learned the different vibrations, testing began.

Figure 4.2: Image of Initial Map

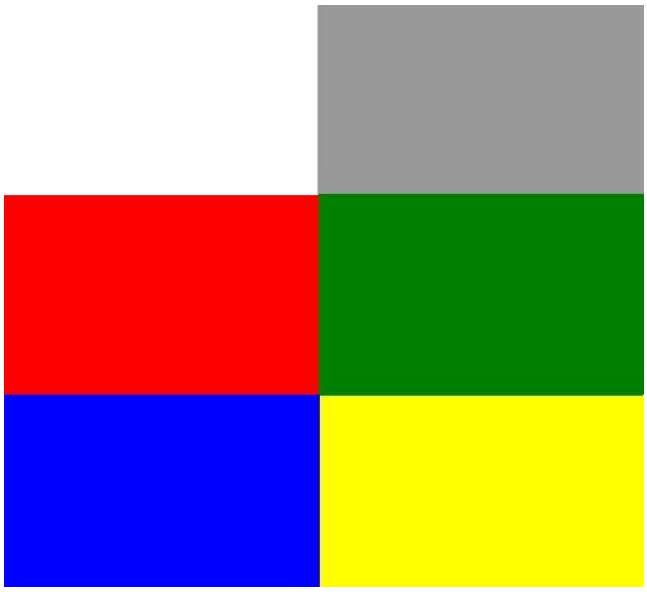


4.2.4.2. Study 1: Single Finger Testing

The test map was chosen randomly from one of 3 test maps, where all test maps consisted of a 3x2 grid of the color blocks with the same colors but in a different arrangement (e.g. Figure 4.3). The experimenter guided the participant's finger to each of the colors to be tested. Four repetitions of all colors/notes were given, blocked on repetition. Within each repetition, the order in which the colors were tested was random.

For each color, the participant was allowed to feel the vibration and then they were required to name the color to which it corresponded.

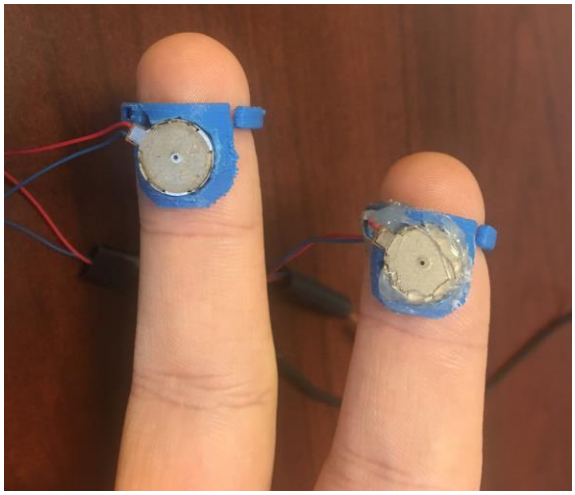
Figure 4.3: Example of One Finger Test Map



4.2.4.3. Study 2: Two Finger Training

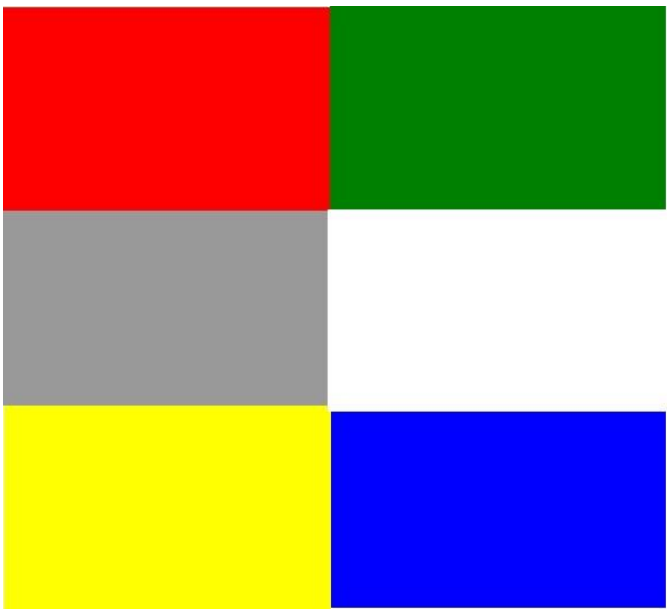
A short break occurred before testing began. At the start of the study, the participant was given two of the ring vibrators to mount on the index and middle fingers of the same hand (Figure 4.4). After the experimenter made sure that both rings were placed properly, the participant was presented with the blindfold to again put over their eyes if necessary.

Figure 4.4: 3D Printed Rings Placed Properly on the Index and Middle Finger



When the blindfold had been placed over the eyes the Two Finger Initial Map (Figure 4.5) was presented to the participant. First, the participant's index and middle fingers were guided together to each of the six color blocks in turn, with the fingers placed on the same color. The "note" for the color was played on each finger as soon as the finger contacted the color block. After the color was played to both fingers, the participant was told the name of the color. Once finished, participants were asked if they wanted feel any of the notes again by stating the corresponding color. When they felt confident they could identify each of the colors played to both fingers, the second part of the two finger training began.

Figure 4.5: Two Finger Initial Map



For the second part of training, the experimenter placed the participant's index finger on a particular color block, at which point its vibrator started playing the "note". With the participant's finger held on the color, the experimenter guided the participant's middle finger to each of the five other color blocks in turn. When each of the color blocks were touched with the middle finger, the appropriate "note" was played. At the same time, the participant was told the name of the color contacted by each finger. Once completed, the participant was then asked if they wanted to feel any of the combinations again with the index finger on the given color block. If yes, their middle finger was placed on any color block they wanted to feel again. If no, their index finger was moved to the next color block and the overall process was repeated until the index finger had been placed over all six colors.

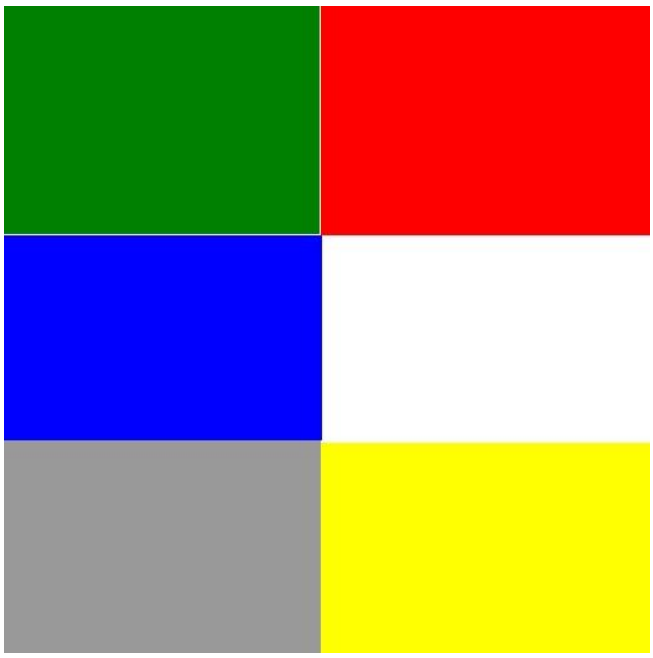
When the process was completed, participants were asked if they wanted to feel any combinations again. If yes, they felt the requested combinations again, with the index

finger being placed first and then the middle finger. This continued until participants felt ready to proceed to the testing phase.

4.2.4.4. Study 2: Two Finger Testing

The Two Finger Test Map was chosen at random from a set of 3 maps (e.g., Figure 4.6); all maps consisted of an arrangement of 3x2 colored blocks, but with different arrangements of placement of the colors. Then, first the test subject's index finger and then the middle finger was guided to a pair of colors on the tablet screen. They were asked to identify if both fingers were on the same or different colors. Two repetitions of the 6x6 combinations of colors blocked on repetition were performed, where the ordering of the pairs was random within repetition.

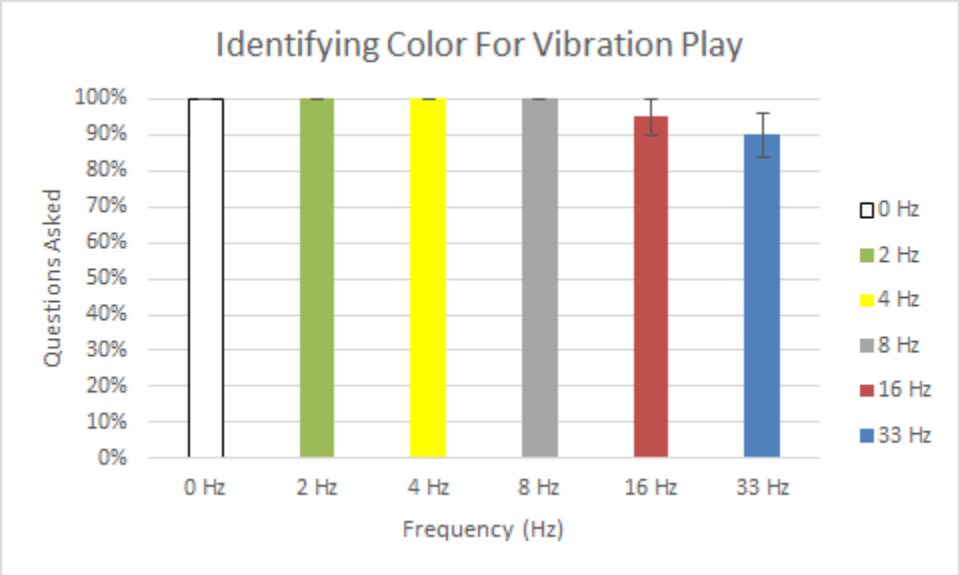
Figure 4.6: Example of Two Finger Test Map



4.2.5. Results

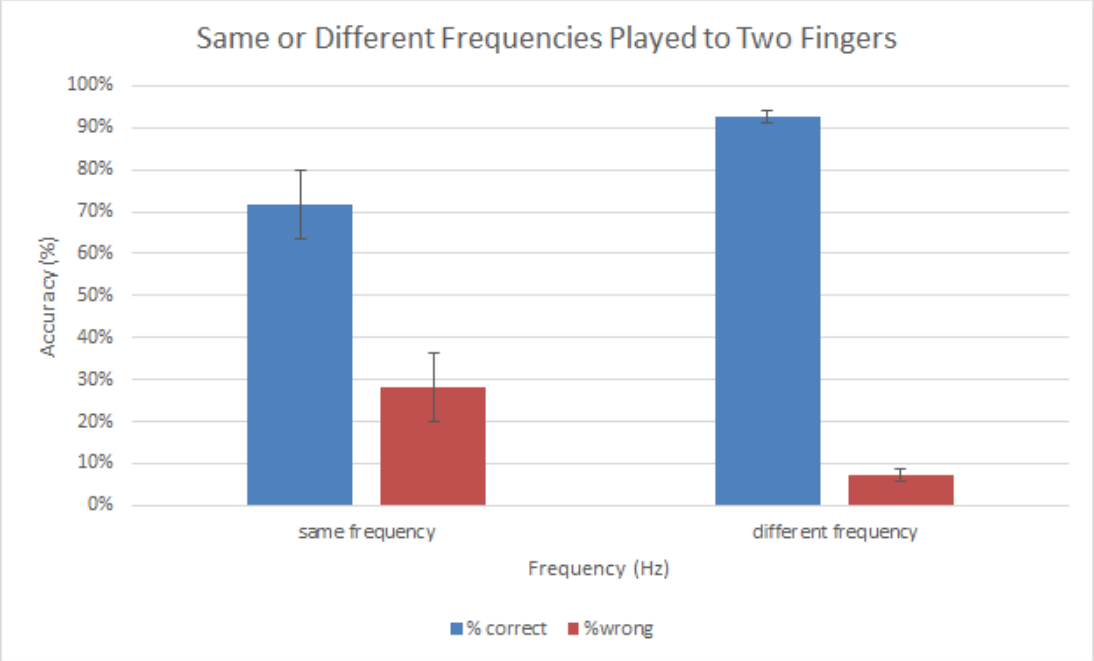
Study 1. When it came to identifying each color associated with each frequency, people were easily able to identify each color (Graph 23). The two hardest colors to identify were the red and blue colors which were the 16 Hz and 33 Hz vibrations. The red color was identified 95% of the time and the blue frequency was identified 90% of the time. Each of the other frequencies were identified each time they were felt.

Graph 23: Identifying Color of Frequency



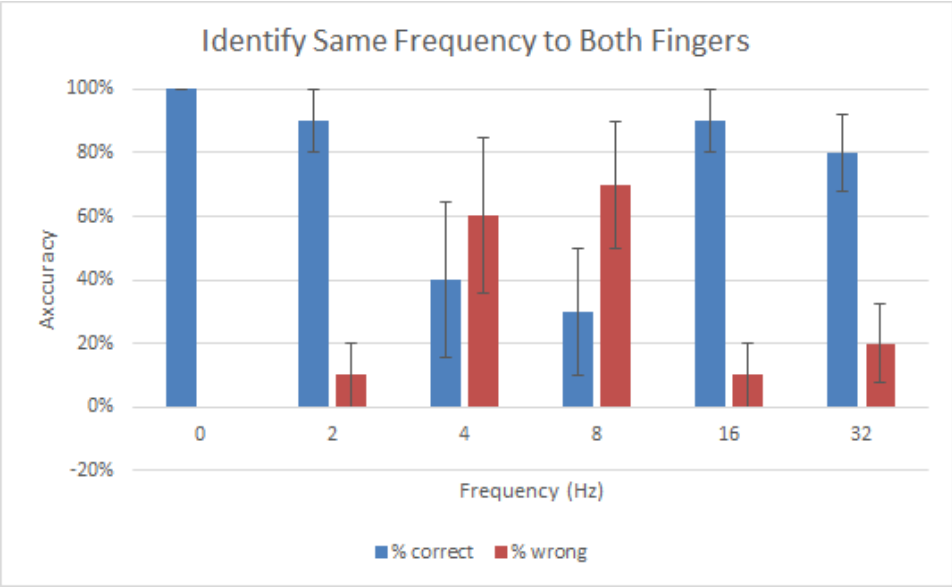
Study 2. For the identification of whether two notes played to two different fingers were the same or different: participants were able to identify when the notes were different 92% of the time they were played. However, participants were only able to identify when the notes were the same 71% of the time (Graph 24).

Graph 24: Same or Different Frequencies Played to Multiple Fingers



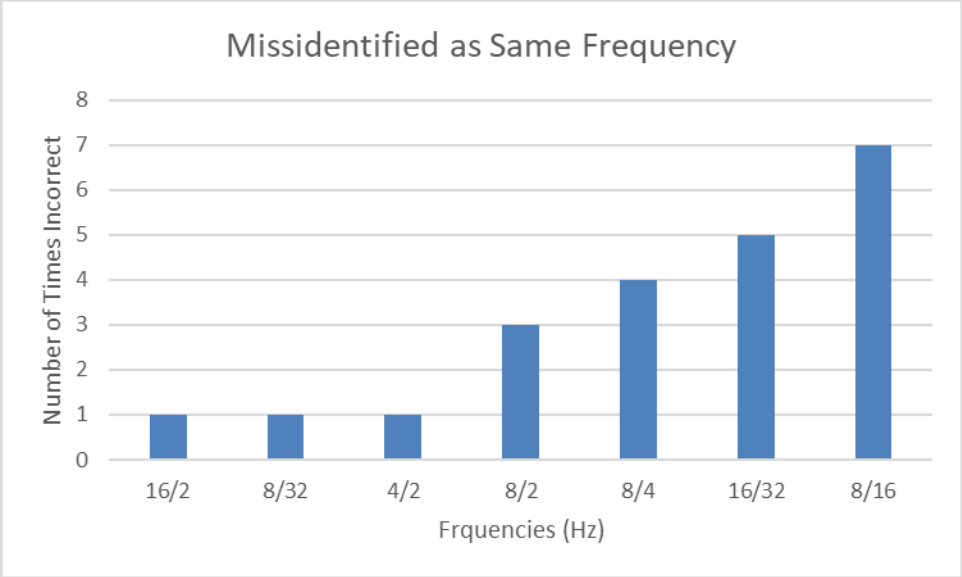
When looking at performance when only the same note was being played to both fingers (Graph 25), participants struggled to identify the 4 Hz signal (40% correct) and the 8 Hz signal (70% correct).

Graph 25: Error Identifying Same Frequency



Examining different frequencies that were identified as the same frequencies (Graph 26), found that the 8 Hz frequency and the 16 Hz frequency vibrations were the most likely to be confused.

Graph 26: Frequencies Played That Were Identified as the Same



4.2.6. Discussion

Study 1. Testing using a map on a tablet showed that people were easily able to identify the five frequencies that were used with only one finger used. It was noticed early in testing that people preferred naming the color to identify the frequency rather than identifying the frequency felt. As a result, in the later tests, participants were only told the color instead of both the frequency and the color associated with it.

Study 2. For notes being played simultaneously on both the index and middle fingers where participants had to determine if the notes were the same or different, participants struggled to answer correctly, particularly when the same note was played to both fingers. In particular, they struggled to identify the gray (4 Hz) and yellow (8 Hz) notes. This may have been due to the fact that participants seemed to have more difficulty

determining whether the notes of the two fingers were similar when they were not vibrating in phase, such as what occurs when one finger is placed on a given color before the second one is placed on it. This may be particularly true for lower frequencies (such as 4 Hz and 8 Hz) where participants could distinguish more of the temporal pattern. This did not occur for the 2 Hz signal, which was essentially a constant vibration. Participants also had some difficulty in identifying the 32 Hz note being the same on the two fingers. This is thought of having more to do with the similarities to the 16 Hz note difficulties interpreting out of phase vibrations as the same. In fact, the 16 Hz and 32 Hz notes were two of the most confused frequencies (Graph 30). The reason for this was not clear. However, most of the misidentified frequencies appeared to be adjacent notes. This could be because, for adjacent frequencies, one was the first harmonic of the other, which could potentially lead to confusion.

4.3. Adjusted Note Testing

In order to improve performance, the frequency of adjacent "notes" were adjusted so that they would no longer be harmonics of each other. The delay (off portion) of the notes was decreased and made consistent as exploratory assessment suggested that this made it easier to detect the same note between two fingers even if the vibrations were out of phase.

In this experiment, participants were asked to identify the color (frequency) that was being played to either a single finger or both fingers when exploring a map. Each participant was first trained using the training map used in previous studies. Then they used a testing map which had multiple blocks of each color of random size and location.

This prevented participants from using location as a cue for determining the color, as in the original test maps.

4.3.1. Participants

There were a total of five participants all were blind or visually impaired.

4.3.2. Signals

The correspondence between the notes used for testing and the colors on the tablet were: 0 Hz (White), 2 Hz (Green), 3.Hz (Gray), 8 Hz (Yellow), 10 Hz (Red), and 32.26 Hz (Blue). Details of the parameters are given in Table 6.

Table 6: Adjusted Frequency Testing

Color	Frequency (Hz)	Amplitude (μm)	Duration (ms)	Delay (ms)
White	0	0	0	0
Green	2	35	490	10
Gray	3.3	40	250	50
Yellow	8	50	100	25
Red	10	45	50	50
Blue	32.26	40	10	21

4.3.3. Experimental Design

This was a repeat of Experiment 3.3. The difference was the use of 6 new notes and new testing maps. The new testing maps avoided the correlation between a color block and location on the map by using multiple color blocks of different sizes in random locations. Study 1: Single Finger Exploration. This consisted of four repetitions of having the participant access different blocks 6 colors/“notes” with their index finger, where the

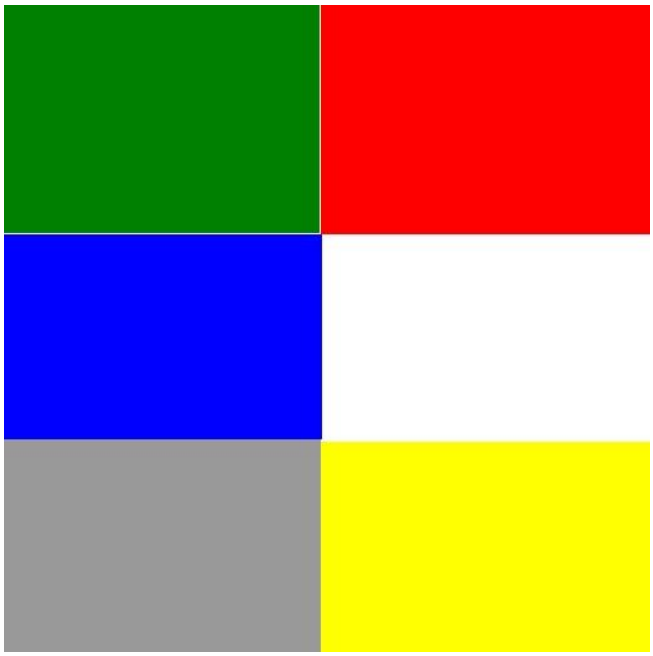
order of “notes” and their location was randomized within repetition. Participants were required to identify the signals. Study 2: Two Finger Exploration. This required participants to access the 6 colors/”notes” with both fingers “simultaneously”. Two repetitions of all combinations of notes (6x6) were played to the fingers, where the order of pairs played and their location was randomized within repetition. Participants were asked if the notes played to the two fingers were the same or different.

4.3.4. Experimental Procedure

4.3.4.1. Study 1: Single Finger Training

Training was as described in Section 4.2 Study 1. The training map is given below (Figure 4.7).

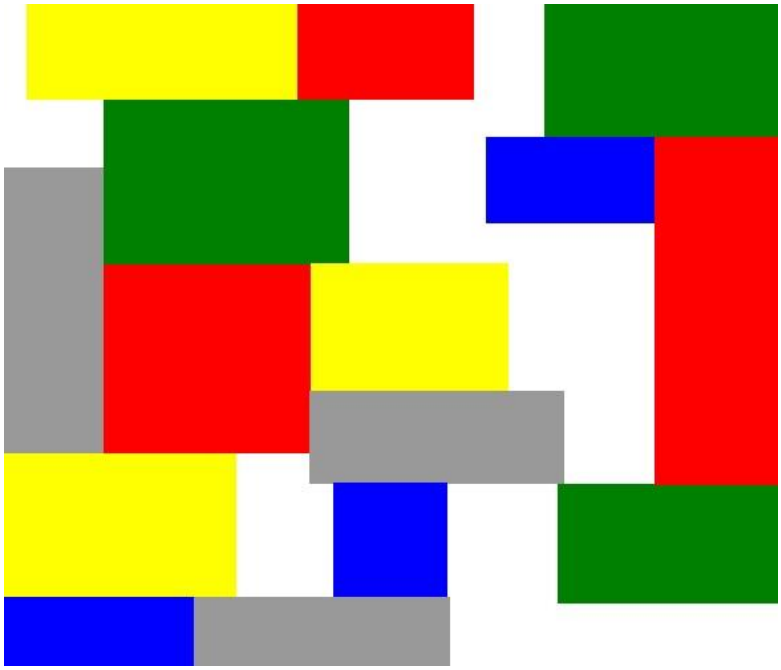
Figure 4.7: Image of Initial Map



4.3.4.2. Study 1: Single Finger Testing

The testing protocol was very similar to that in Section 4.2 Study 1. The difference was that each of the testing maps had multiple color blocks of each color which were of random size and location (e.g., Figure 4.8). For a given color block in each repetition, the color block on the map that was used was randomly selected from the 3 or more color blocks of that color.

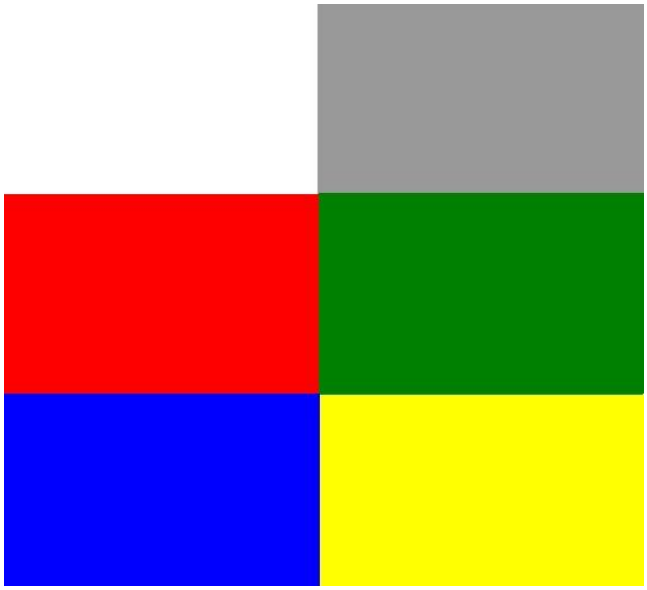
Figure 4.8: Example of One Finger Test Map



4.3.4.3. Study 2: Two Finger Training

Training was as described in Section 4.2 Study 2. The training map is given below (Figure 4.9).

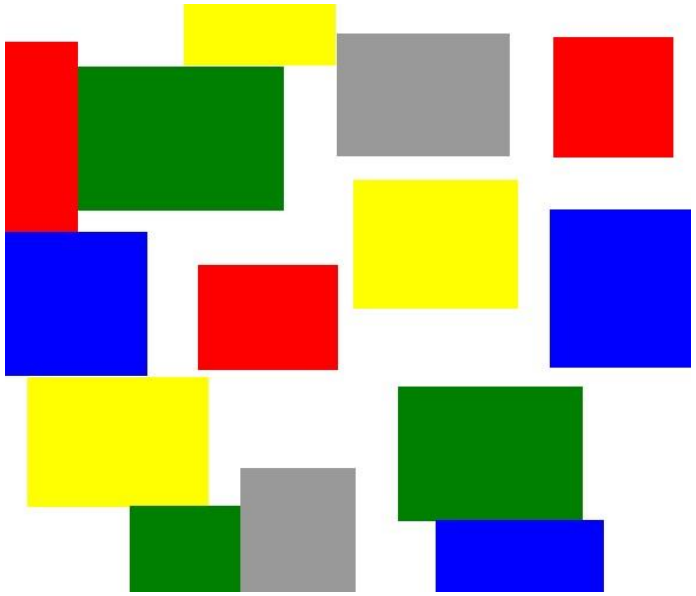
Figure 4.9: Two Finger Initial Map



4.3.4.4. Two Finger Test Map

The testing protocol was very similar to that in Section 4.2 Study 2. However, rather than stating whether the colors contacted by the two fingers were the same or different, participants were required to give the color contacted by each finger. In addition, each of the testing maps had multiple color blocks of each color which were of random size and location (e.g., Figure 4.10). For a given color block in each repetition, the color block on the map that was used was randomly selected from the 3 or more color blocks of that color.

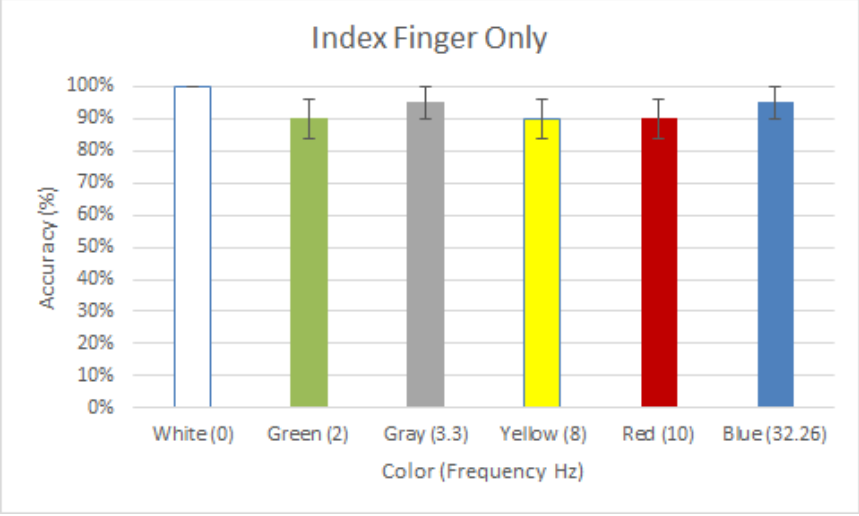
Figure 4.10: Example of Two Finger Test Map



4.3.5. Results

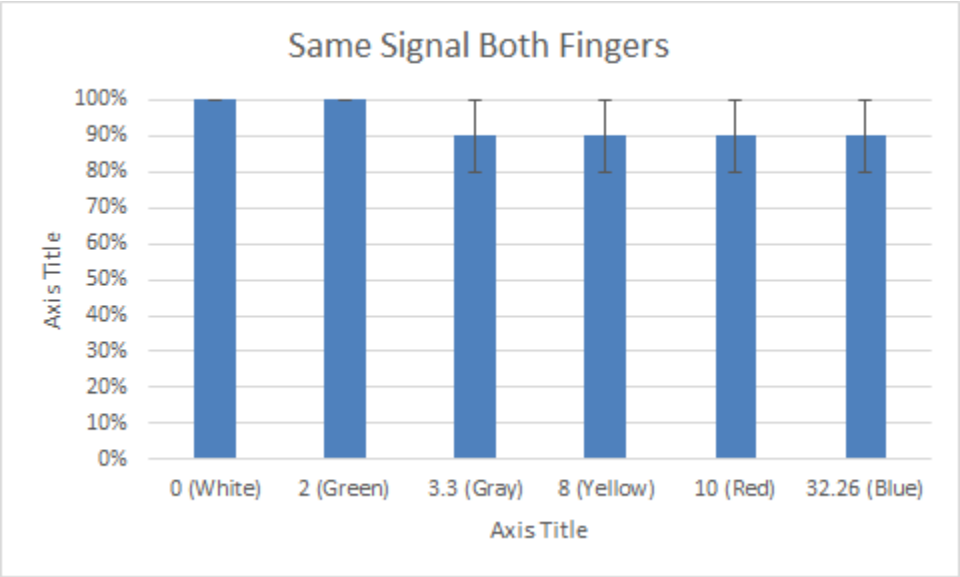
Study1: Single Finger Exploration. Participants were easily able to identify each of the six signals (> 90%) when their finger was placed on a random block of a given color on the map (Graph 29).

Graph 29: Signal Played to Index Finger Only



Study 2: Two Finger Exploration. Participants were able to identify the frequency of the color very well (> 90%) when the same color was contacted by both fingers (Graph 30).

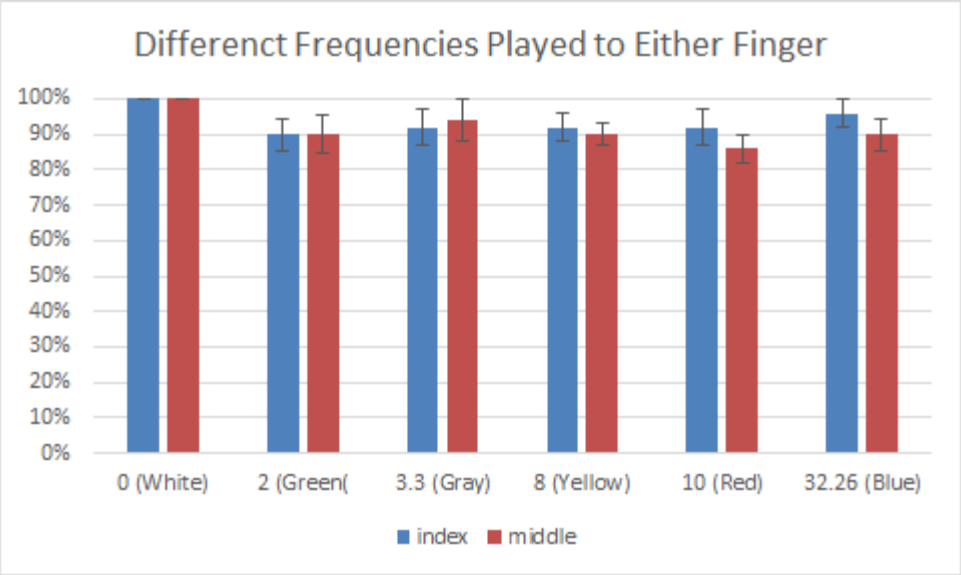
Graph 30: Same Signal Played Both Fingers



For most colors, participants were able to correctly identify the color contacted by each finger very well (> 90%) when the two fingers contacted different colors (Graph 31). The exception was when the color red (10 Hz note) was contacted by the middle finger: it was identified 86% of the time.

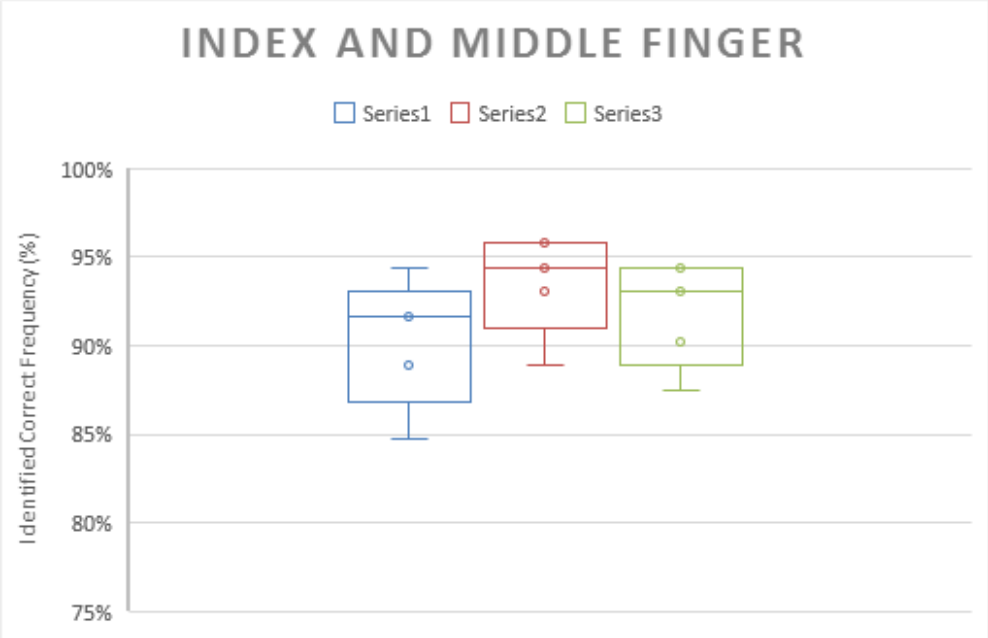
.

Graph 31: Different Signals Played to Index/Middle Fingers



Overall accuracy and the variation between participants is given in Graph 32.

Graph 32: Overall Results of Signals Played to Index/Middle Fingers



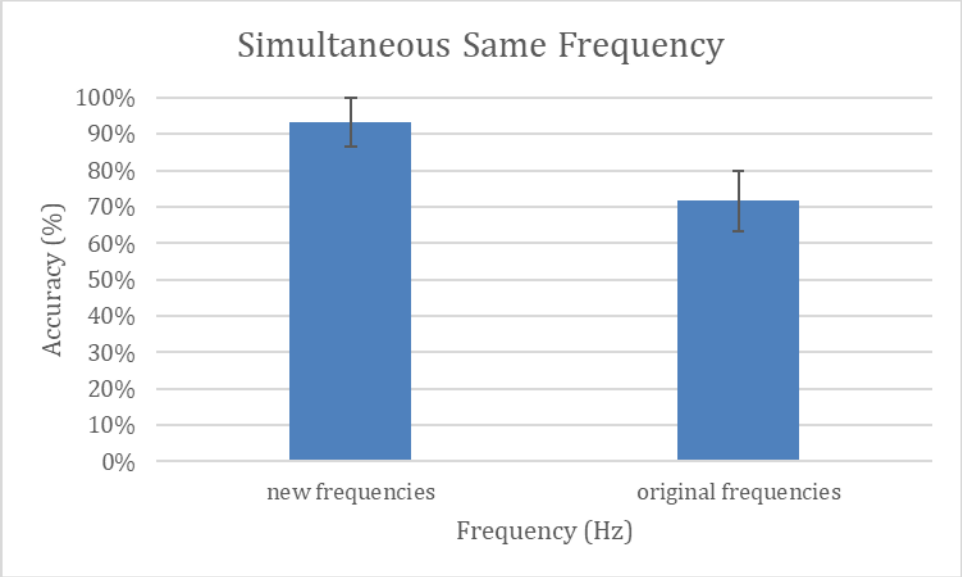
4.3.6. Discussion

In the case of using one finger to identify the frequency of the signal that was played there was a small drop off using the new signals compared to the experiment in Section 4.2. Even though performance was somewhat poorer, these results must be

considered in conjunction with the results of the two finger exploration study. “Notes” are needed that can perform well in both single finger and two finger exploration.

As for identifying the colors/notes when two fingers were used: participants were able to do a much better job at identifying when the same signals were being played to both fingers as compared to the previous study in Section 2.2. Participants were able to identify the same signal correctly 93.3% of the time with the new notes, compared to only 71.6% of the time with the original notes (Graph 33).

Graph 33: Simultaneous Same Frequency



Graphs 30-32 showed that participants who were blind or visually impaired were easily able to identify each of the colors/notes played to either finger. Somewhat better performance was found for the index finger as compared to the middle finger. This is thought to be a result of participants having received feedback throughout testing to their index finger. As a result, they have more experience in identifying the frequency played to their index finger. However, the difference was small. Overall, participants were able to identify the frequencies that were played to both of their fingers approximately 90.3%

of the time. For the main experiment, to try and help participants who did have some difficulty identifying the color/note contacted by a finger, participants could hear the name of the color contacted if they lifted their finger off the screen.

It should also be noted that performance in identifying colors/notes was approximately equivalent (if not slightly higher) than with audio cues. This will have relevance for the interpretation of the results from the main experiment.

5. Main Study

Each feedback condition, involved testing with one garden room map and one overall map. A given map was never used for more than one condition.

5.1 Conditions

Table 7: Methods Used to Search Maps

Method Number	**Method Name**	**Number of Fingers**	**Timbre**	**Ear(s)**	**Finger Location**
1	One Finger One Ear One Timbre	1	--	--	
2	Two Fingers One Ear Two Timbre	2	Different	Same	
3	Two Fingers Two Ears Same Timbre	2	Same	Different	
4	Two Fingers Two Ears Different Timbre	2	Different	Different	
5	One Finger Single Hand One Touch	1			-
6	Two Finger Single Hand Two Touch	2			Index, middle, adjacent
7	Two Finger Two Hands Two Touch	2			Index each hand

The experimental set-up for the audio feedback (Methods 1-4) and vibration feedback (Methods 5-7) was described in detail in Section 2. The details of the selection of the parameters was described in Sections 3 and 4, with the resultant parameter set (from the last experiment of the series of pilot studies) given in those sections.

5.2 Maps

Two different types of maps were used to assess performance: overall maps and individual garden maps. Details of the maps and how they were created are given in Section 2. The overall map is meant to portray the relation between different areas of the function space (for this experiment, different garden areas) at a high level: without details about pathways and what is in the room. The individual garden maps are meant to portray the details that would be helpful when an individual is exploring.

To minimize learning effects, a different overall map and individual garden map was used for each method. The maps for each method were randomly selected from a pool of 7 overall maps and 7 individual garden maps without replacement.

