GIVE-ME: Gamification In Virtual Environments for Multimodal Evaluation - A Framework

Wai Lun Khoo

A dissertation submitted to
the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy,
The City University of New York.

Committee Members:
Professor Zhigang Zhu
Professor Tony Ro
Professor Yingli Tian
Dr. Aries Arditi

August 30, 2016

GIVE-ME: Gamification In Virtual Environments for Multimodal Evaluation

- A Framework

by

Wai Lun Khoo

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Date	**Professor Zhigang Zhu** Chair of Examining Committee
Date	**Professor Robert Haralick** Executive Officer

Supervisory Committee

Professor Zhigang Zhu

Professor Tony Ro

Professor Yingli Tian

Dr. Aries Arditi

The City University of New York

Abstract

GIVE-ME: Gamification In Virtual Environments for Multimodal Evaluation

- A Framework

by

Wai Lun Khoo

Advisor: Professor Zhigang Zhu

In the last few decades, a variety of assistive technologies (AT) have been developed to improve the quality of life of visually impaired people. These include providing an independent means of travel and thus better access to education and places of work. There is, however, no metric for comparing and benchmarking these technologies, especially multimodal systems. In this dissertation, we propose GIVE-ME: Gamification In Virtual Environments for Multimodal Evaluation, a framework which allows for developers and consumers to assess their technologies in a functional and objective manner. This framework is based on three foundations: multimodality, gamification, and virtual reality. It facilitates fuller and more controlled data collection, rapid prototyping and testing of multimodal ATs, benchmarking heterogeneous ATs, and conversion of these evaluation tools into simulation or training tools. Our contributions include: (1) a ***unified evaluation framework***: via developing an evaluative approach for multimodal visual ATs; (2) a ***sustainable evaluation***: by employing virtual environments and gamification techniques to create engaging games for users, while collecting experimental data for analysis; (3) a ***novel psychophysics evaluation***: enabling researchers to conduct psychophysics evaluation despite the experiment being a navigational task; and (4) a ***novel collaborative environment***: enabling developers to rapid prototype and test their ATs with users in an early stakeholder involvement that fosters communication between developers and users. This dissertation first provides a background in assistive technologies and motivation for the framework. This is followed by detailed description of the GIVE-ME Framework, with particular attention to its user

interfaces, foundations, and components. Then four applications are presented that describe how the framework is applied. Results and discussions are also presented for each application. Finally, both conclusions and a few directions for future work are presented in the last chapter.

Acknowledgements

This work is supported by the U.S. National Science Foundation (NSF) through Award # EFRI-1137172 and the CUNY Graduate Center Science Fellowship (2009-2014), for my Ph.D. research. The work has also been supported in part by other grants, such as EFRI-REM supplement (for my mentoring and outreach activities), NSF CBET-1160046 and VentureWell (formerly NCIIA) through Award # 10087-12 (for my collaboration with the capstone class students in surveying and evaluation), and NSF IIP-1416396 (for support of my design for evaluation of VISTA Wearable).

I want to thank my advisor, Prof. Zhigang Zhu, for his constant support, advice, and encouragement. It has been an amazing 10-year journey with Prof. Zhu. I still remember the first piece of advice he gave me when I was in his undergraduate Data Structures class: "You may not be viewed as a good student if you don't know how to solve a problem, but it would be a very good first step if you at least understand why you cannot solve the problem." His research, teaching, mentoring, and collaboration styles have been imprinted on me ever since. And I hope to pass on some of what I've learned to my students as I begin my academic career as a professor.

I also want to thank my committee members, Drs. Tony Ro, YingLi Tian, and Aries Arditi, for their valuable and insightful comments and feedback in my dissertation work and helping me in pursuing my dream job of being a professor.

I also want to thank my colleagues and fellow students/candidates (former and current) Dr. Tao Wang, Dr. Hao Tang, Dr. Edgardo Molina, Ms. Farnaz Abtahi, Mr. Martin Goldberg, Mr. Feng Hu, Mr. Wei Li, and Mr. Greg Olmschenk, for their friendship and collabration, allowing me to brainstorm my ideas with them.

I also want to thank other collaborators both in research and in mentoring students, Dr. Yuying Gosser, Dr. Lei Ai, and Ms. Camille Santistevan, in helping me in more ways than I can count.

Finally, I want thank my mentees Mr. John Settineri, Mr. August Seiple, Ms. Takami Nishimoto, Mr. Joey Knapp, Mr. Joey Pan, and many other unnamed mentees, for helping me in various research projects, and most important of all, for teaching me how to be a better mentor.

To my parents and siblings, Albert, Doris, Lillian, and Jonathan.

To my spouse, Chris, without whom I wouldn't have made it this far.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In 2012, there were 285 million visually impaired people worldwide; 39 million of whom were blind, and the rest (246 million) had low vision. This is approximately a 77% increase from 2002 (161 million visually impaired people)[1,2]. As the world population ages, the number will only increase unless a significant breakthrough is found. In the United States alone, with aging baby boomers, the number will double[3]. Furthermore, as people get older, they will eventually become impaired, one way or another. The need for assistive technology (AT) for safe and efficient navigation, therefore, is and will be there. More importantly, the need for an evaluation system and usability study for AT, to assess its usefulness, to provide scientific measurement of ATs, and to establish benchmarks for heterogeneous systems is imperative. This information make visually impaired people (VIPs) who are looking for AT in the market make an informed decision [32].

[1]World Health Organization, "Magnitude of blindness and visual impairment", July 1 2015, `http://www.who.int/blindness/causes/magnitude/en/`

[2]World Health Organization, "Visual impairment and blindness", July 1 2015, `http://www.who.int/mediacentre/factsheets/fs282/en`

[3]National Federation of the Blind, "Blindness and Low Vision: Fact sheet", July 1 2015, `https://nfb.org/fact-sheet-blindness-and-low-vision`

1.2 Assistive Technology

Assistive (or access) technologies (ATs) include a number of products and services that aid a person who suffers from loss of autonomy (*e.g.,* visual, physical, hearing, *etc.*) [71]. This dissertation focuses on ATs that address navigation assistance for the visually impaired, since this is one of the biggest challenges of independence for VIPs [32]. In other words, we consider sensory substitution or *alternative perception* devices that transform visual information into non-visual signals (*e.g.,* auditory and haptic). We define alternative perception as using machines or devices to sense the environment and present the user with meaningful information about his or her surroundings, allowing the user to navigate the area. To realize the alternative perception, we must determine the kind of sensors (or combination of sensors) that are best suited as "input" devices. In addition, we must also address the inherited limitations of these sensors and neccessary compromised. For instance, infrared has limited sensing distance. An efficient and robust system must be able to present meaningful information to the user without overloading their senses. Sensory overload is one of the downfalls of current electronic travel aid (ETA) technologies [22]. The question boils down to which human senses can best be exploited for alternative perception without overloading the user.

There is a wide range of ATs available for visually impaired people ranging from simple to specialized, from commercial-off-the-shelf to medical. White canes and guide dogs are the most popular, albeit traditional. The white cane is the cheapest and simplest. The guide dog, however, is expensive and needs high maintenance [22]. Since the advent of remote sensing devices and computing technology in the 1960s, scientists and engineers are able to build specialized devices (*e.g.,* ETA, haptic gloves, *etc.*) for the visually impaired. These "alternative perception" devices effectively bypass the retina and optic nerve, and go straight to one's brain via other senses (*i.e.,* hearing and touch). Commercial ETA products

such as UltraCane[4], Miniguide[5], and Trekker Breeze[6] are common examples available to users to purchase. Another type of ATs is retinal prosthesis. Unlike alternative perception devices, retinal implants, such as Argus® II[7], convert incident light ray into electrical signals, and deliver the signals to retinal neurons other than photoreceptors, since it is generally damaged as a result of an eye disorder [87, 88]. The main advantage of retinal prostheses is partial vision restoration, albeit very low resolution (*i.e.,* depends on the electrode array's dimension).

1.3 Problem Statement

Many visual AT designs and a few reviews can be found in the literature [22, 49]. These ATs are categorized into one of following three approaches [49]: physical [1, 8, 10, 12, 16, 37, 45, 56, 58, 68, 72, 75, 76, 91], digital [17–19, 24, 25, 36, 39–41, 46, 51, 62, 79, 83], or medical [28, 34, 42, 44, 53, 57, 64, 81, 89, 93, 94]. The physical category includes tangible devices that users can interact with. Whereas, the digital group comprises software programs that are mostly used as training tools (*e.g.,* preplanning aids). Medical devices are anything related to retinal prosthesis, which could be an implant or simulation software. Despite all this work, there is a lack of *robust evaluation* and *scientific comparison* of the effectiveness and friendliness of ATs. Robust evaluation and scientific comparison enable us to establish benchmarks for heterogeneous systems, and provide informed decisions for consumers shopping for an AT. Often times, ATs are not compared with state-of-the-art systems or algorithms. Mainly because many ATs research systems are either dependent on specific hardware or not fully described,[8] they are not reproducible [24].

[4]Sound Foresight Technology Ltd., "Ultracane", July 29, 2015, `https://www.ultracane.com/`

[5]GDP Research, "The Miniguide mobility aid", July 29, 2015, `http://www.gdp-research.com.au/minig_1.htm`

[6]Humanware, "Trekker Breeze+ handheld talking GPS", November 15, 2015, `http://store.humanware.com/hus/trekker-breeze-plus-handheld-talking-gps.html`

[7]Second Sight, "The Argus® II Retinal Prosthesis System", November 5, 2015, `http://www.secondsight.com/argus-ii-rps-pr-en.html`

[8]The Economist, "How science goes wrong", November 17, 2015, `http://goo.gl/7qQx8Y`

Algorithm benchmarking is a common practice in many research communities. It is prominent in the computer science community, especially in areas such as computer vision and pattern recognition. We can find many national and international competitions that evaluate the performance of an algorithm for specific tasks. For example, there are competitions for video activity recognition[9], social touch gesture recognition[10], emotion recognition[11] [54], object recognition[12], biometrics[13] and multimodal learning and analytics[14]. These competitions allow developers to evaluate their algorithm and compare them to others, which is needed in AT design and development [48, 49]. We started the design of a virtual environment framework in [48]. We then proposed an evaluation matrix for various algorithms and systems in [49]. In this dissertation, instead of focusing on one particular task for data collection and evaluation, a unified framework and platform is proposed and designed for the general goal of mobility and accessibility of visually impaired people, so that developers can tailor the framework for evaluating their algorithms and systems for specific tasks.