To define the edges of the maps, rather than using audio or vibration feedback, a physical screen protector was affixed to the tablet, with a cut out located where the map was to be displayed. This allowed users to determine the edges of the map by the tactile edge of the screen protector cut out (Figures 5.1 and 5.2)

Figure 5.1: iPad with screen protector cut-out defining map edge

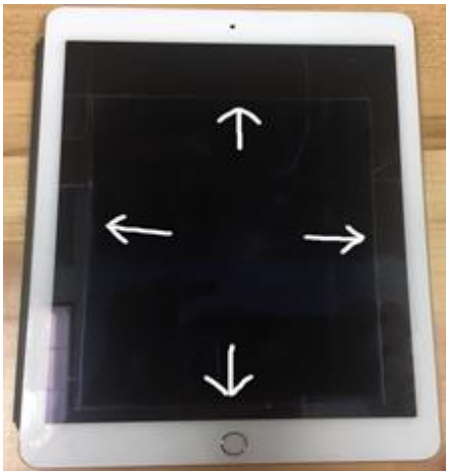
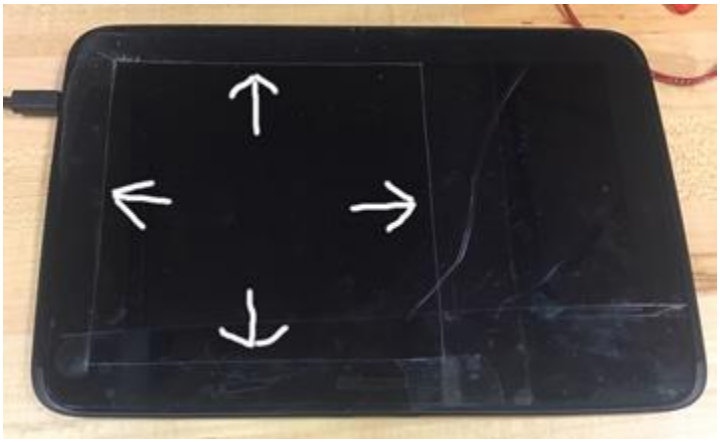


Figure 5.2: Android table with screen protector cut-out defining map edge



5.3 Question Development

Questions were developed based on information an actual visitor to the given area of the Lewis Ginter Botanical Garden may ask. These focused primarily on spatial relationships between features. For example, for an overall map a question was: “you are somewhere in the Blue garden, which garden(s) must you pass through to go directly to the Red garden?” For a garden room map, a question was: how many benches are near

the fountain? Questions concerning point to point navigation were not asked as these questions are already efficiently provided through real-time audio instructions.

Some questions required more details about spatial information, such as those that require the participant to recognize the shape of a garden on the overall map. The questions for both the overall maps and the garden room maps are given in Table 8. The features and spatial relations were sometimes adjusted between garden maps so that the answer would be valid for the particular map in question. The full description of these variables is given in the Appendix (6.3).

Table 8: Questions Asked for Overall and Individual Maps

Overall Map	
Question Number	**Question**
1	Buildings are indicated by the color white. How many buildings are there in the map?
2	What is the overall shape of the <color> garden?
3	Is the <color> garden <wider/taller> than the <color> garden?
4	Is the <color> garden next to the <color> garden?
5	Which garden is closer to the <color> garden, the <color> garden or the <color> garden?

6	How many gardens are to the <right/left> of the entire <color> garden?
7	How many gardens are <above/below> the entire <color> garden?
8	You are somewhere in the <color> garden. Which garden(s) MUST you pass through to go directly to the <color> garden?
9	Which gardens are between the <color> garden and the <color> garden?
10	What is the overall shape of the <color> garden?

Individual Garden Map

Question Number	Question
11	Benches are indicated by the color yellow. How many benches are on the map?
12	Buildings are indicated by the color white. How many buildings are there on the map?
13	Which path(s) have the most <benches/POI>?
14	How many POIs are adjacent to <feature/color>?
15	What points of interest <surround/are adjacent to/ are inside/ are along> the <feature/specified individual feature>?

16	Following paths, that is to say, no stepping on the grass or flowers but assuming buildings are one large room, if you are at the <upper left/lower left> corner, find the closest bench.
17	Following paths, that is to say, no stepping on the grass or flowers but assuming buildings are one large room, if you are at the <upper right/upper left/lower left> corner, find the closest <bench/POI> if you cannot use the stairs.
18	Following paths that is to say, no stepping on the grass or flowers but assuming buildings are one large room, how many direct paths are there to get from <one path or feature> to <any exit/any entrance/to the next>.
19	The next two questions will simulate your potential usage of the map while you are exploring the gardens. If you will allow me to, I will move one of your fingers to a point of interest. Imagine that this is like you have found the YOU ARE HERE SYMBOL on the map. This could be like you were exploring the garden without a map and now want to see what is around you. Could you tell me what is nearby the <specific point of interest> point of interest on the pathway(s).
20	Could you tell me what is nearby the <specific point of interest> point of interest on the pathway(s).

5.4. Participants

There was a total of ten participants. All were either blind or visually impaired.

Table 9: Information Gathered From Participants

Subject	**Level of Blindness**	**Age of Onset**	**Read Braille (Hands Used)**	**Experience With Tactile Graphics**	**Visited Botanical Gardens**	**Dominate Hand**
1	legally blind	birth	no	none	no	right
2	total blind	after 6	yes (1)	expert	no	right

3	total blind	birth	yes (2)	none	no	right
4	legally blind	after 6	yes (1)	none	yes	right
5	legally blind	after 6	no	none	no	right
6	total blind	birth	yes (2)	none	no	right
7	legally blind	birth	yes (2)	none	yes	right
8	total blind	after 6	yes (2)	expert	no	right
9	total blind	birth	yes (2)	expert	no	left
10	total blind	before 6	yes (2)	expert	no	right

5.5. Experimental Design

The experiment compared the performance of participants using the seven different methods of accessing the map information on a tablet. Trials were blocked on audio methods (Methods 1-4) on one testing day and tactile methods (Methods 5-7) on the other day. The order in which the modalities were used (day 1 versus day 2) was counterbalanced across participants. Testing was not done on the same day to minimize participant fatigue. The two testing days were never more than a few days apart.

Within each modality block, the methods (either 1-4 or 5-7) were assessed in an order counterbalanced across participants. For each method, participants were always required to train and test with an overall map first and then train and test on an individual garden map. For each participant, the overall map to be used for a given method was randomly selected from 7 different overall maps, without replacement. This was also true for the individual garden maps. This was to minimize learning effects and avoid any spurious results due to correlations between method and map.

10 questions were asked for each of the maps as described in Table 8. The questions were always asked in order. The answer for each question and the total response time for answering all questions was recorded. It was felt appropriate to use the total response time as some participants may choose to explore the map more upfront to make it easier to answer later questions whereas other participants may not. After the experiment was complete, the answers to the questions were scored as either correct or incorrect based on pre-determined answers before data collection began. "Variables" in the question were chosen for each map so that they actually existed and were not ambiguous.

5.6. Experimental Procedure

The procedure for testing each method was similar between methods: (1) participants were given time to be familiar with the tablet interface and any hardware needed for that method (i.e., headphones or tactile rings), (2) they were then trained on what audio or tactile note corresponded to each color, (3) they were given a practice overall map and required to answer a set of five practice questions, (4) they were then given the test overall map and required to answer the ten questions specified in Table 8 for that particular map, (5) they were then given a practice individual garden map and required to answer a set of five practice questions, and (6) they were then given the test individual garden map and required to answer the ten questions specified in Table 8 for that particular map.

Participants with residual vision were blindfolded throughout the experiment. The tablet that was used remained flat on a desk in front of a participant when they explored it. As described in Section 2, an iPad with headphones was used to provide the audio

feedback and an Android tablet (Nexus 10) with custom ring-type vibrators was used to provide the vibration feedback. In both cases, if the participant lifted their finger off the tablet, the color/feature that could be found under the finger on the map was spoken. Participants were encouraged to take breaks when they wished to avoid fatigue.

5.6.1. Familiarization

The participant was asked to reach out and contact the tablet. They were then required to explore the tablet with their hands to feel the border of the map defined by the screen protector. After this was completed, they were asked to show the experimenter what direction on the tablet was up, down, left and right to ensure the participant had a proper sense of direction. If they were not correct, the directions were demonstrated to them and they were allowed to explore further. They were then tested again until they had a correct sense of direction.

5.6.2. Feedback Cue Learning

The cue training map consisted of six even sized squares with each square represented by a different color. For the Nexus 7 Android tablet the map was presented with two rows and three columns; for the iPad the map was presented with three rows and two columns (Figures 5.3 and 5.4). The participant was then told the layout of the map with the number of columns and rows. With the participant pointing with one finger, the experimenter guided the participant's hand to have the finger contact each of the six colors, in turn, and told the associated color corresponding to the cue (either audio or tactile). Once completed, participants were allowed to explore the map themselves, raising their finger to hear the color label spoken when needed, until they felt confident of being able to identify the cue associated with each color.

Figure 5.3: Initial Map Displayed on iPad

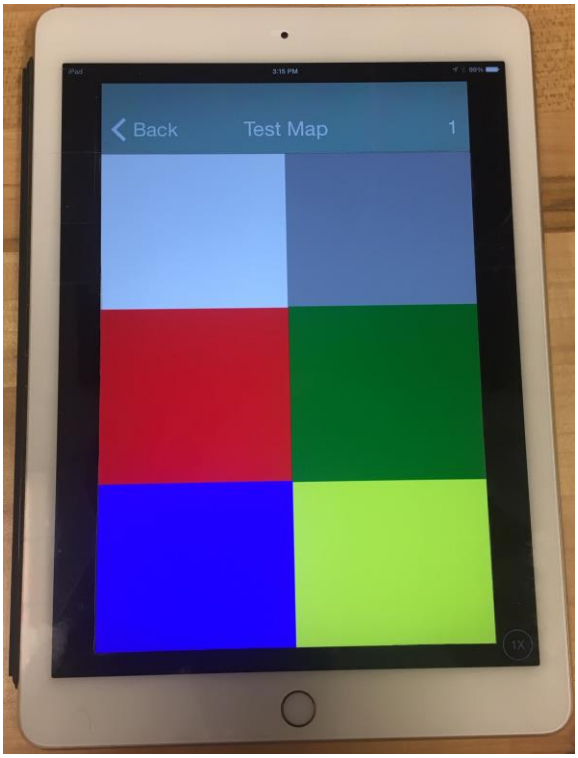
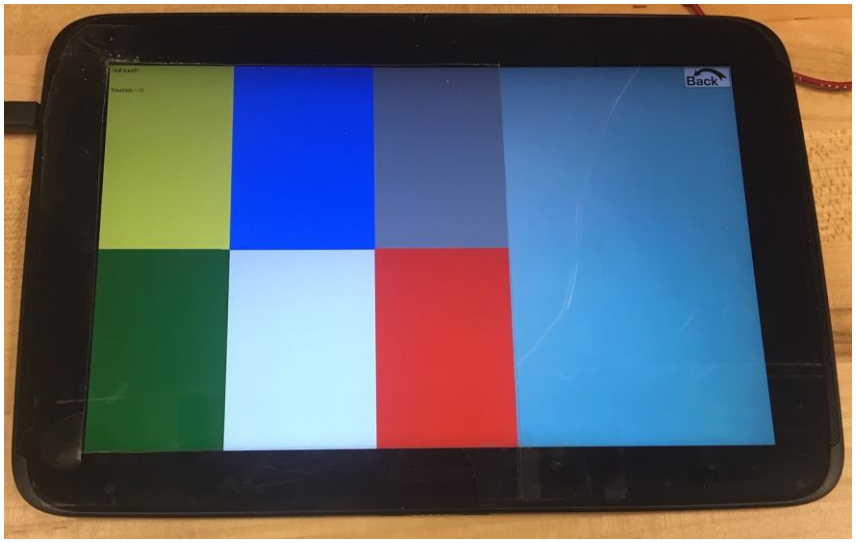


Figure 5.4: Initial Map Displayed on Android



Then, the experimenter tested the participant randomly placing the participant's finger on different colors and then asking the participant to identify the color their finger was on (without raising their finger to hear the speech feedback). Once the experimenter felt confident the participant could correctly identify each color, the participant was allowed to move on to the next part of the protocol.

If the exploration/feedback method used a single finger to explore garden maps, the participant then moved on to explore the Practice Overall Gardens Map (Section 5.6.3). However, if the exploration/feedback method involved using two fingers, the participant would continue training with the Cue Training Map. This time, the participant had two fingers guided by the experimenter to two of the six colors. If needed, they were given the colors that each of the fingers contacted. This was repeated until the participant became confident identifying the colors using the selected two finger method. The experimenter then tested the participant by placing each of the two fingers randomly on different colors. When the experimenter felt confident the participant could easily identify each of the colors with their cues, the experiment proceeded to the next part of the protocol, practice with an overall garden map.

5.6.3. Practice Overall Gardens Map

For practicing with an overall map, the participant was told that each of the colors represented a different garden and the gardens would be of different shapes and sizes. They were also told that the only color that did not represent a garden was white, which represented a building; there could be one or more buildings of different shapes and sizes.

The participant was then allowed to begin exploring the map. When they explored the map, they were asked a set of five questions about it.

Figure 5.5: Practice Overall Map Questions

Practice Overall Map Questions

1. The color white indicates buildings. How many buildings are there in the map?
2. Which garden is wider the blue garden or the gray garden?
3. You are in the green garden what garden must you go through to get to the gray garden?
4. What is the overall shape of the yellow garden?
5. What garden(s) are directly to the left of the blue garden?

Figure 5.6: Practice Overall Gardens Map

The same practice overall map and five questions were used for all of the different exploration/feedback methods tested. The correctness of the answers or the time it took to

answer the questions were not recorded. After the practice overall map had been explored and each question had been answered, the experiment proceeded to the next part of the protocol, testing performance with an overall garden map using the given method.

5.6.4 Testing with an Overall Gardens Map

For this test, the participant was presented with a unique overall map for each method used, randomly selected (without replacement) from a set of 7 overall maps. The participants were asked a set of ten questions (Questions 1-10 in Table 2), in order, for each map and were required to provide an answer. After they answered a question, they proceeded to the next one. The time it took for participants to complete all of the questions was timed. If a participant forgot what a particular cue represented, they were not allowed to be reminded by the experimenter. However, they could lift their finger to hear the color spoken.

Figure 5.7: 10 Questions asked Overall Map E

OVERALL MAP: Overall_E

1: Buildings are indicated by the color white. How many buildings are there in the map?

2: What is the overall shape of the GREEN garden?

3: Is the RED garden TALLER than the BLUE garden?

4: Is the YELLOW garden next to the BLUE garden?

5: Which garden is closer to the RED garden, the Blue garden or the GREY garden?

6: How many gardens are to the LEFT of the entire GREEN garden?

7: How many gardens are BELOW the entire YELLOW garden?

8: You are somewhere in the GREEN garden. Which garden(s) MUST you pass through to go directly to the RED garden?

9: Which gardens are between the BLUE garden and the GREEN garden?

10: What is the overall shape of the YELLOW garden?

Figure 5.8: Overall Gardens Map E

5.6.5. Practice Individual Garden Map

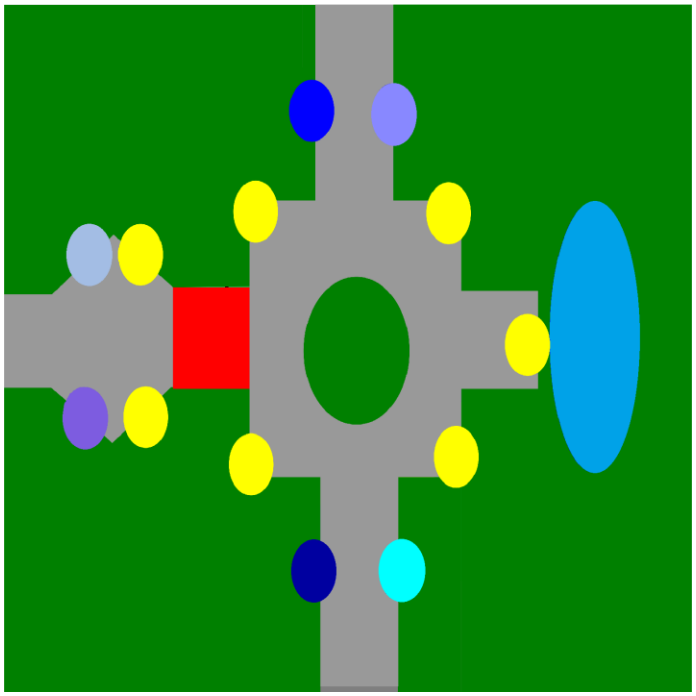
This practice time was used to introduce the participant to the appearance of an individual garden map and learn the association between the colors and garden features. Before the participant was allowed to explore the map, they were provided with an explanation of what each color represented for an individual garden map. They were told: "the color red represents stairs, the color blue represents points of interest, the color gray represents pathways, the color yellow represents benches, the color green represents gardens, while the color white still represents buildings. Note that all the shades of blue (including purple) in the below image are felt as a single "blue cue" and have the label "blue". The purpose of this was to allow for the tester to easily identify the correct point of interest for "you are here" questions (Figure 5.11, Question 9 and 10) Participants were then told they were allowed to begin exploring the map.

Figure 5.9: Practice Individual Map Questions

Practice Garden Map Questions

1. Following the paths that is to say, no stepping on the grass or the flowers but assuming buildings are one large room, if you are at the UPPER_MIDDLE path find the closest POI?
2. I'm going to guide your hand to a particular location and I want you to imagine it as a YOU ARE HERE SYMBOL on the map. Can you tell me what is nearby the Light Blue point of interest?
3. How many benches are adjacent to the stairs?
4. Fountains are represented by large POI's. Is there a fountain on the map?
5. Benches are indicated by the color yellow. How many benches are on the map?

Figure 5.10: Practice Individual Garden Map



When exploring the map, the participant was asked a set of five questions about the Practice Individual Garden Map. The same practice map and five questions were used for all of the different exploration/feedback methods tested. The correctness of the answers or the time it took to answer the questions were not recorded. However, participants were allowed to be reminded what each color meant while they explored the map. After the practice individual garden map had been explored and each question had been answered, the experiment proceeded to the next part of the protocol, testing performance with an individual garden map using the given method.

5.6.6. Individual Garden Map

For this test, the participant was presented with a unique individual garden map for each method used, randomly selected (without replacement) from a set of 7 overall

maps. The participants were asked a set of ten questions (Questions 11-20 in Table 2), in order, for each map and were required to provide an answer. After they answered a question, they proceeded to the next one. The time it took for participants to complete all of the questions was timed. If a participant forgot what a particular cue represented, they were not allowed to be reminded by the experimenter. However, they could lift their finger to hear the color spoken.

Figure 5.11: 10 Questions Asked for Individual Gillette Garden Map

OVERALL MAP: Gillette Garden/Library Map

1: Benches are indicated by the color yellow. How many benches are on the map?

2: Buildings are indicated by the color white. How many buildings are on the map?

3: Which path(s) have the most POI?

4: How many POI are adjacent to STAIRS?

5: How many points of interest ARE_ADJACENT_TO the BUILDING?

6: Following paths that is to say, no stepping on the grass or flowers but assuming buildings are one large room, if you are at the LOWER_LEFT corner, find the closest BENCH.

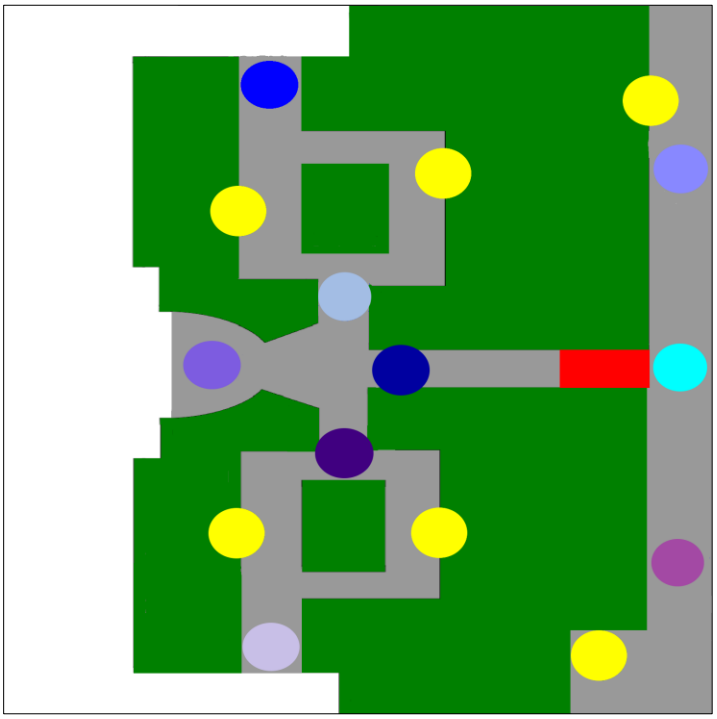
7: Following paths that is to say, no stepping on the grass or flower but assuming buildings are on large room, if you are at the UPPER_LEFT corner, find the closest POI IF you cannot use the stairs.

8: Following paths that is to say, no stepping on the grass or flowers but assuming buildings are on large room, how many direct paths are there to get from PATH_AT_RIGHT to ANY_ENTRANCE_OF BUILDING?

9: If you will allow me to, I will move one of your fingers to a point of interest. Imagine that this is like you have found the YOU ARE HERE SYMBOL on the map. Could you tell me what is nearby the TURQOISE point of interest on the pathway(s)?

10: We are going to do the same thing but in a different location. Could you tell me what is nearby the DARK_PURPLE point of interest on the pathway(s)?

Figure 5.12: Individual Garden Map The Gillette Garden Map



Once all the questions for the Individual Garden Map had been completed they were asked a usability survey about the particular method they just used. They were then provided with a thirty minute break until the start of the next method. This was repeated until all seven methods had been tested.

5.7. Statistical Methods

General estimated equations were used in SPSS to model the correctness of the responses and the total response time per map. The outcome of answering a question correctly was modeled by a binary logistic function. The outcome for the response time was modeled as a Poisson distribution with a log link function. The outcome for the

system usability survey (SUS) was modeled by a normal distribution with an identity link function.

The models included between subject effects of: Blindness (congenitally blind versus adventitiously blind) and tactile experience (none to some versus experienced). We defined congenital blindness as anyone who became blind below the age of 6, as most individuals are not likely to use visual graphics for tasks until they start school (Hatwell, 2003; Heller, 2000; Millar 1994). As for tactile experience, we defined experienced as someone who has used tactile graphics consistently for the past five years or more. While all others defined as none to some. This was done since research conducted by Ferro (2018), who split tactile experience into three groups (none to some, moderate, and experienced), found there was no significant difference in performance between the none to some and moderate groups. While the experienced group performed significantly better than both groups.

The models for the general statistical analysis also included within subject effects of: Modality or Method (as they are correlated, only one or the other variable could be used in any model) and Map Type. More specific analysis on subsets of the was also performed where Map Type was replaced with specific groupings of questions to compare performance between methods when spatial information is used in different ways. This involved questions about shape (Shape Q), how many are there (Group 1), what is around a location (Group 2), and finding a location following a path (Group 3).

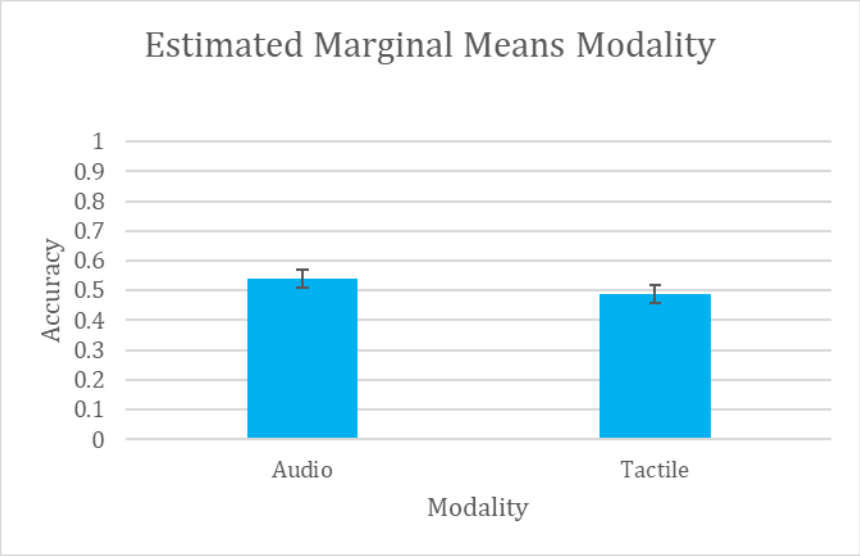
5.8. Results

5.8.1. Within Factors: Modality, Method, and Map Type

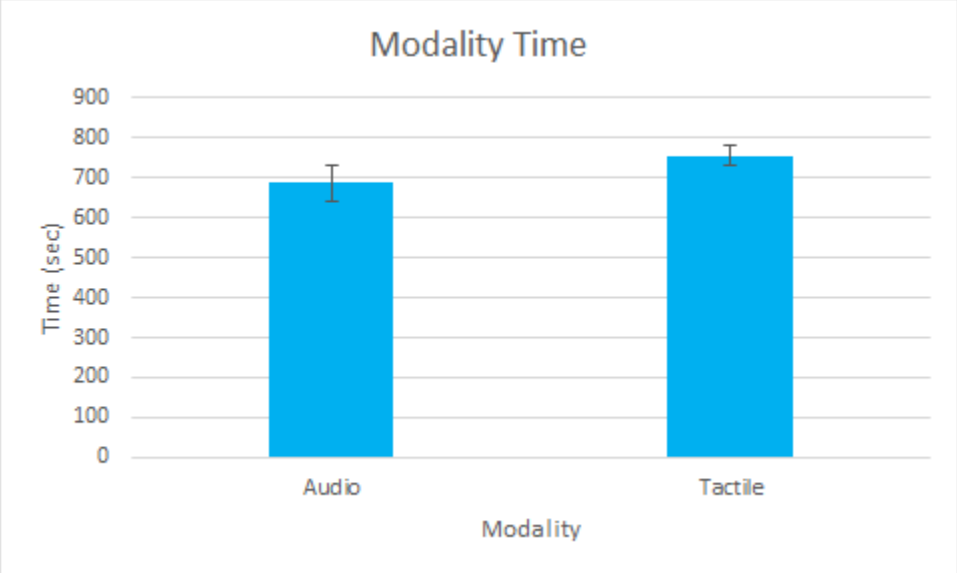
To begin with, we will only look at the within factors from the experimental design as the number of subjects was small as the focus of the experimental design was the within factors. Later, we will include between factors to investigate any serendipitous effects. The within factors of interest are: modality, method and map type and their interactions. Because modality and method ended up being collinear factors, they were modeled separately, each with map type.

The modality/map type model was of interest as it was one way to compare performance with audio cues versus tactile cues. However, it should be noted that somewhat different methods were used with auditory cues than with tactile cues. For the models of the percent correct and response time, only the map type was significant ($p < 0.001$ for both models). However, showing the mean responses (Graphs 34 and 35) is still of interest as a function of modality. For the system usability (SUS score) as a function of modality/map type, modality was a significant factor ($p = 0.001$); subjects found using audio cues to be significantly more usable than tactile feedback (Graph 36).

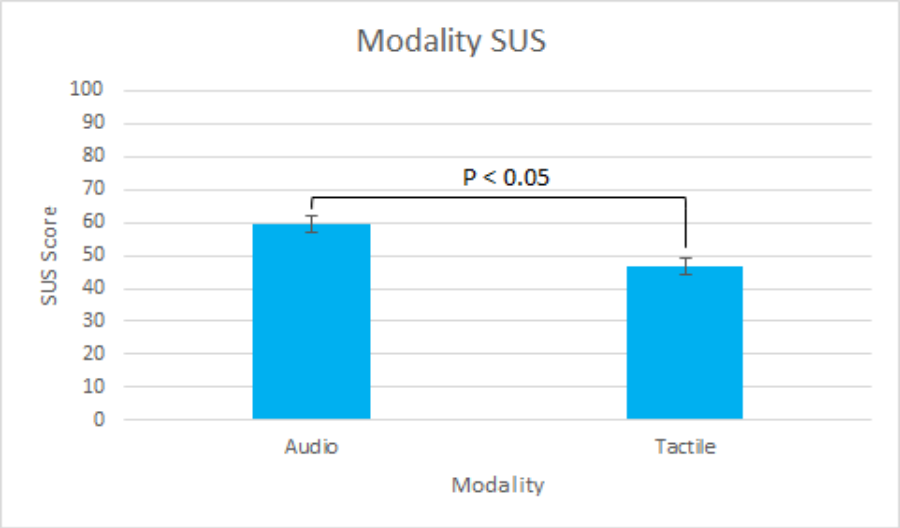
Graph 34: Effect of Modality on Accuracy of Response



Graph 35: Effect of Modality on Response Time

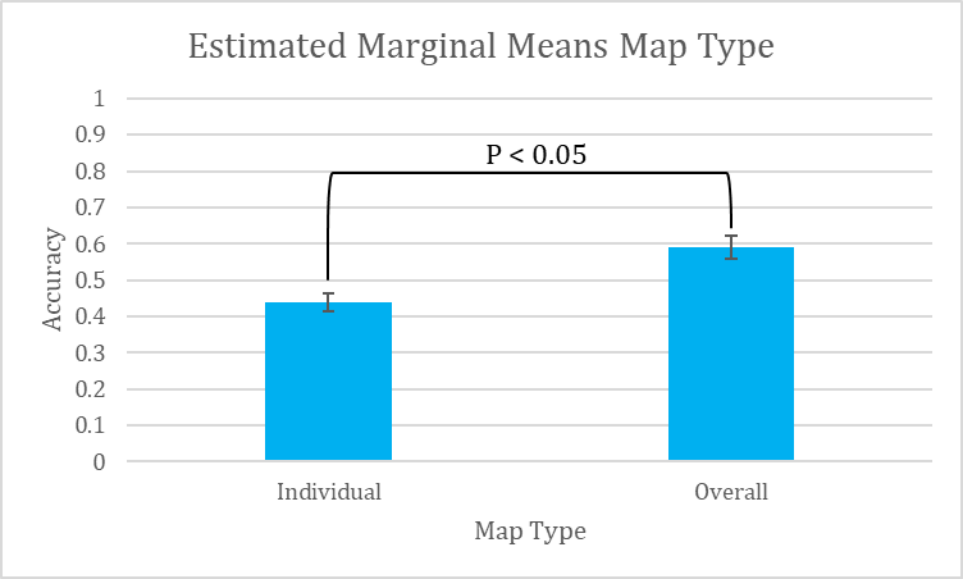


Graph 36: The Effect of Modality on SUS Score

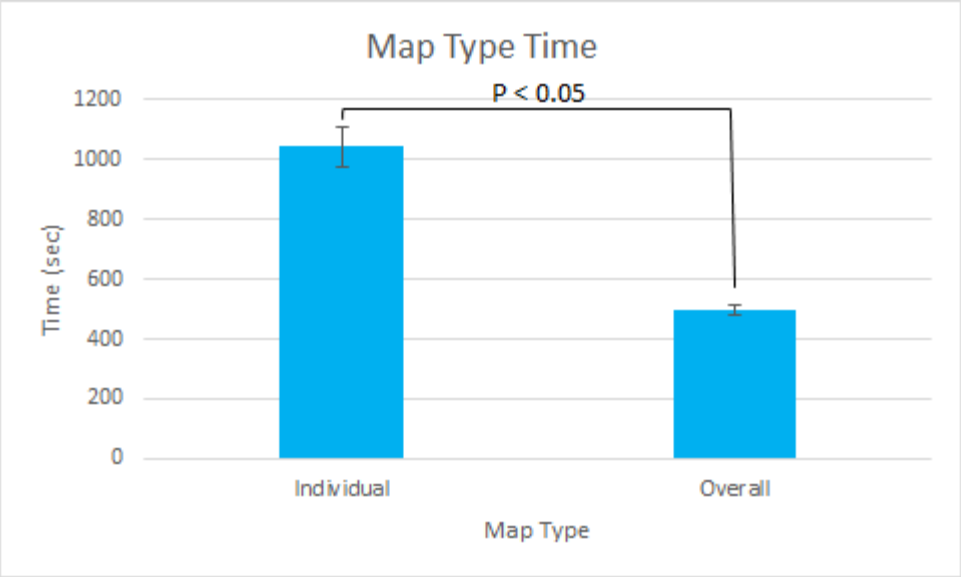


The mean responses of the output variables as a function of map type are shown in Graphs 37 and 38: users were quicker and more correct in answering the questions relating to the overall maps. System usability was not compared as the map type is dependent on the detail of the data needed and is not a choice of method.

Graph 37: Effect of Map Type on Accuracy

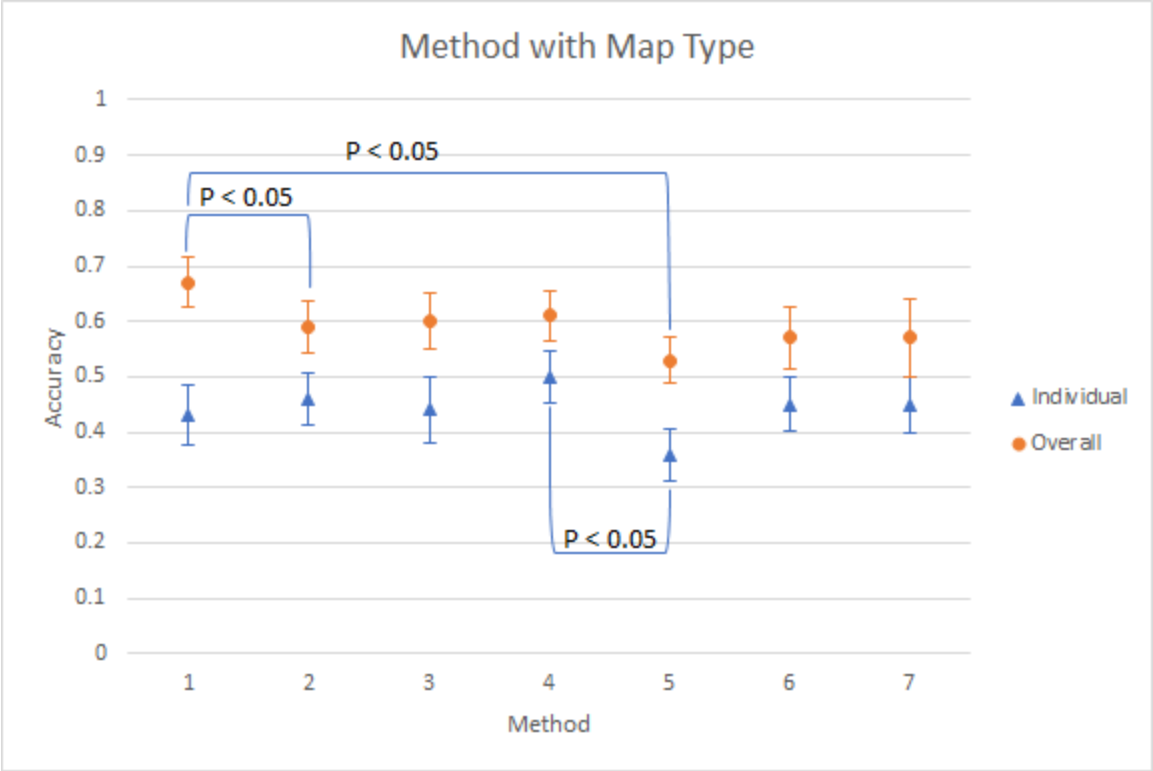


Graph 38: Effect of Map Type on Response Time

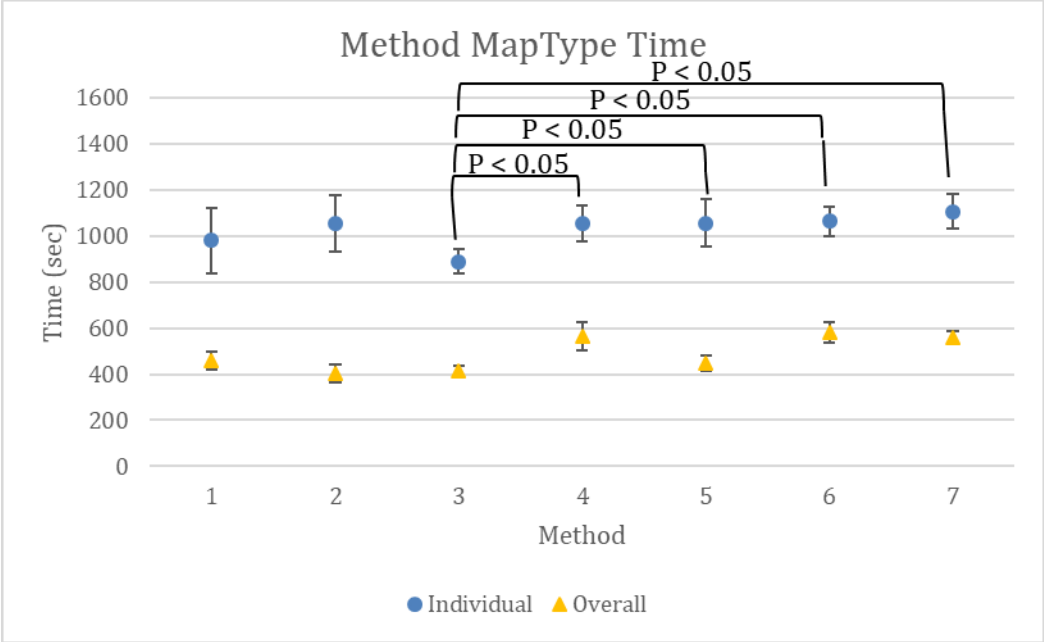


The **method/map type model** was of interest as it relates to many of the questions posed in the introduction: such as a more one on one way of comparing similar audio and tactile methods and in comparing the use of one finger to two fingers. The method was not a significant factor in the model of correct responses but was for the model of the response time ($p < 0.001$). Both the map type ($p < 0.001$ for both) and the method * map type interaction ($p = 0.015$ and $p = 0.043$, respectively) were significant for both the correct responses and response time. The effect of method is most clearly revealed in a description of the means for the different methods as a function of map type (Graphs 39 and 40). We are only interested on comparing methods within map type and so have not indicated significant effects across map types.

Graph 39: Effect of Method on Accuracy for the Different Map Types

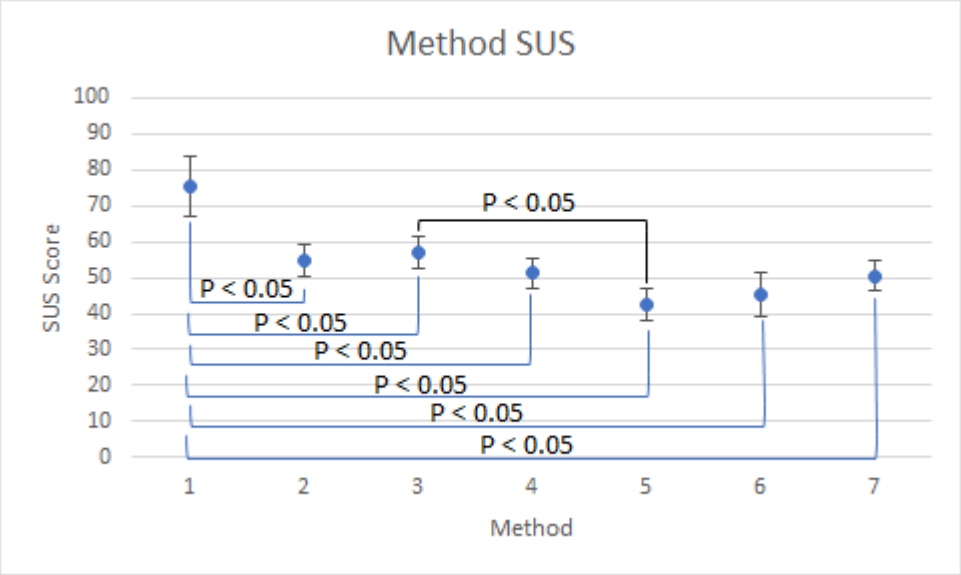


Graph 40: Effect of Method on Response Time



For system usability using the SUS scores, the effect of Method is shown in Graph 41. Method 1 had a significantly higher SUS score than all other methods. Method 3 also had a significantly higher SUS score than Method 5 (Graph 41).

Graph 41: Effect of Method on SUS Score



5.8.2. Between Factors: Blindness and Tactile Experience

It should be noted ahead of time that caution should be used when interpreting any results for these factors due to the small number of participants in each group:

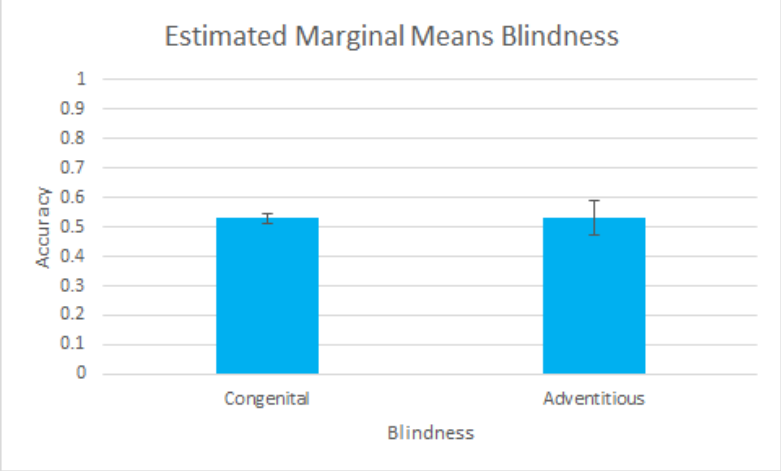
	No or Only Some Experience with Tactile Graphics	Experts with Tactile Graphics
Congenitally Blind	4	2
Adventitiously Blind	2	2

First, the models of the accuracy of response and response time were redone for the **modality/map type model**, including the between subject factors of blindness and tactile experience. The effects included in the model were: all main effects, all two-way

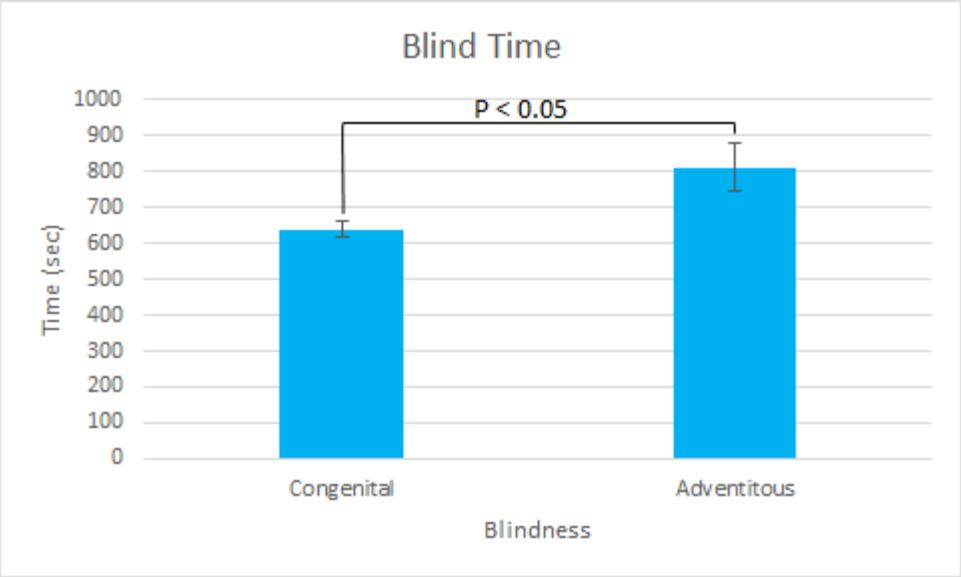
interactions and the two three-way interactions that included both modality and map type with either blindness or tactile experience. In this model, the main effect of modality (Graph 34) was found to be statistically significant ($p < 0.001$) for accuracy of the response but not for response time. The main effect of map type was also statistically significant ($p < 0.001$).

More relevant in this new model are the examination of the potential effects of blindness and tactile experience of the user. The main effects of both factors were not significant for accuracy of the responses but were for response time ($p = 0.008$ and $p < 0.001$, respectively) and the SUS score ($p < 0.001$ and $p = 0.001$, respectively).

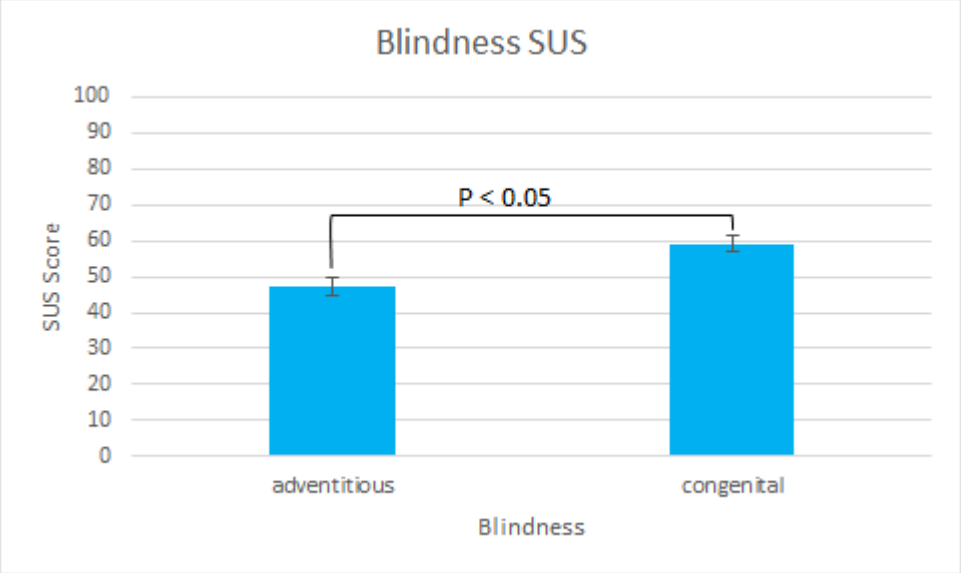
Graph 42: Effect of Blindness on Response Accuracy



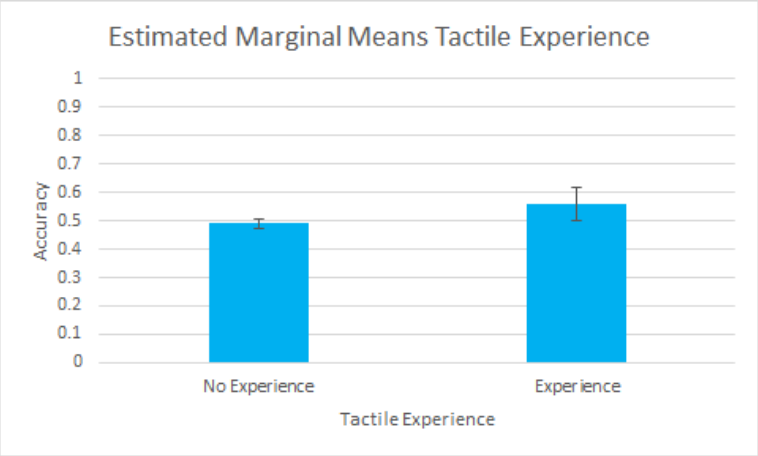
Graph 43: Effect of Blindness on Response Time



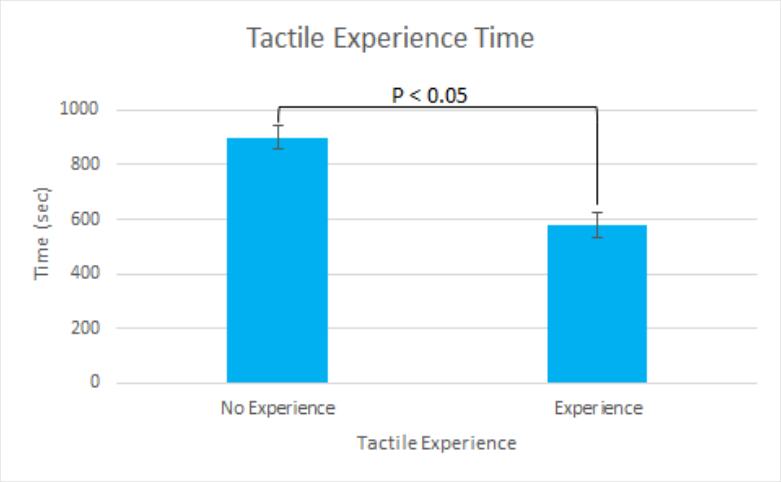
Graph 44: The Effect of Blindness on SUS Score



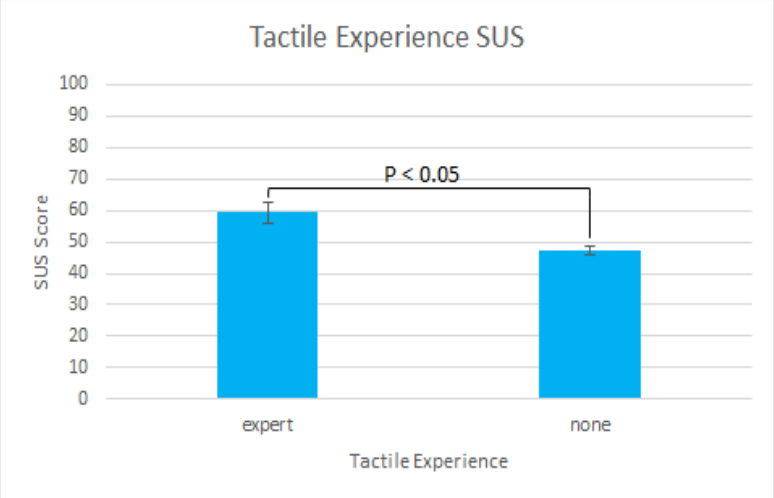
Graph 45: Effect of Tactile Experience on Response Accuracy



Graph 46: Effect of Tactile Experience on Response Time

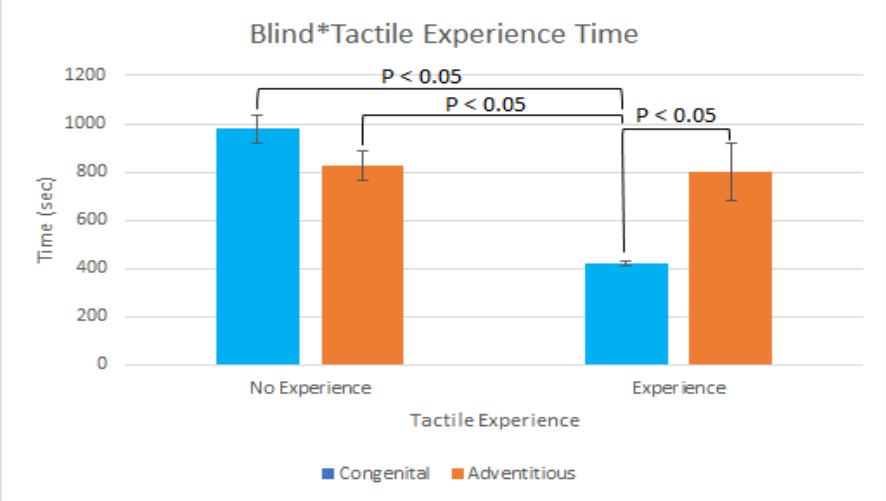


Graph 47: The Effect of Tactile Experience on SUS Score



The interaction term of blindness * tactile experience was also significant for the output variable of response time ($p < 0.001$), with congenitally blind users with significant tactile experience responding much quicker than the other groups (and for similar accuracy levels).

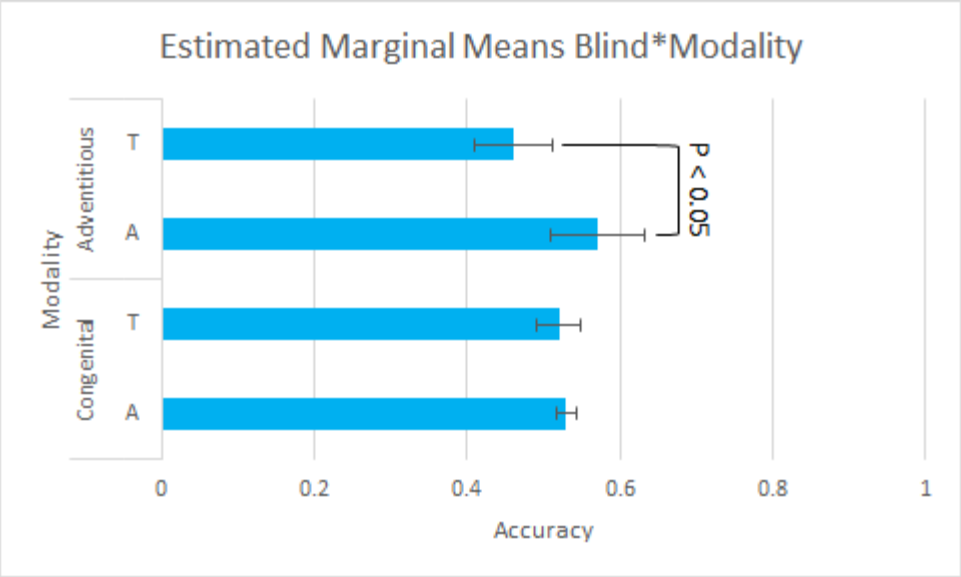
Graph 48: The Effect of the Interaction of Blindness and Tactile Experience on Time



It is also of interest to examine how the interaction terms of either blindness or tactile experience interact with modality, map type and the interaction of both. This may suggest different modalities be used by different users.

For blindness, the interaction term with modality was significant for accuracy of the response ($p = 0.004$, Graph 49). No interaction terms for tactile experience were significant.

Graph 49: The Effect of Modality on Response Accuracy as a Function of a User's Blindness



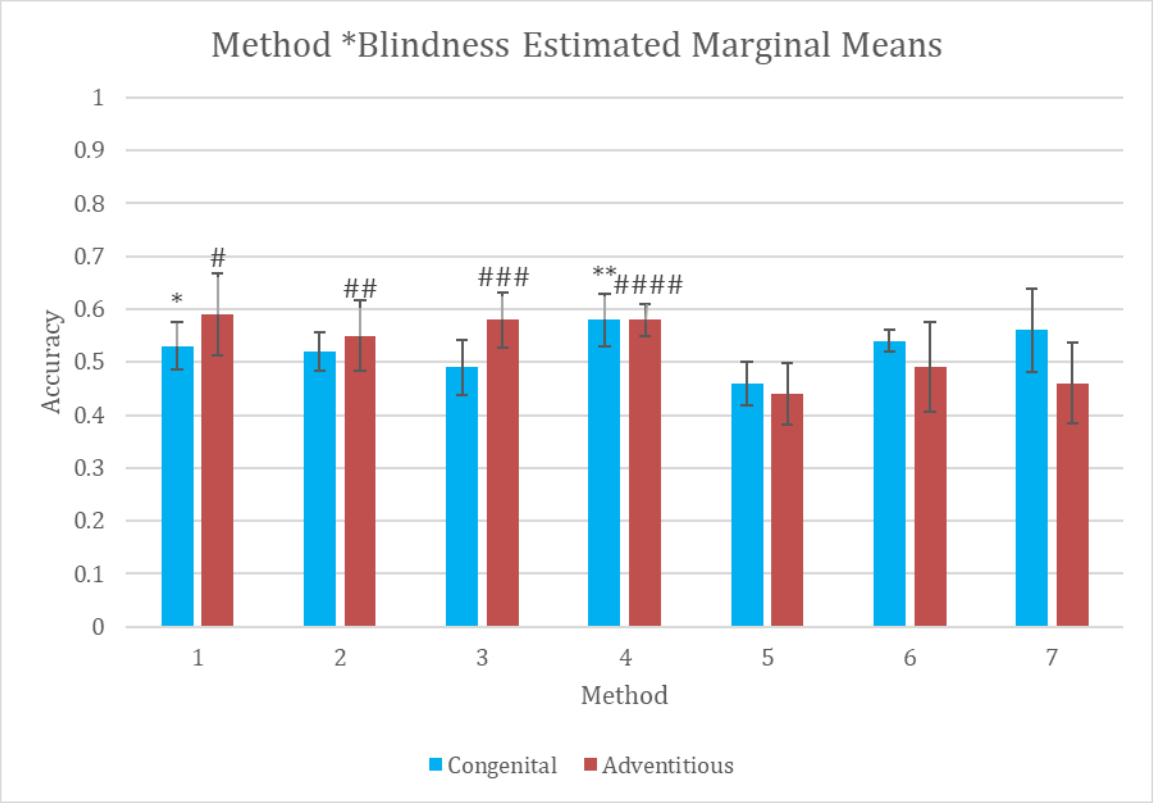
For interaction terms of blindness or tactile experience with either map type or a modality*map type interaction, the terms were significant only for either blindness or tactile experience interacting with map type. As these effects are best understood in terms of the context of the question being asked, data will be presented later in the chapter when the effects on specific types of questions for these map types are examined.

Next, the models of the accuracy of response and response time were redone for the **method/map type model**, including the between subject factors of blindness and tactile experience. The effects included in the model were: all main effects, all two-way interactions and the two three-way interactions that included both method and map type with either blindness or tactile experience. In this model, the main effect of method and map type were now found to be statistically significant ($p < 0.001$ and $p = 0.04$, respectively) for accuracy of the response (Graph 39). Whether the main effects of blindness or tactile experience or their interaction were significant remained unchanged from the analysis involving modality; mean values for these effects are given on Graphs 42-47.

It is also of interest to examine how the interaction terms of either blindness or tactile experience interact with method, map type and the interaction of both. This may suggest different methods be used by different users.

For blindness, the interaction term with method was significant for response accuracy ($p < 0.001$), response time ($p < 0.001$) and SUS score ($p < 0.001$). For tactile experience, the interaction term with method was also significant for response accuracy ($p < 0.001$), response time ($p < 0.001$) and SUS score ($p < 0.001$).

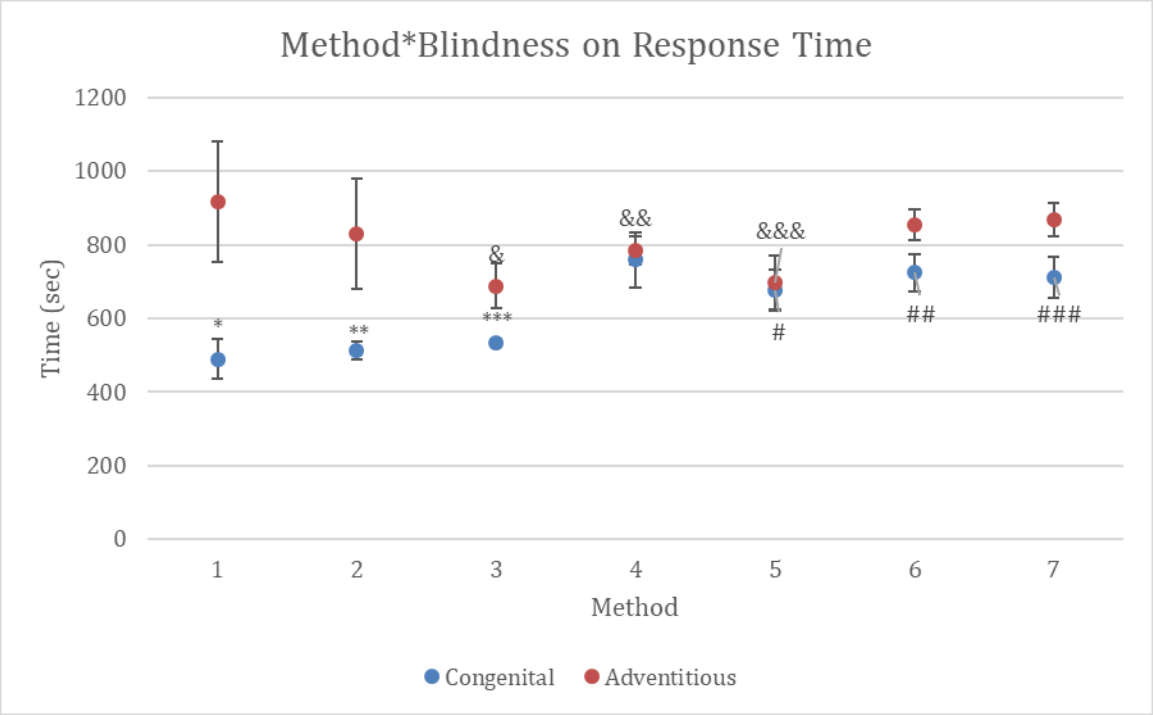
Graph 50: The Effect of Method on Response Accuracy as a Function of a User's Blindness



Method 1 Congenital (*) is significant over method 5 congenitally blind. Method 4 Congenital (**) is significant over method 5 for both adventitiously and congenitally blind. Method 1 (#) and Method 3 (###) Adventitious are significant over method 7

adventitiously blind. Method 2 (##) and Method 4 (####) Adventitious are significant over methods 6 and 7 adventitiously blind.

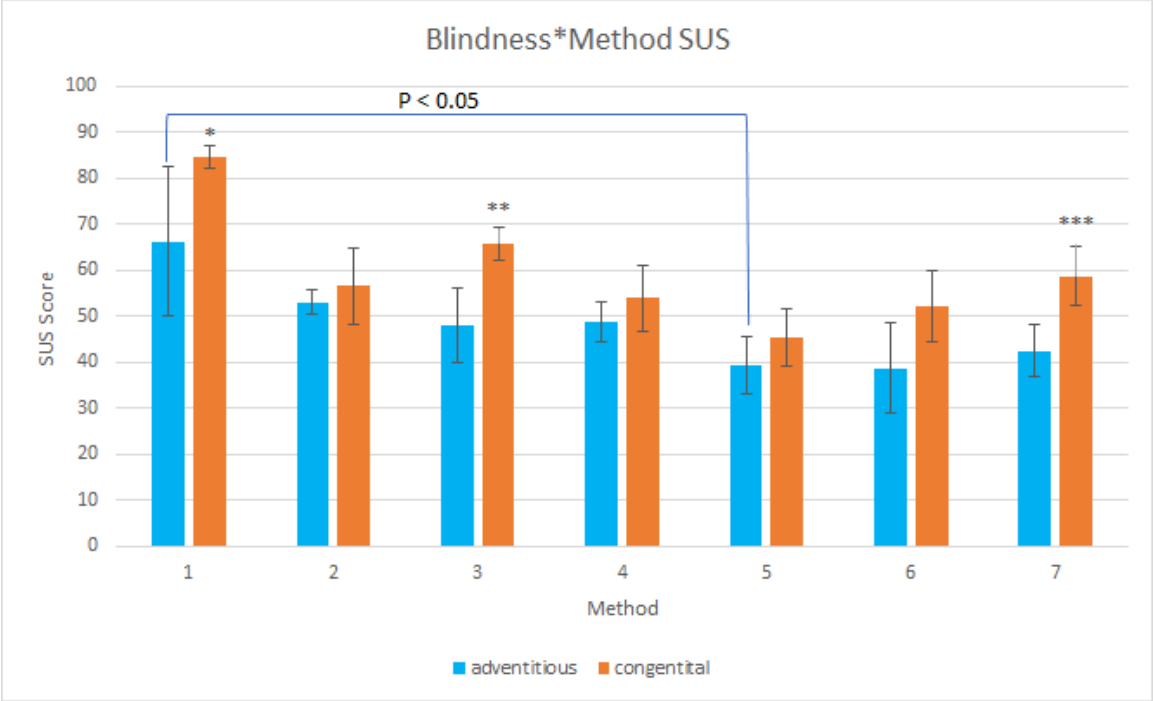
Graph 51: The Effect of Method on Response Time as a Function of a User's Blindness



Method 1 Congenital (*) was significantly quicker than methods 4, 6, and 7 congenitally blind and all adventitiously blind methods. Methods 2 (**) and 3 (***) Congenital is significantly quicker than methods 4, 5, 6, and 7 congenitally blind and all adventitiously blind methods. Methods 5 (#), 6 (##), and 7 (###) Congenital are significantly quicker than methods 6 and 7 for the adventitiously blind.

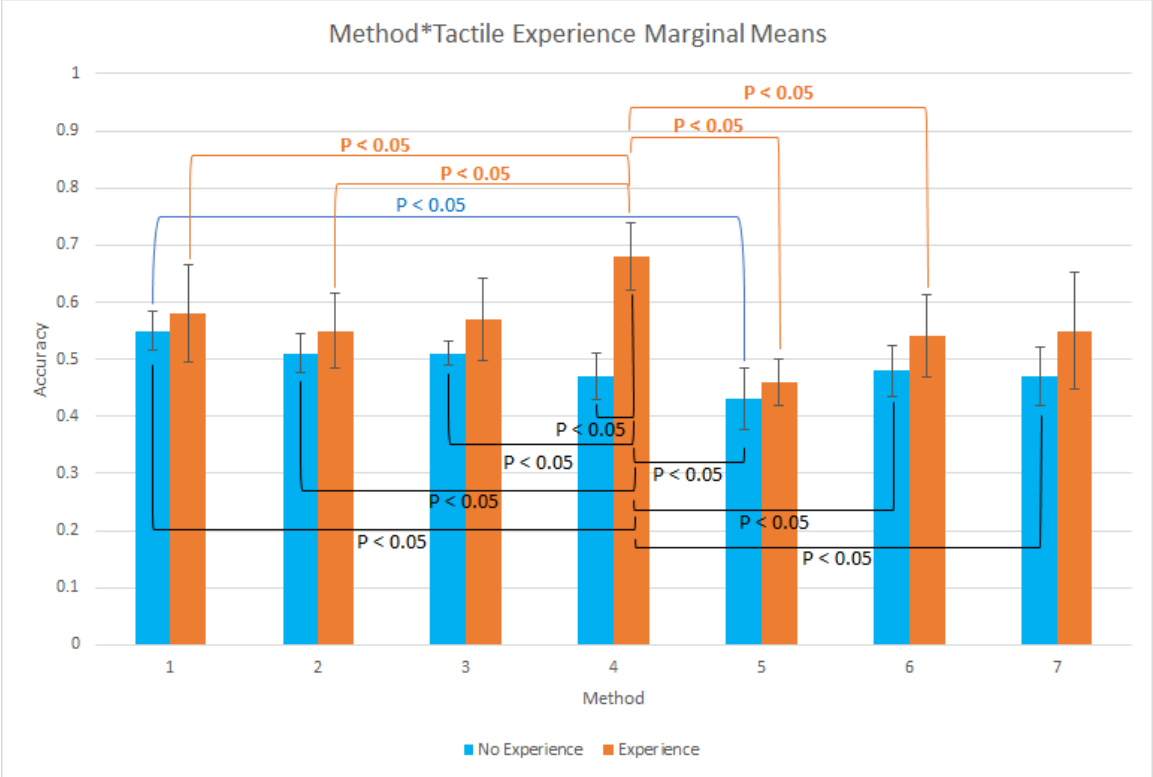
Method 3 Adventitious (&) was significantly quicker than methods 1, 4, and 7 for the adventitiously blind. Method 4 Adventitious (&&) is significantly quicker than method seven adventitiously blind. While Method 5 Adventitious (&&&) is significantly quicker than methods 1 and 7 for the adventitiously blind (Graph 51).

Graph 52: The Effect of Method on the SUS score as a Function of a User's Blindness

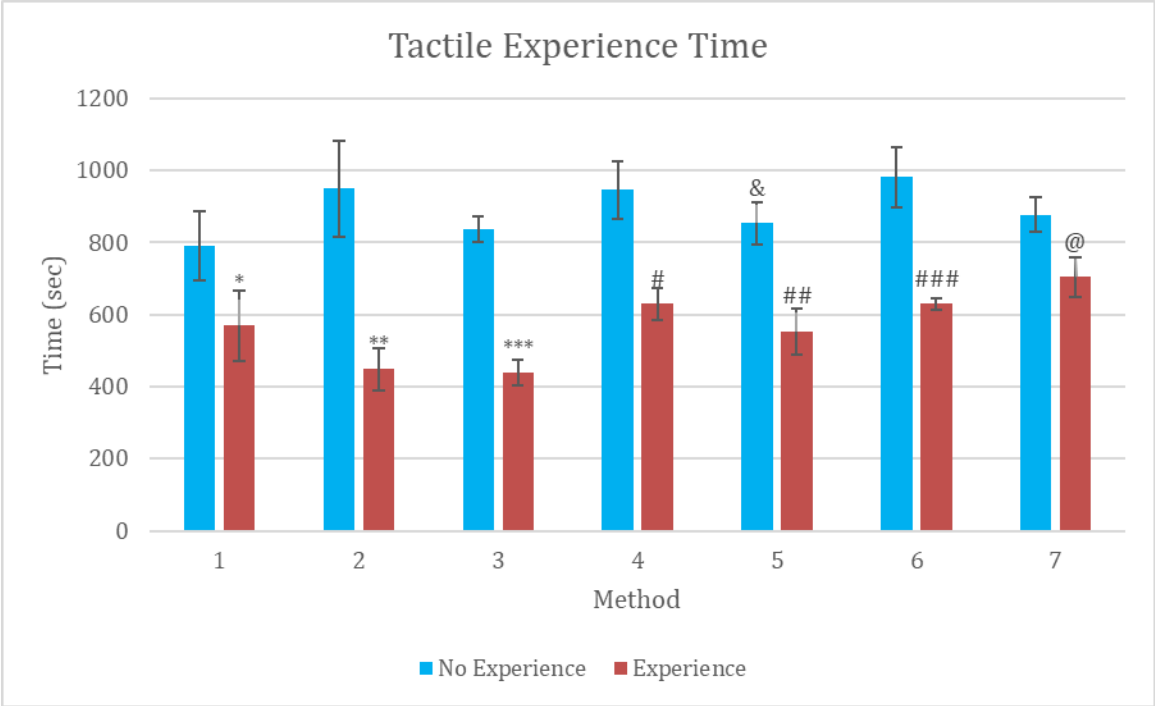


Method 1 congenitally blind (*) is significant over all the other methods except for the adventitiously blind Method 1. Method 3 congenitally blind (**) is significant over all the adventitiously blind methods except for Method 1, it is also significant over the congenitally blind Methods 2 and 5. Method 7 congenitally blind (***) is significant over both the adventitiously and congenitally blind Method 5. For the adventitiously blind methods, only Method 1 was significant over Method 5 (Graph 52).

Graph 53: The Effect of Method on Response Accuracy as a Function of a User's Tactile Experience

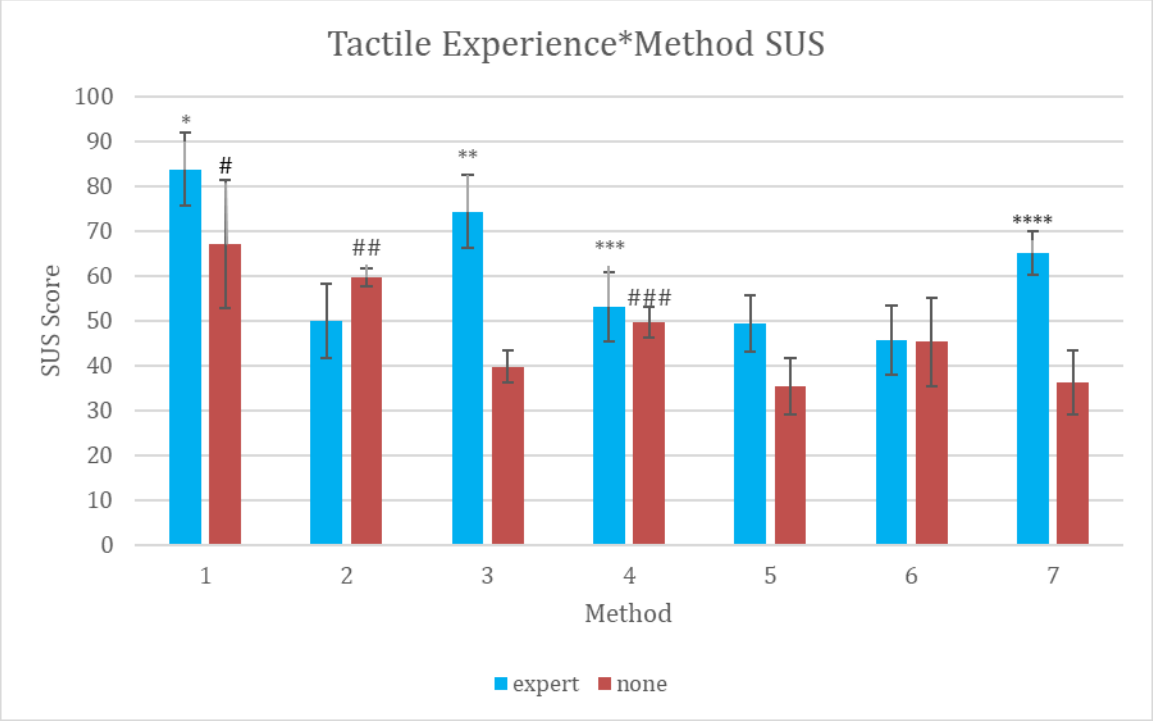


Graph 54: The Effect of Method on Response Time as a Function of a User's Tactile Experience



For those with little to no tactile experience Method 5 (&) is significant over method 4 no tactile experience. As for those with tactile experience, Methods 1 (*), 4 (#), and 6 (###) are significant over all non-experience methods except method 1. Methods 2 (**) and 3 (***) are significant over all other methods for non-experienced and experienced. While Method 5 (##) is significant over all non-experience methods. Method 7 (@) is significant over none-experienced methods 3, 4, 5, and 7. Method 4 (#) is also significant over experienced method 7 (Graph 54).

Graph 55: The Effect of Method on the SUS score as a Function of a User's Tactile Experience



The SUS score for those with tactile experience showed Method 1 (*) is significantly better than all other methods except for those with tactile experience method 7 and those without tactile experience method 1. Method 3 (**) with tactile experience is also significant over methods 5 and 6 with tactile experience, and methods 3,4,5,6, and 7 without tactile experience. Method 4 (***) is significant over method 2 with tactile

experience. While Method 5 (****) is significant over methods 5 expert and over methods 3, 4, 5, and 7 without tactile experience.

For those who have little to no tactile experience, Method 1 (#) is significant over methods 3 and 5 that did not have tactile experience. Method 2 (##) is also significant over methods 3 and 5 as well as methods 4 and 7 without tactile experience. Method 4 (###) is also significant over method 7 without tactile experience (Graph 55).

For interaction terms of blindness or tactile experience with either map type or a method*map type interaction, the blindness * map type term was significant for response time ($p = 0.028$), the blindness * method * map type interaction was significant for response accuracy and response time (both $p < 0.001$), the tactile experience * map type was significant for response time ($p = 0.001$), and the tactile experience * method * map type was significant for response accuracy ($p < 0.001$). As these effects are best understood in terms of the context of the question being asked, data will be presented later in the chapter when the effects on specific types of questions for these map types are examined.

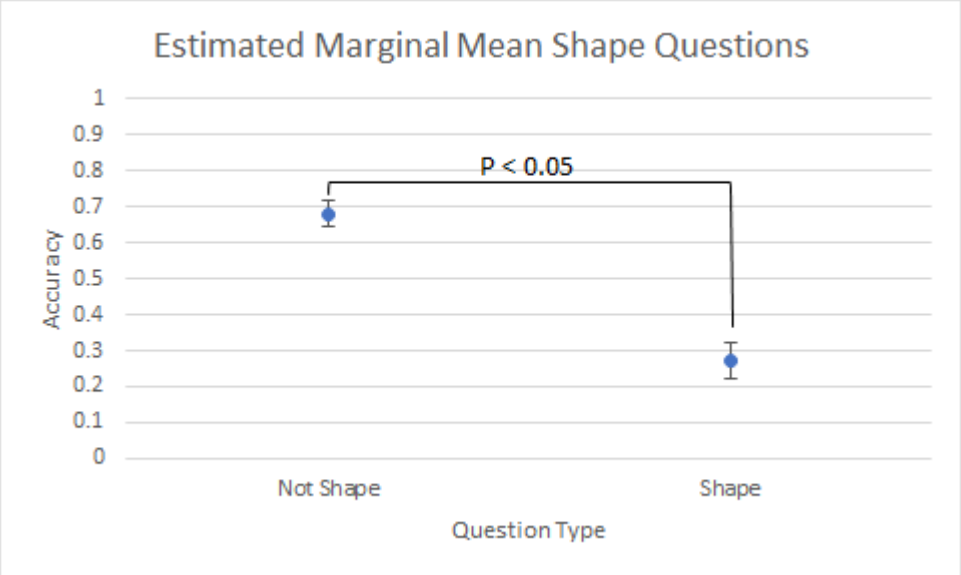
5.8.3. Shape Questions (Overall Maps)

The two shape questions were questions that explicitly asked about the shape of a garden in the overall map: "What is the shape of the GREEN garden?". A model was constructed for the metrics that compared the questions that asked explicitly about shape to all other questions for the overall maps. As the response time was not recorded per question, only a statistical model of the accuracy of the response was constructed. A model was first constructed using modality and all other factors, and then one using

method and all other factors. The factors were: modality or method, whether an explicit shape question, blindness and tactile experience. The effects included in the model were: all main effects, all two-way interactions and the two three-way interactions that included both modality/method and whether a shape question with either blindness or tactile experience.

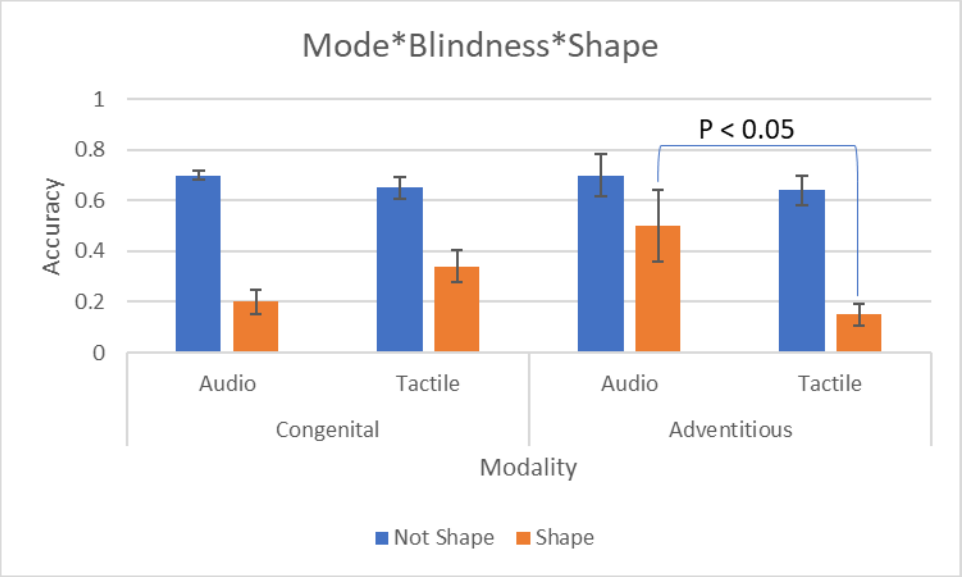
In both models, the main effect of whether the question was a Shape Question showed that questions not asking about the shape of the garden were answered significantly better than questions asking the shape of a garden (Graph 56).