1.4 Overview

In the following chapters, we present our multimodal evaluation approach, in addition to the state of the art, for testing visual navigational assistive technologies for the visually impaired. We first present the proposed framework and then demonstrate four (4) applications of the framework, with details on how each component of the framework is applied to the experiments. In developing the framework, we made the following contributions:

[9]University of Maryland, "CVPR Workshop: Activity Recognition Challenges", November 5, 2015, `http://www.umiacs.umd.edu/conferences/cvpr2011/ARC/`

[10]University of Twente, "Recognition of Social Touch Gestures Challenge 2015", November 5, 2015, `https://www.utwente.nl/ewi/touch-challenge-2015/`

[11]Australian National University, "The Third Emotion Recognition in the Wild Challenge (EmotiW 2015)", November 5, 2015, `https://cs.anu.edu.au/few/emotiw2015.html`

[12]ImageNet, "Large Scale Visual Recognition Challenge 2014 (ILSVRC2014)", November 5, 2015, `http://www.image-net.org/challenges/LSVRC/2014/`

[13]Biometric Recognition Group, "The 6th IAPR International Conference on Biometrics:Competitions", November 5, 2015, `http://atvs.ii.uam.es/icb2013/competitions.jsp`

[14]Society for Learning Analytics Research, "Fourth Multimodal Learning and Analytics", November 5, 2015, `http://sigmla.org/mla2015/`

C.1. **A Unified Evaluation Framework:**

- We developed an evaluation approach for multimodal visual ATs that can handle a variety of sensors and stimulators in addition to the common ones (*i.e.,* infrared, sonar, haptic, and audio). New simulation of sensors can be added, such as magnetic field, gravity, pressure, and thermal. Connecting the framework to new stimulators such as electrical stimulation array, olfactory emitter, and gustatory system is also possible.

C.2. **Sustainable Evaluation:**

- We employed gamification techniques to create fun virtual environments to engage users while collecting experimental data for analysis. In addition to laboratory experiment sessions, a designed virtual environment can be packaged and distributed as an online game. In this mode, data can be collected in a distributed manner and the game can also be used as a simulation or training tool to educate users, or for users to practice using a new AT.

C.3. **Novel Psychophysics Evaluation:**

- We designed a framework that enables researchers to conduct psychophysics evaluation involving navigation tasks. The framework not only allows researchers to collect data on subjects' performance and their subjective assessment of the technology, but also allows brain activity measurements such as EEG. Most importantly, the framework timestamps all data collected, thus all collection mediums are synchronized.

C.4. **Novel Collaborative Environment:**

- We created a framework that not only enables developers to rapid-prototype and test their navigational assistive technology, but also facilitates an early stakeholder

involvement that fosters communication among researchers, developers, and most importantly, users. Since part of the AT is virtually simulated, developers can try out different configurations and solicit feedback from users to refine their prototype in the next iteration. Such a collaborative environment results in a navigational AT that is human-centric.

The rest of this dissertation is organized as follows:

Chapter 2 presents a brief literature review of the state of the art methods in virtual reality, gamification, and multimodality with particular attention to evaluating visual navigational ATs.

Chapter 3 presents the concept and implementation of the Gamification In Virtual Environments for Multimodal Evaluation (GIVE-ME) Framework. First, we present the objectives of the framework. Then we describe each component of the framework in two groups: 1) user interface; and 2) foudations and components. The user interface includes controllers, multimodal stimulators, and measurement device. The foundation and components includes multimodal (virtual) sensors, game mechanics, virtual environment toolbox, and data collection. Lastly, we present its software implementation details and instructions on its usage.

Chapters 4, 5, 6, and 7, describe the application of the framework for three unique ATs (the fourth one is an improved version of one of the three). For each, we first present the experiment's parameters and goals. Then we describe in detail how each component of the framework is implemented for the experiment, followed by results and discussion.

Chapter 4 presents an experiment using Brainport from Wicab, which is a tongue-based electrical stimulation device that conveys the visual scene in front of the user. GIVE-ME is applied to evaluate the Brainport in a navigation task. We present how to build a simple virtual maze, generate navigational directions to reach a destination, send directions to the Brainport, and have users respond based on what they felt on their tongues. Lastly, we show their performance and trajectory data.

Chapter 5 presents an experiment using a smartphone-based crowd-sourced navigation solution. GIVE-ME is applied to simulate the smartphone and enable the evaluation of aggregation methods (of online crowd volunteers' responses) and feedback mechanism. We present how to build randomized mazes and stream a first-person view of the maze to the online crowd volunteers, who direct the avatar to a destination. The alternate mode of avatar control is also presented, in which feedback is given to a real user, who direct the avatar to a destination. Lastly, we show the crowd's performance and avatar's trajectory data.

Chapter 6 presents an experiment using an initial prototype of a wearable system, VISTA (Vibrotactile Intelligent System for Travelling Aid). We present how to simulate the configuration of the range-vibrotactile devices, mimicking the shirt with embedded vibrators that subjects wear. The range-vibrotactile devices convey distance information using vibration intensity. As the distance to obstacle gets smaller, the vibration intensifies. Thus, based on the tactile feedback the subjects felt, they had to avoid obstacles and reach a destination. For this experiment, we built a fictitious L-shaped hallway and a building floor level based on a real layout. Lastly, we present the subjects' performance and trajectory data.

Chapter 7 presents an experiment with the newest version of VISTA. The differences here are the modular design, wireless capability, and additional functionalities of the devices. Instead of wearing the shirt with embedded vibrators, the VISTA devices can be affixed on certain parts of the subject's body using Velco straps. Further, the main purpose of this experiment is to evaluate different configurations (placement and number) of the devices that enable efficient navigation. Lastly, we present the subjects' performance and trajectory data.

We conclude with chapter 8, which summarizes the GIVE-ME Framework, along with a statement of contributions. Then, we discuss the insights and lessons learned during the course of this dissertation work. Lastly, we present some future work left in framework improvement, determining metrics, brain data collection, and comprehensive benchmark.

Chapter 2

Related Works

To address the lack of tools on evaluating ATs, we propose a unified multimodal framework that allows for a formal evaluation and comparison using virtual reality and gamification techniques. Conducting a navigation experiment to evaluate an AT has safety concerns, especially for VIP subjects who are using a new ETA for the first time. Conducting navigation experiments in a virtual environment alleviates such concerns because the tasks and interactions are virtual. The subjects are seated in front of a computer. While an experiment involving repetitive tasks is tedious, gamification techniques can be employed to introduce an element of fun and engagement in the experiment.

2.1 Virtual Reality

In contrast to physical devices, using virtual reality (VR) approaches to design assistive technology can be completely or partially virtual based. Partially virtual is seen as an augmented reality where the user has to interact with physical objects, in order to progress in the virtual environment. Unlike physical devices, VR can receive input ranging from standard keyboard and mouse, body tracking, to interacting and manipulating other objects. Although haptic is a popular feedback mechanism, a VR approach uses a combination of haptic and audio feedback to provide an immersive experience [83]. Software is used to con-

struct an environment (based on reality or otherwise) for free exploration or task completion. A game environment can also be constructed to improve certain skills such as navigation and cognitive mapping [31]. We propose to use ***virtual reality/environment*** (VR/VE)[1] to take advantage of its benefits such as learning transfer, obtaining precise data, and to rapidly simulate ATs and task environments. Virtual reality may be vastly different from the real world, not only its interaction but also the digital content. Thus, users' experiences are different, which affects whether a specific skill is learned and transferred to the real world. These gaps are narrowed by having a realistic and authentic simulation of the real world. Simulations that are based on physical specifications (*e.g.,* range sensing sensor's profile). Furthermore, with intuitive and natural interaction (*i.e.,* gesture), the learned skills are more aligned with the real world.

Research has shown that general skills acquired in a virtual environment are transferable to the real world performing similar tasks, especially if the real environment is adequately represented in the virtual environment [90]. This is evident in driving and flight simulators, where subjects receive driving or flight lessons prior to utilizing a real vehicle or plane. Driving simulators are also used to test compliance of new signage at railroad crossings [52]. In addition to driving and flight simulators, recent developments use virtual reality as a training tool in medical education [80]. Medical students and residents are put through a series of simulator trainings for a medical procedure (*e.g.,* colonoscopy and laparoscopy) before entering an operating room with a patient. The results showed that VR training improved their surgical performance, especially with a warm-up VR training session prior to the scheduled surgery [20, 69]. Beside medical training, VR is also used as a simulation and educational tool to experience, for example, Parkinson's disease psychosis [33]. The advantage of using VR in this example, is the ability to educate, and it conveys to family and friends of patients what the patients are experiencing. While this is insignificant to Parkinson's patients, there are other VR training tools that benefit patients. For example,

[1]For the purpose of this proposal, VR and VE are used interchangeably and both means the same.

using VR to train autistic people in social skills [70]. Parsons and Mitchell used VR tools to allow participants to practice behaviors in a role-playing scenario. The VR not only provided a safe environment for rule learning, but also provided greater insight into the minds of people with autistic spectrum disorders.

Lahav *et al.* [50, 51] studied the exploration process of an unknown space in a virtual environment, the cognitive processing of it, and applying the knowledge in real space. The authors developed a multi-sensory virtual environment, the BlindAid system, using a Microsoft SideWinder Force Feedback Joystick as input and haptic feedback. The virtual unknown environment is based on a real space with randomly placed obstacles. Subjects are asked to freely explore the virtual environments without time limits, and then use building block components to map the layout. Finally, with the mental map in mind, the subjects are asked to perform orientation tasks (*e.g.,* reach and identify an object) in real environments. The only modular design is the input and output device (*i.e.,* the joystick), which can be substituted with any other haptic devices. The virtual environments, however, are not modular, since they are highly dependent on the real space layout. Their experiment results showed that the system is quite useful and easy to use, and helped the subjects navigate the real space easily. The BlindAid system is quite specific to the visually impaired because it helps them cognitively map an environment before visiting the actual space. This is advantageous to the visually impaired, because they do not have to constantly pay attention to their navigational aids and be alert to the surrounding ambient sounds.