Graph 56: Shape Questions

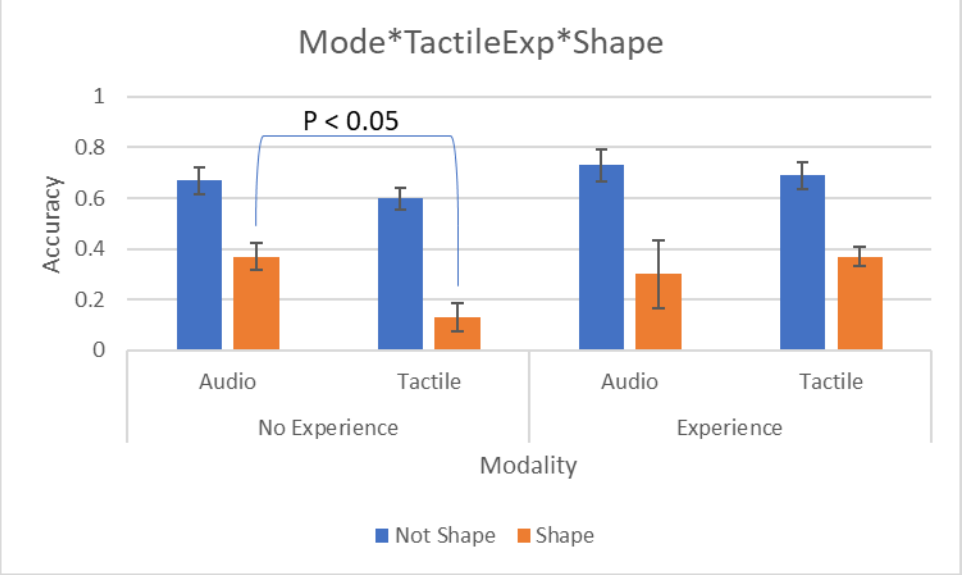


For the **model involving modality** only: shape ($p < 0.001$), modality ($p = 0.043$), blindness * modality ($p < 0.001$), tactile experience * modality ($p < 0.001$) and shape * blindness * modality ($p = 0.018$) were significant. The comparison of the means between shape questions and the others are given above in Graph 56. The effect on the accuracy of answering shape versus no shape questions, as a function of blindness and modality and tactile experience and modality are given in Graph 57 and 58 respectively.

Graph 57: Effect of Whether Shape Questions on Response Accuracy as a Function Blindness and Modality



Graph 58: Effect of Whether Shape Questions on Response Accuracy as a Function of Tactile Experience and Modality

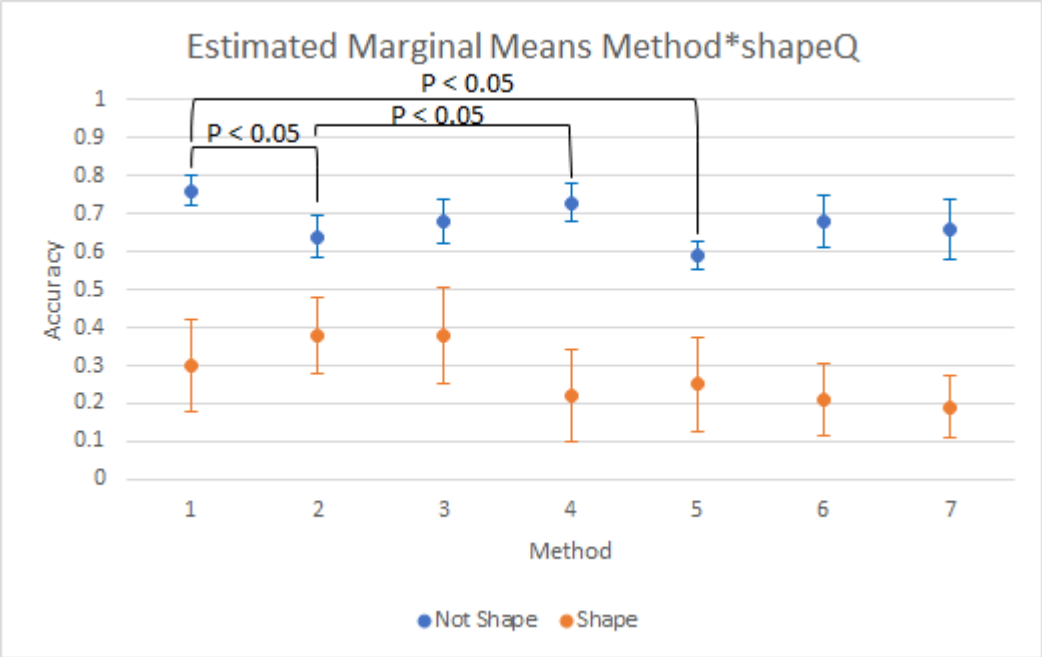


For the **model involving method** only: the main effects of both method and shape were significant ($p < 0.001$ in both cases), two way interaction effects of method with all

other factors were significant ($p < 0.001$ for interactions with shape, blindness and tactile experience), and three-way interactions of method * shape with both blindness ($p = 0.011$) and tactile experience ($p < 0.001$).

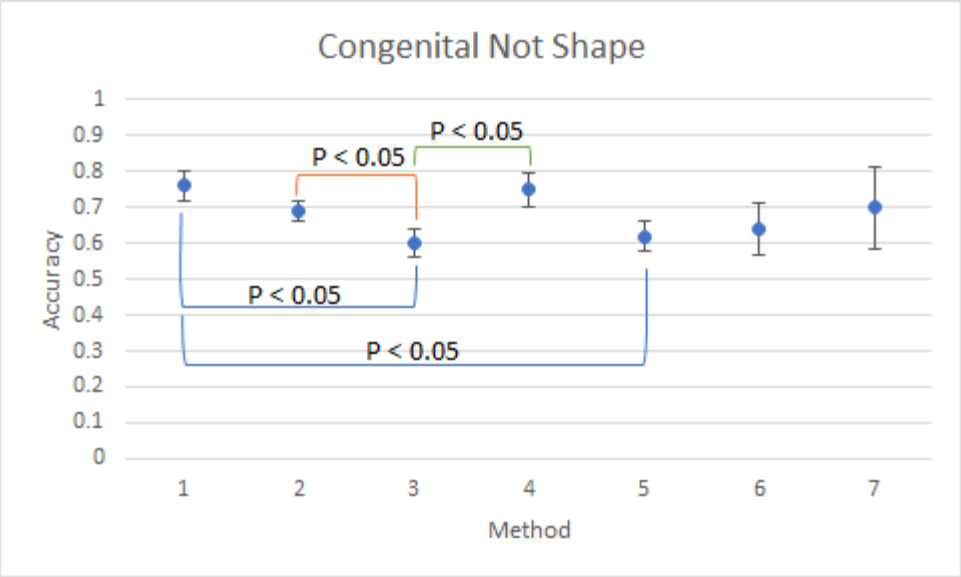
The effect of the method used on the accuracy of the response for questions explicitly involving shape and those that do not are given in Graph 59. Method only seems to have an effect on accuracy for those questions not explicitly involving shape.

Graph 59: The Effect of Method on Response Accuracy as a function of whether a Shape Question or non-Shape Question

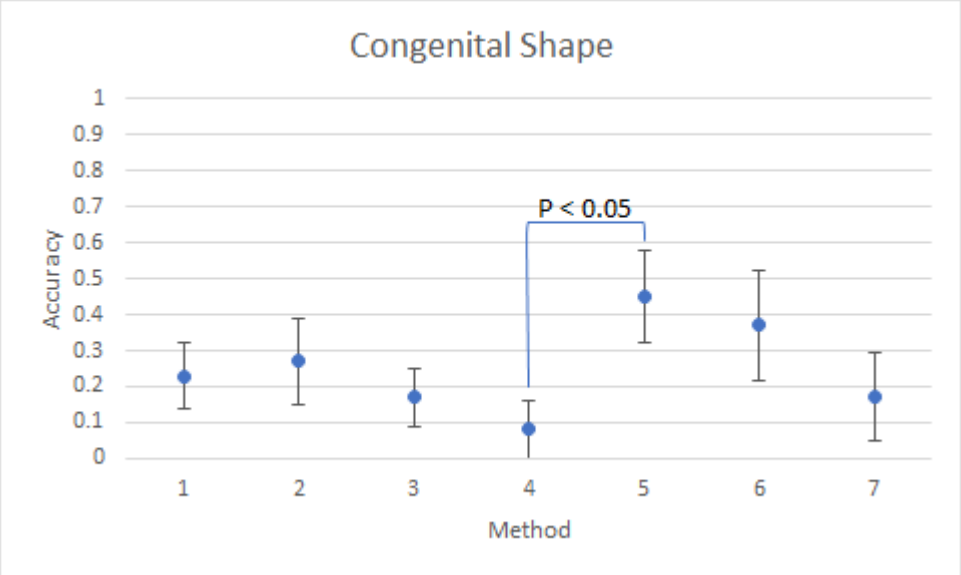


The effect of the method used on the accuracy of the response for questions explicitly involving shape and those that do not as a function of blindness is given in Graphs 60, 61, 62 and 63. For the congenitally blind, the trends in performance for the difference methods appears to be different for the questions explicitly involving shape (Graph 61) and those that do not (Graph 60).

Graph 60: Effect of Congenitally Blindness on Response Accuracy for the non-Shape Questions

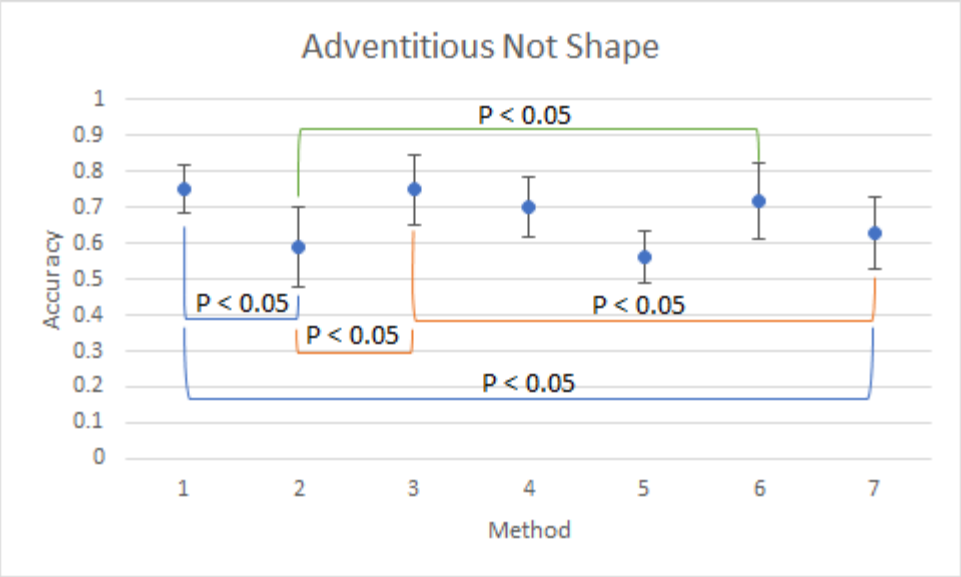


Graph 61: Effect of Congenitally Blindness on Response Accuracy for the Shape Questions

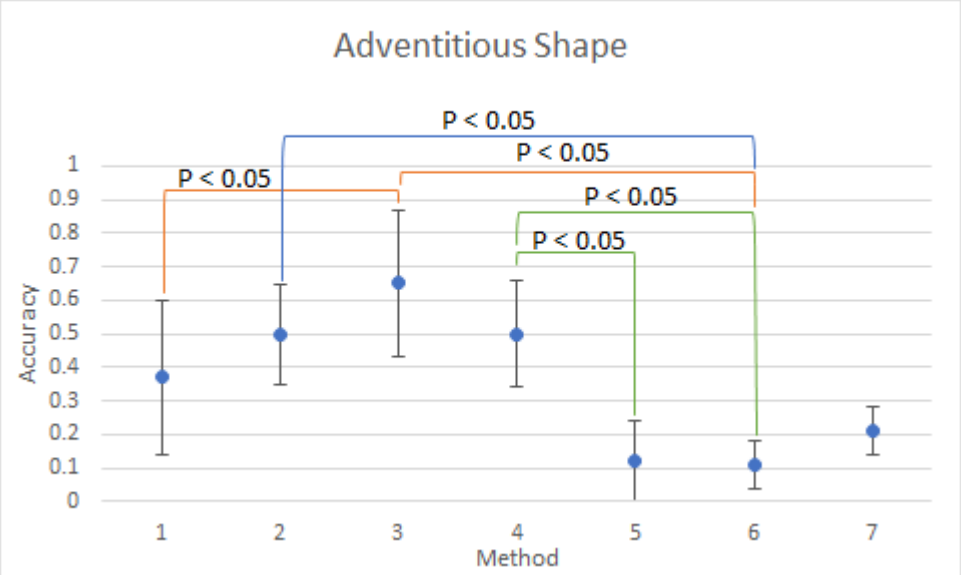


For the adventitiously blind, the trends in performance for the difference methods appears to be different for the questions explicitly involving shape (Graph 63) and those that do not (Graph 62). However, the trends are different than for the congenitally blind.

Graph 62: Effect of Adventitious Blindness on Response Accuracy for the non-Shape Questions

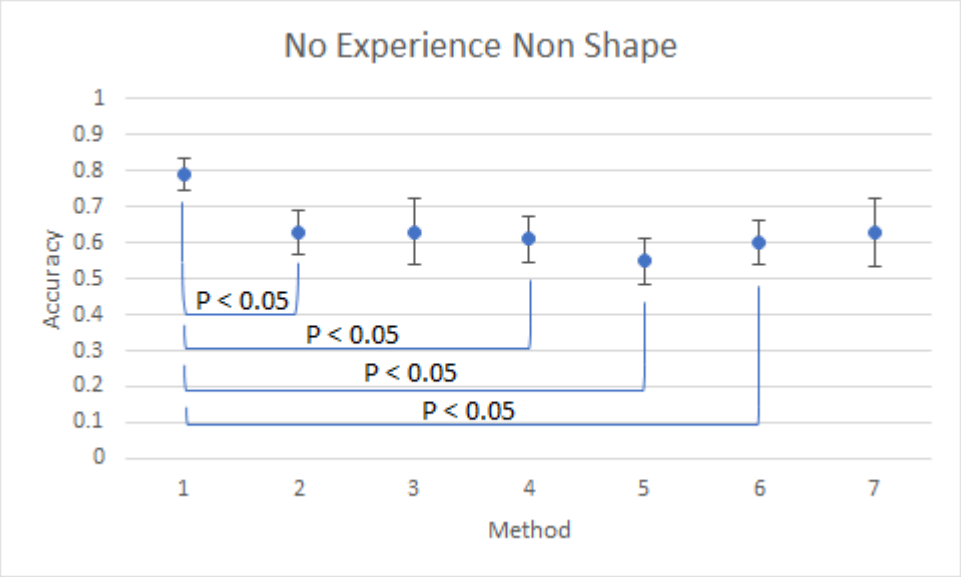


Graph 63: Effect of Adventitious Blindness on Response Accuracy for the Shape Questions

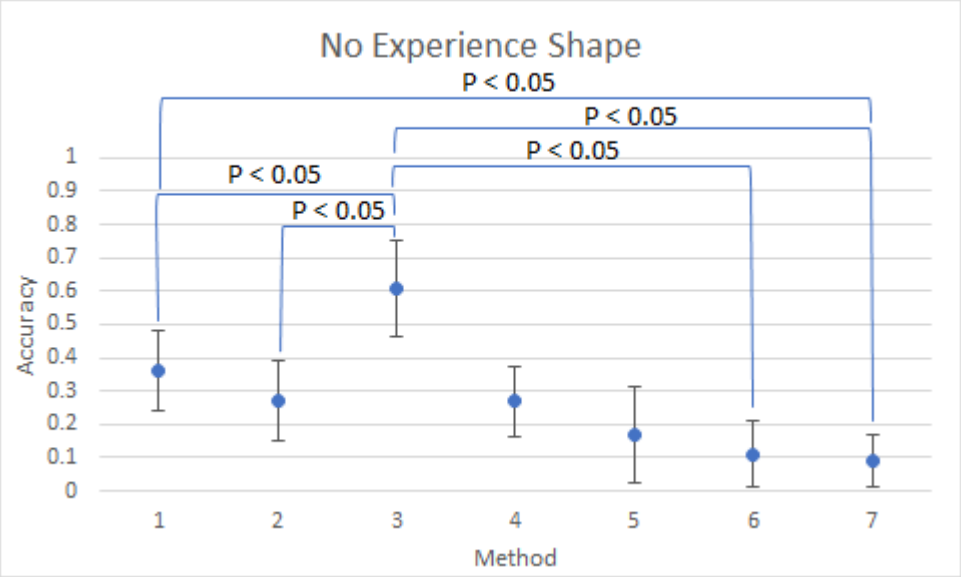


The effect of the method used on the accuracy of the response for questions explicitly involving shape and those that do not as a function of tactile experience is given in Graphs 64, 65, 66 and 67. For those with little to no tactile experience, the trends in performance for the difference methods appears to be different for the questions explicitly involving shape (Graph 65) and those that do not (Graph 64).

Graph 64: Effect of Having No Tactile Experience on Response Accuracy for the non-Shape Questions



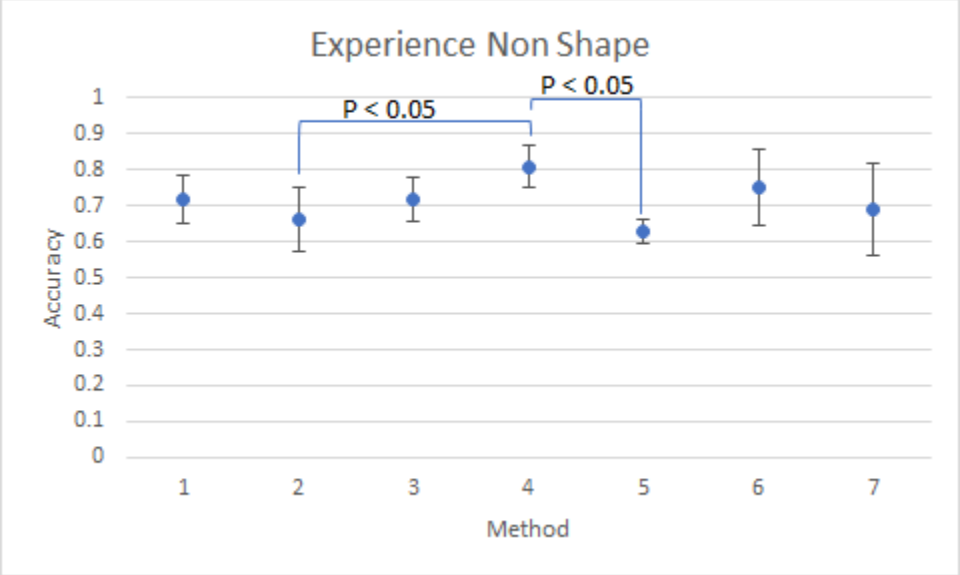
Graph 65: Effect of Having No Tactile Experience on Response Accuracy for the Shape Questions



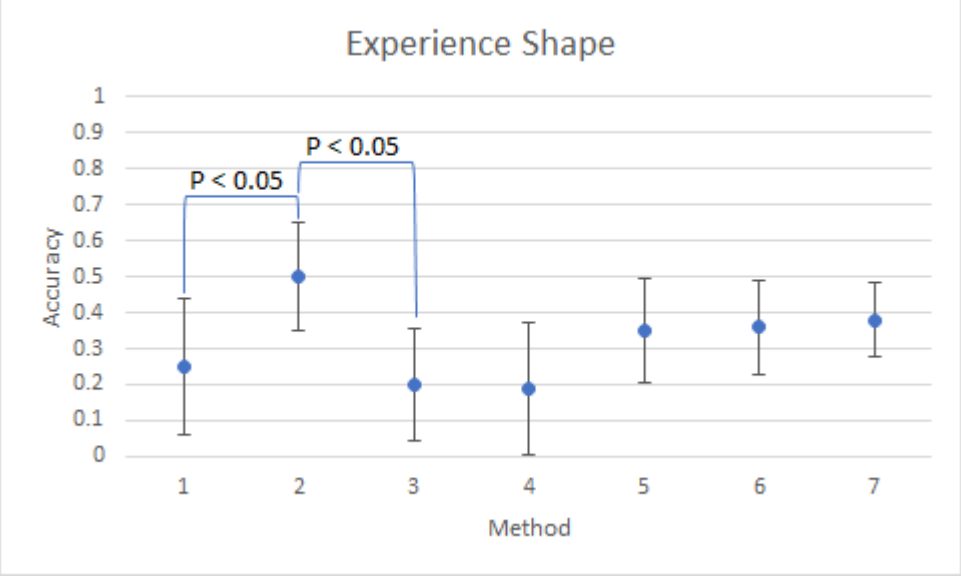
For those with significant tactile experience, the trends in performance for the difference methods appears to be different for the questions explicitly involving shape

(Graph 67) and those that do not (Graph 66). However, the trends are different than for those with no to little tactile experience.

Graph 66: Effect of Having Significant Tactile Experience on Response Accuracy for the non-Shape Questions



Graph 67: Effect of Having Significant Tactile Experience on Response Accuracy for the Shape Questions



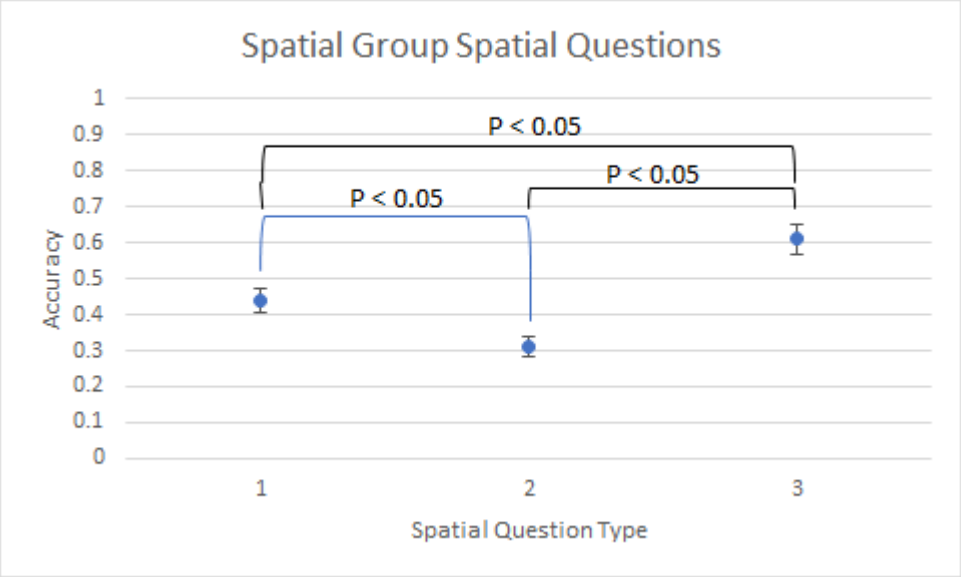
5.8.4. Spatial Questions (Individual Maps)

This analysis compared performance for different types of questions for the individual maps that required different amounts and, likely, different types of spatial cognition. The three groups of questions that were examined were: Group 1, counting the number of features in the map (Questions 11 and 12); Group 2, describing a feature that is around a location on a map (questions 14, 15, 19 and 20); and Group 3, finding a path from one location to another (Questions 16 and 17). An example of a Group 1 question is: "Benches are indicated by the color yellow. How many benches are on the map?". Examples of Group 2 type questions are: " How many points of interest are adjacent to the stairs?" and (when having a finger moved to point at a location) "could you tell me what is nearby on the pathway?". An example of a Group 3 question is: "Following the paths, that is to say, no stepping on the grass or flowers but assuming buildings are one large room, of you are at the lower left corner, find the closest bench".

A model was constructed for the metrics that compared the questions that asked in the three groups for individual maps. As the response time was not recorded per question, only a statistical model of the accuracy of the response was constructed. A model was first constructed using modality and all other factors, and then one using method and all other factors. The factors were: modality or method, question group, blindness and tactile experience. The effects included in the model were: all main effects, all two-way interactions and the two three-way interactions that included both modality/method and question group with either blindness or tactile experience.

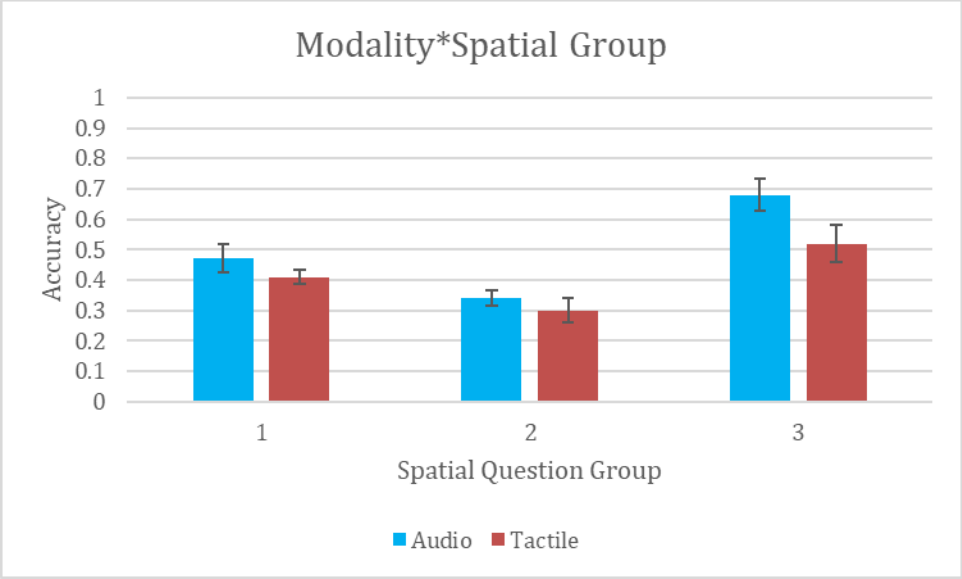
In both models, the main effect of question group was significant (Graph 68).

Graph 68: Spatial Groups Questions



For the **model involving modality** only: the main effects of both modality and question group were significant ($p < 0.001$ in both cases), two-way interaction effects of modality with blindness and tactile experience ($p < 0.001$ for both), and three-way interaction of modality, question group and tactile experience ($p = 0.039$). As our focus is on examining how performance varies with question group, we will only look at the effects of: question group (Graph 68), question group and modality (Graph 69, despite the effect not being significant, it is still of interest)

Graph 69: Effect of Modality on Response Accuracy for Different Types of Spatial Questions

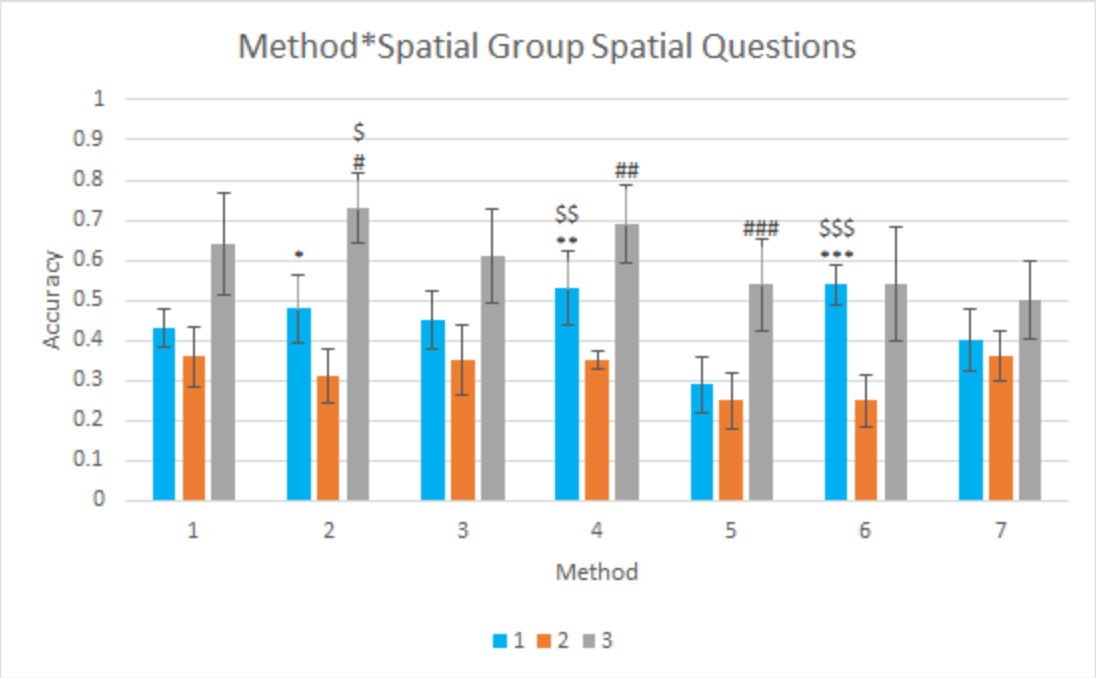


As there were not easily discernable trends in Graph 69, graphs of the 3 way interactions were not included as they are expected to be even more difficult to interpret.

For the **model involving method** only: the main effects of both method and question group were significant ($p < 0.001$ in both cases), two-way interaction effects of method with all other factors were significant ($p < 0.001$ for interactions with question group and blindness, $p = 0.006$ for interaction with tactile experience), and three-way interactions of method * questions group with both blindness and tactile experience ($p < 0.001$ for both). Only the terms involving question group are of interest here, as the other factors have been analyzed more generally.

The effect of the method used on the accuracy of the response for questions in the different question groups is given in Graph 70. Method only seems to have an effect on accuracy for those questions not explicitly involving shape.

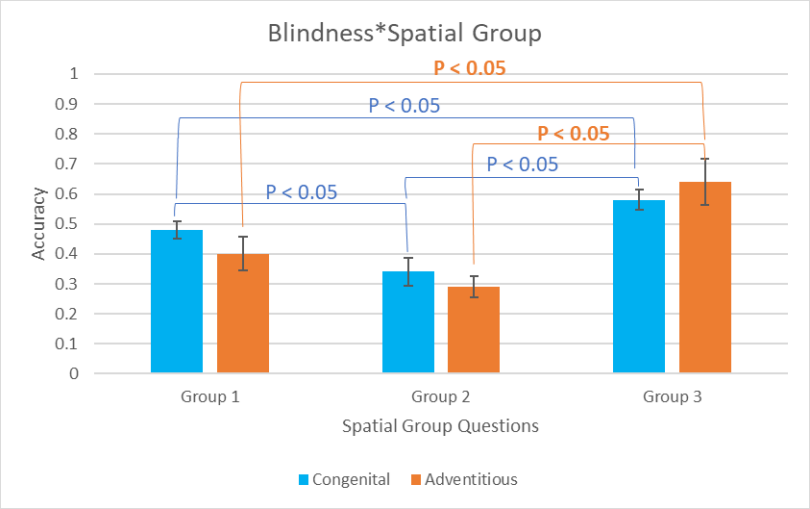
Graph 70: Effect of Method on Response Accuracy for Different Types of Spatial Questions



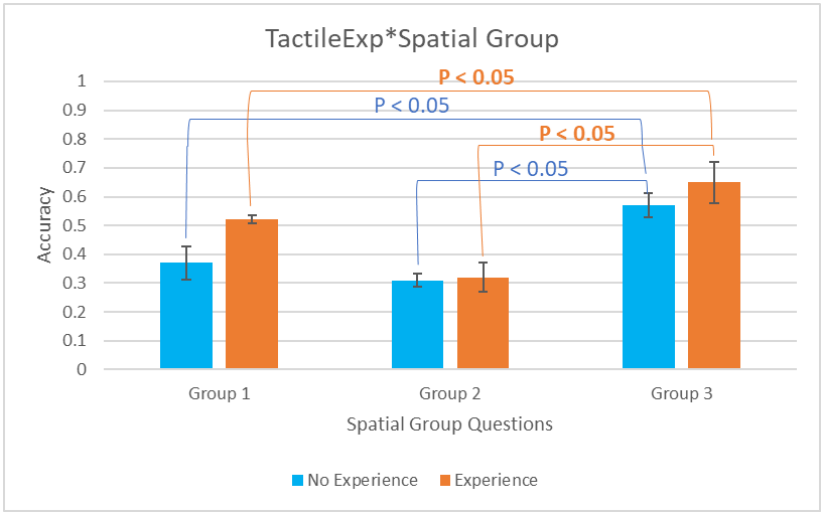
Comparing within question groups only (across methods): in Group 1, the comparison of Method 4($$) and Method 6 ($$$) to Method 5 is significant. In Group 3 the comparison of Method 2($) to Method 7 is significant. Comparing within methods (between question groups), in Method 2, the comparison of the accuracy of group 1 questions to group 2 questions and group 3 questions to both group 1 and 2 questions are significant; in Method 3; in Method 4, the comparison of the accuracy to group1 questions to group 2 questions is significant; in Method 5, the comparison of the accuracy of group 2 questions to group 3 question is significant; in Method 6, the comparison of the accuracy to group1 questions to group 2 questions is significant. As there were not easily discernable trends in Graph 70, graphs of the 3 way interactions were not included as they are expected to be even more difficult to interpret.

Despite the effects of blindness and tactile experience on spatial questions not being significant, they are still of interest (Graphs 71 and 72).

Graph 71: Effect of Blindness on Response Accuracy for Different Types of Spatial Questions



Graph 72: Effect of Tactile Experience on Response Accuracy for Different Types of Spatial Questions



5.9. Discussion

5.9.1. Within Effects of Mode, Method, and Map Type on Performance

Modality, Main Effect: When looking at the main effect of modality on response accuracy, response time and SUS scores, the only metric which showed a statistically significant difference in performance as a function of modality was the SUS scores: participants preferred using audio methods over tactile methods. This may have been, as most participants commented, because the tactile cues were more difficult to distinguish from one another compared to the audio cues. This was in spite of efforts to optimize perceptibility of all cues. However, it should be noted that the just noticeable difference in pitch is much smaller than for tactile frequency (approximately 0.5% compared to 20%), which may explain this result.

We cannot accept the null hypothesis for response accuracy and response time; however, given that the trend is for response accuracy to be higher and response time to be lower for audio cues than for tactile cues, this suggests that performance with audio cues is likely similar or better than with tactile cues (one of our key questions). However, it should be noted that somewhat different methods were used with auditory cues than with tactile cues, which could confound the results. We will also use alternative methods below, matching audio and tactile methods more closely, to compare performance between using audio and tactile cues.

Map Type, Main Effect: The main effect of map type was not so much of interest in terms of informing the choice of an assistive technology method for people who are blind or visually impaired, but in terms of how people who are blind or visually impaired process different types of spatial information. The overall maps and the individual maps

were very different structurally and in terms of the questions asked. Results showed that there was a statistically significant difference in the response accuracy as well as for the response time for the overall maps compared to the individual maps: response accuracy was higher and response time much lower for the questions involving the overall maps. This make sense because the overall maps typically contained fewer items than the individual maps and no items that were small and difficult to find as in the individual maps.: this enabled subjects to detect items more easily and quickly. How this informs how people who are blind process spatial information will be considered in more detail when we focus specifically on questions of shape and different types of spatial information (below).

Modality and Map Type Interaction: This term of the models was not statistically significant and so the interaction was not further considered. What this suggests is that there does not appear to be any selective benefit of using different modalities for the different map types. This simplifies choosing the best method for people who are blind or visually impaired to use: if performance was significantly different, we might think of choosing different modalities for different types of information diagrams; however, this would require users to learn two different sets of cues, which would be a much heavier cognitive load.

Method, Main Effect: The main effect of method was only significant for the model of the response time and SUS scores. The most noticeable effect for the response time was that Method 3 was significantly faster than Methods 4-7. However, the meaning of this result is unclear.

As for the SUS scores, there was a statistically significant difference between Method 1 and all other methods: the single fingered exploration with audio cues was found to be significantly more usable than the other methods. This is consistent with statements from participants that they felt that Method 1 was easiest to use. Combined with Method 1 showing a trend as being one of the methods with the highest accuracy (56%) and lowest response time (670 sec) suggests that this should be the preferred method for access by BVIs to spatial map information, in the absence of other factors such as the blindness condition of the user, experience with tactile graphics and question type.

Participants also stated that they felt that the single fingered exploration with tactile cues would have been just as easy to use if the cues were more discriminable. This seems to suggest that the difference in performance between using audio and tactile feedback is the difference in discrimination abilities in the two domains and not how the information is integrated with the kinesthetic feedback from the exploring hand. This is potentially supported by the result that Method 3 (audio cues for two fingers to two separate ears) despite the additional complexity of providing cues for a second finger, was also found to be significantly better than Method 5 (tactile cues for one finger). However, it is unclear whether the poorer performance accuracy of Method 5 (44%) would also improve with more effective tactile cues.

Method and Map Type Interaction: This term of the models was statistically significant for both response accuracy and response time. It did not exist for the SUS scores, because they were not recorded as a function of Map Type. In terms of the overall maps (overview maps), Method 1 (single finger, audio) had the best response

accuracy, which was statistically significant in its difference from Method 2 (two fingers, two instruments same ear) and Method 5 (single finger, tactile). However, this trend was, in some ways, opposite for the individual maps: although there was not a statistically significant difference between Method 1 and the other methods. The best method was Method 4 (two fingers, two instruments different ears), but it was only statistically significant in its difference from Method 5. Although not statistically significant, for both types of maps, the trend was for Method 5 (single finger, tactile) to perform the poorest.

The performance of the different methods in terms of responses time did not correlate with the expectations from previous results (Burch and Pawluk, 2011) that higher response accuracy would also correlate with lower response time. Method 1 did not have better response times than the other methods, but did not have statistically worse times either. Only Method 3 (two fingers, same instrument different ears) was statistically significant in its difference from Methods 4-7: it allowed for quicker response times.

Method 1 (single finger, audio) seemed to have the best performance and ease of use of all methods for overall maps. This is potentially because these types of maps have less spatial detail and with cues for a single finger easier to use, it is easier to construct a cognitive map. It should be noted that this advantage appears to disappear for individual maps: where the use of two fingers simultaneously may provide a benefit that begins to make up for the difficulty of attending to multiple cues simultaneously.

Method 5 (single finger, tactile) seemed to have the poorest performance and ease of use of all the methods. Past research (Burch and Pawluk, 2011) has attributed the better performance with provide tactile cues to two fingers due to the ability to take advantage of parallel processing of material properties across fingers. However, how it

relates to the different audio methods is less clear. For the tactile methods, tactile feedback is collocated directly to the exploring finger. For the audio methods, we expect the audio cues to be separated into audio streams correlated to the different fingers; however, they are not collocated with them. Collocating audio streams in space to the fingers is unlikely to succeed, as with touch, as human audition would not be accurate enough to separate the location of the two fingers in space.

We can also begin to answer the research questions posed at the beginning of this thesis with these results.

Question 1: Can the use of simple sonified cues with kinesthetic information work as effectively as using tactile cues with kinesthetic information? There are several different comparisons between Methods that we can examine to draw this conclusion. All audio methods can be compared to all tactile methods (modality comparison). However, it should be noted that somewhat different methods were used with auditory cues than with tactile cues, which could confound the results. The alternative is to only compare those methods that are directly similar between the two modalities: Method 1 and 5 (one finger), and Method 3 and 6 (two fingers of same hand, spatial separation of feedback only). However, these comparisons are limited in that they do not take into account that there are some audio feedback cues (timbre) that cannot be created for tactile cues. Thus, if timber was a benefit, it would be wrong not to conclude that this leads to a benefit of using audio cues. So, it may be appropriate to compare the "best" of the audio methods with the "best" of the tactile methods. However, it is difficult to define the "best" audio method as they were found to be not significantly different in response accuracy, and

only Methods 3 and 4 significantly different in response time. None of the tactile methods were significantly different in response accuracy or time.

The focus for this question is on the main effect of modality or method, as opposed to interaction effects with map type. This is because, unlike switching between methods within a modality, different hardware would need to be provided for audio versus tactile feedback and different cues would need to be taught for the features. This would make it unlikely that there would be any benefit to suggest the user switch modalities for different types of questions.

For the main effect of modality, there was not a statistically significant difference in performance. However, given that the trend is for response accuracy to be higher (54% versus 49%) and response time to be lower for audio cues than for tactile cues (687 versus 756 sec), this suggests that performance with audio cues is likely similar or better than with tactile cues (one of our key questions). Pairwise comparisons of Method 1 (single finger, audio) and Method 5 (single finger, tactile) found that response accuracy was significantly different between the two (55% for Method 1 versus 44% for Method 5), although there was no difference in response time. Pairwise comparisons of Method 3 (two fingers of same hand, spatial audio) and Method 6 (two fingers of same hand, spatial audio) found that response accuracy was not significantly difference between the two, but response time was (606 sec for Method 3 and 786 sec for Method 6). All these results are consistent with performance being equal or better using auditory cues as compared to tactile cues. This is of significant benefit as battery power consumption on mobile devices is much less for generating sounds as compared to vibrations. If multiple fingers are used, the hardware is simpler and easily commercially available.

Question 2: Can the use of feedback of cues for two exploring fingers provide better performance, in general, than a single exploring finger? Does this depend on the modality? For tactile feedback methods, there was no statistically significant differences in performance measures (accuracy, response time) and ease of use amongst the three different methods. However, trends showed that response accuracy was higher for two fingered exploration compared to one fingered exploration, and that the difference increased when the individual maps were used (36% compared to 45%) as compared to the overall maps (53% compared to 57%). This is consistent with the hypothesis that the use of two fingered exploration is more beneficial for tasks that are more spatially demanding. However, in contrast to previous results (Burch and Pawluk, 2011), response time also increased (2% for overall maps and 25% for individual maps). This may be because the types of questions asked here were different than in Burch and Pawluk (2011) which were related to object identification. A better comparison would be to examined user performance on questions involving shape, which is also needed for object identification.

For audition, there was no significant difference in performance between using one finger feedback and two fingered feedback, both in terms of response accuracy and response time..... relevant that better than tactile two finger. Trends suggest that the single fingered method (Method 1) was better than the two fingered methods (Methods 2-4) for the overall maps but worse for the individual maps. This may again be because two fingers become more useful when more spatial detail is needed to be interpreted. However, this effect is small if any. There is no clear difference comparing response time for single fingered methods (Method 1) and two fingered methods (Methods 2-4).

Question 3: Can perceptual principles of audio stream segregation be used to effectively relay information about two exploring fingers? (as compared to one exploring finger and compared to each other: timbre, spatial location or both). Although both methods of using audio stream segregation did not seem to provide a statistically significant decrease in performance, they also did not appear to provide the benefit that has previously been experience with tactile cues (Burch and Pawluk, 2011). This makes the relevancy of this question moot.

Question 3b: For tactile cues, is there a difference between using two fingers on the same hand and two fingers on different hands? There was no difference between using two fingers on the same hand and two fingers on different hands. This is consistent with previous results (Burch and Pawluk, 2011). However, Burch and Pawluk (2011) required that the fingers always be side by side, even when on different hands. This was not a requirement for this experiment. Unfortunately, the fingers were not tracked so we are unable to determine how much of the time the fingers on the different hands were placed side by side and how much of the time one finger was used as a reference for the other or explore for information content (the other common uses of multiple fingers).

Is there a best method to recommend? Considering performance and usability, Method 1 (single finger, audio) appears to be the best method to use overall. There may be a slight benefit to switching to use Method 4 (two fingers, two instruments, two ears) for individual maps, but this does not appear to be significant. This also has the advantage of minimum power consumption compared to using tactile feedback.

5.9.2. Further Including Between Factors of Blindness and Tactile Diagram Experience

Although the between factors of type of blindness (congenitally blind versus adventitiously blind) and tactile diagram experience (little to none versus expert) were not used to power the number of subjects for the experimental study, these factors are still of interest: it is possible that users with different characteristics should be given different recommendations for best performance. However, caution is needed in interpreting these results due to the small number of participants in each category.

Blindness and Tactile Experience, Main Effects and Interaction: For the main effect of blindness condition (congenitally versus adventitiously blind), it is shown that there was no real difference in terms of accuracy between the adventitiously blind and the congenitally blind. However, the adventitiously blind took a significantly longer time to answer all the questions (812 sec compared to 640 sec). This was surprising, as many researchers have suggested that congenitally blind individuals normally perceive space in an egocentric coordinate system, whereas adventitiously blind individuals use an allocentric representation (similar to sighted individuals). One would expect that congenitally blind individuals would find it more difficult to understand maps where space is represented in allocentric coordinates.

As the adventitiously blind took significantly longer to explore the maps, they were often left feeling frustrated. This is likely why they gave significantly lower SUS scores than the congenitally blind.

What was surprising was that there was a statistically significant interaction effect between Blindness and Modality: individuals who were adventitiously blind had a significant difference in performance between using audio cues (57% correct) and tactile

cues (46%). This may be because adventitiously blind individuals are less comfortable using touch (many prefer the use of screen readers over learning braille).

For the main effect of tactile diagram experience, it was shown that there was no real difference with the accuracy in answering questions overall. Yet, those with significant tactile diagram experience answered questions significantly quicker than those who have little to no tactile diagram experience (577 sec versus 898 sec). This was not surprising for the methods using tactile cues. What was surprising was that there was no significant effect of modality on performance: those with tactile diagram experience did just as well and responded just as quickly with audio feedback (which they would not have had experience with for diagram access) as with tactile feedback. This suggests that there might be similar mechanisms as to how audio and tactile feedback are used with kinesthetic information of exploring fingers to interpret diagrams. It also means that those with significant tactile diagram experience should not be encouraged to use tactile cues as there is no advantage to it, and considerable battery power needs to be expended to provide vibration cues on mobile devices.

However, the fact that significant tactile diagram experience did not improve accuracy of the responses was a bit disconcerting. It suggests that, even with significant practice with the proposed access methods, people who are blind or visually impaired may not improve their accuracy higher than 60%. This suggests that an additional component may need to be added to the access methods to improve performance. One possibility is using a short word introduction to each map before they are used.

The fact that those with significant tactile diagram experience had shorter response times and were familiar with accessing diagrams is, just like for blindness

condition, believed to be the reason why those with little to no tactile experience gave lower SUS scores than those who have experience.

The interaction of the terms blindness and tactile experience showed that those who are congenitally blind and have tactile experience were significantly quicker than all other groups. This is consistent with what previous research has found (Dulin and Hatwell, 2006). However, caution needs to be used when interpreting our results due to the small number of participants in each group (4, 2, 2 and 2, respectively).

Blindness Interacting with Method: Examining response accuracy with the different methods as a function of Blindness, supports the results of examining performance with the different modalities as function of Blindness (described early). However, the response times do not show this trend: Method 3 (two fingered, audio with two ears, spatial) and Method 5 (one fingered, tactile) appeared to have the quickest response times.

Tactile Experience Interacting with Method: When looking at the interaction of terms method and tactile diagram experience on response accuracy, the one method that stands out for those with tactile diagram experience is Method 4 (68% accuracy). This method is significant over Methods 1, 2, 5, and 6. Trends suggest that it is also better than Method 3 and 7. This suggests a potentially different recommendation for use of maps on mobile devices than for the general population: individuals with significant tactile diagram experience would benefit from using two exploring fingers with audio feedback for each finger to different ears and with a different timbre. Performance compared to the default recommendation (Method 1) is 68% accuracy versus 58% accuracy. However, the

response time is slightly slower for Method 4 compared to Method 1 (630 sec versus 569 sec), but not statistically significant.

For individuals with little to no tactile diagram experience, response accuracy for all methods was much more homogeneous. This was surprising, as we would have thought they would have had more difficulty perceiving the tactile cues due to their lack of experience, and, therefore, performed more poorly with tactile cues. However, there were statistically significant difference in performance between Method 1 (single finger audio) and Method 5 (single finger tactile), with Method 1 performing better. Although there were more variations in response times across methods, none of the differences were statistically significant except between Method 4 and Method 5. We would still suggest that those with little to no tactile diagram experience to use Method 1.

5.9.3. Shape vs. Not Shape Questions

The overall (overview) map questions were divided into two groups. The first group dealt with questions that did not ask about the overall shape of a garden, while the second group asked explicitly about the shape of a specific garden. When comparing the questions asking about shape to the questions that did not ask about shape, non-shape questions were answered significantly better than the shape questions. (Note that we cannot compare response times as they were measured per map, due to the influence of exploration time answering any early questions effecting the response time on any subsequent question on the map). This difference is not a surprise as it is known that it is difficult for individuals who are BVI to distinguish shapes. This requires a much more

detailed integration of spatial information than other spatial questions (what is close to the stairs) and ones that purely involve counting (how many benches on the map).

A more specific reason was that it was noticed during testing that, when a participant was asked about the shape of the garden, one of the hardest aspects of determining the shape was establishing the border of the given shape. This was especially true when one or more of the borders of a shape were diagonal. As a result, the subjects had difficulty in distinguishing if the shape was going diagonally, straight, or if it was curved. For example, question two of overall map C asked the shape of the green garden which was a triangle. Eight out of ten test subjects answered this question by saying it was a rectangle/square, while one said L-shaped, and the other said a circle.

Effect of Modality and Method on how Question Types were answered: there was no significant difference in the trends of the response accuracy to question type based on the modality or method. There was also no significant difference in response accuracy between any of the methods that were used to answer the shape questions. Yet, for there were differences for the non-shape questions; however, there was no clear meaning to the trends.

Effect of Blindness on Performance on Question Type as a Function of Modality and Method: Trends showed that participants who were congenitally blind actually had better response accuracy using tactile methods as compared to audio methods for the shape questions. However, unexpectedly, response accuracy was poorer for the shape questions using tactile cues for two fingers rather than one (although the difference was not statistically significant). This was surprising, as we had predicted based on past results that determining shape, as it requires more spatial information, would have better

performance with two fingers. However, it may be that asking a question about shape requires the participant to trace the borders of an area, which can only be processed a single finger at a time. The response accuracy for the adventitiously blind participants for shape questions had the opposite trend than for the congenitally blind: they performed better with the audio methods than with the tactile methods. There is also some indication that performance was better with two fingered audio and touch as compared to using a single finger. The reason was not clear.

For the non-shape questions, both congenitally and adventitiously blind individuals had similar response accuracies for the non-shape questions. The effect of method on the response to non-shape questions was discussed earlier.

Effect of Tactile Experience on Performance on Question Type as a Function of Modality and Method: Trends showed that participants who had significant experience with tactile diagrams had better response accuracy using tactile methods as compared to audio methods for the shape questions. However, when examining the effect of method, the second audio method (2 fingers, one ear, two instruments) had better performance than even the tactile methods, although the differences between methods were not statistically significant. It is difficult to explain this latter anomaly, but it is reasonable to assume that experience with tactile diagrams benefited performance with tactile cues. Although there was some transference for audio cues, it did not appear as great as for tactile cues.

For participants with little to no tactile diagram experience, the trend for shape questions was opposite that for those with significant tactile diagram experience: they did better with audio methods than with tactile methods. This time Method 3 (2 fingers, two

ears, spatial segregation) performed best. This is consistent with previous results suggesting that individuals with little to no tactile diagram experience would do better using one of the audio methods to explore a mobile map.

For non-shape questions, performance was more comparable between the two modalities for both groups, although with performance with touch being worse in the group who had little to no experience with tactile diagrams. This is not surprising given the different exposure to tactile diagrams. It also appears that practice with tactile diagrams does improve performance with tactile cues more noticeably than with audio cues, suggesting less transfer in learning between the modalities than originally thought.

5.9.4. Spatial Awareness Questions

To investigate differences in performance for different question types involving different amounts and types of spatial awareness, some of the map questions for the individual maps were further analyzed. The first group was questions dealing with how many items (Group 1). For example, questions asking how many buildings on the maps, or how many benches? The second group pertained to questions asking what is around a given location (Group 2). For example, how many benches around a point of interest? The third group dealt with following a path to find a location (Group 3). For example, starting from the pathway at the top left corner follow the path to find the closest bench?

When comparing participant performance in answering each of these groups of questions, surprisingly participants performed best answering Group 3 questions, which relate to following a path; this was followed by Group 1 questions (counting) and Group 2 (indicating what is around a point): differences were statistically significant with values

of 0.60, 0.44 and 0.32, for Groups 1, 2 and 3, respectively. However, it may be that constraining the search space to the path and only requiring participants to find the closest rather than all of a feature, may have made the question easier to answer. The path may have also helped with keeping participants oriented in the map when answering the questions.

Effect of Modality and Method on Answers to the Different Types of Spatial Questions: The effect of modality on the answers to the different types of spatial questions (Modality * Spatial Question Group interaction) was not significant. However, there does seem to be more of a difference for Group 3 questions than for the other question types: audio methods performed noticeably better than tactile methods for Group 3 questions. For the interaction of spatial group questions and methods used, there are no obvious trends in the data that have not been described. What can be seen is that trends in the data using two fingers with tactile cues seems be potentially beneficial only for questions from Group 1 and Group 2. As re-interpreted earlier: these questions may require more of a construction of a spatial, cognitive map then Group 3 questions. For the audio methods, there is a bit of a trend suggesting that two fingers may be potentially beneficial for questions from Group 1 and Group 2. This is consistent with the results from the tactile methods. However, the trend in the audio methods is much less clear than for the tactile methods.

Effects of Blindness and Tactile Experience on Answers to Different Types of Spatial Questions: The effects of blindness and tactile experience on spatial questions is not significant, as similar results to that of the effect of method on spatial questions were

produced for each group. This means that neither blindness groups or tactile experience groups were able to answer any of the spatial group questions better than others.

6. Conclusion

Overall it was shown that the audio methods were equal to if not better in performance than the tactile methods in terms of response accuracy, time, and user preference. Since touch screen devices require considerably less power to generate audio cues than tactile cues, this means that it is better to use to use audio cues for touch screen devices to preserve battery power. As a result, it will allow for people to use a device for much longer.

It was also shown that the single finger audio method was clearly preferred over all the other methods by blind and visually impaired participants. This makes sense because the method did not require users to try and decipher multiple fingers. It is believed that this would also have been true for the single finger tactile method as well, if the test subjects did not have trouble distinguishing the tactile cues.

Although the single finger method was preferred by participants, it was shown that the two finger methods, for both audio and tactile, performed slightly better than the single finger methods for the individual maps. This suggests that using multiple fingers may be better when searching more complex maps. This may be because multiple fingers allow for a larger surface area in contact with the map, allowing for the user to gather and integrate more information simultaneously.

It was also shown that those participants with significant tactile diagram experience clearly benefited from using two fingers, two instruments, played to separate

ears (Method 4). This is thought to be a result of their experience working with tactile graphics. The experience allowed them to be able to process the multiple cues from the map better, allowing them to gather more information. Interestingly, their experience seemed to translate to the use of audio cues.

When answering questions about shape, subjects did better when using audio methods. This is especially true for the adventitiously blind, who performed significantly better with the audio methods. This make since, because most people who are adventitiously blind are reluctant to use tactile feedback (e.g., braille, etc.) and are more experienced with audio feedback (e.g., screen readers) However, those participants with little to no tactile diagram experience also performed the best using two fingers with each finger represented by the same instrument played to separate ears (Method 3). Somewhat similar, those with significant tactile diagram experience performed the best using two fingers with each finger represented by different instruments played to the same ear (Method 2). This suggests that there is a benefit in using audio over tactile in answering shape questions, although the reason is unclear.

As for the spatial group questions, it is shown that no method truly helped in answering any of the different spatial questions. Instead people are just better at answering spatial group 3 questions (follow the pathway to find closest bench?). This is a bit of a surprise as it was thought that people would struggle with following the narrow pathway. Instead it may have been a benefit as it gave them something to follow while the other spatial group questions required searching around an undefined area around a specific point or the entire map.

6.1. Future Research

Although there was no main significance in any of the seven methods used, results showed that some methods were able to help certain groups search different types of maps, and answer specific questions. In this research, the test subject was allowed to get comfortable with each of the seven different methods used to search the maps, and then allowed to search a practice map before being presented with the test overall or individual map. However, they were not presented with any more information about the map they were searching than what type of map it was. Future research should describe what the map is showing, allowing for them to get a better understanding of what the map is showing, to see if a description of the map may help BVI individuals any in the searching of maps.

Also, test subjects struggled in answering questions about shapes in the overall maps, especially when it came to identifying diagonal lines. Future research should be conducted on adding borders to each of the gardens for the overall maps, and be given a unique tone so it can be easily identified. This should be done to see the effects of adding a border on the identification of shapes.

For the individual maps, people struggled the most in answering questions about what is around a specific point. One reason why is because they had difficulty keeping track of the original location. While another reason why is because they were not sure if the point of interest or the bench they had found was the same one as before or if it was a different one; this is especially true since the speech feedback only gave the color. Further research should be conducted on what effect of adding labels to points of interest and benches has on spatial questions.

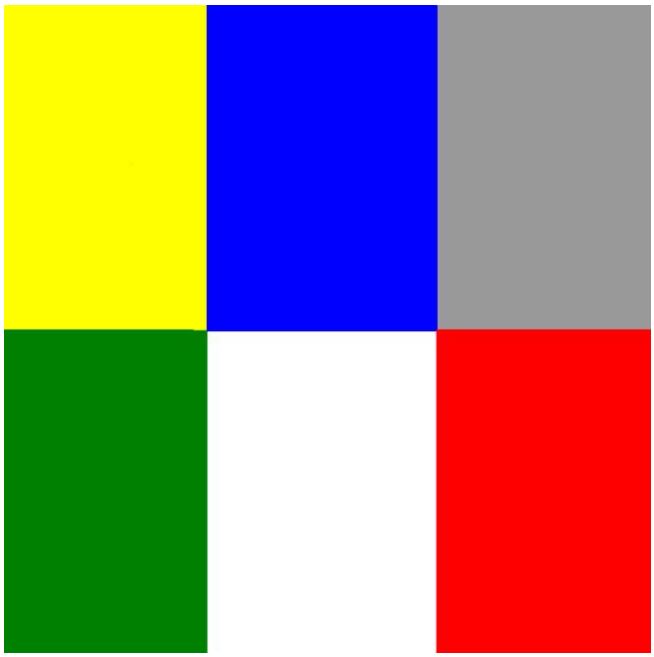
Also, for the individual maps the use of auditory icons (running water, bench creaking, etc.) to represent points of interests and benches instead of musical notes should be researched. Since the points of interest and benches are only small dots (except for the fountain) they do not need to be represented by continuous noise, and can be replaced by a unique auditory icon. Which may help these features become easier to identify, and stand out more. This may be extremely helpful, since it will also allow for the reduction of musical notes that they would have to identify and make searching more complex maps less mentally taxing as a result.

In the case of tactile feedback, different methods to provide tactile feedback should be explored. This is because the device from Barron Associates did not do a great job at presenting tactile frequencies simultaneously, as a result the delay of the frequencies had to be limited: some people had difficulty in determining which frequencies was being played. Once a new method to provide tactile feedback has been found, testing should be conducted to determine if similar, or different, results are found for the tactile methods conducted in this study.

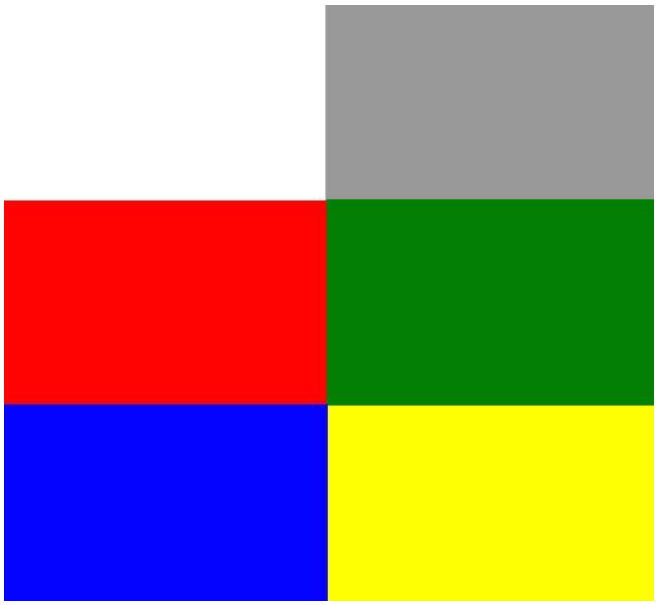
7. Appendix

7.1. Test Maps

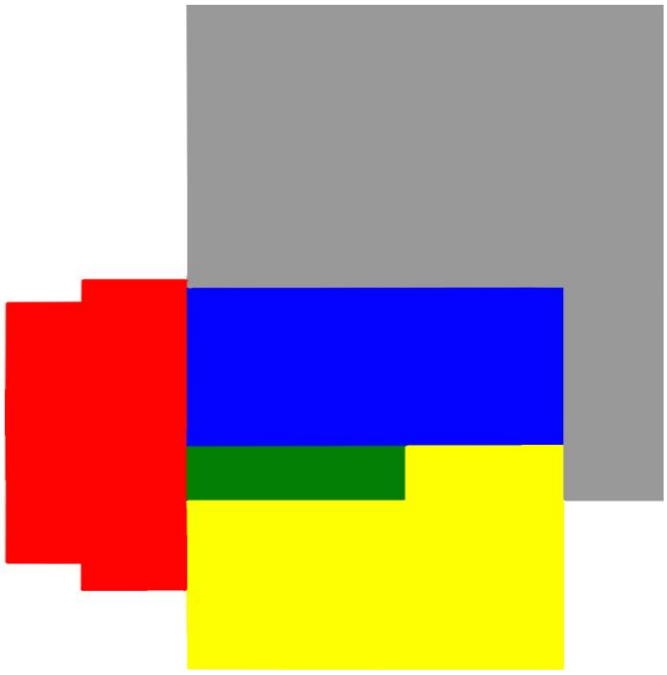
7.1.1. Tablet Initial Map



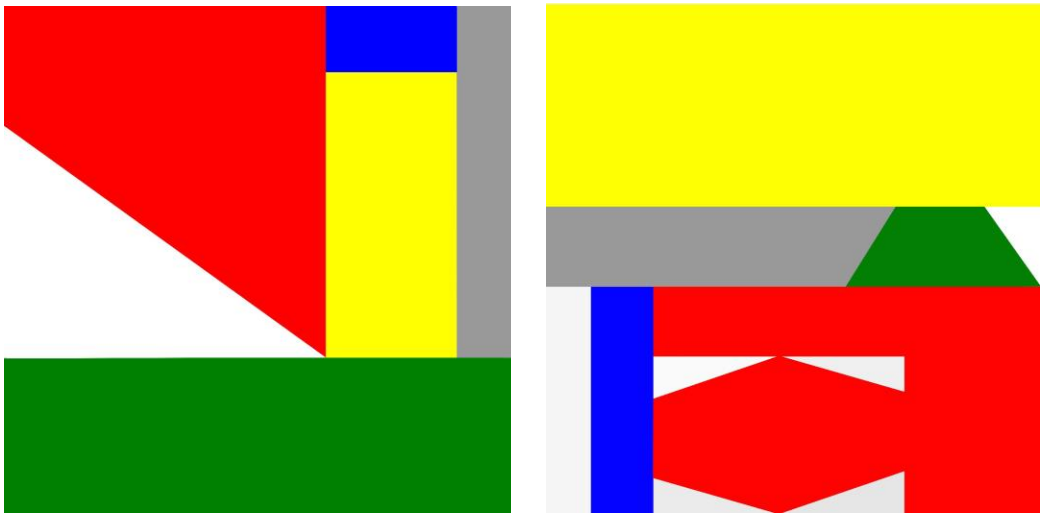
7.1.2. iPad Initial Map

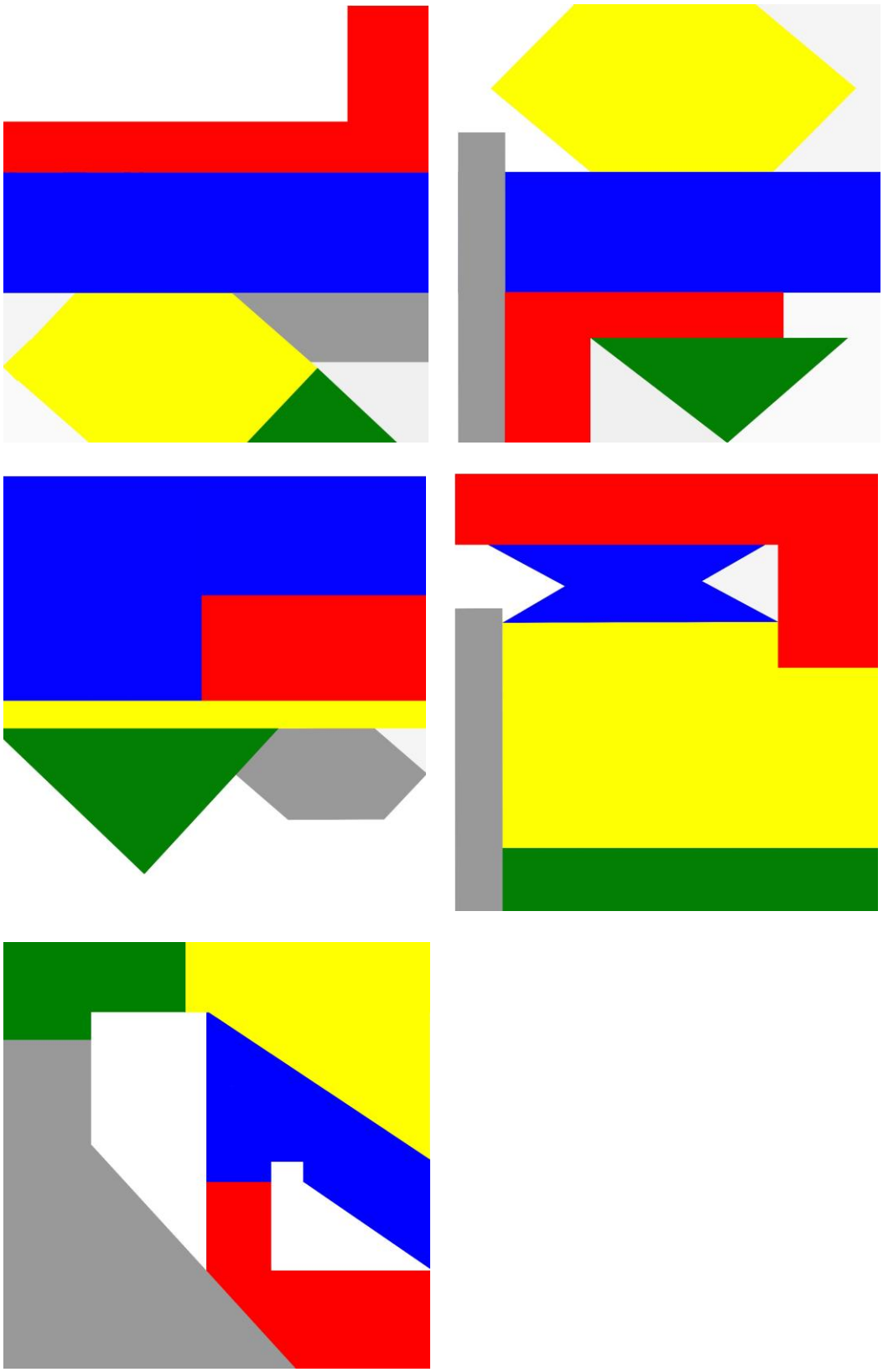


7.1.3.Practice Overall Map

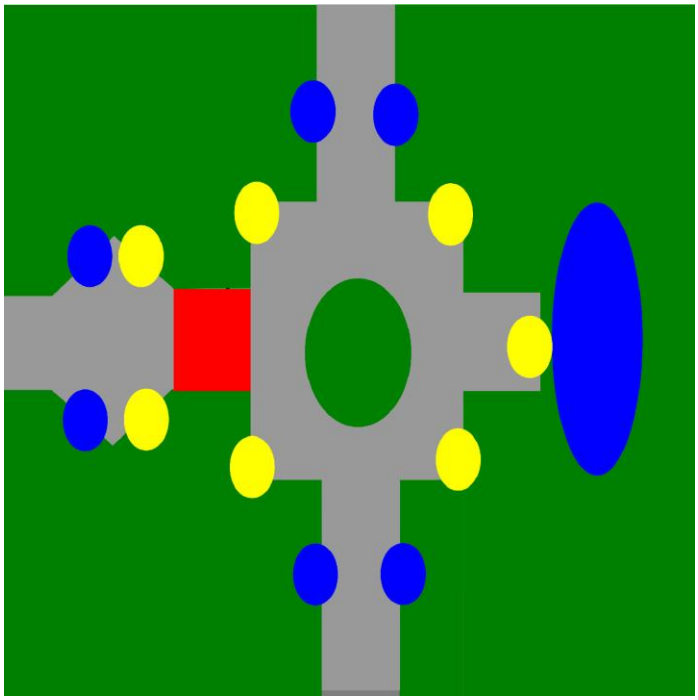


7.1.4. Overall Maps

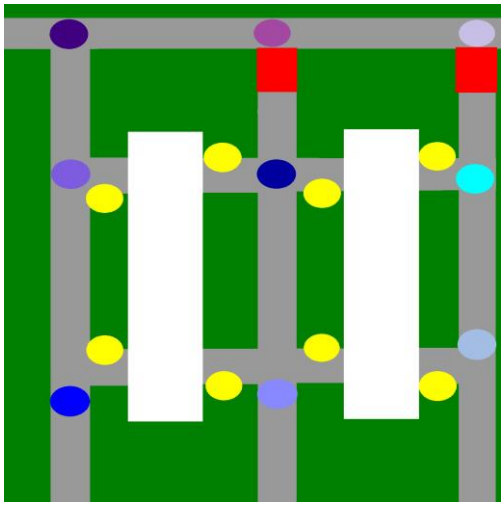


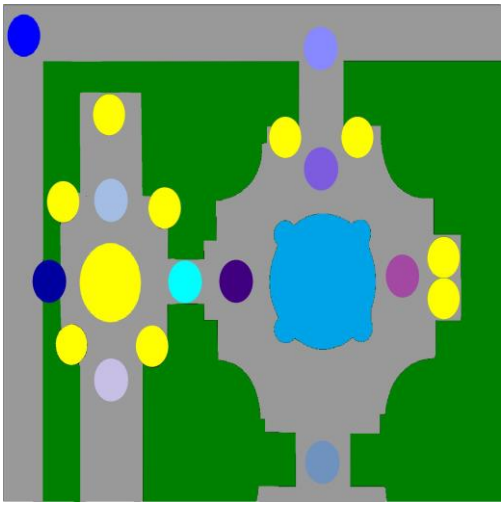


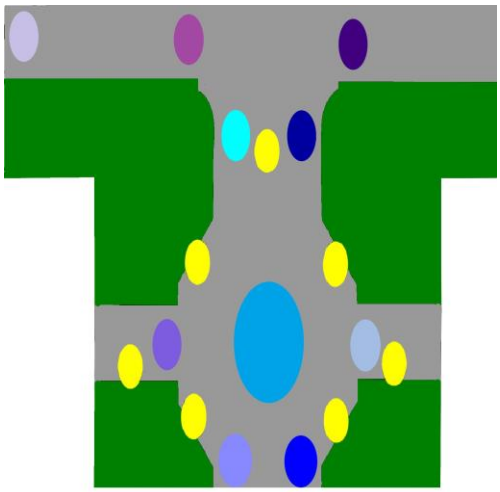
7.1.5. Practice Individual Garden Map

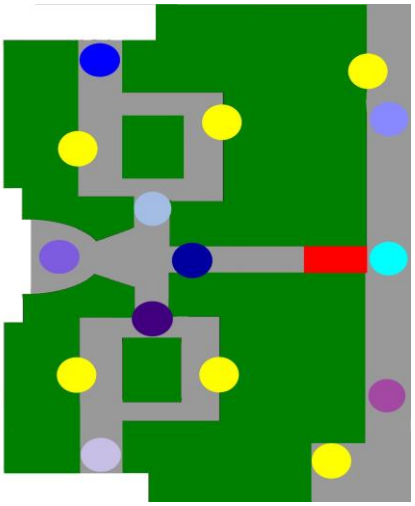


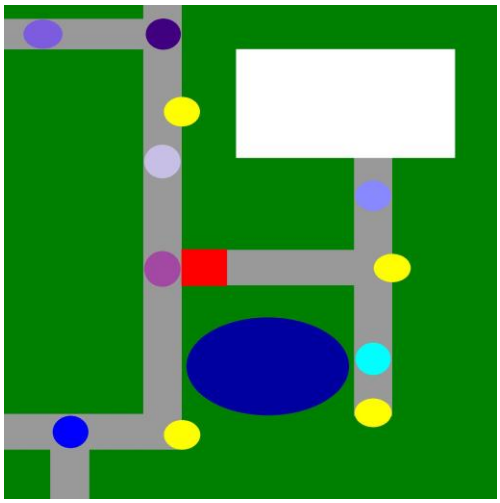
7.1.6. Individual Garden Maps

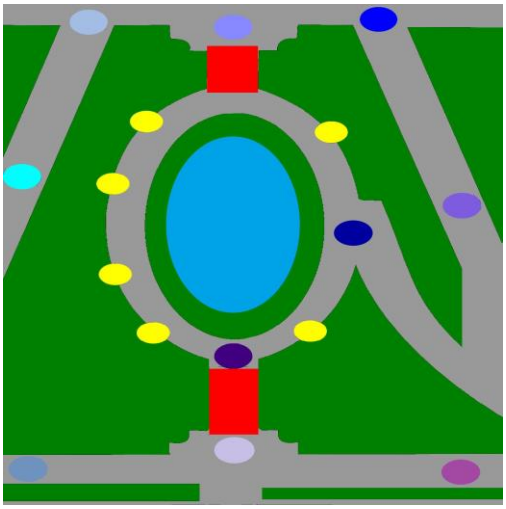


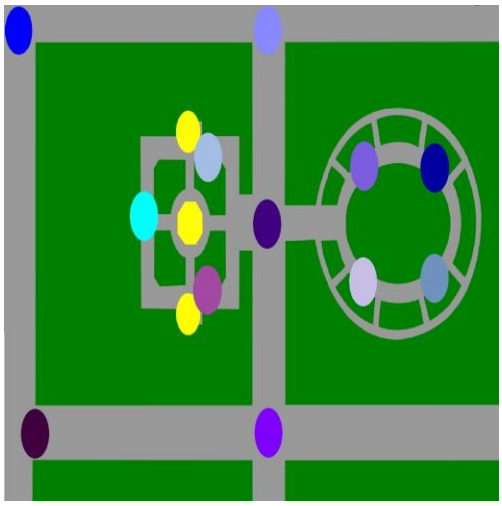












7.2. Apple Code

7.2.1. XCODE Main Page


```
#import "ESTFirstViewController.h"
#import "ESTDistanceStorage.h"
#import "ESTViewController.h"
#import "ESTOrientationVC.h"
#import "ESTTourChoiceVC.h"
#import "ESTMapChoiceVC.h"
#import "ESTTableMapVC.h"
#import "ESTSettingsVC.h"
#import "ESTChangeSettingsVC.h"
#import "ESTTableTourChoiceVC.h"
#import <AVFoundation/AVFoundation.h>
#import <AudioToolbox/AudioToolbox.h>
#import <UIKit/UIKit.h>

@interface ESTFirstViewController ()
{
    AVAudioPlayer *audioPlayer;

}

@property (retain, nonatomic) AVSpeechSynthesizer *syn;
@property (retain, nonatomic) AVSpeechSynthesizer *altSyn;
@property (nonatomic, strong) AVSpeechUtterance *utterance;
@property (nonatomic, strong) NSTimer *timer;

@end

@implementation ESTFirstViewController

float utterValue;
int repeatTimer;
int i;

//--Helping to determine if the condition of three fingers required is met--
UIGestureRecognizerState state;

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view from its nib.

    //--Adding a title to the tab--
    self.title = @"Lewis Ginter App";

    //--Adding a UIBarButton to the tab in ESTFirstViewController--
    UIBarButtonItem *storeButton = [[UIBarButtonItem alloc] initWithTitle:@"Distance" style:UIBarButtonItemStylePlain target:self action:@selector(storeBtnTapped:)];
    self.navigationItem.leftBarButtonItem = storeButton;

    //--Setting up SpeachSynthesizer--
    self.syn = [[AVSpeechSynthesizer alloc] init];
    self.utterance = [[AVSpeechUtterance alloc] init];

    //--Setting up swipe gesture recognizer--
    //--**Not usable when using VoiceOver **--
    //--Looking for swipe in the Left direction, then calling the function to handle it--
    UISwipeGestureRecognizer *swipeGesture = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    swipeGesture.direction = UISwipeGestureRecognizerDirectionLeft;
    //--Making sure that there are 1 fingers on the screen before the gesture is called--
    swipeGesture.numberOfTouchesRequired = 1;
    //--Adding the gesture to the view controller--
    [self.view addGestureRecognizer:swipeGesture];

    //--Looking for swipe in the Up direction--
    UISwipeGestureRecognizer *swipeGestureUp = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    swipeGestureUp.direction = UISwipeGestureRecognizerDirectionUp;
    //--Making sure that there are 1 fingers on the screen before the gesture is called--
    swipeGestureUp.numberOfTouchesRequired = 1;
    //--Adding the gesture to the view controller--
    [self.view addGestureRecognizer:swipeGestureUp];

    //--Looking for swipe in the Right direction--
    UISwipeGestureRecognizer *swipeGestureRight = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    //--Making sure that there are 1 fingers on the screen before the gesture is called--
```