Similar to previous works, Khoo *et al.* [46, 48] studied what combination of sensors and stimulators are best suited for multimodal perception. They designed a virtual floor on campus that includes virtualized infrared sensors connected to physical vibrotactile units. As a user uses a joystick (to control the avatar) to navigate the environment, virtual sensor readings are transduced to vibrotactile units, which allow the user to feel vibration as they get closer to an obstacle . Currently, the sensor configuration is on the arm; three on each arm. The goal of this system is to evaluate the actual sensor-stimulator units. Implementing

the sensors and display devices in a virtual environment not only reduces the cost and time in development (*i.e., rapid prototyping*), but also provides an opportunity to examine how the human brain responds to and processes stimuli using the devices, in a well-controlled environment that minimizes head movements. The system design is modular; developers can quickly implement a virtual sensor of one modality and connect it to a stimulator of another modality. Their experiment results showed that the system/unit is useful for the visually impaired; users can get some training in the virtual environment and then use it in the real world. Users find the system/unit intuitive and easy to use. The system/unit can be extended to other applications, such as search and rescue, or ones that require situation awareness. The current design is bulky and wired, but they have plans to make it more aesthetic, wireless, and wearable. The system/unit operates in real time and is built from commercial-off-the-shelf parts and quick software prototyping.

Similarly, Ai and Ro [4] studied tactile perception in response to a stimulus. We wish to, however, extend similar study for navigation using alternative perception. In a well-controlled VR environment, for example, it allows neuroscientists to collect functional neuroimaging (*e.g.,* electroencephalography (EEG) and magnetoencephalography (MEG)) data for navigation tasks, which traditionally was extremely difficult due to head movements [46, 48]. Furthermore, due to the rapid prototyping nature of VR, *stakeholders* (*i.e.,* users, researchers, practitioners, engineers, *etc.*) can be involved in the process early on. Thus, they increased the probability of producing an AT that is truly desirable by the targeted audience. Lastly, VR is an excellent *evaluation tool* before beta testing an AT with users.

In contrast to previous works, Hara *et al.* [36] designed an augmented reality to evaluate multimodal feedback strategies. In this work, the authors used a Wiimote[2] to provide audio and vibrotactile feedback to subjects. The Wiimote is attached to an aluminum stick (like a long cane) as a handle. On the other end of the stick is an active marker. Similarly, there is an active marker on a cap that subjects wear. The active markers are tracked by an optical

[2]Nintendo. "Wiimote", March 5, 2014, `http://www.nintendo.com/wii/`

tracking system and mapped onto the virtual environment which is rendered using OpenGL. As subjects get close to a virtual wall, the Wiimote will alert users via audio and/or haptic feedback. Even though the entire system can be expensive (mainly the optical tracking system), each of the components is modular and wireless. One can experiment with various feedback devices along with other approaches of tracking (*i.e.,* using a camera with computer vision technique) and rendering methods. Though the system is quite useful and easy to use for navigation for the visually impaired, it can also be extended for immersive gaming. The system is available to those who know how to set it up and ready for further experiments.

2.2 Gamification

Gamification is a fairly new concept in the research community. Gamification is the application of game mechanics and elements in a non-gaming context. Traditionally, laboratory experiments are *repetitive* and *tedious*. Incorporating gaming concepts enrich experiments by providing *fun* and *engaging* sessions [35]. This will encourage users to participate again, thus producing a sustainable evaluation and data collection [9, 55]. It is also a good motivator for child subjects [13, 78]. Game logic, along with scoring, can be used as objective assessment; evaluating every action users took in the game. Games can also be distributed to other users as a kind of crowd-sourcing data collection method [26]. Lastly, experiments can be nicely packaged into a *simulation or training tool*, since the VE is already designed. Some of evaluation procedures, though might have to be modified or removed before using it as a simulation or training tool [27].

MOVA3D is a game that helps visually impaired children improve their navigation skills [79]. A virtual environment is developed using Visual Studio .NET, C#, and Microsoft SDK development tools for XNA. The game receives inputs from a customized Digital Clock Carpet (DCC) or keyboard. With a keyboard, users navigate using the arrow keys. With DCC, users use the clock system to step in the direction they want to take, with 12 O'clock

always being in front of you. The spatial sound is relative to users' orientation. As evident, hardware component of the system is modular (DCC *vs.* keyboard); the software, however, is specific to Windows. Their experiment results showed that the system is useful and easy to use. Visually impaired children, furthermore, can play the game along with their sighted friends since MOVA3D GUI also renders normal textures for walls and floors. MOVA3D also has an option to render in high contrast, which makes it very accessible and promotes social interaction between sighted and non-sighted children. The system is relatively low cost since the customized haptic device is built from commercial-off-the-shelf parts and the game is developed in-house. The DCC needs a USB connection to the computer; use of a keyboard, however, can be wired or wireless. Overall, the system has good performance and it had captured subjects' interest from the beginning.

Connors *et al.* [19] developed an audio-based virtual environment called Audio-based Environment Simulator (AbES) that allows simulated navigation and exploration of an existing physical building. 3D audio is important in allowing subjects to perceive and localize a sound source in the virtual environment [65]. The AbES can render an environment from a building floor plan or virtual rendering of a modern building. AbES uses a keyboard as input and spatialized sound cues as feedback. AbES is conceptualized as a game. Users have to collect as many jewels as possible hidden in various rooms while avoiding roving monsters that can steal the jewels from them. This is an easy to use, yet modular, game, and it can load various floor plans for gameplay. This is beneficial for visually impaired people in improving their navigation skills and building a cognitive map of a place. It is also accessible to other people who are interested in exploring a new place, either for fun or in anticipation of a visit. Good physical navigation performance is achieved after some training in virtual navigation. The authors claim that the game is still in an initial phase of the study, but have plans to include a large-scale study.

2.3 Multimodality

We take it for granted that sensory data from disparate sources gives better results when compared to sensory data coming from a single source. In a multimodal platform, sensory data from one source can be verified or supported by sensory data obtained from another source. How does one know, however, whether the multimodal sensory data is beneficial in the design of ATs, since it is all up to users to interpret sensory inputs and make an informed decision based on it?

Multimodality is of particular interest because Proulx *et al.* [73] showed that our brain areas are not described by the source of inputs (*e.g.,* eyes, ears, *etc.*), but by the cognitive activation and physical type of processing required by a task [60]. Sometimes, this is called *cross-modal plasticity*. This implies that even though visually impaired people cannot see in the usual sense of the word, but they can still "see" in a way with their non-visual senses (*i.e., alternative perception*) [11, 77]. Furthermore, multimodality allows for a flexible framework setup such that any multimodal sensors (inputs) can be paired with any multimodal stimulators (outputs). This enables the study of how best to transduce information to VIP, through the use of combinations of sensors and stimulators. The following shows examples of sensors and stimulators being used in AT.

Palmer *et al.* designed wearable range-vibrotactile units by placing range sensors (*e.g.,* infrared) coupled with a haptic unit (*e.g.,* vibrator) throughout a user's body. The units allow users to "feel" objects and obstacles in close proximity without having to physically touch them. The tactile feedback is achieved via vibration intensity such that the vibration intensifies as the distance of the user to an object or obstacle decreases. Due to the limited field of view of sensors, the authors envisioned having sufficient number of units placed on the body such that it has complete coverage of users' surroundings [68]. The work is of modular design since each unit can be placed on any body's location, and the sensors and stimulators are interchangeable. With its simplistic design, it is functionally acceptable, easy to use, and in real time. The units can also be expanded to the general population since its application

is not limited to the visually impaired. With a bit of fashion design, it can be aesthetic and unobtrusive as well. The units are designed from cheap commercially-off-the-shelf parts. For the current implementation, as reported in Palmer *et al.* [68], each unit is wired to each other and to a microcontroller. Despite that, the unit is fully implemented, available, and has overall good performance.

Another sensory substitution device that is similar to Palmer *et al.* is the Tactile-Sight (TSIGHT). This device receives input from a Microsoft Kinect that is attached to the torso with a belt. The depth data from Microsoft Kinect is then processed, capturing the distance of objects in one's surroundings. Through the XBee wireless device, commands are sent to an inter-integrated circuit network (control module) that receives activation intensity for each stimulator, and resends the information to the corresponding stimulators. In Cancar *et al.* experiment, 72 vibrotactile stimulators are housed in a waistband placed on the abdominal area of users. The stimulators are arranged in 6 rows with 12 columns [16]. The only modular design in the system is their power and control modules. This system is friendly, real time and easy to use, as evidenced by results of the experiments where participants can perceive incoming objects and predict when it will impact their respective bodies. This system has the potential to be extended to the general population, especially to those who need situation awareness such as first responders. In their current implementation, however, the device is bulky and requires users to carry a backpack. In addition, it is socially awkward carrying a Microsoft Kinect on your waist. Part of the system is also wireless (*i.e.,* going from XBee wireless device to I^2C module), while other parts are not (*i.e.,* Kinect to the processing unit and I^2C to stimulators). The system is constructed from low cost, commercially-off-the-shelf parts such as Microsoft Kinect, coin-motor type vibrators, I^2C module, and XBee wireless devices. Even though the current version is available and has overall good performance, the authors wish to improve wearability [16].

In 2012, a group of researchers from Lebanon designed a prototype of rehabilitative shoes paired with eyeglasses for the blind [1]. The devices are used to detect obstacles at ground

and head levels, and include stair ascent and descent; situations which traditional white cane users often have trouble. They achieved this in two parts. First, they retrofitted each shoe with three pairs of ultrasonic transducers (transmitter and receiver) placed on the toe cap with spacing in between (in a way that allows maximum ultrasonic beam coverage for detecting ground level obstacles of various height as well as pits and holes). Vibrating motors are attached inside the shoes underneath the corresponding locations of the transducers. For the second part, they retrofitted a pair of ultrasonic transducers above the bridge of a pair of Ray-Ban eyeglasses, and a buzzer is mounted at one of the temples. They used a microcontroller-based belt pack unit to control each pair of transducers. While this is an interesting approach to assist the visually impaired in navigation, their design is not modular because it is specific to the shoes and spectacles (though the sensors and stimulators could be of different types). This design is useful in navigation, especially in situations such as detecting potholes and knowing when you're at the top (or bottom) of stairs. We imagine that with some training, it would be easy to use the shoes. We are not sure about having a buzzer (or any kind of vibrating units) at one's temple, as that could be annoying, and one may be sensitive to having a device on the head. We believe this design is accessible to other populations such as the mobility-challenged. The shoes also help to maintain a proper gait. In its current implementation, with wires connecting each component and a belt pack unit, it has a low social integration value. It is also characterized by low cost, functioning in real time, and availability.