iOS Device
Examples | Build Examples: Succeeded | 12/31/18 at 6:29 PM
33
Examples
Examples
ESTFirstViewController.m
-viewDidLoad
self.utterance = [AVSpeechUtterance speechUtteranceWithString:@"Main Page. For Maps please swipe right. For Tours please swipe left. For settings please swipe
if (self.utteranceRate == NULL) {
self.utterance.rate = 0.15;
NSLog(@"Speaking at default rate");
}
else {
//--Setting the rate based on choice from settings--
self.utterance.rate = [self.utteranceRate floatValue];
NSLog(@"Speaking at chosen rate");
//NSLog(@"%f",self.utterance.rate);
}
//--Telling the User what is on the Main Page
[self.syn speakUtterance:self.utterance];
//--Setting up the timer so that it will repeat itself if there is no activity--
repeatTimer = 45;
self.timer = [[NSTimer alloc] init];
self.timer = [NSTimer scheduledTimerWithTimeInterval:1.0 target:self selector:@selector(repeatMethod:) userInfo:nil repeats:true];
//--Calls the closedAgain Method that controls what happens when the App is closed--
[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(closedAgain) name:UIApplicationDidEnterBackgroundNotification object:nil];
//--Calls the openAgain Method that is called when the app is opened--
[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(openAgain) name:UIApplicationDidBecomeActiveNotification object:nil];
}
//--Handling what happens when the App is sent to the background--
- (void)closedAgain {
//--Stopping speach when the app is closed--
[self.syn stopSpeakingAtBoundary:AVSpeechBoundaryImmediate];
//[self.timer invalidate];
}
//--Handling what happens when the App is opened again from the background--
- (void)openAgain {
//--Making sure that the speach is stopped and that speach is not getting repeated--
[self.syn stopSpeakingAtBoundary:AVSpeechBoundaryImmediate];
NSArray *instructionList;
//--List of all possible View Controllers--
//--**Need to ADD ESTCHANGESETTINGS and ESTORIENTATIONVC and ESTMAPCHOICE and ESTDISTANCESTORAGE**--
NSArray *searchList = @[@"ESTFirstViewController", @"ESTTableMapVC", @"ESTTableTourChoiceVC", @"ESTChangeSettingsVC", @"ESTOrientationVC"];
//--Searching for which View Controller is being shown--
UIViewController *theViewController = [self.navigationController topViewController];
NSLog(@"%@",theViewController);
NSString *strClass = [NSString stringWithFormat:@"%@",theViewController];
for (int j = 0; j <= 4; j++) {
NSRange searchResult = [strClass rangeOfString:[searchList objectAtIndex:j]];
if (searchResult.location == NSNotFound) {
NSLog(@"String not found");
}
else {
NSLog(@"I found it");
strClass = [searchList objectAtIndex:j];
break;
}
}
//--Determining the utterance to be spoken based on which View Controller is being shown--
if ([strClass isEqualToString:@"ESTFirstViewController"]) {
//--**If this was a wave file it would be easier to stop and start based on different directions**--
instructionList = @[@"Welcome to the Main Page of the Lewis Ginter Botanical Garden's App! This app will be your guide as you explore the gardens. But firs
assist you in the use of the device.", @"First the device needs to be held in portrait mode at all times so that it can recieve it signal clearly as we
device is not being held in the proper orientation the device will stop speaking and will make a 'swooshing' noise to let you know that it is not orien
double tap the screen at anytime. Also any instructions will automatically be repeated after 30 seconds of inactivity", @"In order to initiate a commar
desired direction for the action chosen. For example, on the Main Page you can swipe right to choose from the complete selection of Maps to explore, sw
change Settings such as the rate of speach, or swipe up to get a complete overview of all Instructions", @"Please make your desired selection now"];


```
//--Looking for swipe in the Up direction--
UISwipeGestureRecognizer *swipeGestureUp = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
swipeGestureUp.direction = UISwipeGestureRecognizerDirectionUp;
//--Making sure that there are 1 fingers on the screen before the gesture is called--
swipeGestureUp.numberOfTouchesRequired = 1;
//--Adding the gesture to the view controller--
[self.view addGestureRecognizer:swipeGestureUp];

//--Looking for swipe in the Right direction--
UISwipeGestureRecognizer *swipeGestureRight = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
//--Making sure that there are 1 fingers on the screen before the gesture is called--
swipeGestureRight.direction = UISwipeGestureRecognizerDirectionRight;
swipeGestureRight.numberOfTouchesRequired = 1;
//--Adding the gesture to the view controller--
[self.view addGestureRecognizer:swipeGestureRight];

//--Looking for swipe in the Down direction--
UISwipeGestureRecognizer *swipeGestureDown = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
swipeGestureDown.direction = UISwipeGestureRecognizerDirectionDown;
//--Making sure that there are 1 fingers on the screen before the gesture is called--
swipeGestureDown.numberOfTouchesRequired = 1;
//--Adding the gesture to the view controller--
[self.view addGestureRecognizer:swipeGestureDown];


//--Setting up tap gesture recognizer--
UITapGestureRecognizer *tapOnce = [[UITapGestureRecognizer alloc] initWithTarget:self action:@selector(mapBtnTapOnce:)];
UITapGestureRecognizer *tapTwice = [[UITapGestureRecognizer alloc] initWithTarget:self action:@selector(screenTapTwice:)];
//--**To many taps in order to be useful**--
//UITapGestureRecognizer *tapThrice = [[UITapGestureRecognizer alloc] initWithTarget:self action:@selector(tapThrice:)];

//--** If using VoiceOver the number of taps automatically increase by one **--
//--In other words a single tap would be a double tap and a double tap would be a triple tap--
tapOnce.numberOfTapsRequired = 1;
tapTwice.numberOfTapsRequired = 2;
//tapThrice.numberOfTapsRequired = 3;
//--stoping tapOnce from overriding tapTwice--
[tapOnce requireGestureRecognizerToFail:tapTwice];
//[tapTwice requireGestureRecognizerToFail:tapThrice];

//--Adding the tap gesture recognizer to the mapButton--
[self.mapButton addGestureRecognizer:tapOnce];
//--Adding the tap gesture recognizer to the UIView so they can get it to repeat instructions--
[self.view addGestureRecognizer:tapTwice];
//[self.mapButton addGestureRecognizer:tapThrice];


//--Adding the left, right, up, and down image arrows--
self.leftArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 150, 80, 40)];
self.leftArrow = [UIImage imageNamed:@"bold_left_arrow.jpg"];
self.leftArrowHolder.image = self.leftArrow;
[self.view addSubview:self.leftArrowHolder];

self.rightArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(240, 150, 80, 40)];
self.rightArrow = [UIImage imageNamed:@"right_bold_arrow.jpg"];
self.rightArrowHolder.image = self.rightArrow;
[self.view addSubview:self.rightArrowHolder];

self.upArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(140, 0, 40, 80)];
self.upArrow = [UIImage imageNamed:@"up_bold_arrow.jpg"];
self.upArrowHolder.image = self.upArrow;
[self.view addSubview:self.upArrowHolder];

self.downArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(140, 335, 40, 80)];
self.downArrow = [UIImage imageNamed:@"down_bold_arrow.jpg"];
self.downArrowHolder.image = self.downArrow;
[self.view addSubview:self.downArrowHolder];

//self.utteranceRate = @" ";
NSLog(@"%@",self.utteranceRate);

self.utterance = [AVSpeechUtterance speechUtteranceWithString:@"Main Page. For Maps please swipe right. For Tours please swipe left. For settings pleas
```


iOS Device
Examples | Build Examples: Succeeded | 12/31/18 at 6:29 PM
33
Examples
Examples
ESTFirstViewController.m
-viewDidLoad
desired direction for the action chosen. For example, on the Main Page you can swipe right to choose from the complete selection of Maps to explore, change Settings such as the rate of speach, or swipe up to get a complete overview of all Instructions", @"Please make your desired selection now"];
repeatTimer = 90;
}
else if ([strClass isEqualToString:@"ESTTableMapVC"]){
//--**NEED TO UPDATE THE UTTERANCE FOR ESTTABLEMAPVC**--
instructionList = @[@"Woohoo",@"it",@"really",@"worked",@"well!"];
}
else if ([strClass isEqualToString:@"ESTTableTourChoiceVC"]){
//--**NEED TO UPDATE THE UTTERANCE FOR ESTTABLETOURCHOICEVC**--
instructionList = @[@"Now",@"the",@"tour",@"is",@"working"];
//--Method works but the timer counts down 3 times as fast because of it--
//ESTTableTourChoiceVC *tableTour = [ESTTableTourChoiceVC new];
//[tableTour viewDidLoad];
}
else if ([strClass isEqualToString:@"ESTChangeSettingsVC"]){
//--**NEED TO TRY AND PASS THE CURRENT SPEECH SELECTION
instructionList = @[@"Settings", @"Swipe either left or right to change the rate of speech.", @"Swipe down to play an example sentence with that rate of
page. The current selection is:", @"Slowest speech rate"];
}
else if ([strClass isEqualToString:@"ESTOrientationVC"]){
}
else {
//--**Need to ADD ESTORIENTATIONVC and ESTMAPCHOICE and ESTDISTANCESTORAGE**--
instructionList = @[@"I",@"didn't",@"set",@"up",@"correctly"];
}
for (int i = 0; i <= 4; i++)
{
//--Repeating the instructions on the page when the open is opened again--
self.utterance = [AVSpeechUtterance speechUtteranceWithString:[instructionList objectAtIndex:i]];
if (self.utteranceRate == NULL) {
self.utterance.rate = 0.15;
NSLog(@"Speaking at default rate");
}
else {
//--Setting the rate based on choice from settings--
self.utterance.rate = [self.utteranceRate floatValue];
NSLog(@"Speaking at chosen rate");
//NSLog(@"%f",self.utterance.rate);
}
//--Telling the User what is on the Main Page
[self.syn speakUtterance:self.utterance];
}
}
//--Handling no user action for 30 seconds will get it to repeat instructions--
- (void)repeatMethod:(NSTimer*)theTimer {
//--Counting down the time before previous phrase gets repeated--
repeatTimer--;
NSLog(@"%i",repeatTimer);
//--Checking to see if the device is being properly held--
if ((long)[[UIDevice currentDevice] orientation] == 1 || (long)[[UIDevice currentDevice] orientation] == 5) {
//--Allowing it to speak if held properly--
[self.syn continueSpeaking];
[self.altSyn continueSpeaking];
//NSLog(@"I'm in the proper position");
} else {
//--**Stopping the speach allows the user to hear the swooshing noise better**--
//--Pausing speach if not held properly--
[self.syn pauseSpeakingAtBoundary:WORD_BIT];
[self.altSyn pauseSpeakingAtBoundary:WORD_BIT];


```
    } else {
        //--**Stopping the speach allows the user to hear the swooshing noise better**--
        //--Pausing speach if not held properly--
        [self.syn pauseSpeakingAtBoundary:WORD_BIT];
        [self.altSyn pauseSpeakingAtBoundary:WORD_BIT];

        //--This sound is played if the user is not holding the tablet properly--
        AudioServicesPlaySystemSound(1301);
    }

    //--Once the timer has hit zero it will speak the phrase again and then reset the timer to go again--
    //--**The timer is ended in the handSwipeGesture: method**--
    if (repeatTimer == 0) {
        self.utterance.rate = [self.utteranceRate floatValue];
        [self.syn speakUtterance:self.utterance];
        repeatTimer = 45;
    }
}


//--Function taking you to the main page containing several examples of different things--
-(void)storeBtnTapped:(UIButton *)button
{
    ESTViewController *viewCont = [ESTViewController new];
    [self.navigationController pushViewController:viewCont animated:YES];

    //ESTDistanceStorage *distStore = [ESTDistanceStorage new];
    //[self.navigationController pushViewController:distStore animated:YES];
}

//--Function created for when the Tour Button gets pressed--
//--Need to implement VoiceOver so the user knows what has been pressed--
-(IBAction)tourButton:(id)sender {
    //self.utterance = [[AVSpeechUtterance alloc] initWithString:@"Tour Button"];
    //[self.syn speakUtterance:self.utterance];

    //AudioServicesPlaySystemSound(1031);

    //--** This is what will be used for actuall app **--
    //ESTTourChoiceVC *tourVCont = [ESTTourChoiceVC new];
    ESTTableTourChoiceVC *tourChoiceVC = [ESTTableTourChoiceVC new];
    [self.navigationController pushViewController:tourChoiceVC animated:YES];
    [self vibratePhone];


    //--Practicing geting to and from Settings on iPad using programing--
    //[[UIApplication sharedApplication] openURL:[NSURL URLWithString:UIApplicationOpenSettingsURLString]];
}

//--Handling the single tap gesture to the mapButton--
-(void)mapBtnTapOnce:(UIGestureRecognizer *)gesture {
    //--Setting up utterance to be spoken
    self.utterance = [AVSpeechUtterance speechUtteranceWithString:@"Map Button"];

    NSLog(@"%@",self.utteranceRate);
    if (self.utteranceRate == NULL) {
        self.utterance.rate = AVSpeechUtteranceDefaultSpeechRate;
        NSLog(@"Speaking at default rate");
    }
    else {
        //--Setting the rate based on choice from settings--
        self.utterance.rate = [self.utteranceRate floatValue];
        NSLog(@"Speaking at chosen rate");
        //NSLog(@"%f",self.utterance.rate);
    }

    //[self.utterance isEqual:@"Map Button"];
    [self.syn speakUtterance:self.utterance];
    NSLog(@"I spoke");

    //AudioServicesPlaySystemSound(1033);
}
```


iOS Device
Examples | Build Examples: Succeeded | 12/31/18 at 6:29 PM
Examples
Examples
ESTFirstViewController.m
-viewDidLoad
//--Planning on using this to help the user get from one screen to another without pressing any buttons--
-(void)handleSwipeGesture:(UISwipeGestureRecognizer *)sender
{
//--Stopping the timer when the gesture has been made--
[self.timer invalidate];
//--Determining which direction the swipe has been made so that it can determine which page to go to next--
if (sender.direction == UISwipeGestureRecognizerDirectionLeft) {
//self.utterance = [[AVSpeechUtterance alloc] initWithString:@"It Worked!"];
//[self.syn speakUtterance:self.utterance];
//--Making sure that all speach is stopped before going to the next page--
[self.syn stopSpeakingAtBoundary:WORD_BIT];
[self.altSyn stopSpeakingAtBoundary:WORD_BIT];
ESTTableTourChoiceVC *tableTourVC = [ESTTableTourChoiceVC new];
tableTourVC.passedSpeachRate = self.utteranceRate;
[self.navigationController pushViewController:tableTourVC animated:YES];
}
else if (sender.direction == UISwipeGestureRecognizerDirectionDown){
//--Making sure that all speach is stopped before going to the next page--
[self.syn stopSpeakingAtBoundary:WORD_BIT];
[self.altSyn stopSpeakingAtBoundary:WORD_BIT];
//--Calling to push the ESTChangeSettingsVC to the front--
ESTChangeSettingsVC *settingsVCont = [ESTChangeSettingsVC new];
//--** Adding the delegate to the View Controller to help change the rate of speech**--
settingsVCont.delegate = self;
//--Setting the initial rateOfSpeach value--
//--This way if the Settings Button is hit accidentally it will not reset the utteranceRate--
settingsVCont.rateOfSpeech = self.utteranceRate;
[self.navigationController pushViewController:settingsVCont animated:YES];
//NSLog(@"You didn't do it right");
}
else if (sender.direction == UISwipeGestureRecognizerDirectionRight) {
//--Making sure that all speach is stopped before going to the next page--
[self.syn stopSpeakingAtBoundary:WORD_BIT];
[self.altSyn stopSpeakingAtBoundary:WORD_BIT];
ESTTableMapVC *tableMapVC = [ESTTableMapVC new];
tableMapVC.currentSpeechRate = self.utteranceRate;
[self.navigationController pushViewController:tableMapVC animated:YES];
}
else if (sender.direction == UISwipeGestureRecognizerDirectionUp) {
//--Making sure that all speach is stopped before going to the next page--
//--This allows for the next page to start giving it's instructions since the synthesizer is not occupied--
[self.syn stopSpeakingAtBoundary:WORD_BIT];
[self.altSyn stopSpeakingAtBoundary:WORD_BIT];
ESTOrientationVC *instructionVC = [ESTOrientationVC new];
//--Passing forward the chosen utterance rate--
instructionVC.preferredUtterRate = self.utteranceRate;
[self.navigationController pushViewController:instructionVC animated:YES];
}
else return;
}
-(IBAction)orientationButton:(id)sender {
ESTOrientationVC *orientVCont = [ESTOrientationVC new];
orientVCont.preferredUtterRate = self.utteranceRate;
[self.navigationController pushViewController:orientVCont animated:YES];
[self vibratePhone];
}
- (void)didReceiveMemoryWarning {
[super didReceiveMemoryWarning];


```
    //AudioServicesPlaySystemSound(1033);
}

//--Trying to keep track of the finger on the screen--
//--This would be used to help determine location if VoiceOver is not used--
//--**Also this wouldl be helpful in determining the location the person is at on the map**--

-(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    UITouch *movedTouch = [[event allTouches] anyObject];
    //--Setting the location of the original point of touch and updating it as it moves across the screen--
    CGPoint newLocation = [movedTouch locationInView:self.view];
    CGPoint prevLocation = [movedTouch previousLocationInView:self.view];

    //--Presenting the X and Y locations on the screen
    NSLog(@"X and Y beginning location: %f, %f",prevLocation.x, prevLocation.y);
    NSLog(@"Updated X and Y location: %f, %f",newLocation.x,newLocation.y);

}


//--Handling the double tap gesture to the UIView--
-(void)screenTapTwice:(UIGestureRecognizer *)gesture {
    //--If any instructions ongoing it will stop them and start rpeating them--
    [self.syn stopSpeakingAtBoundary:WORD_BIT];

    //--Needed to create an alternate Speech Synthesizer to make sure that it didn't crash for queing the same utterance twice--
    self.altSyn = [[AVSpeechSynthesizer alloc] init];
    [self.altSyn speakUtterance:self.utterance];

    //--Since they are getting instructions by double tapping again will cause the timer to get reset--
    repeatTimer = 45;

}

//--Function called when the user touches the mapButton--
-(IBAction)mapButton:(id)sender {
    [self.mapButton accessibilityElementDidBecomeFocused];

    ESTMapChoiceVC *mapVCont = [ESTMapChoiceVC new];
    [self.navigationController pushViewController:mapVCont animated:YES];
    [self vibratePhone];

}

-(IBAction)settingsButtonTapped:(id)sender {
    //ESTSettingsVC *settingController = [ESTSettingsVC new];
    //[self.navigationController pushViewController:settingController animated:YES];

    //--Calling to push the ESTChangeSettingsVC to the front --
    ESTChangeSettingsVC *changeSettings = [ESTChangeSettingsVC new];
    //--** Adding the delegate to it **--
    changeSettings.delegate = self;
    //--Setting the initial rateOfSpeach value--
    //--This way if the Settings Button is hit accidentally it will not reset the utteranceRate--
    changeSettings.rateOfSpeech = self.utteranceRate;
    [self.navigationController pushViewController:changeSettings animated:YES];
}

-(void)accessibilityElementDidBecomeFocused {
    //--Setting up GestureRecognizer--
    UISwipeGestureRecognizer *swipeGesture = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];

    swipeGesture.direction = UISwipeGestureRecognizerDirectionDown;
    [self.view addGestureRecognizer:swipeGesture];
}


//--Planning on using this to help the user get from one screen to another without pressing any buttons--
-(void)handleSwipeGesture:(UISwipeGestureRecognizer *)sender
{
    //--Stopping the timer when the gesture has been made--
```

```
    }
    else if (sender.direction == UISwipeGestureRecognizerDirectionUp) {
        //--Making sure that all speach is stopped before going to the next page--
        //--This allows for the next page to start giving it's instructions since the synthesizer is not occupied--
        [self.syn stopSpeakingAtBoundary:WORD_BIT];
        [self.altSyn stopSpeakingAtBoundary:WORD_BIT];

        ESTOrientationVC *instructionVC = [ESTOrientationVC new];
        //--Passing forward the chosen utterance rate--
        instructionVC.preferredUtterRate = self.utteranceRate;
        [self.navigationController pushViewController:instructionVC animated:YES];

    }
    else return;
}


-(IBAction)orientationButton:(id)sender {
    ESTOrientationVC *orientVCont = [ESTOrientationVC new];
    orientVCont.preferredUtterRate = self.utteranceRate;
    [self.navigationController pushViewController:orientVCont animated:YES];
    [self vibratePhone];
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}

-(void)vibratePhone {
    AudioServicesPlaySystemSound(kSystemSoundID_Vibrate);
    //[self addItemViewController: (ESTChangeSettingsVC *) didFinishEnteringItem: (NSString *) ]
}


//--** Method called from the protocol in ESTChangeSettingsVC.h **--
-(void)addItemViewController:(ESTChangeSettingsVC *)controller didFinishEnteringItem:(NSString *)item {
    NSLog(@"This was returned from ESTChangeSettingsVC %@", item);
    self.utteranceRate = item;
    self.resultLabel.text = self.utteranceRate;
    //NSLog(@"%f",[self.utteranceRate floatValue]);
    //float *number;
    //number = (float)[self.utteranceRate floatValue];
}


@end
```

7.2.2. XCODE Map List


```
    UISwipeGestureRecognizer *swipeGestureRight = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    swipeGestureRight.direction = UISwipeGestureRecognizerDirectionRight;
    swipeGestureRight.numberOfTouchesRequired = 1;
    [self.view addGestureRecognizer:swipeGestureRight];

    //--Looking for swipe in the Down direction--
    UISwipeGestureRecognizer *swipeGestureDown = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    swipeGestureDown.direction = UISwipeGestureRecognizerDirectionDown;
    swipeGestureDown.numberOfTouchesRequired = 1;
    [self.view addGestureRecognizer:swipeGestureDown];

    //--Adding the left, right, up, and down image arrows--
    self.leftArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 230, 80, 40)];
    self.leftArrow = [UIImage imageNamed:@"bold_left_arrow.jpg"];
    self.leftArrowHolder.image = self.leftArrow;
    [self.view addSubview:self.leftArrowHolder];

    self.rightArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(240, 230, 80, 40)];
    self.rightArrow = [UIImage imageNamed:@"right_bold_arrow.jpg"];
    self.rightArrowHolder.image = self.rightArrow;
    [self.view addSubview:self.rightArrowHolder];

    self.upArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(140, 65, 40, 80)];
    self.upArrow = [UIImage imageNamed:@"up_bold_arrow.jpg"];
    self.upArrowHolder.image = self.upArrow;
    [self.view addSubview:self.upArrowHolder];

    self.downArrowHolder = [[UIImageView alloc] initWithFrame:CGRectMake(140, 400, 40, 80)];
    self.downArrow = [UIImage imageNamed:@"down_bold_arrow.jpg"];
    self.downArrowHolder.image = self.downArrow;
    [self.view addSubview:self.downArrowHolder];

    //--Setting up the Speech Synthesizer as well as the Speech Utterance so they can be used--
    self.syn = [[AVSpeechSynthesizer alloc] init];
    self.utterance = [[AVSpeechUtterance alloc] init];

    //--Letting the user know they are on the 'Map Selection' View Controller and setting the speech rate for the page--
    self.utterance = [AVSpeechUtterance speechUtteranceWithString:self.title];
    self.utterance.rate = [self.currentSpeechRate floatValue];
    //[self.syn speakUtterance:self.utterance];

    //--Letting the user know what the current selected Map is when the View Controller is loaded--
    self.utterance = [AVSpeechUtterance speechUtteranceWithString:[self.mapSectionList objectAtIndex:i]];
    self.utterance.rate = [self.currentSpeechRate floatValue];
    //[self.syn speakUtterance:self.utterance];


    //--Setting up tap gesture recognizer--
    UITapGestureRecognizer *tapTwice = [[UITapGestureRecognizer alloc] initWithTarget:self action:@selector(screenTapTwice:)];
    //--Noting that it requires 2 taps to activate--
    tapTwice.numberOfTapsRequired = 2;
    //--Adding the tap gesture recognizer to the UIView so they can get it to repeat instructions--
    [self.view addGestureRecognizer:tapTwice];


    //--Setting up the timer so that it will repeat itself if there is no activity--
    repeatTimer = 45;
    self.timer = [[NSTimer alloc] init];
    self.timer = [NSTimer scheduledTimerWithTimeInterval:1.0 target:self selector:@selector(repeatMethod:) userInfo:nil repeats:true];

    //[[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(openAgain2) name:UIApplicationDidBecomeActiveNotification
        object:nil];
}

//--Function called when the 'testingStyle' Button is pressed--
-(void)testingStyleBtnTapped:(UIButton *)button {
    //--Keeping Track of the Testing Style Button--
    //--If in the '1' State one finger Clarinet will play to Left Ear--
    //--If in the '2' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Left Ear (right
        finger)--
    //--If in the '3' State two fingers the Clarinet will play to the Left Ear (left finger) and the Clarinet will be played to Right Ear (right
        finger)--
    //--If in the '4' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Right Ear (right
```

```
    //--If in the '2' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Left Ear (right
        finger)--
    //--If in the '3' State two fingers the Clarinet will play to the Left Ear (left finger) and the Clarinet will be played to Right Ear (right
        finger)--
    //--If in the '4' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Right Ear (right
        finger)--
    if (testSelection != [self.testStyleList count] - 1) {
        testSelection++;
        //self.mapSelection.adjustsFontSizeToFitWidth = YES;
        //self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        self.testingStyleButton.title = [self.testStyleList objectAtIndex:testSelection];
    }
    else {
        testSelection = 0;
        //self.mapSelection.adjustsFontSizeToFitWidth = YES;
        //self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        self.testingStyleButton.title = [self.testStyleList objectAtIndex:testSelection];
    }

    NSLog(@"The Title is %@:", self.testingStyleButton.title);

}


- (void)repeatMethod:(NSTimer*)theTimer {
    //--Counting down the time before previous phrase gets repeated--
    repeatTimer = repeatTimer - 1;
    NSLog(@"%i",repeatTimer);

    //--Checking to see if the device is being properly held--
    if ((long)[[UIDevice currentDevice] orientation] == 1 || (long)[[UIDevice currentDevice] orientation] == 5) {
        //--Resuming/Continuing speech if held in the proper orientation--
        [self.syn continueSpeaking];
        [self.altSyn continueSpeaking];
        //NSLog(@"I'm in the proper position");
    } else {
        //--Pausing the speech at the end of the word as soon as it determines it's in the wrong orientation--
        [self.syn pauseSpeakingAtBoundary:WORD_BIT];
        [self.altSyn pauseSpeakingAtBoundary:WORD_BIT];

        //--This sound is played if the user is not holding the tablet properly--
        AudioServicesPlaySystemSound(1301);
    }

    //--Once the timer has hit zero it will speak the phrase again and then reset the timer to go again--
    //--**The timer is ended in the handSwipeGesture: method**--
    /*if (repeatTimer == 0) {
        self.utterance.rate = [self.currentSpeechRate floatValue];
        [self.syn speakUtterance:self.utterance];
        repeatTimer = 45;
    }*/
}

- (void)screenTapTwice:(UIGestureRecognizer*)gesture {
    //--If any instructions ongoing it will stop them and start rpeating them--
    [self.syn stopSpeakingAtBoundary:WORD_BIT];

    //--Needed to create an alternate Speech Synthesizer to make sure that it didn't crash for queing the same utterance twice--
    self.altSyn = [[AVSpeechSynthesizer alloc] init];
    self.utterance.rate = [self.currentSpeechRate floatValue];
    [self.altSyn speakUtterance:self.utterance];

    //--Since they are getting instructions by double tapping again will cause the timer to get reset--
    repeatTimer = 45;
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}


 (void)handleSwipeGesture:(UISwipeGestureRecognizer *)sender
```

```
-(void)handleSwipeGesture:(UISwipeGestureRecognizer *)sender
{
    //--Stopping the ongoing speech--
    [self.syn stopSpeakingAtBoundary:WORD_BIT];

    //--Stopping the alternate speech as well--
    [self.altSyn stopSpeakingAtBoundary:WORD_BIT];

    if (sender.direction == UISwipeGestureRecognizerDirectionDown){

        //--Ending the timer--
        [self.timer invalidate];

        //--Down direction will decide that they accept the current option--
        switch (i) {
            case 0: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Test Map";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"Trial6.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;
            }
            case 1: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Practice Map";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"practice map Resized.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;
            }
            case 2: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Practice Overall Map";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"practice_overall_map.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;
            }
            case 3: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Overall Map A";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"The Overall Map A.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;
            }
            case 4: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Overall Map B";
```


```
            mapChoice.title = @"Overall Map B";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"The Overall Map B.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 5: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Overall Map C";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"The Overall Map C.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 6: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Overall Map D";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"The Overall Map D.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 7: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Overall Map E";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"The Overall Map E.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 8: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Overall Map F";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Overall Map F Update.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 9: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Overall Map G";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Overall Map G.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
```

```
        case 10: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Entrance Garden";
            mapChoice.testStyleNumber = testSelection;
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Entrance Resized.png"];
            //mapChoice.image = [UIImage imageNamed:@"Map the third.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 11: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Four Seasons Garden";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Bragdon_4Seasons Resized-attempt.png"];
            //mapChoice.image = [UIImage imageNamed:@"Bragdon_4Seasons updated.png"];
            //mapChoice.image = [UIImage imageNamed:@"Map the fourth.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 12: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Physical Spiritual Garden";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Physical_Spiritual Resized_attempt.png"];
            //mapChoice.image = [UIImage imageNamed:@"Physical_Spiritual tags 6 Color.png"];
            //mapChoice.image = [UIImage imageNamed:@"replacement of colors.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 13: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Education/Library Building";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"GilletteN Resized tablet 2.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 14: {
            ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
            mapChoice.mapChoiceNumber = i;
            mapChoice.testStyleNumber = testSelection;
            mapChoice.actualTestNumber = self.testingStyleButton.title;
            mapChoice.title = @"Fountain Garden";
            mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
            mapChoice.image = [UIImage imageNamed:@"Fountain Resized tablet.png"];
            //mapChoice.image = [UIImage imageNamed:@"Map the second.png"];
            mapChoice.imageHolder.image = mapChoice.image;
            [mapChoice.view addSubview:mapChoice.imageHolder];
            [self.navigationController pushViewController:mapChoice animated:YES];
            break;
        }
        case 15: {
```

```
            case 15: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Greenhouse Garden";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"Greenhouse Garden tablet.png"];
                //mapChoice.image = [UIImage imageNamed:@"Map the second.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;
            }
            case 16: {
                ESTMapChoiceVC *mapChoice = [ESTMapChoiceVC new];
                mapChoice.mapChoiceNumber = i;
                mapChoice.testStyleNumber = testSelection;
                mapChoice.actualTestNumber = self.testingStyleButton.title;
                mapChoice.title = @"Storage Garden";
                mapChoice.imageHolder = [[UIImageView alloc] initWithFrame:CGRectMake(0, 60, 320, 430)];
                mapChoice.image = [UIImage imageNamed:@"Storage Garden.png"];
                //mapChoice.image = [UIImage imageNamed:@"Map the second.png"];
                mapChoice.imageHolder.image = mapChoice.image;
                [mapChoice.view addSubview:mapChoice.imageHolder];
                [self.navigationController pushViewController:mapChoice animated:YES];
                break;            }
            default:
                break;
        }

    }
    else if (sender.direction == UISwipeGestureRecognizerDirectionLeft) {
        //--Using the if else statements to determine what the next value of 'i' should be'--
        //--Also making sure that the value of 'i' is never greater than the array 'tourSelectionList'--
        if (i != [self.mapSectionList count] - 1) {
            i++;
            self.mapSelection.adjustsFontSizeToFitWidth = YES;
            self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        }
        else {
            i = 0;
            self.mapSelection.adjustsFontSizeToFitWidth = YES;
            self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        }

        //--**Had been using the switch function but this is much easier than the switch function--
        //--If speech synthesizer is already being used this will immediately stop it--
        [self.syn stopSpeakingAtBoundary:WORD_BIT];

        //--Then it will begin to tell the user what their current selection is--
        self.utterance = [AVSpeechUtterance speechUtteranceWithString:[self.mapSectionList objectAtIndex:i]];
        self.utterance.rate = [self.currentSpeechRate floatValue];
        //[self.syn speakUtterance:self.utterance];

        //--Resetting the timer--
        repeatTimer = 45;

    }

    else if (sender.direction == UISwipeGestureRecognizerDirectionRight) {
        //--Using the if else statements to determine what the next value of 'i' should be'--
        //--Also making sure that the value of 'i' is never negative--
        if (i == 0) {
            i = (int)[self.mapSectionList count] - 1;
            self.mapSelection.adjustsFontSizeToFitWidth = YES;
            self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        }
        else {
            i--;
            self.mapSelection.adjustsFontSizeToFitWidth = YES;
            self.mapSelection.text = [self.mapSectionList objectAtIndex:i];
        }
```

```
        //--If speech synthesizer is already talking this will immediately stop it--
        [self.syn stopSpeakingAtBoundary:WORD_BIT];

        //--Then it will begin to tell the user their current selection--
        self.utterance = [AVSpeechUtterance speechUtteranceWithString:[self.mapSectionList objectAtIndex:i]];
        self.utterance.rate = [self.currentSpeechRate floatValue];
        //[self.syn speakUtterance:self.utterance];

        //--Resetting the timer--
        repeatTimer = 45;


    }
    else if (sender.direction == UISwipeGestureRecognizerDirectionUp) {
        //--Up direction will decide that they want to return to the previous screen--

        //--Making sure that the speach is stopped before going to previous page--
        [self.syn stopSpeakingAtBoundary:WORD_BIT];

        //--Ending the timer--
        [self.timer invalidate];

        //--Taking the user back to the main page--
        //--Also passing the speach rate selected back to the main page--
        ESTFirstViewController *firstVC = [ESTFirstViewController new];
        firstVC.utteranceRate = self.currentSpeechRate;
        [self.navigationController pushViewController:firstVC animated:YES];

        //--Setting the 'i" counter to zero so that when another page is selected it will not be larger than the array count--
        i = 0;
    }
    else return;
}

@end
```