A long cane is one of the trusted tools that visually impaired people use to navigate known or unknown environment. While the cane is useful to detect different objects from the ground up to the waistline, it fails to detect those above the waistline. To address this limitation, Ramirez *et al.* [76] used ultrasonic technology to develop a device to achieve better spatial exploration. This device, an electronic long cane, is realized by retrofitting electronics to a traditional long cane's handle. The embedded device consists of an ultrasonic sensor, a micro-motor (vibration stimulator), a microcontroller, and a 9-V battery, all of which

are easy to purchase. The device is not of a modular design because users cannot easily substitute one component for another. It is, however, self-contained and wireless. When an obstacle is detected by the ultrasonic sensor, a haptic response (*i.e.,* vibration) is triggered inside the cane. The vibration becomes increasingly frequent in real time as users approach an obstacle. The device is also easy to use and useful to the visually impaired. This device is not accessible to other population groups, however, because it has little or no use to non-visually impaired people. The long cane is a common tool among the visually impaired, and it is not unusual to see one walking with a cane. With a simple retrofitted device to the long cane, it is easily "camouflaged" like a regular long cane. The device is available now and has satisfactory performance; users were able to bypass obstacles and resume their original routes with ease.

Khan *et al.* [45] developed an obstacle avoidance navigation system using Xtion Pro[3]. Like Microsoft Kinect, Xtion Pro uses depth sensing sensors, such as infrared, to obtain a depth map of the scene, while the device is strapped to a user's waist. Using the depth map, the system computes distance to nearest colliding obstacles, if any. If there is a colliding obstacle, the system computes the direction (far-left, left, middle, right, far-right) to an obstacle-free path. The direction is conveyed using text-to-speech. The system is not modular and requires depth sensing sensors. Furthermore, users have to carry the sensor on their waist, which reduces its aesthetic value. Although the system is easy to use and somewhat useful in helping visually impaired people navigate an environment in real time, its design is bulky because users need to carry a laptop backpack with wires directly connecting to Xtion Pro. Initial results showed good promise of its performance and the system is available for further experimentation.

Polacek *et al.* [72] used a WOz testing approach to develop their WOz Prototype, which is part of the Contextual Wizard (ConWiz) system. The WOz prototype is an Android smartphone application that is capable of synthesizing voice commands for navigation and

[3]ASUS, "Xtion pro live", March 3, 2014, `http://www.asus.com/Multimedia/Xtion_PRO_LIVE/`

controlling a vibra-wristband for tactile commands. The human wizard is controlling a separate device, Mobile Wizard, that sends navigational and vibration commands to the WOz prototype. Both the Mobile Wizard and WOz prototype are connected to a server that has data and an evaluator which monitors the experiment progress. The wizard also does note-taking on the device to log important events. The vibration is used to alert users about any dangerous situation. The choice of employing the WOz testing approach is to simplify the design and focus on investigation of navigational commands and usability issues. It is easy for anyone to use a smartphone to navigate an environment, especially in urban settings. Although all components are wireless, the server (which is a laptop) is carried by a third person among them. Because the system needs a human wizard to give navigational commands, it is currently not available. The study, however, received positive feedback about the usability of the Mobile Wizard.

Table 2.1 summarizes the aforementioned multimodal assistive technologies. The proposed evaluation framework enhances and complements these technologies. For the range-vibrotactile, the framework allows user study in a safe and controlled environment, and gathers valuable feedback for the next design iteration. More details about this are in Chapter 6. Similar to the range-vibrotactile, this framework provides a safe (albeit virtual) navigation environments for TSIGHT, Khan *et al.* system, and the rehab shoes and glasses. Virtual sensory data can be generated and fed to the TSIGHT's vibrotactile array placed on one's abdominal area to study the efficacy of the system, without the need of a bulky system. Better yet, they do not have to rely on a Microsoft Kinect, but instead can try out a variety of depth sensing units. Similarly, for Khan *et al.* system, virtual directions can be generated and transduced to users. For the rehab shoes and glasses, there are risks in user testing, where blind subjects can accidentally hit an overhang object or trip over gaps. Using our proposed framework for testing, we eliminate such risks because the subjects will be safely seated in a lab, navigating in a virtual environment. Our framework complements the electronic long cane because the framework not only provides an evaluation platform, but

Table 2.1: Multimodal ATs

Devices	Sensors	Actuators	Processing Units	Functions
Range-vibrotactile [68]	infrared	vibrator; whole body adjustable	linear mapping between distance and sensor reading	As user nears an obstacle, vibration intensifies
TSIGHT[16]	depth	vibrator; 72 units placed on abdominal area	integrated circuit	Capturing distance of objects and convey it via an array of vibrators on waist
Rehab shoes and glasses[1]	ultrasonic	vibrator / buzzer; 3 pairs on toe cap & 1 pair on glasses	micro-controller	Detect gaps on grounds and obstacles on or above head level
Electronic long cane[76]	ultrasonic	vibrator; 1 unit embedded in handle	linear mapping between distance and sensor reading	As user nears an obstacle, vibration intensifies
Khan *et al.* [45]	depth	text-to-speech	using depth map to compute distance to nearest colliding obstacle	compute direction to an obstacle-free path
ConWiz[72]	smartphone	text-to-speech	WOz prototype	synthesizing voice commands for navigation

also a simulation and training tool. Different types of ultrasonic sensors can be simulated with obstacles of various heights to see which sensors work best. New users can use the simulation tool to try out devices before purchasing and outfit them onto their long canes. In term of the Wizard of Oz testing approach, our framework is similar to ConWiz. Both involved orchestrated data. Our framework can achieve the testing scenarios in ConWiz in a virtual environment, thus reducing testing and setup times, and alleviating safety concerns.

Chapter 3

GIVE-ME: A Multimodal Evaluation Framework

This chapter presents our Gamification In Virtual Environments for Multimodal Evaluation (GIVE-ME) framework. Chapter 3.1 discusses the objectives of this framework. The user interfaces and foundations of this framework are described in Chapters 3.2 and 3.3, respectively. Lastly, the framework's intended uses and advantages are presented in Chapter 3.4.

This proposed framework differs from Degara *et al.* [24, 25] such that our framework allows for additional transducing methods, in addition to sonification. Furthermore, it differs from Lahav *et al.* [50, 51] and Huang [38, 39] in that their work was focused on cognitive mapping in unknown space, and they studied how best to utilize haptic and audio cues for spatial information in collaborative environments. Conversely, our framework focuses on evaluating multimodal ATs for navigation and studies what combinations of sensors and stimulators in ATs support optimal navigation performance.

3.1 Objectives

The primary goal of this framework is to provide a robust evaluation and scientific comparison of navigational ATs. More specifically, this framework enables the following:

- Benchmark heterogeneous ATs.
- Study of how best to transduce information to VIPs, through the use of combinations of sensors and actuators.
- Rapid-prototype and testing of sensors and stimulators before integrating into AT.
- Early stakeholder involvement.
- Psychophysics evaluation on navigation tasks.
- Fun, engaging, and motivating experiment sessions.
- Conversion of an evaluation tool into a simulation or training tool.

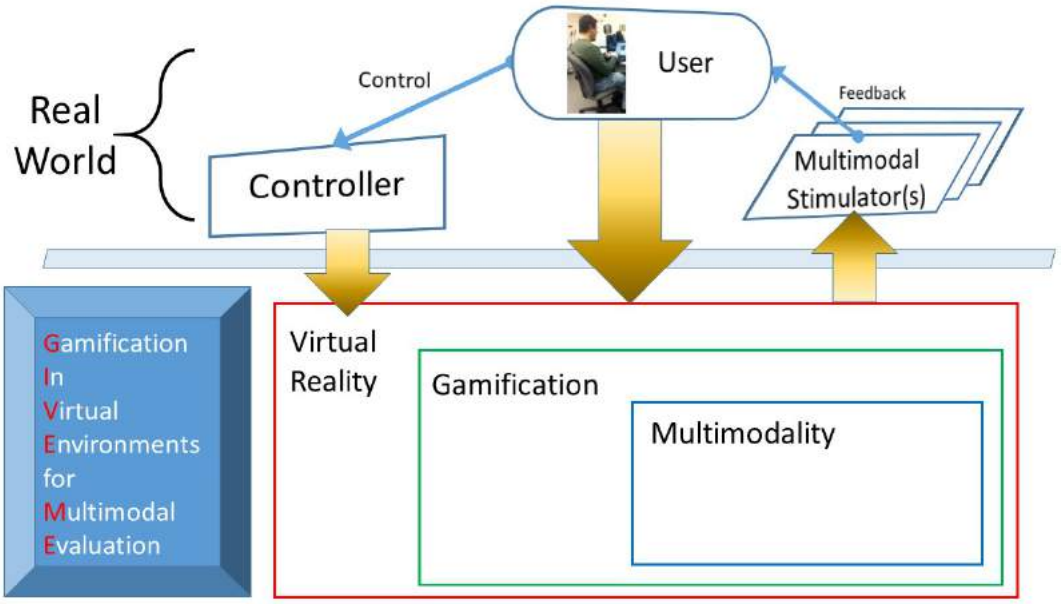


Figure 3.1: **G**amification **I**n **V**irtual **E**nvironments for **M**ultimodal **E**valuation Framework

The proposed GIVE-ME Framework - **G**amification **I**n **V**irtual **E**nvironments for **M**ultimodal **E**valuation - is illustrated in Figure 3.1. The framework is described in two parts. The top part, residing in the real world, is the framework's **User Interface** (including controller, multimodal stimulators, and measurement device). The bottom part, residing in

a virtual world, is the framework's **Foundations and Components** (including multimodal sensors, game mechanics, data collection, and VE toolbox). Briefly, a user guides a virtual avatar from point A to B via a controller. As the avatar is moving in the virtual world, virtual sensory measurement of nearby environment information is transduce to the user via a multimodal stimulator (or a set of stimulators). The following two sections detail the framework in two parts.

3.2 Framework: User Interface

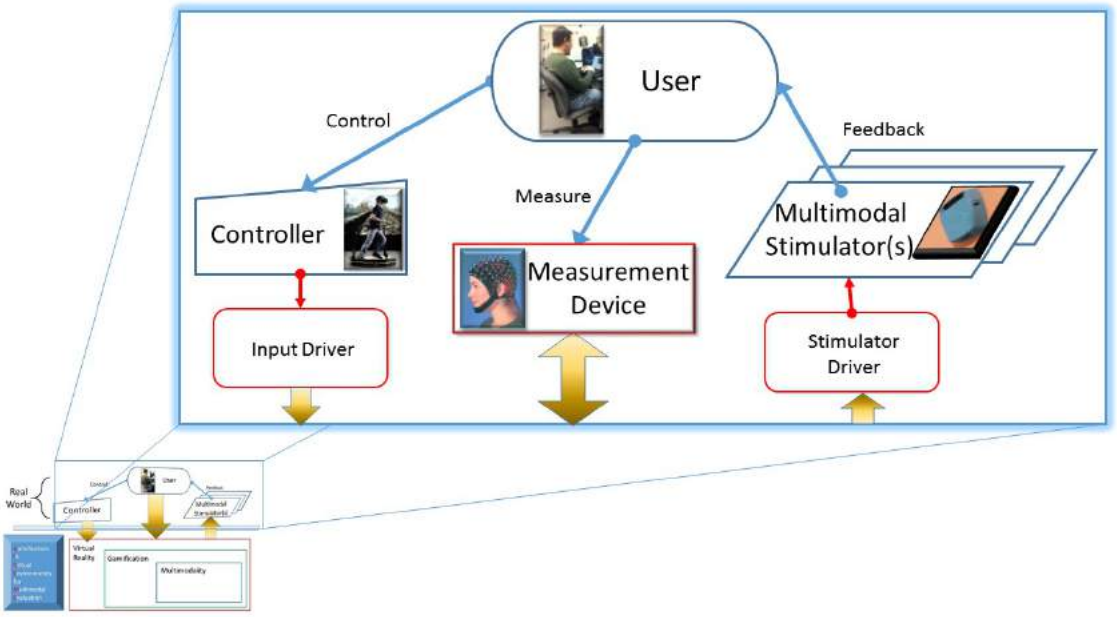


Figure 3.2: User Interface

Figure 3.2 shows the user interface of the framework. It consists of the following:

1. Controller: an input to the VE
2. Multimodal Stimulator(s): an output mechanism to the user
3. Measurement Device: to collect user's brain/behavioral data

Using a ***controller*** (see Chapter 3.2.1), a user controls a virtual avatar while receiving feedback about a VE through a set of ***stimulators*** (see Chapter 3.2.2). Specifically, control

data are passed from the controller to the VE via the input driver, and VE data (*e.g.,* virtual sensor range readings, see Chapter 3.3.1) are sent out to the stimulators via the stimulator driver. The VEs in this dissertation are built using Unity3D[1], a game engine (see Chapter 3.3.3). Like any hardware, drivers (*i.e.,* input and stimulator drivers in Figure 3.2) are needed to operate the hardware. Drivers are installed natively on the host machine via plug-and-play or as third-party programs that provide interfaces to the hardware. In the following sections, we discuss what types of controllers and stimulators are available or have been used by other people. Then we discuss which controllers and stimulators GIVE-ME supports and how we connected them to VEs.

Psychophysics evaluation can also be conducted at the same time through the ***Measurement Device*** (see Chapter 3.2.3). This component is meant to be generic in such a way that in addition to quantitative data, qualitative data (*e.g.,* survey and questionnaire) can also be collected, whether during an experiment (online), or after an experiment (offline). Offline data are then aggregated with online data for further study.

3.2.1 Controllers

A *controller* is a mechanism to navigate an avatar inside VEs. Traditionally, controllers are keyboard, mouse, joystick, and game pad (*e.g.,* Xbox 360 Controller[2]). In order to have an immersive and sustainable experience, however, we suggest using more intuitive controllers such as Oculus Rift VR[3] and Virtuix Omni[4], in conjunction with a game pad controller (Figure 3.3). Arguably, the game pad controller can be substituted with other gesture recognition sensors such as Leap Motion [5] and Kinect for Windows [6]. They need to have 360° coverage, however, because Virtuix Omni is an omnidirectional treadmill that

[1]Unity Technologies, "Unity3D", November 5, 2015, `http://unity3d.com/`

[2]Wikipedia, "Xbox 360 Controller", November 5, 2015, `https://en.wikipedia.org/wiki/Xbox_360_controller`

[3]Oculus VR LLC, "Press Kit", September 7, 2016, `https://www.oculus.com/press-kit/hardware/`

[4]Virtuix, "Press", September 7, 2016, `http://www.virtuix.com/press/`

[5]Leap Motion Inc, "Leap Motion", November 5, 2015, `https://www.leapmotion.com/`

[6]Microsoft, "Kinect for Windows", November 5, 2015, `https://dev.windows.com/en-us/kinect`

automatically generates input data of both rotation and translation of users to control the rendering of a VE. Therefore, a game pad controller is sufficient for experiments. While we understand the Oculus Rift is a visual experience that blind individuals cannot experience, it can still be used for low-vision users (with enlarged visual content). Furthermore, Oculus Rift has state-of-the-art head trackers that can be leveraged to simulate realistic 3D and binaural sounds. With gamification, particularly using realistic 3D and binaural sounds and the head tracker, users (players) will have a fuller and realistic immersion in virtual environments.

(a) Oculus VR[3]

(b) Virtuix Omni[4]

(c) Xbox 360 Controller[2]

Figure 3.3: Example of controllers

Connecting controllers to VEs happens in two levels: 1) operating systems (OS) and 2) application. In the OS level, most standard hardware (*e.g.,* keyboard, mouse, and joystick) are connected to the PC via plug-and-play. In other words, drivers for these devices are installed automatically, without user intervention, once plugged in. Other (non-standard) hardware such as game pad, Oculus Rift VR, Virtuix Omni, Leap Motion, Kinect for Windows, *etc.*, may require manual installation of drivers and a runtime environment. For example, in order to use the Kinect, the Kinect for Windows SDK 2.0[7] needs to be installed. The SDK includes a driver for the hardware, APIs, and code samples.

Once the controller hardware is connected, installed, and recognized by the OS, its functionalities can be accessed via applications. For the purpose of this dissertation, we want

[7]Microsoft, "Kinect for Windows SDK 2.0", November 23, 2015, `http://www.microsoft.com/en-us/download/details.aspx?id=44561`

to access the controllers via Unity3D. The advantage of using a game engine is that it provides a uniform API to access the connected devices. For example, Unity3D has an Input class[8] that allows developers to access the state of the mouse/joystick ($GetAxis$), buttons ($GetButton$), and keys ($GetKey$). Connecting non-standard hardware to Unity3D, however, may require an additional step. Unity3D has expanded their development to include integrated support for Oculus, thus, using Oculus in Unity3D does not require an additional step beyond installing its driver and runtime. On the contrary, Kinect for Windows and others are not directly supported by Unity3D. To access these devices, custom programs are needed to provide interfaces between devices' APIs and Unity3D. To connect Kinect for Windows to Unity3D, for example, we developed a server-client program. The server and client communicate with each other via a TCP/IP socket (*i.e.,* localhost) with a specified port number. The server program is written in C++ using Visual Studio and is connected directly to the Kinect, while the client program is written in Unity3D C# as part of the VE. Starting the server program first, its task (as defined by the programmer) is to send skeleton joint data (*i.e.,* set of X-Y coordinates for each joint) upon a client request. Of course, other data can also be sent depending on what is needed in the VE. The simplest way to send data from server to client is to encode the data as a custom string, for example, "joint#, jointX, jointY, joint#, . . . , jointY/". Finally, the client program task is to send a request to the server whenever new data are needed, and to parse (*e.g.,* split by comma) the received string. The server-client model also eliminates the issue of compatibility because the Kinect APIs require a .NET 4.5 Framework, but Unity3D only supports up to .NET 2.0.

3.2.2 Multimodal Stimulators

This section covers the various ways of conveying information to users of ATs through actual stimulators. For sighted users, information is generally conveyed *visually.* While a picture is worth a thousand words to sighted users, it is almost worthless to the visually impaired and

[8]Unity Technologies, "Input", November 23, 2015, `http://docs.unity3d.com/ScriptReference/Input.html`

thus they require *alternative perception* mechanisms to perceive their environment. Furthermore, it has been shown that VIPs preferred audio (*i.e.,* speech) for navigation. This frees their hands for other tasks, especially for those who use a white cane or guide dog [5].

For each of the following subsections, stimulators that are supported by GIVE-ME is listed: sound cue generation, text-to-speech (TTS), vibration motors, and Brainport. Sound cue generation is interfaced via a host's OS (and then via Unity3D audio class). TTS is interfaced via a plugin. Vibration motors are interfaced via serial port/Arduino. Brainport is interfaced via a TCP/IP socket (localhost). Interfaces to these stimulators are described in details in the individual subsections.

3.2.2.1 Audio devices

The most common type of stimulators is audio devices. Audio is used to provide feedback for action performed, whether correct or an error [79]. Sonification and visualization using sound have been studied widely for decades. The aim is to display, for example, graphics, line graphs or even pictures using non-speech sound [31]. Furthermore, sound synthesis can be used to convey color; for example, by playing a sine wave to convey gray color or by creating a tremolo effect to convey redness [8]. Talking Points 3 (TP3), a smartphone-based system, uses TTS technology to convey information to users about nearby points of interest in an environment [91]. Khan *et al.* [45] developed an obstacle avoidance navigation system that also uses TTS technology to convey direction to users.

GIVE-ME supports sound cue generation and text-to-speech. Both types of audio require a real speaker to transduce the sensory information. For a sound cue generation, Unity3D provides a class, AudioSource[9], that represents an audio source in 3D space. The audio source also contains an audio clip (*e.g.,* "ouch" and footstep sounds) that is played when triggered. Text-to-speech, however, is not as trivial as a sound cue generation. Instead of

[9]Unity Technologies, "AudioSource", November 23, 2015, `http://docs.unity3d.com/ScriptReference/AudioSource.html`

writing a TTS engine from scratch, we used a Win32 TTS plugin[10] to provide an interface. For other OS types, different versions of TTS plugin are needed. The plugin provides a set of interfaces to control the host's TTS. To make a speech, we call a function with a string of text, and the PC will pronounce the sentence.