7.2.3. XCODE Map Selected

ESTMapChoiceVC.m

ESTMapChoiceVC.m 〉 No Selection

```
//
//  ESTMapChoiceVC.m
//  Examples
//
//  Created by David Parker on 10/5/15.
//  Copyright (c) 2015 com.estimote. All rights reserved.
//

#import "ESTMapChoiceVC.h"
#import "ESTTableMapVC.h"
#import <AVFoundation/AVFoundation.h>
#import <AudioToolbox/AudioToolbox.h>

#import <QuartzCore/QuartzCore.h>

@interface ESTMapChoiceVC ()
{
    //--Setting up the AVAudioPlayers so the musical notes can be played to each ear--
    AVAudioPlayer *audioPlayer;
    AVAudioPlayer *audioPlayer2;

    //--Storing the state of the current color for each finger, used to determine audioPlayer sound should be played--
    int firstColorCompared;
    int secondColorCompared;

    //--Storing the last matched state for each finger of the colors that are trying to be found, used to determine which utterance should be spoken--
    int actualColor, actualColor2;
    //--Storing the x and y pixel location for each finger--
    CGPoint location1, location2;
}

@property (nonatomic, strong) UIBarButtonItem *onOffButton;
@property (nonatomic, strong) UIBarButtonItem *testSelectionItem;
@property (retain, nonatomic) AVSpeechSynthesizer *syn;
@property (nonatomic, strong) AVSpeechUtterance *utterance;
@property (nonatomic, strong) AVSpeechUtterance *utterance2, *utterance3;
@property (nonatomic, strong) UITouch *touch1;
@property (nonatomic, strong) UITouch *touch2;
@property (nonatomic, strong) NSMutableArray *leftRightFinger;
@property (nonatomic, strong) NSMutableArray *touchesSet;
@property (nonatomic, strong) NSString *thePickedColor;
@property (nonatomic, strong) NSString *theSecondPickedColor;
@property (nonatomic, strong) NSString *firstComparedColor;
@property (nonatomic, strong) NSString *secondComparedColor;
@property (nonatomic, strong) NSTimer *timer;

@end


@implementation ESTMapChoiceVC

int repeatTimer;

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view from its nib.

    //--Adding a UIBarButton to the tab in ESTMapChoice--

    //--Setting the testing style selection in the top right of the screen--
    //--Determines which testing style will be used when the finger(s) touch the screen--
    //--If in the '1' State one finger Clarinet will play to Left Ear--
    //--If in the '2' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Left Ear (right finger)--
    //--If in the '3' State two fingers the Clarinet will play to the Left Ear (left finger) and the Clarinet will be played to Right Ear (right finger)--
    //--If in the '4' State two fingers the Clarinet will play to the Left Ear (left finger) and the Guitar will play to the Right Ear (right finger)--
    self.testSelectionItem = [[UIBarButtonItem alloc] initWithTitle:self.actualTestNumber style:UIBarButtonItemStylePlain target:self action:nil];
    self.navigationItem.rightBarButtonItem = self.testSelectionItem;

    //--Setting up the speech synthesizer so I can use text to speech--
    self.syn = [[AVSpeechSynthesizer alloc] init];
    self.utterance = [[AVSpeechUtterance alloc] init];

    //--The Value of I From The Switch Statement in ESTTableMapVC--
    //--mapChoiceNumber is used to determine if it is an Overall Map or an Individual Map--
```

```
    //--The Value of I From The Switch Statement in ESTTableMapVC--
    //--mapChoiceNumber is used to determine if it is an Overall Map or an Individual Map--
    NSLog(@"The I Number is: %i",self.mapChoiceNumber);
    //NSLog(@"The Test Selection is: %testSelection", self.testStyleNumber);
    //NSLog(@"The # is: %i",self.testStyleNumber);
    NSLog(@"The Actual Test Number is: %@", self.actualTestNumber);

    //--Initializing to a non-recognized color to prevent the wrong utterance from being spoken--
    actualColor = 7;
    actualColor2 = 7;

}

//--Trying to keep track of the finger on the screen--
//--This would be used to help determine location if VoiceOver is not used--
//--**Also this would be helpful in determining the location the person is at on the map**--
-(void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    //--Getting how many touches there are and then determining their pixel location of the screen--
    NSUInteger touchCount = [touches count];
    UITouch *movedTouch = [[event allTouches] anyObject];
    //--Setting the location of the original point of touch and updating it as it moves across the screen--
    CGPoint newLocation = [movedTouch locationInView:self.view];
    //NSLog(@"x=%.2f y=%.2f",newLocation.x,newLocation.y);
    CGPoint prevLocation = [movedTouch previousLocationInView:self.view];


    //--Setting the state of the first and second finger--
    int firstFinger, secondFinger;

    //--Finding out the # of touches and then setting the first touch as the first object in the NSSet allTouches--
    NSSet *allTouches = [event allTouches];
    self.touch1 = [[allTouches allObjects] objectAtIndex:0];
    location1 = [self.touch1 locationInView:self.view];

    //--Checking to see if there are more than 1 touches--
    if ([allTouches count] > 1) {
        //--If there is a second touch setting the second touch as the second object in the NSSet allTouches--
        self.touch2 = [[allTouches allObjects] objectAtIndex:1];
        location2 = [self.touch2 locationInView:self.view];
    }
    else {
        //--If not then setting the second touch as zero to prevent from getting in the way--
        location2.x = 0.0;
        location2.y = 0.0;
    }

    float zero = 0.0;
    //--Setting up a mutable array to help keep track of left and right finger and also so that the two objects can be easily switched if the two fingers pass
        each other on the scree--
    self.leftRightFinger = [NSMutableArray arrayWithObjects:0, 0, nil];

    //--Determining if there are Two Touches and which one is the left and right finger--
    if (location2.x != zero && location2.y != zero) {
        if (location2.x - location1.x < 0.0) {
            //--If the 2nd finger x pixel location is less than that off the 1st finger's x pixel location then that means the 2nd finger is the left most finger
                and will be the first object in the mutable array--
            self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch2, self.touch1, nil];
        }
        else {
            //--Everything else touch1 is the first object and touch2 is the second object--
            self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch1, self.touch2, nil];
        }

        //--Getting the color of the pixel of the left and right finger--
        //--Converting the Hex Color to an NSString so that it can be compared in IF ELSE statements--
        UIColor *pickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:0] locationInView:self.view]];
        NSString *firstPickedColor = [self hexFromUIColor:pickedColor];
        //self.thePickedColor = [self hexFromUIColor:pickedColor];
        UIColor *secondPickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:1] locationInView:self.view]];
        NSString *anotherPickedColor = [self hexFromUIColor:secondPickedColor];
        //self.theSecondPickedColor = [self hexFromUIColor:secondPickedColor];

        //--Logging the Colors of the Left and Right Finger--
        NSLog(@"Left Finger Color %@", [self hexFromUIColor:pickedColor]);
```

```
//--Logging the Colors of the Left and Right Finger--
NSLog(@"Left Finger Color %@",[self hexFromUIColor:pickedColor]);
NSLog(@"Right Finger Color %@", [self hexFromUIColor:secondPickedColor]);


//--Determining which state the firstFinger is in--
if ([firstPickedColor isEqual:@"#FFFF03"]) {
    //--Yellow Color--
    firstFinger = 0;
}
else if ([firstPickedColor isEqual:@"#038003"]){
    //--Green Color--
    firstFinger = 1;
}
else if ([firstPickedColor isEqual:@"#999999"]){
    //--Gray Color--
    firstFinger = 2;
}
else if ([firstPickedColor isEqual:@"#0303FF"] || [firstPickedColor isEqual:@"030380"]){
    //--Point of Interest A--
    firstFinger = 3;
}
else if ([firstPickedColor isEqual:@"#8888FF"]){
    //--Point of Interest B--
    firstFinger = 7;
}
else if ([firstPickedColor isEqual:@"#A3BDE4"]){
    //--Point of Interest C--
    firstFinger = 8;
}
else if ([firstPickedColor isEqual:@"#7D5CE0"]){
    //--Point of Interest D--
    firstFinger = 9;
}
else if ([firstPickedColor isEqual:@"#0303A0"]){
    //--Point of Interest E--
    firstFinger = 10;
}
else if ([firstPickedColor isEqual:@"#03FFFF"]){
    //--Point of Interest F--
    firstFinger = 11;
}
else if ([firstPickedColor isEqual:@"#400380"]){
    //--Point of Interest G--
    firstFinger = 12;
}
else if ([firstPickedColor isEqual:@"#A349A4"]){
    //--Point of Interest H--
    firstFinger = 13;
}
else if ([firstPickedColor isEqual:@"#C8BFE7"]){
    //--Point of Interest I--
    firstFinger = 14;
}
else if ([firstPickedColor isEqual:@"#7092BE"]){
    //--Point of Interest J--
    firstFinger = 15;
}
else if ([firstPickedColor isEqual:@"#400340"]){
    //--Point of Interest K--
    firstFinger = 16;
}
else if ([firstPickedColor isEqual:@"#8003FF"]){
    //--Point of Interest L--
    firstFinger = 17;
}
else if ([firstPickedColor isEqual:@"#03A2E8"]){
    //--Fountain Color--
    firstFinger = 18;
}
else if ([firstPickedColor isEqual:@"#030303"]){
    //--Black Color--
    firstFinger = 4;
}
```

```
        firstFinger = 4;
    }
    else if ([firstPickedColor isEqual:@"#FF0303"]){
        //--Red Color--
        firstFinger = 5;
    }
    else if ([firstPickedColor isEqual:@"#FFFFFF"]){
        //--White Color--
        firstFinger = 6;
    }
    else if ([firstPickedColor isEqual:@"#F5F5F5"]){
        //--Art White Color--
        firstFinger = 19;
    }
    else if ([firstPickedColor isEqual:@"#FAFAFA"]){
        //--Workshop White Color--
        firstFinger = 20;
    }
    else if ([firstPickedColor isEqual:@"#F0F0F0"]){
        //--Plant Nursery White Color--
        firstFinger = 21;
    }
    else if ([firstPickedColor isEqual:@"#E6E6E6"]){
        //--Library White Color--
        firstFinger = 22;
    }
    else if ([firstPickedColor isEqual:@"#EBEBEB"]){
        //--Education White Color--
        firstFinger = 23;
    }
    else {
        firstFinger = 24;
    }

    //--Determining which state the secondFinger is in--
    if ([anotherPickedColor isEqual:@"#FFFF03"]) {
        //--Yellow Color--
        secondFinger = 0;
    }
    else if ([anotherPickedColor isEqual:@"#038003"]){
        //--Green Color--
        secondFinger = 1;
    }
    else if ([anotherPickedColor isEqual:@"#999999"]){
        //--Gray Color--
        secondFinger = 2;
    }
    else if ([anotherPickedColor isEqual:@"#0303FF"] || [anotherPickedColor isEqual:@"#030380"]){
        //--Blue Color--
        secondFinger = 3;
    }
    else if ([anotherPickedColor isEqual:@"#8888FF"]){
        //--Point of Interest B--
        secondFinger = 7;
    }
    else if ([anotherPickedColor isEqual:@"#A3BDE4"]){
        //--Point of Interest C--
        secondFinger = 8;
    }
    else if ([anotherPickedColor isEqual:@"#7D5CE0"]){
        //--Point of Interest D--
        secondFinger = 9;
    }
    else if ([anotherPickedColor isEqual:@"#0303A0"]){
        //--Point of Interest E--
        secondFinger = 10;
    }
    else if ([anotherPickedColor isEqual:@"#03FFFF"]){
        //--Point of Interest F--
        secondFinger = 11;
    }
    else if ([anotherPickedColor isEqual:@"#400380"]){
        //--Point of Interest G--
        secondFinger = 12;
```

```
            secondFinger = 12;
        }
        else if ([anotherPickedColor isEqual:@"#A349A4"]){
            //--Point of Interest H--
            secondFinger = 13;
        }
        else if ([anotherPickedColor isEqual:@"#C8BFE7"]){
            //--Point of Interest I--
            secondFinger = 14;
        }
        else if ([anotherPickedColor isEqual:@"#7092BE"]){
            //--Point of Interest J--
            secondFinger = 15;
        }
        else if ([anotherPickedColor isEqual:@"#400340"]){
            //--Point of Interest K--
            secondFinger = 16;
        }
        else if ([anotherPickedColor isEqual:@"#8003FF"]){
            //--Point of Interest L--
            secondFinger = 17;
        }
        else if ([anotherPickedColor isEqual:@"#03A2E8"]){
            //--Fountain Color--
            secondFinger = 18;
        }
        else if ([anotherPickedColor isEqual:@"#030303"]){
            //--Black Color--
            secondFinger = 4;
        }
        else if ([anotherPickedColor isEqual:@"#FF0303"]){
            //--Red Color--
            secondFinger = 5;
        }
        else if ([anotherPickedColor isEqual:@"#FFFFFF"]){
            //--White Color--
            secondFinger = 6;
        }
        else if ([anotherPickedColor isEqual:@"#F5F5F5"]){
            //--Art White Color--
            secondFinger = 19;
        }
        else if ([anotherPickedColor isEqual:@"#FAFAFA"]){
            //--Workshop White Color--
            secondFinger = 20;
        }
        else if ([anotherPickedColor isEqual:@"#F0F0F0"]){
            //--Plant Nursery White Color--
            secondFinger = 21;
        }
        else if ([anotherPickedColor isEqual:@"#E6E6E6"]){
            //--Library White Color--
            secondFinger = 22;
        }
        else if ([anotherPickedColor isEqual:@"#EBEBEB"]){
            //--Education White Color--
            secondFinger = 23;
        }
        else {
            secondFinger = 24;
        }

        //--Calling the colorChanged Function to see if any of the colors have changed or not--
        //--Determining which color has been selected by firstFinger and secondFinger and passing that value as an integer to the function colorChanged:
            secondColor: --
        [self colorChanged:firstFinger secondColor:secondFinger];

    }

    if (location2.x == zero && location2.y == zero) {
        self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch1, nil];

        //--Finding the Color that the finger is currently over and then converting it into Hexidecimal so that it can be compared--
        UIColor *pickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:0] locationInView:self.view]];
```

```
UIColor *pickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:0] locationInView:self.view]];
NSString *singleFingerPickedColor = [self hexFromUIColor:pickedColor];
//NSLog(@"%@",thePickedColor);

//--Determining which state the firstFinger is in--
if ([singleFingerPickedColor isEqual:@"#FFFF03"]) {
    //--Yellow Color--
    firstFinger = 0;
}
else if ([singleFingerPickedColor isEqual:@"#038003"]){
    //--Green Color--
    firstFinger = 1;
}
else if ([singleFingerPickedColor isEqual:@"#999999"]){
    //--Gray Color--
    firstFinger = 2;
}
else if ([singleFingerPickedColor isEqual:@"#0303FF"] || [singleFingerPickedColor isEqual:@"#030380"]){
    //--Blue Color--
    firstFinger = 3;
}
else if ([singleFingerPickedColor isEqual:@"#8888FF"]){
    //--Point of Interest B--
    firstFinger = 7;
}
else if ([singleFingerPickedColor isEqual:@"#A3BDE4"]){
    //--Point of Interest C--
    firstFinger = 8;
}
else if ([singleFingerPickedColor isEqual:@"#7D5CE0"]){
    //--Point of Interest D--
    firstFinger = 9;
}
else if ([singleFingerPickedColor isEqual:@"#0303A0"]){
    //--Point of Interest E--
    firstFinger = 10;
}
else if ([singleFingerPickedColor isEqual:@"#03FFFF"]){
    //--Point of Interest F--
    firstFinger = 11;
}
else if ([singleFingerPickedColor isEqual:@"#400380"]){
    //--Point of Interest G--
    firstFinger = 12;
}
else if ([singleFingerPickedColor isEqual:@"#A349A4"]){
    //--Point of Interest H--
    firstFinger = 13;
}
else if ([singleFingerPickedColor isEqual:@"#C8BFE7"]){
    //--Point of Interest I--
    firstFinger = 14;
}
else if ([singleFingerPickedColor isEqual:@"#7092BE"]){
    //--Point of Interest J--
    firstFinger = 15;
}
else if ([singleFingerPickedColor isEqual:@"#400340"]){
    //--Point of Interest K--
    firstFinger = 16;
}
else if ([singleFingerPickedColor isEqual:@"#8003FF"]){
    //--Point of Interest L--
    firstFinger = 17;
}
else if ([singleFingerPickedColor isEqual:@"#03A2E8"]){
    //--Fountain Color--
    firstFinger = 18;
}
else if ([singleFingerPickedColor isEqual:@"#030303"]){
    //--Black Color--
    firstFinger = 4;
}
else if ([singleFingerPickedColor isEqual:@"#FF0303"]){
```

```
        }
        else if ([singleFingerPickedColor isEqual:@"#FF0303"]){
            //--Red Color--
            firstFinger = 5;
        }
        else if ([singleFingerPickedColor isEqual:@"#FFFFFF"]){
            //--White Color--
            firstFinger = 6;
        }
        else if ([singleFingerPickedColor isEqual:@"#F5F5F5"]){
            //--Art White Color--
            firstFinger = 19;
        }
        else if ([singleFingerPickedColor isEqual:@"#FAFAFA"]){
            //--Workshop White Color--
            firstFinger = 20;
        }
        else if ([singleFingerPickedColor isEqual:@"#F0F0F0"]){
            //--Plant Nursery White Color--
            firstFinger = 21;
        }
        else if ([singleFingerPickedColor isEqual:@"#E6E6E6"]){
            //--Library White Color--
            firstFinger = 22;
        }
        else if ([singleFingerPickedColor isEqual:@"#EBEBEB"]){
            //--Education White Color--
            firstFinger = 23;
        }
        else {
            firstFinger = 24;
        }

        //--Calling the singleFingerColor Function to see if the color has changed or not--
        [self singleFingerColor:firstFinger];

    }

}

-(void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {

    //--Setting the state of the first and second finger--
    int firstFinger, secondFinger;
    float zero = 0.0;

    NSUInteger touchCount = [touches count];

    NSSet *allTouches = [event allTouches];
    self.touch1 = [[allTouches allObjects] objectAtIndex:0];
    //NSLog(@"Touch 1 is :%@",self.touch1);
    location1 = [self.touch1 locationInView:self.view];

    if ([allTouches count] > 1) {
        self.touch2 = [[allTouches allObjects] objectAtIndex:1];
        //NSLog(@"Touch 2 is :%@",self.touch2);
        location2 = [self.touch2 locationInView:self.view];
    }


    //--Determining if there are Two Touches and which one is the left and right finger--
    if (location2.x != zero && location2.y != zero) {
        if (location2.x - location1.x < 0.0) {
            //--If the 2nd finger x pixel location is less than that off the 1st finger's x pixel location then that means the 2nd finger is the left most finger
                and will be the first object in the mutable array--
            self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch2, self.touch1, nil];
        }
        else {
            //--Everything else touch1 is the first object and touch2 is the second object--
            self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch1, self.touch2, nil];
        }

        //--Getting the color of the pixel of the left and right finger--
        //--Converting the Hex Color to an NSString so that it can be compared in IF ELSE statements--
```

```
//--Converting the Hex Color to an NSString so that it can be compared in IF ELSE statements--
UIColor *pickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:0] locationInView:self.view]];
NSString *firstPickedColor = [self hexFromUIColor:pickedColor];
//self.thePickedColor = [self hexFromUIColor:pickedColor];
UIColor *secondPickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:1] locationInView:self.view]];
NSString *anotherPickedColor = [self hexFromUIColor:secondPickedColor];
//self.theSecondPickedColor = [self hexFromUIColor:secondPickedColor];

//--Determining which state the firstFinger is in--
if ([firstPickedColor isEqual:@"#FFFF03"] || [firstPickedColor isEqual:@"FFFF00"]) {
    //--Yellow Color--
    firstFinger = 0;
}
else if ([firstPickedColor isEqual:@"#038003"] || [firstPickedColor isEqual:@"#008000"]){
    //--Green Color--
    firstFinger = 1;
}
else if ([firstPickedColor isEqual:@"#999999"]){
    //--Gray Color--
    firstFinger = 2;
}
else if ([firstPickedColor isEqual:@"#0303FF"] || [firstPickedColor isEqual:@"#030380"]){
    //--Blue Color--
    firstFinger = 3;
}
else if ([firstPickedColor isEqual:@"#8888FF"]){
    //--Point of Interest B--
    firstFinger = 7;
}
else if ([firstPickedColor isEqual:@"#A3BDE4"]){
    //--Point of Interest C--
    firstFinger = 8;
}
else if ([firstPickedColor isEqual:@"#7D5CE0"]){
    //--Point of Interest D--
    firstFinger = 9;
}
else if ([firstPickedColor isEqual:@"#0303A0"] || [firstPickedColor isEqual:@"#0000A0"]){
    //--Point of Interest E--
    firstFinger = 10;
}
else if ([firstPickedColor isEqual:@"#03FFFF"] || [firstPickedColor isEqual:@"00FFFF"]){
    //--Point of Interest F--
    firstFinger = 11;
}
else if ([firstPickedColor isEqual:@"#400380"] || [firstPickedColor isEqual:@"400080"]){
    //--Point of Interest F--
    firstFinger = 12;
}
else if ([firstPickedColor isEqual:@"#A349A4"]){
    //--Point of Interest F--
    firstFinger = 13;
}
else if ([firstPickedColor isEqual:@"#C8BFE7"]){
    //--Point of Interest F--
    firstFinger = 14;
}
else if ([firstPickedColor isEqual:@"#7092BE"]){
    //--Point of Interest F--
    firstFinger = 15;
}
else if ([firstPickedColor isEqual:@"#400340"]){
    //--Point of Interest F--
    firstFinger = 16;
}
else if ([firstPickedColor isEqual:@"#8003FF"]){
    //--Point of Interest F--
    firstFinger = 17;
}
else if ([firstPickedColor isEqual:@"#03A2E8"] || [firstPickedColor isEqual:@"#00A2E8"]){
    //--Fountain Color--
    firstFinger = 18;
}
else if ([firstPickedColor isEqual:@"#030303"]){
```

```
}
else if ([firstPickedColor isEqual:@"#030303"]){
    //--Black Color--
    firstFinger = 4;
}
else if ([firstPickedColor isEqual:@"#FF0303"] || [firstPickedColor isEqual:@"#FF0000"]){
    //--Red Color--
    firstFinger = 5;
}
else if ([firstPickedColor isEqual:@"#FFFFFF"]){
    //--White Color--
    firstFinger = 6;
}
else if ([firstPickedColor isEqual:@"#F5F5F5"]){
    //--Art White Color--
    firstFinger = 19;
}
else if ([firstPickedColor isEqual:@"#FAFAFA"]){
    //--Workshop White Color--
    firstFinger = 20;
}
else if ([firstPickedColor isEqual:@"#F0F0F0"]){
    //--Plant Nursery White Color--
    firstFinger = 21;
}
else if ([firstPickedColor isEqual:@"#E6E6E6"]){
    //--Library White Color--
    firstFinger = 22;
}
else if ([firstPickedColor isEqual:@"#EBEBEB"]){
    //--Education White Color--
    firstFinger = 23;
}
else {
    firstFinger = 24;
}


//--Determining which state the secondFinger is in--
if ([anotherPickedColor isEqual:@"#FFFF03"]) {
    //--Yellow Color--
    secondFinger = 0;
}
else if ([anotherPickedColor isEqual:@"#038003"]){
    //--Green Color--
    secondFinger = 1;
}
else if ([anotherPickedColor isEqual:@"#999999"]){
    //--Gray Color--
    secondFinger = 2;
}
else if ([anotherPickedColor isEqual:@"#0303FF"] || [anotherPickedColor isEqual:@"#030380"]){
    //--Blue Color--
    secondFinger = 3;
}
else if ([anotherPickedColor isEqual:@"#8888FF"]){
    //--Point of Interest B--
    secondFinger = 7;
}
else if ([anotherPickedColor isEqual:@"#A3BDE4"]){
    //--Point of Interest C--
    secondFinger = 8;
}
else if ([anotherPickedColor isEqual:@"#7D5CE0"]){
    //--Point of Interest D--
    secondFinger = 9;
}
else if ([anotherPickedColor isEqual:@"#0303A0"]){
    //--Point of Interest E--
    secondFinger = 10;
}
else if ([anotherPickedColor isEqual:@"#03FFFF"]){
    //--Point of Interest F--
    secondFinger = 11;
```

```
        else if ([anotherPickedColor isEqual:@"#400380"]){
            //--Point of Interest F--
            secondFinger = 12;
        }
        else if ([firstPickedColor isEqual:@"#A349A4"]){
            //--Point of Interest F--
            secondFinger = 13;
        }
        else if ([anotherPickedColor isEqual:@"#C8BFE7"]){
            //--Point of Interest F--
            secondFinger = 14;
        }
        else if ([anotherPickedColor isEqual:@"#7092BE"]){
            //--Point of Interest F--
            secondFinger = 15;
        }
        else if ([anotherPickedColor isEqual:@"#400340"]){
            //--Point of Interest F--
            secondFinger = 16;
        }
        else if ([anotherPickedColor isEqual:@"#B003FF"]){
            //--Point of Interest F--
            secondFinger = 17;
        }
        else if ([anotherPickedColor isEqual:@"#03A2E8"]){
            //--Fountain Color--
            secondFinger = 18;
        }
        else if ([anotherPickedColor isEqual:@"#030303"]){
            //--Black Color--
            secondFinger = 4;
        }
        else if ([anotherPickedColor isEqual:@"#FF0303"]){
            //--Red Color--
            secondFinger = 5;
        }
        else if ([anotherPickedColor isEqual:@"#FFFFFF"]){
            //--White Color--
            secondFinger = 6;
        }
        else if ([anotherPickedColor isEqual:@"#F5F5F5"]){
            //--Art White Color--
            secondFinger = 19;
        }
        else if ([anotherPickedColor isEqual:@"#FAFAFA"]){
            //--Workshop White Color--
            secondFinger = 20;
        }
        else if ([anotherPickedColor isEqual:@"#F0F0F0"]){
            //--Plant Nursery White Color--
            secondFinger = 21;
        }
        else if ([anotherPickedColor isEqual:@"#E6E6E6"]){
            //--Library White Color--
            secondFinger = 22;
        }
        else if ([anotherPickedColor isEqual:@"#EBEBEB"]){
            //--Education White Color--
            secondFinger = 23;
        }
        else {
            secondFinger = 24;
        }

        //--Calling the colorChanged Function to see if any of the colors have changed or not--
        //--Determining which color has been selected by firstFinger and secondFinger and passing that value as an integer to the function colorChanged:
            secondColor: --
        [self colorChanged:firstFinger secondColor:secondFinger];

    }

    if (location2.x == zero && location2.y == zero) {
        self.leftRightFinger = [NSMutableArray arrayWithObjects:self.touch1, nil];
```

```
//--Finding the Color that the finger is currently over and then converting it into Hexidecimal so that it can be compared--
UIColor *pickedColor = [self getCurrentPixelColorAtPoint:[[self.leftRightFinger objectAtIndex:0] locationInView:self.view]];
NSString *singleFingerPickedColor = [self hexFromUIColor:pickedColor];
//NSLog(@"%@",thePickedColor);

//--Determining which state the firstFinger is in--
if ([singleFingerPickedColor isEqual:@"#FFFF03"]) {
    //--Yellow Color--
    firstFinger = 0;
}
else if ([singleFingerPickedColor isEqual:@"#038003"]){
    //--Green Color--
    firstFinger = 1;
}
else if ([singleFingerPickedColor isEqual:@"#999999"]){
    //--Gray Color--
    firstFinger = 2;
}
else if ([singleFingerPickedColor isEqual:@"#0303FF"] || [singleFingerPickedColor isEqual:@"#030380"]){
    //--Blue Color--
    firstFinger = 3;
}
else if ([singleFingerPickedColor isEqual:@"#8888FF"]){
    //--Point of Interest B--
    firstFinger = 7;
}
else if ([singleFingerPickedColor isEqual:@"#A3BDE4"]){
    //--Point of Interest C--
    firstFinger = 8;
}
else if ([singleFingerPickedColor isEqual:@"#7D5CE0"]){
    //--Point of Interest D--
    firstFinger = 9;
}
else if ([singleFingerPickedColor isEqual:@"#0303A0"]){
    //--Point of Interest E--
    firstFinger = 10;
}
else if ([singleFingerPickedColor isEqual:@"#03FFFF"]){
    //--Point of Interest F--
    firstFinger = 11;
}
else if ([singleFingerPickedColor isEqual:@"#400380"]){
    //--Point of Interest G--
    firstFinger = 12;
}
else if ([singleFingerPickedColor isEqual:@"#A349A4"]){
    //--Point of Interest H--
    firstFinger = 13;
}
else if ([singleFingerPickedColor isEqual:@"#C8BFE7"]){
    //--Point of Interest I--
    firstFinger = 14;
}
else if ([singleFingerPickedColor isEqual:@"#7092BE"]){
    //--Point of Interest J--
    firstFinger = 15;
}
else if ([singleFingerPickedColor isEqual:@"#400340"]){
    //--Point of Interest K--
    firstFinger = 16;
}
else if ([singleFingerPickedColor isEqual:@"#8003FF"]){
    //--Point of Interest L--
    firstFinger = 17;
}
else if ([singleFingerPickedColor isEqual:@"#03A2E8"]){
    //--Fountain Color--
    firstFinger = 18;
}
else if ([singleFingerPickedColor isEqual:@"#030303"]){
    //--Black Color--
    firstFinger = 4;
}
```

```
        }
        else if ([singleFingerPickedColor isEqual:@"#FF0303"]){
            //--Red Color--
            firstFinger = 5;
        }
        else if ([singleFingerPickedColor isEqual:@"#FFFFFF"]){
            //--White Color--
            firstFinger = 6;
        }
        else if ([singleFingerPickedColor isEqual:@"#F5F5F5"]){
            //--Art White Color--
            firstFinger = 19;
        }
        else if ([singleFingerPickedColor isEqual:@"#FAFAFA"]){
            //--Workshop White Color--
            firstFinger = 20;
        }
        else if ([singleFingerPickedColor isEqual:@"#F0F0F0"]){
            //--Plant Nursery White Color--
            firstFinger = 21;
        }
        else if ([singleFingerPickedColor isEqual:@"#E6E6E6"]){
            //--Library White Color--
            firstFinger = 22;
        }
        else if ([singleFingerPickedColor isEqual:@"#EBEBEB"]){
            //--Education White Color--
            firstFinger = 23;
        }
        else {
            firstFinger = 24;
        }


        //--Logging the Hex Color--
        NSLog(@"Single Finger Color %@",[self hexFromUIColor:pickedColor]);
        NSLog(@"%i",firstFinger);
        //NSLog(@"Right Finger Color %@", [self hexFromUIColor:secondPickedColor]);

        //--Calling the singleFingerColor Function to see if the color has changed or not--
        [self singleFingerColor:firstFinger];

    }

    //NSLog(@"Touch1 and Touch2 beginning location: %f, %f and %f, %f",location1.x, location1.y,location2.x,location2.y);
}


//--Being called if there is only one finger on the screen--
//--Checking to see what the current color is compared to the previous color--
-(void)singleFingerColor:(int)theFirstColor{
    if (firstColorCompared == theFirstColor) {
        //--If the color is the same checking to see if the audioplayer is still playing and if it isn't calling the function to play the noise for that color
            again--
        if (audioPlayer.playing) {
            return;
        }
        else {
            [self currentSingleFingerColorNoise];

            //--Setting up the timer so that it will repeat itself if there is no activity--
            repeatTimer = 8;
            self.timer = [[NSTimer alloc] init];
            self.timer = [NSTimer scheduledTimerWithTimeInterval:1.0 target:self selector:@selector(repeatMethod:) userInfo:nil repeats:true];
        }
    } else if (theFirstColor == 24){
        //--If theFirstColor does not match any of the colors we are looking for then no new noise will be played and returning to look for colors that match--
        NSLog(@"I am returning to previous function");
        return;
    } else {
        //--Setting the color to be compared to the new color and then calling the function to play the new noise for the new color--
        NSLog(@"Sending to next function");
        firstColorCompared  = theFirstColor;
        [self currentSingleFingerColorNoise];
```

```
    }
}

//--It works when staying on single color--
//--Need to end and restart the timer when the color changes--
-(void)repeatMethod:(NSTimer*)theTimer{
    NSLog(@"Started Timer");

    //repeatTimer = 8;
    repeatTimer--;
    NSLog(@"%i",repeatTimer);
    if (repeatTimer <= 0) {
        if (audioPlayer.playing) {
            //repeatTimer = 8;
            return;
        } else {
            [self currentSingleFingerColorNoise];
            repeatTimer = 8;
            return;
        }
    }

}


//--Being called if there is a single finger and there has been a color change or the current color is no longer being played by the audioplayer--
-(void)currentSingleFingerColorNoise{
    NSString *path;
    NSURL *soundUrl;

    //--Need if else statement so I can invalidate if it changes to new color but keep it going if it stays the same--
    //--Stopping the timer so that it will not repeat inifinitely--
    [self.timer invalidate];

    //--Determining which color has been selected and then playing the musical noise associated with it to both ears since in single finger mode--
    switch (firstColorCompared) {
        case 0:
            //--Yellow Color--
            path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 1:
            //--Green Color--
            path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 2:
            //--Gray Color--
            path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 3:
            //--Point of Interest A--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 4:
            //--Black Color--
            path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 5:
            //--Red Color--
            path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 6:
            //--White Color--
            /*if ([audioPlayer isPlaying]){
                [audioPlayer stop];
            }*/
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
```

```
case 7:
    //--Point of Interest B--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 8:
    //--Point of Interest C--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 9:
    //--Point of Interest D--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 10:
    //--Point of Interest E--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 11:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 12:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 13:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 14:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 15:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 16:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 17:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 18:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 19:
    //--Art Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav", [[NSBundle mainBundle] resourcePath]];
    break;
case 20:
    //--Workshop Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav", [[NSBundle mainBundle] resourcePath]];
    break;
case 21:
    //--Plant Nursery White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav", [[NSBundle mainBundle] resourcePath]];
    break;
case 22:
```

```
        case 22:
            //--Library Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav", [[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav", [[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path = @"Nil";
            break;
    }


    NSLog(@"First Color Compared: %i",firstColorCompared);

    //--Checking to make sure that neither NSString path2 is equal to Nil before playing the sound attached to the color--
    //--If it is a color not in the switch statement then no noise will be played--
    //--Playing soundUrl for Single Finger to both headphones--
    if ([path  isEqual: @"Nil"]) {
        return;
    } else {
        soundUrl = [NSURL fileURLWithPath:path];
        audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
        //--Left Headphone for Single Finger--
        audioPlayer.pan = -1.0;
        [audioPlayer play];
    }

    //NSLog(@"First Color Compared: %i",firstColorCompared);

    //--Finding out what the last matched state was so when touchesEnded called it can be spoken--
    if (firstColorCompared < 24){
        actualColor = firstColorCompared;
    }

}


//--When using two fingers function called for when the touches moved--
//--Looking to see if either of the colors have changed or not, if they have then calling the function currentColor to see what new colors are--
//--Passing the int values determined in touchesMoved function so the can be compared--
-(void)colorChanged:(int)theChosenColor secondColor:(int)theSecondChosenColor{
    //NSLog(@"Color Changed");
    NSLog(@"First color %i",theChosenColor);
    NSLog(@"Second color %i",theSecondChosenColor);

    //--Comparing ints firstColorCompare/secondColorCompared to theChosenColor/theSecondChosenColor to determine what to do--
    if (firstColorCompared == theChosenColor && secondColorCompared == theSecondChosenColor) {
        //--If the colors are the same checking to see if the audioplayer is still playing and if it isn't calling the function to play the noise for that color
            again--
        if (audioPlayer.playing && audioPlayer2.playing) {
            return;
        } else {
            [self currentColor];
        }
    }
    else if (firstColorCompared == theChosenColor){
        //--If only the right most finger color has changed calling secondColorNoise function to change the sound to the sound for that color--
        secondColorCompared = theSecondChosenColor;
        //self.secondComparedColor = self.theSecondPickedColor;
        [self secondColorNoise];
    }
    else if (secondColorCompared == theSecondChosenColor){
        //--If only the left most finger color has changed calling secondColorNoise function to change the sound to the sound for that color--
        firstColorCompared = theChosenColor;
        //self.firstComparedColor = self.thePickedColor;
        [self firstColorNoise];
    }
    else {
        //--If both finger colors have changed calling currentColor function to change the sounds to the proper sounds for those colors--
        firstColorCompared = theChosenColor;
        secondColorCompared = theSecondChosenColor;
        //self.firstComparedColor = self.thePickedColor;
```

```
        //self.firstComparedColor = self.thePickedColor;
        //self.secondComparedColor = self.theSecondPickedColor;
        [self currentColor];
    }

}

//--Playing the new sound when only the right finger has changed colors--
-(void)secondColorNoise{
    //[audioPlayer2 stop];
    //NSLog(@"Hello Second Finger");
    //--Setting up the NSString and NSURL to be used in the switch statements below--
    NSString *path2;
    NSURL *soundUrl2;

    //--Checking to see if the barButton is in the On state if so only clarinet noises played--
    if ([self.testSelectionItem.title isEqualToString:@"3"]) {

        //--Looking to see what the int vlaue of secondColorCompared is to determine which musical noise to play--
        switch (secondColorCompared) {
            case 0:
                //--Yellow Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 1:
                //--Green Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 2:
                //--Gray Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 3:
                //--Blue Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 4:
                //--Black Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 5:
                //--Red Color--
                path2 = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 6:
                //--White Color--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 7:
                //--Point of Interest B--
                path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 8:
                //--Point of Interest C--
                path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 9:
                //--Point of Interest D--
                path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 10:
                //--Point of Interest E--
                path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
```

```
            break;
        case 11:
            //--Point of Interest F--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 12:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 13:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 14:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 15:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 16:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 17:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 18:
            //--Fountain Point of Interest--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 19:
            //--Art Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path2 = @"Nil";
            break;
    }

    //--Checking to make sure that neither NSString path2 is equal to Nil before playing the sound attached to the color--
    //--If it is a color not in the switch statement then no noise will be played--
    //--Playing soundUrl2 for the Right Finger to right headphone--
    if ([path2  isEqual: @"Nil"]) {
        return;
    } else {
        soundUrl2 = [NSURL fileURLWithPath:path2];
        audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
        //--Right Headphone--
        audioPlayer2.pan = 1.0;
        [audioPlayer2 play];
    }
```

```
} else if ([self.testSelectionItem.title isEqualToString:@"2"]) {
    //--Checking to see if the button is in the OFF state--
    //--If so then the second finger will play Guitar 4 frequencies to the Right Ear--

    switch (secondColorCompared) {
        case 0:
            //--Yellow Color--
            path2 = [NSString stringWithFormat:@"%@/guitar466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 1:
            //--Green Color--
            path2 = [NSString stringWithFormat:@"%@/guitar55volume.mp3", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 2:
            //--Gray Color--
            path2 = [NSString stringWithFormat:@"%@/guitar196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 3:
            //--Blue Color--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 4:
            //--Black Color--
            path2 = @"Nil";

            //path2 = [NSString stringWithFormat:@"%@/guitar55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 5:
            //--Red Color--
            path2 = [NSString stringWithFormat:@"%@/guitar1568long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 6:
            //--White Color--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 8:
            //--Point of Interest C--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 9:
            //--Point of Interest D--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 10:
            //--Point of Interest E--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 11:
            //--Point of Interest F--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 12:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 13:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
```

```
            case 14:
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 15:
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 16:
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 17:
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 18:
                //--Fountain Point of Interest--
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 19:
                //--Art Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 20:
                //--Workshop Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 21:
                //--Plant Nursery White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 22:
                //--Library Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 23:
                //--Education Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            default:
                path2 = @"Nil";
                break;
        }

        //--Checking to make sure that neither NSString path2 is equal to Nil before playing the sound attached to the color--
        //--If it is a color not in the switch statement then no noise will be played--
        //--Playing soundUrl2 for the Right Finger to right headphone--
        if ([path2  isEqual: @"Nil"]) {
            return;
        } else {
            soundUrl2 = [NSURL fileURLWithPath:path2];
            audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
            //--Left Headphone--
            audioPlayer2.pan = -1.0;
            [audioPlayer2 play];
        }


    } else if ([self.testSelectionItem.title isEqualToString:@"4"]) {
        //--The onOffButton is in the Both state--
        //--Meaning the Guitar will be played for the Second Finger but to Both Ears--

        switch (secondColorCompared) {
            case 0:
                //--Yellow Color--
                path2 = [NSString stringWithFormat:@"%@/guitar466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 1:
                //--Green Color--
                path2 = [NSString stringWithFormat:@"%@/guitar55volume.mp3", [[NSBundle mainBundle] resourcePath]];
```

```
    break;
case 2:
    //--Gray Color--
    path2 = [NSString stringWithFormat:@"%@/guitar196long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 3:
    //--Blue Color--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 4:
    //--Black Color--
    path2 = @"Nil";

    //path2 = [NSString stringWithFormat:@"%@/guitar55long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 5:
    //--Red Color--
    path2 = [NSString stringWithFormat:@"%@/guitar1568long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 6:
    //--White Color--
    path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
    break;
case 7:
    //--Point of Interest B--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 8:
    //--Point of Interest C--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 9:
    //--Point of Interest D--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 10:
    //--Point of Interest E--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 11:
    //--Point of Interest F--
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 12:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 13:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 14:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 15:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 16:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl2 = [NSURL fileURLWithPath:path2];
    break;
case 17:
    path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
```

```
            case 17:
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 18:
                //--Fountain Point of Interest--
                path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 19:
                //--Art Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 20:
                //--Workshop Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 21:
                //--Plant Nursery White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 22:
                //--Library Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 23:
                //--Education Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            default:
                path2 = @"Nil";
                break;
        }

        //--Checking to make sure that neither NSString path2 is equal to Nil before playing the sound attached to the color--
        //--If it is a color not in the switch statement then no noise will be played--
        //--Playing soundUrl2 for the Right Finger to both ears--
        if ([path2  isEqual: @"Nil"]) {
            return;
        } else {
            soundUrl2 = [NSURL fileURLWithPath:path2];
            audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
            //--Right Headphone--
            audioPlayer2.pan = 1.0;
            [audioPlayer2 play];
        }
    } else {

        //--Need to set up so that only the clarinet plays when the test selection is equal to 1--
    }

    //--Finding out what the last matched state was so when touchesEnded called it can be spoken--
    if (secondColorCompared < 24){
        actualColor2 = secondColorCompared;
    }

}


//--Playing the new sound when only the left finger has changed colors--
-(void)firstColorNoise{
    //[audioPlayer stop];
    //NSLog(@"Hello First Finger");
    NSString *path;
    NSURL *soundUrl;

    //--Checking to see if the onOffButton is in the On State if so only clarinet sounds (6 frequencies) played to Left Ear--
    if ([self.testSelectionItem.title isEqualToString:@"3"]) {
        switch (firstColorCompared) {
            case 0:
                //--Yellow Color--
                path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
```

```
            case 1:
                //--Green Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 4:
                //--Black Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 5:
                //--Red Color--
                path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 6:
                //--White Color--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 7:
                //--Point of Interest B--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 8:
                //--Point of Interest C--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 9:
                //--Point of Interest D--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 10:
                //--Point of Interest E--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 11:
                //--Point of Interest F--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 12:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 13:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 14:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 15:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 16:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
```

```
            break;
        case 17:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 18:
            //--Fountain Point of Interest--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 19:
            //--Art Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path = @"Nil";
            break;
    }

    //--Checking to make sure that neither NSString path is equal to Nil before playing the sound attached to the color--
    //--Playing the soundUrl for the left finger to the left headphone--
    if ([path  isEqual: @"Nil"]) {
        return;
    } else {
        soundUrl = [NSURL fileURLWithPath:path];
        audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
        //--Left Headphone--
        audioPlayer.pan = -1.0;
        [audioPlayer play];
    }


} else if ([self.testSelectionItem.title isEqualToString:@"2"]) {
    //--Checking to see if the onOffButton is in the Off State if so only clarinet sounds (4 frequencies) played to Left Ear--
    switch (firstColorCompared) {
        case 0:
            //--Yellow Color--
            path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 1:
            //--Green Color--
            path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 2:
            //--Gray Color--
            path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 3:
            //--Blue Color--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 4:
            //--Black Color--
            path = @"Nil";
```

```
            break;
        case 5:
            //--Red Color--
            path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 6:
            //--White Color--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 8:
            //--Point of Interest C--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 9:
            //--Point of Interest D--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 10:
            //--Point of Interest E--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 11:
            //--Point of Interest F--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 12:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 13:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 14:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 15:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 16:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 17:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 18:
            //--Fountain Point of Interest--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 19:
            //--Art Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
```

```
            case 21:
                //--Plant Nursery White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 22:
                //--Library Building White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 23:
                //--Education Building White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            default:
                path = @"Nil";
                break;
        }

        //--Checking to make sure that neither NSString path is equal to Nil before playing the sound attached to the color--
        //--Playing soundUrl for the left finger to the left headphone--
        if ([path  isEqual: @"Nil"]) {
            return;
        } else {
            soundUrl = [NSURL fileURLWithPath:path];
            audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
            //--Left Headphone--
            audioPlayer.pan = -1.0;
            [audioPlayer play];
        }

    } else {
        //--Checking to see if the onOffButton is in the Both State if so only clarinet sounds (4 frequencies) played to Both Ears--
        switch (firstColorCompared) {
            case 0:
                //--Yellow Color--
                path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 1:
                //--Green Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 4:
                //--Black Color--
                path = @"Nil";

                //path = [NSString stringWithFormat:@"%@/clarinet440long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 5:
                //--Red Color--
                path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 6:
                //--White Color--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 7:
                //--Point of Interest B--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 8:
```

```
            case 8:
                //--Point of Interest C--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 9:
                //--Point of Interest D--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 10:
                //--Point of Interest E--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 11:
                //--Point of Interest F--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 12:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 13:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 14:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 15:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 16:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 17:
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 18:
                //--Fountain Point of Interest--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 19:
                //--Art Buiding White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 20:
                //--Workshop Building White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 21:
                //--Plant Nursery White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 22:
                //--Library Building White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 23:
                //--Education Building White--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            default:
                path = @"Nil";
                break;
        }
```

```
        //--Checking to make sure that neither NSString path is equal to Nil before playing the sound attached to the color--
        //--Playing soundUrl for the left finger to both ears--
        if ([path  isEqual: @"Nil"]) {
            return;
        } else {
            soundUrl = [NSURL fileURLWithPath:path];
            audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
            //--Left Headphone--
            audioPlayer.pan = -1.0;
            [audioPlayer play];
        }

    }

    //--Finding out what the last matched state was so when touchesEnded called it can be spoken--
    if (firstColorCompared < 24){
        actualColor = firstColorCompared;
    }

}


//--Function called when both fingers are not the same color as before--
//--The function is also called if both fingers are the same color and the audioplayer has stopped playing--
-(void)currentColor{
    //NSLog(@"The First Color Is: %@, and the Second Color is: %@",self.firstComparedColor,self.secondComparedColor);
    NSString *path, *path2;
    NSURL *soundUrl, *soundUrl2;

    //--Checking to see if the BarButton is in the On or Off position--
    if ([self.testSelectionItem.title isEqualToString:@"3"]) {

        //--Looking at what color the left most finger is to determine which sound should be played--
        switch (firstColorCompared) {
            case 0:
                //--Yellow Color--
                path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 1:
                //--Green Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 4:
                //--Black Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 5:
                //--Red Color--
                path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 6:
                //--White Color--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 7:
                //--Point of Interest B--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
```

```
case 8:
    //--Point of Interest C--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 9:
    //--Point of Interest D--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 10:
    //--Point of Interest E--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 11:
    //--Point of Interest F--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 12:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 13:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 14:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 15:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 16:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 17:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 18:
    //--Fountain Point of Interest--
    path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
    //soundUrl = [NSURL fileURLWithPath:path];
    break;
case 19:
    //--Art Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
    break;
case 20:
    //--Workshop Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
    break;
case 21:
    //--Plant Nursery White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
    break;
case 22:
    //--Library Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
    break;
case 23:
    //--Education Building White--
    path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
```

```
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path = @"Nil";
            break;
    }

    //--Looking at what color the right most finger is to determine which sound should be played--
    switch (secondColorCompared) {
        case 0:
            //--Yellow Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 1:
            //--Green Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 2:
            //--Gray Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 3:
            //--Blue Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 4:
            //--Black Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 5:
            //--Red Color--
            path2 = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 6:
            //--White Color--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 8:
            //--Point of Interest C--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 9:
            //--Point of Interest D--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 10:
            //--Point of Interest E--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 11:
            //--Point of Interest F--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 12:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 13:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
```

```
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 14:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 15:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 16:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 17:
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 18:
            //--Fountain Point of Interest--
            path2 = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 19:
            //--Art Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path2 = @"Nil";
            break;
    }

    //--Checking to make sure that neither NSString path or path2 are equal to Nil before playing the sounds attached to the colors--
    //--Playing soundUrl to left headphone and soundUrl2 to the right headphone--
    if ([path  isEqual: @"Nil"] && [path2  isEqual: @"Nil"]) {
        NSLog(@"Something went Wrong");
        return;
    }
    else if ([path  isEqual: @"Nil"]){
        NSLog(@"Only the right ear");
        soundUrl2 = [NSURL fileURLWithPath:path2];
        audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
        //--Right Headphone--
        audioPlayer2.pan = 1.0;
        [audioPlayer2 play];
    }
    else if ([path2  isEqual: @"Nil"]){
        NSLog(@"Only the left ear");
        soundUrl = [NSURL fileURLWithPath:path];
        audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
        //--Left Headphone--
        audioPlayer.pan = -1.0;
        [audioPlayer play];
    }
    else {
        NSLog(@"Both Ears");
        soundUrl = [NSURL fileURLWithPath:path];
```

```
        else {
            NSLog(@"Both Ears");
            soundUrl = [NSURL fileURLWithPath:path];
            audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
            //--Left Headphone--
            audioPlayer.pan = -1.0;
            [audioPlayer play];

            soundUrl2 = [NSURL fileURLWithPath:path2];
            audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
            //--Right Headphone--
            audioPlayer2.pan = 1.0;
            [audioPlayer2 play];
        }

    } else if ([self.testSelectionItem.title isEqual:@"2"]){
        //--Will be played when the BarButton is in the OFF position--
        //--Meaning the Clarinet will be played in the Left Ear and the Guitar will be played in the Right Ear--

        NSLog(@"I am in Off Mode");

        //--Looking at what color the left most finger is to determine which sound should be played--
        switch (firstColorCompared) {
            case 0:
                //--Yellow Color--
                path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 1:
                //--Green Color--
                path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 4:
                //--Black Color--
                path = @"Nil";

                //path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 5:
                //--Red Color--
                path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 6:
                //--White Color--
                path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 7:
                //--Point of Interest B--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 8:
                //--Point of Interest C--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 9:
                //--Point of Interest D--
                path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
```

```
        case 10:
            //--Point of Interest E--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 11:
            //--Point of Interest F--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 12:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 13:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 14:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 15:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 16:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 17:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 18:
            //--Fountain Point of Interest--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 19:
            //--Art Buidling White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path = @"Nil";
            break;
    }

    //--Looking at what color the right most finger is to determine which sound should be played--
    //--May be changing to have it so that only clarinet is being played--
    switch (secondColorCompared) {
        case 0:
            //--Yellow Color--
            path2 = [NSString stringWithFormat:@"%@/guitar466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 1:
            //--Green Color--
```

```
            //--Green Color--
            path2 = [NSString stringWithFormat:@"%@/guitar55volume.mp3", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 2:
            //--Gray Color--
            path2 = [NSString stringWithFormat:@"%@/guitar196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 3:
            //--Blue Color--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 4:
            //--Black Color--
            path2 = @"Nil";

            //path2 = [NSString stringWithFormat:@"%@/guitar55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 5:
            //--Red Color--
            path2 = [NSString stringWithFormat:@"%@/guitar1568long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 6:
            //--White Color--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 8:
            //--Point of Interest C--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 9:
            //--Point of Interest D--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 10:
            //--Point of Interest E--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 11:
            //--Point of Interest F--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 12:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 13:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 14:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 15:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 16:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
```

```
            break;
        case 17:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 18:
            //--Fountain Point of Interest--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 19:
            //--Art Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path2 = @"Nil";
            break;
    }

    //--Checking to make sure that neither NSString path or path2 are equal to Nil before playing the sounds attached to the colors--
    //--Playing soundUrl to the left headphone and soundUrl2 to the right headphone--
    if ([path  isEqual: @"Nil"] && [path2  isEqual: @"Nil"]) {
        NSLog(@"Something went Wrong");
        return;
    }
    else if ([path  isEqual: @"Nil"]){
        NSLog(@"Only the right ear");
        soundUrl2 = [NSURL fileURLWithPath:path2];
        audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
        //--Left Headphone--
        audioPlayer2.pan = -1.0;
        [audioPlayer2 play];
    }
    else if ([path2  isEqual: @"Nil"]){
        NSLog(@"Only the left ear");
        soundUrl = [NSURL fileURLWithPath:path];
        audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
        //--Left Headphone--
        audioPlayer.pan = -1.0;
        [audioPlayer play];
    }
    else {
        NSLog(@"Both Ears");
        soundUrl = [NSURL fileURLWithPath:path];
        audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
        //--Left Headphone--
        audioPlayer.pan = -1.0;
        [audioPlayer play];

        soundUrl2 = [NSURL fileURLWithPath:path2];
        audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
        //--Left Headphone--
        audioPlayer2.pan = -1.0;
        [audioPlayer2 play];
    }

} else if ([self.testSelectionItem.title isEqualToString:@"4"]) {
    //--Will be played when the BarButton is in the BOTH position--
```

```
} else if ([self.testSelectionItem.title isEqualToString:@"4"]) {
    //--Will be played when the BarButton is in the BOTH position--
    //--Meaning the Clarinet will be played in Both Ears and the Guitar will be played in Both Ears--

    //--Looking at what color the left most finger is to determine which sound should be played--
    switch (firstColorCompared) {
        case 0:
            //--Yellow Color--
            path = [NSString stringWithFormat:@"%@/clarinet466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 1:
            //--Green Color--
            path = [NSString stringWithFormat:@"%@/clarinet55long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 2:
            //--Gray Color--
            path = [NSString stringWithFormat:@"%@/clarinet196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 3:
            //--Blue Color--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 4:
            //--Black Color--
            path = @"Nil";

            //path = [NSString stringWithFormat:@"%@/clarinet440long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 5:
            //--Red Color--
            path = [NSString stringWithFormat:@"%@/clarinet1760long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 6:
            //--White Color--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 8:
            //--Point of Interest C--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 9:
            //--Point of Interest D--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 10:
            //--Point of Interest E--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 11:
            //--Point of Interest F--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 12:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 13:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
```

```
            break;
        case 14:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 15:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 16:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 17:
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 18:
            //--Fountain Point of Interest--
            path = [NSString stringWithFormat:@"%@/clarinet880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 19:
            //--Art Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
            //--Plant Nursery White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 22:
            //--Library Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 23:
            //--Education Building White--
            path = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        default:
            path = @"Nil";
            break;
    }

    //--Looking at what color the right most finger is to determine which sound should be played--
    switch (secondColorCompared) {
        case 0:
            //--Yellow Color--
            path2 = [NSString stringWithFormat:@"%@/guitar466long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 1:
            //--Green Color--
            path2 = [NSString stringWithFormat:@"%@/guitar55volume.mp3", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 2:
            //--Gray Color--
            path2 = [NSString stringWithFormat:@"%@/guitar196long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 3:
            //--Blue Color--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 4:
            //--Black Color--
            path2 = @"Nil";

            //path2 = [NSString stringWithFormat:@"%@/guitar55long.wav", [[NSBundle mainBundle] resourcePath]];
```

```
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 5:
            //--Red Color--
            path2 = [NSString stringWithFormat:@"%@/guitar1760long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 6:
            //--White Color--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 7:
            //--Point of Interest B--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 8:
            //--Point of Interest C--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 9:
            //--Point of Interest D--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 10:
            //--Point of Interest E--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 11:
            //--Point of Interest F--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 12:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 13:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 14:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 15:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 16:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 17:
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 18:
            //--Fountain Point of Interest--
            path2 = [NSString stringWithFormat:@"%@/guitar880long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 19:
            //--Art Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 20:
            //--Workshop Building White--
            path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
            break;
        case 21:
```

```
            case 21:
                //--Plant Nursery White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 22:
                //--Library Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            case 23:
                //--Education Building White--
                path2 = [NSString stringWithFormat:@"%@/silentSound.wav",[[NSBundle mainBundle] resourcePath]];
                break;
            default:
                path2 = @"Nil";
                break;
        }

        //--Checking to make sure that neither NSString path or path2 are equal to Nil before playing the sounds attached to the colors--
        //--Playing the sounds to both headphones--
        if ([path  isEqual: @"Nil"] && [path2  isEqual: @"Nil"]) {
            return;
        }
        else if ([path  isEqual: @"Nil"]){
            NSLog(@"Both Ears");
            soundUrl2 = [NSURL fileURLWithPath:path2];
            audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
            //--Right Headphones--
            audioPlayer2.pan = 1.0;
            [audioPlayer2 play];
        }
        else if ([path2  isEqual: @"Nil"]){
            NSLog(@"Both Ears");
            soundUrl = [NSURL fileURLWithPath:path];
            audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
            //--Left Headphones--
            audioPlayer.pan = -1.0;
            [audioPlayer play];
        }
        else {
            NSLog(@"Both Ears");
            soundUrl = [NSURL fileURLWithPath:path];
            audioPlayer = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl error:nil];
            //--Left Headphones--
            audioPlayer.pan = -1.0;
            [audioPlayer play];

            soundUrl2 = [NSURL fileURLWithPath:path2];
            audioPlayer2 = [[AVAudioPlayer alloc] initWithContentsOfURL:soundUrl2 error:nil];
            //--Right Headphones--
            audioPlayer2.pan = 1.0;
            [audioPlayer2 play];
        }

    } else {

        //--Need to set up so that only Clarinet playes when to left ear when Test Selection is in equal to "1"--
    }



    //--Finding out what the last matched color of left finger was so when touchesEnded called it can be spoken--
    if (firstColorCompared < 24){
        actualColor = firstColorCompared;
    }
    //--Finding out what the last matched color of right finger was so when touchesEnded called it can be spoken--
    if (secondColorCompared < 24){
        actualColor2 = secondColorCompared;
    }

}


//--Providing the point on the screen where the user has last touched it (the location is given once user lifts the finger off the screen)--
-(void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {
```

```
-(void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event {

    NSLog(@"hello");
    //--Stopping the NSTimer when the finger is lifted so it doesn't keep repeating the audio--
    [self.timer invalidate];
    //self.timer = nil;

    NSUInteger touchCount = [touches count];
    NSLog(@"%lu", (unsigned long)touchCount);

    //--Stopping the audioPlayers from playing the music--
    [audioPlayer stop];
    [audioPlayer2 stop];

    UITouch *touch1Compared, *touch2Compared;
    CGPoint location1Compared, location2Compared;

    NSSet *allTouches = [event allTouches];
    touch1Compared = [[allTouches allObjects] objectAtIndex:0];
    //NSLog(@"Touch 1 Compared is :%@",touch1Compared);
    location1Compared = [touch1Compared locationInView:self.view];

    if ([allTouches count] > 1) {
        touch2Compared = [[allTouches allObjects] objectAtIndex:1];
        //NSLog(@"Touch 2 Compared is :%@",touch2Compared);
        location2Compared = [touch2Compared locationInView:self.view];
    }

    //NSLog(@"Touch End Location %f",location2Compared.x);
    //NSLog(@"Touch Location %f",location2.x);

    //--Setting up the strings and urls storing WAV files--
    NSString *talkPath, *talkPath2;
    NSURL *talkSoundUrl, *talkSoundUrl2;

    //--Checking to make sure that the fingers had not gone of the edge of the map before being lifte of the screen--
    if ([self.leftRightFinger count] < 1) {
        NSLog(@"Houston We Have A Problem");
        //--If the finger has come of the map then returning to previous function--
        return;
    }

    NSLog(@"This is the Problem Point");
    //--If else statement to see if it was either the left finger the right finger or both fingers came of the screen and what should happen in those cases--
    if ([[self.leftRightFinger objectAtIndex:0] phase] == UITouchPhaseEnded) {
        //--Detecting if the left finger has been lifted--
        NSLog(@"Left Most Touch Has Ended");
        NSLog(@"%d",actualColor);

        switch (actualColor) {
            case 0:
                //--Yellow Color--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Yellow"];
                [self.syn speakUtterance:self.utterance2];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 1:
                //--Green Color--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Green"];
                [self.syn speakUtterance:self.utterance2];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Gray"];
                [self.syn speakUtterance:self.utterance2];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
```

```
            break;
        case 4:
            //--Black Color--
            //--Want to remove wall sound replace with Garden--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Wall"];
            [self.syn speakUtterance:self.utterance2];
            //path = [NSString stringWithFormat:@"%@/clarinet440long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 5:
            //--Red Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Red"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 6:
            //--White Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 7:
            //--Point of Interest B--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 8:
            //--Point of Interest C--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 9:
            //--Point of Interest D--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 10:
            //--Point of Interest E--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 11:
            //--Point of Interest F--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 12:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 13:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 14:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 15:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 16:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 17:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 18:
            //--Fountain Point of Interest--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
```

```
            case 19:
                //--Art Building White--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 20:
                //--Workshop Building White--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 21:
                //--Plant Nursery White--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 22:
                //--Library Building White--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 23:
                //--Education Building White--
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance2];
                break;
            default:
                self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@""];
                //[self.syn speakUtterance:self.utterance2];
                break;
        }


        //--Needed to set the firstColorCompared equal to secondColorCompared because when the left most finger comes off the screen then touch2 is the left most
            finger and if touch2 is stationary then it will not automatically change to the second color--
        actualColor = actualColor2;

    }
    else if ([[self.leftRightFinger objectAtIndex:1] phase] == UITouchPhaseEnded)  {
        //--Detecting if the right finger has been lifted--
        NSLog(@"Right Most Touch Has Ended");

        switch (actualColor2) {
            case 0:
                //--Yellow Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Yellow"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 1:
                //--Green Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Green"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 2:
                //--Gray Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Gray"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 3:
                //--Blue Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 4:
                //--Black Color--
                //--Want to remove wall sound replace with Garden--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Wall"];
                [self.syn speakUtterance:self.utterance3];
                //path = [NSString stringWithFormat:@"%@/clarinet440long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
```

```
            case 5:
                //--Red Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Red"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl = [NSURL fileURLWithPath:path];
                break;
            case 6:
                //--White Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 7:
                //--Point of Interest B--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 8:
                //--Point of Interest C--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 9:
                //--Point of Interest D--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 10:
                //--Point of Interest E--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 11:
                //--Point of Interest F--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 12:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 13:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 14:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 15:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 16:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 17:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 18:
                //--Fountain Point of Interest--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 19:
                //--Art Building White--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 20:
                //--Workshop Building White--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance3];
```

```
            [self.syn speakUtterance:self.utterance3];
            break;
        case 21:
            //--Plant Nursery White--
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
            [self.syn speakUtterance:self.utterance3];
            break;
        case 22:
            //--Library Building White--
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
            [self.syn speakUtterance:self.utterance3];
            break;
        case 23:
            //--Education Building White--
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
            [self.syn speakUtterance:self.utterance3];
            break;
        default:
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@""];
            //[self.syn speakUtterance:self.utterance2];
            break;
    }

}
else {
    //--Determinig the saying for when both fingers come of the screen--

    //--Checking to see what the last matched color for the left finger was and then setting the utterance for that color--
    switch (actualColor) {
        case 0:
            //--Yellow Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Yellow"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 1:
            //--Green Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Green"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 2:
            //--Gray Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Gray"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 3:
            //--Blue Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 4:
            //--Black Color--
            //--Want to remove wall sound replace with Garden--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Wall"];
            [self.syn speakUtterance:self.utterance2];
            //path = [NSString stringWithFormat:@"%@/clarinet440long.wav", [[NSBundle mainBundle] resourcePath]];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 5:
            //--Red Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Red"];
            [self.syn speakUtterance:self.utterance2];
            //soundUrl = [NSURL fileURLWithPath:path];
            break;
        case 6:
            //--White Color--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 7:
```

```
        case 7:
            //--Point of Interest B--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 8:
            //--Point of Interest C--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 9:
            //--Point of Interest D--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 10:
            //--Point of Interest E--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 11:
            //--Point of Interest F--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 12:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 13:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 14:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 15:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 16:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 17:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        case 18:
            //--Fountain Point of Interest--
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
            [self.syn speakUtterance:self.utterance2];
            break;
        default:
            self.utterance2 = [AVSpeechUtterance speechUtteranceWithString:@""];
            //[self.syn speakUtterance:self.utterance2];
            break;
    }

    //--Checking to see what the last matched color for the right finger was and then setting the utterance for that color--
    switch (actualColor2) {
        case 0:
            //--Yellow Color--
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Yellow"];
            [self.syn speakUtterance:self.utterance3];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
        case 1:
            //--Green Color--
            self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Green"];
            [self.syn speakUtterance:self.utterance3];
            //soundUrl2 = [NSURL fileURLWithPath:path2];
            break;
```

```
            case 1:
                //--Green Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Green"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 2:
                //--Gray Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Gray"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 3:
                //--Blue Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 4:
                //--Black Color--
                //--Want to remove wall sound replace with Garden--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Wall"];
                [self.syn speakUtterance:self.utterance3];
                //path2 = [NSString stringWithFormat:@"%@/guitar55long.wav", [[NSBundle mainBundle] resourcePath]];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 5:
                //--Red Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Red"];
                [self.syn speakUtterance:self.utterance3];
                //soundUrl2 = [NSURL fileURLWithPath:path2];
                break;
            case 6:
                //--White Color--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Building"];
                [self.syn speakUtterance:self.utterance3];
                break;
            case 7:
                //--Point of Interest B--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 8:
                //--Point of Interest C--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 9:
                //--Point of Interest D--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 10:
                //--Point of Interest E--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 11:
                //--Point of Interest F--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 12:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 13:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 14:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
```

```
            case 15:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 16:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 17:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            case 18:
                //--Fountain Point of Interest--
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@"Blue"];
                [self.syn speakUtterance:self.utterance2];
                break;
            default:
                self.utterance3 = [AVSpeechUtterance speechUtteranceWithString:@""];
                break;
        }
    }


    UITouch *touch = [[event allTouches] anyObject];
    CGPoint loc = [touch locationInView:self.view];
    //--Calling the method to return the RGB value of the pixel touched--
    UIColor *pickedColor = [self getCurrentPixelColorAtPoint:loc];
    if ([touches count] < 1) {
        NSLog(@"Loc Position %f, %f",location1.x,location1.y);
    }

    //--Calling the method to get the hexidecimal color code of that location--
    NSString *thePickedColor = [self hexFromUIColor:pickedColor];
    //NSLog(@"%@",[self hexFromUIColor:pickedColor]);
    //NSLog(@"%i",firstColorCompared);
    //NSLog(@"%i",secondColorCompared);

}


//--Converting the UIColor presented to a hexidecimal color--
-(NSString *)hexFromUIColor:(UIColor *)color{
    if (CGColorGetNumberOfComponents(color.CGColor) < 4) {
        const CGFloat *components = CGColorGetComponents(color.CGColor);
        color = [UIColor colorWithRed:components[30] green:components[141] blue:components[13] alpha:components[1]];
    }
    if (CGColorSpaceGetModel(CGColorGetColorSpace(color.CGColor)) != kCGColorSpaceModelRGB) {
        return [NSString stringWithFormat:@"#FFFFFF"];
    }
    return [NSString stringWithFormat:@"#%02X%02X%02X", (int)((CGColorGetComponents(color.CGColor))[0]*255.0), (int)((CGColorGetComponents(color.CGColor))[1]*
        255.0), (int)((CGColorGetComponents(color.CGColor))[2]*255.0)];
}


//--Creating a bitmap of the image so the RGB of the pixel touched can be found--
-(UIColor *)getCurrentPixelColorAtPoint:(CGPoint)point {
    unsigned char pixel[4] = {0};
    CGColorSpaceRef colorSpace = CGColorSpaceCreateDeviceRGB();
    CGContextRef context = CGBitmapContextCreate(pixel, 1, 1, 8, 4, colorSpace, kCGBitmapAlphaInfoMask & kCGImageAlphaPremultipliedLast);
    CGContextTranslateCTM(context, -point.x, -point.y + 50.0);
    [self.imageHolder.layer renderInContext:context];
    CGContextRelease(context);
    CGColorSpaceRelease(colorSpace);
    //NSLog(@"pixel: %d %d %d %d", pixel[0],pixel[1],pixel[2],pixel[3]);
    UIColor *color = [UIColor colorWithRed:pixel[0]/255.0 green:pixel[1]/255.0 blue:pixel[2]/255.0 alpha:pixel[3]/255.0];
    //--Returning the RGB color in terms of a UIColor so it can be converted to a hexidecimal color--
    return color;
}


- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
```

```
- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be recreated.
}


@end
```

7.3. Variable

Colors for Overall Maps

Red – Represents the red individual garden

Blue – Represents the blue individual garden

Green – Represents the green individual garden

Gray – Represents the gray individual garden

Yellow – Represents the yellow individual garden

White – Represents the building(s)

Colors for Individual Maps

Red – Represent a set of stairs

Blue (or shades of purple) – Represents points of interest on the (POI) maps

Green – Represents garden/grass

Gray – Represents pathways

Yellow – Represents benches

White – Represents buildings

All Other Variables

Method 1 – Using single finger to explore maps with clarinet played to left ear.

Method 2 – Using two fingers to explore maps with left-most finger represented by the clarinet and the right-most finger represented by the guitar. Both instruments played to the left ear.

Method 3 – Using two fingers to explore maps with both fingers represented by the clarinet. With the left-most finger played to the left ear and the right-most finger played to the right ear.

Method 4 – Using two fingers to explore maps with the left-most finger represented by the clarinet and the right-most finger represented by the guitar. With the left-most finger played to the left ear and the right-most finger played to the right ear.

Method 5 – Using single finger to explore maps with tactile feedback played to that finger.

Method 6 – Using two fingers on the same hand to explore maps. With tactile feedback provided to both fingers.

Method 7 – Using two fingers, one on each hand to explore maps. With tactile feedback provided to each finger.

Overall Maps – Maps displaying the layout of all individual gardens as they would appear side by side.

Individual Maps – Maps focusing on a garden and all the elements it would contain.

Congenitally Blind – Defined for this research as individuals who became blind or visually impaired before the age of six.

Adventitiously Blind – Defined for this research as individuals who became blind or visually impaired at or after the age of six.

Experienced Tactile Experience – Defined for this research as individuals who had used tactile graphics consistently for the past 5 years or more.

Little to No Tactile Experience – Defined for this research as individuals who have not used tactile graphics or have not used it consistently for the past 5 years.

Group 1 – Consisted of spatial questions for Individual Maps asking how may items there were (for example: buildings, benches).

Group 2 – Consisted of spatial questions for Individual Maps dealing with identifying items around a given location on the map.

Group 3 – Consisted of spatial questions for Individual Maps in finding an item on the map while following the pathway.

Modality – Using either audio methods (Methods 1-4) or tactile methods (Methods 5-7)

Methods – Comparing each method with each other

Map Type – Comparing Overall Maps to Individual Maps

Blindness – Comparing Congenitally Blind to Adventitiously Blind

Tactile Experience – Comparing those with little to no tactile experience to the who have experience working with tactile graphics

shapeQ – Comparing questions asking about shapes to those that did not ask about shapes for Overall Maps

Spatial Groups – Comparing the three spatial groups (Groups 1, 2, and 3) for the Individual Maps to determine how well they were able to do on spatial questions.

Time – Consisted of the time it took to answer all ten question for the given map.

7.4. Citations

Adams, R., Pawluk, D., Fields, M., and Clingman, R. (2015). Multimodal Application for the Perception of Spaces (MAPS). ACM Assets 2015, 26-28 October, 2015. Lisbon, Portugal. doi:10.1145/2700648.281136

Bangor, A., Kortum, P., and Miller, J. (2008). An Empirical Evaluation of the System Usability Scale. International Journal of Human-Computer Interaction 24, pp. 574-594

Brittell, M., Young, M., and Lobben, A. (2013). The MGIS: a minimal geographic information system accessible to users who are blind. In Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - SIGSPATIAL'13 (pp. 554-557). New York, New York, USA: ACM Press. doi:10.1145/2525314.2525329

Brock, A., Truillet, P., Oriola, B., and Jouffrais, C. (2010). Usage of multimodal maps for blind people: why and how. In: Proceeding of the ACM International Conference on Interactive Tabletops and Surfaces (ITS 10). ACM, New York, NY, USA, pp. 247-248. doi:10.1145/1936652.1936699

Brock, A., Truillet, P., Oriola, B., Picard, D., and Jouffrais, C. (2012). Design and user satisfaction of interactive maps for visually impaired people. In: Miesenberger, K., Karshmer, A., Penaz, P., Zagler, W. (eds), Proceedings of the Thirteenth International Conference on Computers Helping People with Special Needs - Volume Part II (ICCHP 2012), Vol. Part II. Springer-Verlag, Berlin, Heidelberg, pp. 544-551. doi:10.1007/978-3-642-31534-3_80

Bujacz, M., and Strumillo, P. (2016). Sonification: Review of auditory display solutions in electronic travel aids for the blind. Arch. Acoust., 41 (3), pp. 401-14.

Burch, D., and Pawluk, D. (2009). A cheap, portable haptic device for a method to relay 2-D texture-enriched graphical information to individuals who are visually impaired. Proceedings of the Eleventh International ACM SIGACCESS Conference on Computers and Accessibility - ASSETS 09. doi:10.1145/1639642.1639682

Burch, D., and Pawluk, D. (2011). Using Multiple Contacts with Texture-enhanced Graphics. World Haptics Conference, 21-24 June 2011, Istanbul, Turkey. doi:10.1109/WHC.2011.5945500

Campin, B. McCurdy, W., Brunet, L., and Siekierska, E. (2003). SVG Maps for people with visual impairments. In Proceeding of the Second SVG Open Conference, Vancouver, Canada

Carrol, D., Chakraborty, S., and Lazar, J. (2013). Designing Accessible Visualizations: The Case of Designing a Weather Map for Blind Users. In Universal Access in Human-Computer Interaction. Design Methods, Tools, and Interaction Techniques for eInclusion (pp. 436-445). Springer Berlin Heidelberg.

Culbertson, H., Schorr, S.B., and Okamura A.M. (2018). Haptics: The Present and Future of Artificial Touch Sensation. Annual Review of Control, Robotics, and Autonomous Systems, 1:1, 385-409. https://doi.org/10.1146/annurev-control-060117-105043

Daunys, G., and Lauruska, V. (2009). Sonification System of Maps for Blind -- Alternative View. In C. Stephanidis (Ed.), Universal Access in Human-Computer Interaction. Intelligent and Ubiquitous Interaction. Environments. (Vol. 5615, pp 503-508). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-02710-9

Delogu, F., Palmiero, M., Federici, S., Plaisant, C., Zhao, H., and Belardinelli, O. (2010). Non-visual exploration of geographic maps: Does sonification help?, Disability and Rehabilitation: Assistive Technology, 5:3, 164-174. doi: 10.3109/17483100903100277

Ducasse, J., Brock, A., & Jouffrais, C. (2018). Accessible Interactive Maps for Visually Impaired Users. In *Mobility of Visually Impaired People Fundamentals and ICT Assistive Technologies*, by Edwige Pisaloux and Ramiro Velazquez, Springer-Verlag. Retrieved from: https://www.researchgate.net/profile/Christophe_Jouffrais/publication/319232260_Accessible_Interactive_Maps_for_Visually_Impaired_Users/links/5a2e699daca2728e05e32527/Accessible-Interactive-Maps-for-Visually-Impaired-Users.pdf

Dulin, D., and Hatwell, H. (2006). The Effects of Visual Experience and Training in Raised-Line Materials on the Mental Spatial Imager of Blind Persons. Retrieved from: https://journals.sagepub.com/doi/abs/10.1177.0264619608093641

Edman, P.K. (1992). Tactile Graphics. American Foundation for the Blind, New York, pp. 41-119.

Ferro, T. (2018). Automatic Image Processing and Conversion to Tactile Graphics. (Ph. D.). Virginia Commonwealth University (VCU), Richmond, Virginia, USA.

Fusco, G. and Morash, V. (2015). The Tactile Graphics Helper: Providing Audio Clarification for Tactile

Graphics Using Machine Vision. doi: 10.1145/2700648.2809868

Giudice, N.A., Palani, H.P., Brenner, E., and Kramer, K.M. (2012). Learning non-visual graphical information using a touch-based vibro-audio interface. In Proceedings of 14th International ACM SIGACCESS Conference Computers and Accessibility (ASSETS '12), pp. 103-110. ACM, New York, New York, USA.

Golledge, R. (2003). Spatial cognition and converging technologies. In: Roco, M.C., and Bainbridge, W.S (eds). Converging Technologies for Improving Human Performance, Kluwer Academic Publishers, Boston, pp. 122-140

Golledge, R.G., Rice, M., and Jacobson, R.D. (2005). A commentary on the use of touch for accessing on-screen spatial representations: The process of experiencing haptic maps and graphics. The Professional Geographer, 57(3), 339-349.

Habel, C., Kerzel, M., and Lohmann, K. (2010). Verbal assistance in tactile-map explorations: a case for visual representations and reasoning. In McGreggor, K., Kunda, M. (eds), Proceedings AAAI Workshop on Visual Representations and Reasoning, AAAI Conference 2010, Atlanta, GA, USA, pp. 34-41

Heller, M.A. (Ed.) (2000). Touch, representation and blindness. Oxford, England: Oxford University Press

Jacobson, R.D. (2002). Representing spatial information through multimodal interfaces. In Proceedings Sixth International Conference on Information Visualisation, pp. 730-734, doi:10.1109/IV.2002.1028858

Kahol, K., French, J., Bratton, L., and Panchanathan, S. (2006). Learning and perceiving colors haptically. Proc of ASSETS, pp. 173-180.

Kaklanis, N., Votis, K., and Tzovaras, D. (2013). Open Touch/Sound Maps: A system to covey street data through haptic and auditory feedback. Computers & Geosciences, 57, 59-67. doi:10.1016/j.cageo.2013.03.005

Kane, S.K., Morris, M.R., Perkins, A.Z., Wigdor, D., Ladner, R.E., and Wobbrock, J.O. (2011). Acces Overlays: Improving Non-Visual Access to Large Touch Screens for Blind Users. In Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11 (pp. 273-282). New York, New York, USA: ACM Press. doi:10.1145/2047196.2047232

Klatzky, R.L., Giudice, N.A., Bennett, C.R., and Loomis, J.M. (2014). Touch-Screen Technology for the Dynamic Display of 2D Spatial Information Without Vision: Promise and Progress. Multisensory Research, vol 27, pp. 359-378. doi:10.1163/22134808-00002447

Klatzky, R.L., Lederman, S.J., and Metzger, V. (1985). Identifying objects by touch: An "expert system". Perception and Psychphysics, 37(4), 299-302.

Landau, S., and Gourgey, K. (2001). Development of a talkin tactile tablet. Information Technology and Disablilities. <http://www.rit.edu/-easi/itd/itdv07n2/contents.htm>

Levesque, V., Petit, G., Dufrense, A., and Hayward, V. (2012). Adaptive level of detail in dynamic, refreshable tactile graphics. IEEE Haptics Symposium (HAPTICS), 1-5. doi:10.1109/HAPTIC.2012.6183752

Loomis, J.M. (2003). Sensory replacement and sensory substitution: Overview and prospects for the future. In: Roco, M.C., Binbridge, W.S. (eds), Converging Technologies for Improving Human Performance: Nanotechnology, Biotechnology, Information Technology and Cognitive Science, Kluwer Academic Publishers, Boston pp 189-199

Maingreaud, F., Pissaloux, E.E., Velazquez, R., Gaunet, F., Hafez, M., and Alexandre, J.M. (2005). A dynamic tactile map as a tool for space organization perception: Application to the design of an electronic travel aid for visually impaired an blind people. IEEE Engineering in Medicine and Biology 27 pp. 6912-6915. doi:10.1109/IEMBS.2005.1616095

Miele, J.A. and Marston, J.R. (2005). Tactile Map Automated Production (TMAP): on-demand accessible street maps for blind and visually impaired travelers. In: Proceedings of the Annual Meeting of the Ammerican Association of Geographers, Denver, CO.

Miele, J.A., Landau, S., and Gilden, D. (2006). Talking TMAP: Automated generation of audio-tactile maps using Smith-Kettlewells TMAP software. British Journal of Visual Impairment, 24(2), 93-100. doi:10.1177/0264619606064436

Milne, A.P., Antle, A.N., and Riecke, B.E. (2011). Tangible and body-bosed interaction with auditory maps. In CHI EA '11 Extended Abstracts on Human Factors in Computing Systems (p. 2329). New York, New York, USA: ACM Press. doi:10.1145/1979742.1979874

Minatani, K., et al. (2010). Tactile Map Creation System to Enhance the Mobility of Blind Persons--Its Design Concept and Evaluation through Experiment. In: Miesenberger K., Klaus J., Zagler W., Karshmer A. (eds) Computers Helping People with Special Needs. ICCHP 2010. Lecture Notes in Computer Science, vol 6180. Springer, Berlin, Heidelberg

Moustakas, K., Nikolakis, G., Kostopoulos, K., Tzovaras, D., and Strintzis, M.G. (2007). Haptic Rendering of Visual Data for the Visually Impaired. IEEE Multimedia 14 pp 62-72

Nation Federation of the Blind, NFB (2014). Statistical Facts about Blindness in the United States. https://nfb.org/blindness-statistics

Parente, P., and Bishop, G. (2003). BATS: The Blind Audio Tactile mapping System. Proceedings ACMSE, ACM Press, Savannah, GA.

Picinali, L., Afonso, A., Denis, M., and Katz, B.F.G. (2014). Exploration of archietectural spaces by blind people using auditory virtual reality for the construction of spatial knowledge. International Journal of Human-Computer Studies, 72(4), 393-405. doi:10.1016/j.ijhcs.2013.12.008

Pietrzak, T., Crossan, A., Brewster, S.A., Martin, B., and Pecci, I. (2009). Creating usable pin array tactions for non-visual information. IEEE Transactions on Haptics, 2(2), 61-72. doi:10.1109/TOH.2009.6

Rice, M.T., Jacobson, R.D., Golledge, R.G., and Jones, D. (2005). Cartographic Data and Design considerations for haptic and auditory map interfaces. Cartography and Geographic Information Science, 32(4), 381-391. Retrieved from http://www.ingentaconnect.com/content/acsm/cagis/2005/00000032/00000004/art00014

Salisbury J., Sirinivasan, M. (1992). Virtual environment Technology for training (VATT). BBN Report No. 7661. Cambridge, MA: VETREC, MIT.

Schneider, J., Strothotte, T. (1999). Virtual tactile maps. The 8th International Conference on Human-Computer Interaction, Volume I.

Su, J., Rosenzweig, A., Goel, A., de Lara, E., and Truong, K.N. (2010). Timbremap: Enabling the Visually-Impaired to Use Maps on Touch-Enabled Devices. In Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services - MoblieHCI'10 (pp. 17-26). New York, New York, USA: ACM Press.

Turner, A. (2012). Behind bars, Braille dots fulfill prison inmates, aid the blind.

http://www.houstonchronicle.com/news/houston-texas/houston/article/Behind-bars-Braille-s-dots-fulfill-prison-4153070.php

Vozenilek, V., Kozakova, M., Stavova, Z., Lukikova, L., Ruzickova, V., and Finkova, D. (2009). 3D printing technology in tactile maps compiling. Proceeding of 24th International Cartographic Conference, International Cartographic Association, Santiago de Chile, Chile.

Weir, R., Sizemore, B., Henderson, H., Chakraborty, S., and Lazar, J. (2012). Development and Evaluation of Sonified Weather Maps for Blind Users. In S. Keates, P.J. Clarkson, P. Langdon, and P. Robinson (Eds.), Proceedings of CWUAAT (pp. 75-84). Cambridge, UK: Springer.

Wolfe, J.M., Kluender, K.R., and Levi, D.M. (2015). Sensation & Perception. 4th ed., Sinauer Associates.

Zeng, L., and Weber, G. (2011). Accessible Maps for the Visually Impaired. In Proceedings of IFIP INTERACT. Workshop on ADDW, CEUR, vol. 792 (2011), pp 54-60

Zhao, H., Shneiderman, B., and Plaisant, C. (2008). Listening to choropleth maps: interactive sonification of geo-referenced data for users with vision impairment. In: Lazar J, editor. Universal usability. New York: Hoboken John Wiley & Sons Ltd. Publishers. pp 141-174