3.2.2.2 Haptic/tactile devices

Tactile modality is another way visually impaired players acquire game information [19, 31]. Using tactile stimulation as a way to convey information is ubiquitous; for example, cellphones vibrate to indicate incoming calls or messages. Haptic devices are generally used to convey direction or distance to an obstacle. For example, SpaceSense uses an array of 3×3 vibration motors to give a cardinal direction for users to walk towards [92]. Vibration motors are also used to help users walk in a safe path by providing corrective feedback [89]. Vibration motors as distance stimulators are generally activated by increasing vibration intensity as one gets closer to an obstacle. This is achieved with Pulse Width Modulation (PWM) control [68]. Because of its small size, it can fit densely on different parts of our skin such as the chest, the ankles, and the wrists [16], giving the illusion of a whole body "range field" [46, 68]. In a classroom setting, Quek and Oliveira (2013) use a haptic glove to allow visually impaired students to follow the instructor's deictic focus [75].

GIVE-ME also supports other types of haptic devices, in addition to vibration motors. Vibration motors come in various forms, from unpackaged motors to motors integrated onto a circuit board. Unpackaged motors mean motors that at a minimum need to be connected to a breadboard, such as Arduino (which is connected to a PC via USB), using jumper wires to control them. For these type of motors, we control them by sending data voltage values through the connected serial port to the motors. The voltage values are determined based on sensory information to be transduced. For motors that are integrated onto a circuit board,

[10]Unity Technologies, "Text to speech DLL for Win32", November 23, 2015, `http://forum.unity3d.com/threads/text-to-speech-dll-for-win32.56038/`

if the board has a network module such as Bluetooth, voltage values can be written to the motors via the Bluetooth communication protocol.

3.2.2.3 Electrode devices

Another modality that has been used to convey information to VIPs is electrical stimulation. This is common in medical devices such as Argus® II [2, 3, 21], where a 60-electrode array is placed on the epiretinal, and electrical signals are sent through the optic nerve. Another similar device is the Brainport[11], where a 400-electrode array is placed on the tongue, and users have to feel the electrical stimulation to perceive the scene in front of them. More invasive devices, such as MVG Cortical Visual Prosthesis [53], place an implant in the brain, directly stimulating the visual cortex.

For electrode devices, GIVE-ME supports the Brainport's electrode array. We have not tested the framework with other implanted devices such as Argus. For the electrode array, we used a server-client model to transduce the sensory information. The server and client programs are communicating with each other via a TCP/IP socket (*i.e.,* localhost) with a specified port number. Specifically, we set up a VE as a server program. That is, the VE is generating sensory information such as direction to be transduced. The client program is directly connected to the Brainport. It receives information from the server, which needs to be parsed, and generates 20×20 pixel images to be sent to the 400-electrode array.

3.2.3 Measurement Device

To enable psychophysics evaluation of navigation tasks, this Measurement Device allows us to connect to devices such as an EEG to collect brain activity measurements, in addition to other qualitative data such as questionnaires and observations. A custom procedure is needed in order to connect to EEG machines because the machine we used has a parallel port interface, and such interface is neither provided as a standard library in Unity3D nor in

[11]Wicab Inc, "Brainport", November 5, 2015, `http://www.new.wicab.com/`

C#. Most systems now use USB. There is also a commercially available wireless and mobile EEG technology called EMOTIV[12]. To resolve this, a parallel port plugin[13] is used to enable such interface. This also means that a VE needs to be deployed in a machine that has a parallel port. Once connected, the EEG device needs to be synchronized with the VE such that when a stimulus is presented (from the VE), we expect the EEG measurements from that time forward to be a response to that stimulus, until a new stimulus is presented. To synchronize, a trigger (8-bit unsigned integer) is sent to the EEG device. A trigger "tells" the EEG what stimulus has been presented, and marks the measurement accordingly. For example, in navigation, the VE tells a subject to turn left or right. The trigger values are chosen as 128 (binary: 10000000) for a left turn, and 64 (binary: 01000000) for a right turn. Finally, the trigger is sent by calling the plugin's $DlPortWritePortUshort$.

Furthermore, a Measurement Device is not restricted to a mechanical device. The device can simply be a person (*i.e.,* experimenter or researcher) observing subjects, collecting behavioral data and conducting post-experiment questionnaires. For more details on what types of data GIVE-ME can collect, see Chapter 3.3.4. To distinguish the two types of measurement devices, we shall call the former type (*i.e.,* with a mechanical device) "online measurement," and the latter "offline measurement." Online measurement is connected and synchronized with the virtual world, such that the virtual world triggers the measurement device to collect data in response to an action. Offline measurement has to be aggregated and synchronized, if necessary, to online measurement. With the aggregated data, we can evaluate the task performed and establish ground truth. In addition, these data can also be used for man-machine learning and modeling to derive a better sensorimotor model. The goal of the model is to provide a better insight of the neural mechanisms of sensorimotor integration and motor learning, leading to new design concepts, formulation of required information, and development of cost-effective and revolutionary human-friendly mechatronic devices to assist VIPs.

[12]EMOTIV Inc., "EMOTIV", August 31, 2016, `http://www.emotiv.com`

[13]Logix4u.net, "Inpout32.dll for Windows", November 23, 2015, `http://www.logix4u.net/index.php`

3.3 Framework: Foundations and Components

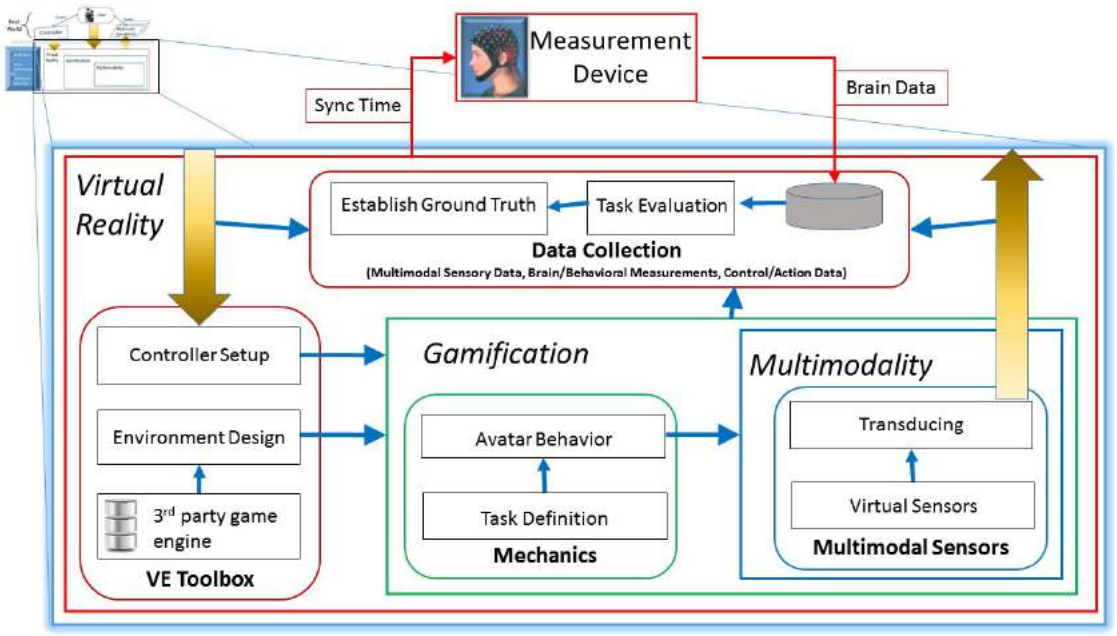


Figure 3.4: Foundations of the framework

Figure 3.4 shows the foundations and components of the framework. It consists of the following components:

- Virtual Reality
  - VE Toolbox
    - Third party game engine: building block for VE.
    - Environment Design: setup VE and layout for experiment.
    - Controller Setup: interface to controller and filter inputs.
  - Data Collection
    - Task Evaluation: collect quantitative and qualitative data.
    - Establish Ground Truth: contribute above data to a benchmark.
- Gamification
  - Mechanics

        * Task Definition: experiment objectives and game logic.
        * Avatar Behavior: design avatar to enable task completion.

- Multimodality
    - Multimodal Sensors
        * Virtual Sensors: virtualize sensors based on real specification.
        * Transducing: convert virtual sensor output data of one modality to feedback of an other modality.

The three foundations in this framework are virtual reality, gamification, and multimodality (Chapter 2) and they are the bedrocks that achieve the objectives of this framework. Each foundation also has components, and each component is exemplified by the foundation that it is in. The components are the building blocks, with specific tasks, in this framework. While a component is pigeonholed into a foundation, each component also reflects the spirit of the other two foundations that it is not in. For example, even though the task definition is in the Gamification foundation, it also relies on the Virtual Reality foundation to be realized and collaborated with the Multimodality foundation to allow subjects to achieve tasks.

The following sections discuss how the interaction of these components help to achieve the objectives of GIVE-ME. For the sake of clarity, we discuss each component in reverse order. First, we discuss how to virtualize a multimodal sensor. Then, we discuss how to define mechanics in a game. Lastly, we discuss how the VE toolbox provides an environment (*i.e.,* a game), and what data are collected for analysis.

3.3.1 Multimodal (Virtual) Sensors

Classically, sensors and stimulators are considered as user interfaces or human-computer interface. However, we also subscribe to the novel definition of *alternative perception* as defined in Khoo *et al.* [46, 48]; that is, to use devices of multimodalities to sense the user's

surroundings (such as camera and RGB-D sensor), and present spatial information to the user non-visually (*e.g.,* auditory, haptic, electrode). Thus, "interface" and "alternative perception" are interchangeable in this dissertation. In this section, we describe some common sensors, how to virtualize these sensors based on their real specifications, and how these *virtual sensors* data are transduced into *real stimulation* to the user.

3.3.1.1 Virtual Sensors

The keyword here is "virtual," where we leverage the power of virtual reality to *simulate* multimodal sensors based on real ones. Real sensors generally have some physical limitations such as noisy data. Simulating real sensors in VEs allows developers to control the limitation by introducing a known noise signal (*i.e.,* Gaussian noise with known means and standard deviations) into the noise-free data. Sensors also include conventional input devices such as a keyboard, a mouse, and a touchscreen. This dissertation focuses on some common multimodal sensors that can be virtualized as part of the GIVE-ME framework. The following sections discuss some of the common sensors, including but not limited to, infrared and sonar range, and RGB-D sensors.

Infrared range sensors

Infrared range (IR) sensors are light-based sensors, with a very narrow beam angle (a few degrees) and a short range, about a meter. For an IR sensor that can detect objects up to 80 cm away, it has beam width of 12 cm (spec data obtained from Acroname[14]), which is about 8.5°. An IR sensor also has a minimum sensor range (*e.g.,* 10 cm); this is where an object is so close to the sensor that it cannot get an accurate reading. It may be misconstrued as very far away. While the narrowness of the beam is an advantage in reading sharp details with high accuracy, the major issue is if it is not pointed exactly at the object, the object is invisible. If two IR sensors are so close together that their beams overlap, it will create

[14]Acroname, "Sharp GP2Y0A21YK0F Wide Beam IR Distance Sensor Kit", December 14, 2015, `https://acroname.com/products/SHARP-GP2Y0A21YK0F-IR-PACKAGE?sku=R301-GP2Y0A21YK`

cross-interference. This is when a signal emitted by one sensor is read by other sensors, and therefore produces erroneous readings.

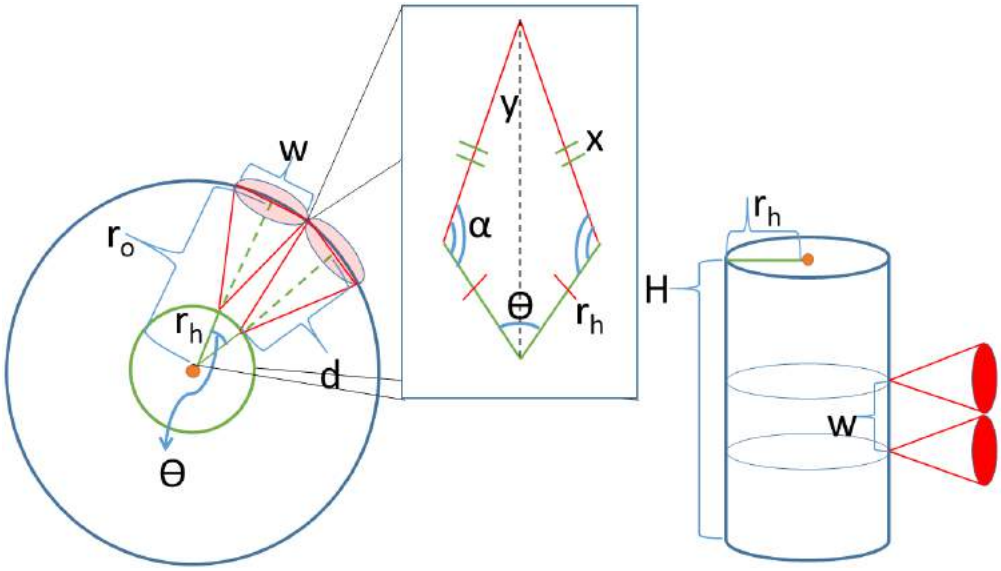


Figure 3.5: Estimation of number of infrared sensors on the body

In Figure 3.5, we illustrate the arrangement of infrared sensor such that there is no overlapping coverage. We model the person as a cylinder of r_h cm radius and H cm height, and the sensors are oriented perpendicular to the cylinder. Using this model, we theoretically estimated the upper bounds of the number of infrared sensors that can be placed on a human body. Given a height H and a maximum beam width of sensor w, there are $\frac{H}{w}$ rings of sensors without vertical overlaps (right side of the figure). To estimate the number of sensors in each ring, we compute the angle θ between each sensor such that they do not overlap in coverage. θ and the angle between two non-overlapping beams' edges form a kite polygon (in middle of figure.) First, given the beam's width w and sensor maximum reading distance d, the angle α is computed using the tangent function (Eq. 3.1) and the length of beam's edge x using the Pythagorean theorem (Eq. 3.2). Then, combining Eqs 3.1 and 3.2, and the cylinder's radius r_h, the kite's longest diagonal y is computed using the Law of Cosines (Eq. 3.3).

$$\alpha = 180 - \tan^{-1}\left(\frac{w}{2d}\right) \tag{3.1}$$

$$x = \sqrt{d^2 + \left(\frac{w}{2}\right)^2} \tag{3.2}$$

$$y = \sqrt{x^2 + r_h^2 - 2r_h x \cos\alpha} \tag{3.3}$$

Finally, given the kite polygon has two congruent triangles (all sides are equal), the minimum angle of sensor separation is computed using the Law of Sines (Eq. 3.4).

$$\begin{aligned} \theta' &= \frac{\theta}{2} \\ \frac{\sin\theta'}{x} &= \frac{\sin\alpha}{y} \\ \theta' &= \sin^{-1}\left(\frac{x\sin\alpha}{y}\right) \\ \theta &= 2\sin^{-1}\left(\frac{x\sin\alpha}{y}\right) \end{aligned} \tag{3.4}$$

Using statistical data from the CDC [30], an average male has a height of 175.9 cm and a girth of 100.9 cm (32.12 cm in diameter), and an average female has a height of 162.1 cm and a girth of 95.2 cm (30.3 cm in diameter). Putting these data into the equations, and using Eq. 3.5, we computed the theoretical upper bounds of the number of infrared sensors we can place on a human body, assuming a cylindrical model. A male can have no more than 732 sensors on his body, and a female can have no more than 661 sensors on her body.

$$totalSensor = \left\lfloor \left(\frac{360^\circ}{\theta}\right) \times \left(\frac{H}{w}\right) \right\rfloor \tag{3.5}$$

Despite the limitations of range, beam narrowness, and beam interference, IR is a highly cost-effective sensor (a few dollars) that can be worn on the whole body. Coupled with a stimulator, it can be mounted on hands for reaching tasks, on arms and legs for obstacle detection, and on feet for stair detection [46, 68]. Given the narrow beam coverage of an infrared sensor, a group of infrared sensors (perhaps in the hundreds), with different orientations and no beams overlapped, can be used to generate a wide field-of-view (FOV) coverage. This is of no problem for Unity3D[1] because it leverages the host's GPU for

processing power and can run upwards of 60 Hz. We had also tested a simulation of a thousand infrared sensors (see Figure 3.6), which is more than the theoretical upper bound we computed earlier.

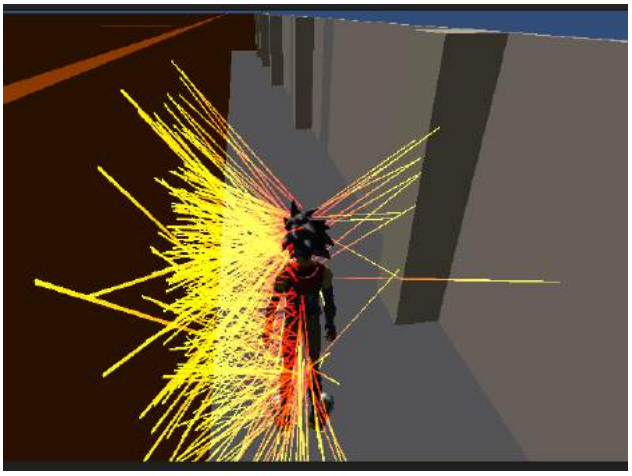


Figure 3.6: 1,000 IR range sensors on an avatar. The rays show the distances to the closest walls.

Algorithm 1 Virtual Infrared Sensor

Require: $maxRange > 0.0, direction \leftarrow Vector3()$
Ensure: $0 \leq distance \leq maxRange$

```
1:  procedure IR
2:      hitInfo ← RaycastHit()                          ▷ Initialize hitInfo
3:      distance ← 0.0
4:      loop
5:          startPt ← transform.position                ▷ Changes when avatar is walking
6:          if Raycast(startPt, direction, hitInfo, maxRange) then
7:              distance ← hitInfo.distance
8:          else
9:              distance ← 0.0
10:         end if
11:     end loop
12: end procedure
```

Algorithm 1 shows how an IR sensor is virtualized in Unity3D. Due to the IR beam narrowness, we decided to simulate it with a single ray, instead of its actual beam profile (*i.e.,* 8.5°). In order to simulate the sensor, the algorithm requires the maximum range the sensor can detect, and the norm vector to the placement surface. The maximum range in this

case is 1 meter for the real sensor. The simulated sensor has its own coordinate system; thus, the norm vector is the same as the positive y-axis of the virtual sensor, using the right-hand rule and assuming the positive axis points away from the body. The algorithm is guaranteed to return a distance value between 0 and the maximum range value. Sensing happens in an infinite loop (frequency of this loop depends on the VE's frame rate), where a raycast[15] is constructed with the position of the virtual sensor, the norm vector, the maximum range, and a reference parameter ($hitInfo$) that contains any hit result. Note that since the virtual sensor is mounted on the avatar's body, its position changes as the avatar moves around. If the raycast hits an obstacle within the maximum range, $hitInfo$ contains a reference to that obstacle, in addition to distance to it. This distance is assigned to the public variable *distance*, which other classes can query for the sensor output. These classes can also perform additional processing such as threshold before transducing to a real stimulator.

Ultrasonic/sonar range sensors

Ultrasonic/sonar range sensors have been widely used in robotic applications and the technology is very mature and the cost is very low so it could be used for daily obstacle detection for a VIP. Abu-Faraj *et al.* [1] created an ultrasonic sensor prototype that is specific to shoes and spectacles. Their prototype uses three pairs of ultrasonic transducers placed on the medial, central, and lateral aspects of the toe-cap. These sensors are used to detect ground-level obstacles as well as pits and holes. The ultrasonic transducers are also mounted above the bridge of a spectacle to detect head-level obstacles [1]. Ultrasonic sensors can also be retrofitted to traditional white canes to address some of their limitations such as detecting overhangs. While the trusted white canes can detect irregularities and obstacles on the ground, they fail to detect those above users' waistlines. Ultrasonic sensors can be mounted near the handle of the canes, such that when an obstacle is detected, a haptic response is triggered inside the cane, with increasing intensity as users approach the obstacle [76].

[15]A ray is a line that starts at a point and continues to the maximum range parallel to the norm vector. A raycast involves intersecting a ray with objects in the VE; raycast tells you what objects in the VE the ray runs into, and may return additional information as well, *i.e.,* intersection point

Similar to aforementioned sensors, a virtual ultrasonic sensor is realized as a cone-shaped beam profile range sensor (range up to 8 meters, angle up to 150°) with the tip of the cone pointing at the body. The beam profile is based on a real sensor's specification.

Algorithm 2 Virtual Sonar Sensor

Require: $maxRange > 0.0, radius > 0.0, norm \leftarrow Vector3()$
Ensure: $0 \leq distance \leq maxRange$

```
1: procedure SONAR
2:     hitInfo ← RaycastHit()                          ▷ Initialize hitInfo
3:     distance ← 0.0
4:     loop
5:         startPt ← transform.position                ▷ Changes when avatar is walking
6:         minDist ← maxRange
7:         for i,j ← 0, radius do
8:             if radius^2 ≥ i^2 + j^2 then
9:                 direction ← vector from startPt to (i,j,maxRange), offset from norm
10:                if Raycast(startPt, direction, hitInfo, maxRange) then
11:                    if hitInfo.distance < mindDist then
12:                        minDist ← hitInfo.distance
13:                    end if
14:                end if
15:            end if
16:        end for
17:        distance ← minDist
18:    end loop
19: end procedure
```

Algorithm 2 shows how a sonar sensor is virtualized in Unity3D, which is similar to IR with the exception of the beam profile. In order to simulate the sonar sensor, the algorithm requires the maximum range the sensor can detect, the radius of the beam, and the norm vector to the placement surface. The maximum range in this case is 8 meters for the real sensor. The simulated sensor has its own coordinate system, thus, the norm vector is the same as the positive y-axis of the virtual sensor, using the right-hand rule and assuming that the positive axis points away from the body. The algorithm is guaranteed to return a distance value between 0 and the maximum range value. Imagine the cone-shaped beam has a circle opposite of the tip, with a radius variable called *radius*. Instead of a single raycast, the virtual sonar sensor uses multiple raycasts in a single frame. Each raycast is constructed

with the position of the virtual sensor, the direction from $startPt$ to $(i, j, maxRange)$ ($\forall i, j$ *i.e.,* $i^2 + j^2 \leq radius^2$), the maximum range, and a reference parameter ($hitInfo$) that contains any hit result. $\forall i, j$ *i.e.,* $i^2 + j^2 \leq radius^2$, the public variable *distance* is assigned $min\_distance(hitInfo_{i,j})$.

RGB-D sensors

With the advent of commercial-off-the-shelf and portable RGB-D sensors such as Xtion PRO LIVE [16] and Microsoft Kinect [17], researchers are using such devices to identify people, detect obstacles, and avoid objects, since they can capture both color and depth (approximately 4 or 6 meters) information in real time [16, 45]. These depth-sensing input devices usually use infrared pattern or time-of-light lasers to provide real time distance information. While they are fast, they are limited in range and dependent on surface reflectance of the object. Alternatively, a traditional approach (which may be expensive in price or computational resources) is to use a stereo camera (or cameras in general), which generates dense 3D maps of color image pairs of the scene similar to a range-finder. Developers then feed the maps into algorithms that perform obstacle detection and avoidance by computing the shortest or safest path through the scene [89].

Regardless of how the depth map is generated, there are two sensors involved: optical and depth sensors. Virtualizing the optical sensor is the simplest since the camera (game) view can be used directly. That is, for every frame, it captures the view and then sends it to another Unity3D script for processing and transducing. The depth sensor, however, requires a similar approach to the virtual sonar sensor, but instead of a cone-shaped beam, it is a cube profile. To simulate a depth sensor with images of size $width \times height$ pixel, $width \times height$ raycasts are constructed at each "pixel" location with a direction that is parallel to the avatar's positive z-axis (assuming the right-hand rule and the negative y-axis points to the ground) and a maximum range of 4 meters. Since raycasts operate with 3D

[16] ASUS, "Xtion PRO LIVE", November 23, 2015, `https://www.asus.com/us/3D-Sensor/Xtion_PRO_LIVE/`

[17] Microsoft, "Kinect for Windows", November 23, 2015, `https://dev.windows.com/en-us/kinect`

vectors, each "pixel" location ($startPt$) can be computed by the x-y offset (as determined by $width \times height$) from the avatar and a z-value the same as the avatar's.

3.3.1.2 Transducing

Regardless of which range sensor is to be virtualized, a range profile is simulated and range information extracted. Using a third-party game engine – such as Unity3D – simulating IR, sonar, and RGB-D sensors is achieved via raycasting in the VE. All these virtual sensors generated sensory data such as range and depth data. The next steps are processing and *transducing*.

It is generally ill-advised to transduce raw data to users as that will be overwhelming. It is recommended instead to process the data into meaningful information before transducing it to another modality and to users. Processing is as simple as thresholding. To transduce from range data to vibration, for example, the raw data is thresholded into three intervals (or vibration levels and intensity). The intervals could signify an obstacle that is close, near, and far, respectively, and vibrate accordingly.

Transducing occurs when meaningful information (processed data) is sent to the stimulators of another modality, using a communication protocol that is compliant to the stimulators' hardware. Examples of communication protocols are USB/serial ports, Bluetooth low energy, and TCP/IP. For complex protocols where the required software libraries are not supported by the game engine, transducing has to happen in two parts. The first part resides in the VE where range data is extracted from virtual sensors, processed, and sent (outside of the VE) to the second part (this could be on the same machine as the VE or in another networked machine). Data transformation happens in the second part; it then transmits (using one of the communication protocols above) a command to stimulators. The applications in Chapters 4 – 7 contain more specific transducing examples.

Even though, in this dissertation, we focus on depth and intensity values in our simulation, other properties of the physical world, such as brightness, temperature, and color, can also be simulated and transduced to audio and haptic.

3.3.2 Game Mechanics

As discussed in Chapter 2, gamification is the application of game mechanics to a non-gaming context. In this section, we discuss how to define game mechanics in the GIVE-ME Framework. Without loss of generality, game mechanics provides a set of goals to achieve and define how users interact with the game. Furthermore, game mechanics also enables early stakeholder involvement by defining tasks that are typical of a scenario and observing how users behave using the AT. The spirit of game mechanics is to engage users while studying how well they perform in navigation tasks using the AT.

3.3.2.1 Task Definition

Task definitions or goals provide a context for the game or experiment. It describes what needs to be done in order to finish. The task is defined based on the experiment's objectives. The simplest task definition is navigating from point A to point B. More advanced tasks can also be defined, such as visiting rooms X, Y, and Z before reaching point B, or collecting as many collectibles as possible before reaching point B. Point B could also be on a different floor than point A, thus requiring a multi-floor navigation strategy (*i.e.,* how do I get to another floor?). Task definition can also be as vague as "go explore this layout," for the purpose of familiarizing oneself of the layout. A termination condition, however, is needed for such a task. Termination conditions can be time-outs, a certain set of actions performed, or something else at the discretion of the experimenter.

Of course, in an experiment, a single trial of a layout is not enough for any analysis. Multiple variations of the task definitions, therefore, are needed. For example, each subject in an experiment is required to complete two different sets of tasks: simple and complex. We

have a few simple layouts of different variations (*e.g.,* a straight hallway or a hallway with one corner) and a few complex layouts (*e.g.,* partial modeling of a few floors of a building). They can all have task definitions such as go from A to B. For complex layouts, A and B can be random locations for each trial. In addition, if the task definition includes obstacle avoidance (with the exception of wall collision), the layout could include stationary obstacles (but in random locations for each trial), or dynamic obstacles.

3.3.2.2 Avatar Behavior

To complete a task, users have to guide an avatar to completion. The avatar behavior dictates how users can interact with the avatar and what the avatar can interact with or perceive from its environs. Specifically, this describes what controller commands are valid and thus what the avatar can do (*e.g.,* Can the avatar run? Can it spin? Can the user query for more information for an object in front of the avatar?) and what environ cues the avatar can receive (*e.g.,* Ambient sounds? Sounds in response to an action? Some visual cues for low vision users?)

In a controlled experiment, there may be a need to limit some actions or variables. For example, the experiment may require subjects (vicariously the avatar) to walk in constant speed and turn or rotate in place, so as to remove the walking speed factors among subjects and ascertain their intention in making a turn. To achieve this, the avatar needs to be programmed to conform to these requirements. The avatar can process the controller commands such that no matter how hard one pushes the joystick forward, the translation speed is the same, and when one pushes the joystick left or right to turn, translational motion is nullified.

Since the involved multimodal ATs are for VIPs, sound cues can also be used to aid subjects in navigation and enhance the real stimulator of a different modality (*i.e.,* haptic feedback). This is related to avatar behavior, because, in Unity3D, sound sources, listeners, and sound profiles need to be defined. The listener is the avatar and it needs to know which sound sources are constantly audible (*i.e.,* ambient), and which are audible in response to

an action. It also needs to know how far away it will begin to detect the sound and whether it will gradually increase in volume as it gets closer. All these are defined through Unity3D's Audio Listener[18] and Audio Source[19] classes.

3.3.3 Virtual Environment Toolbox

The VE toolbox is responsible for setting up a virtual or gaming environment, receiving inputs from the controller, and passing the control data to other components. The VE toolbox has three components: 1) third-party game engine; 2) Environment Design; and 3) Controller Setup.

3.3.3.1 Third party game engine

The building block within the GIVE-ME framework is VE created with a game engine. We decide to use Unity3D[1] because of its popularity, excellent documentation, and tutorials. Using Unity3D, developers and experimenters can collaboratively design any environment they want for testing. Developers can program in C# or JavaScript, using Unity3D. In particular, Unity3D supports an open source version of Microsoft's .NET Framework 2.0 functionalities. These functionalities are part of MonoDevelop, Unity3D's integrated development environment (IDE). Unity3D is also a multiplatform development tool. It supports various PC operating systems, mobile devices, gaming consoles, and virtual reality headsets such as Oculus Rift. Thus, using Unity3D as a research tool allows us to leverage the power of the PC through the .NET Framework, and to connect with various sensors, stimulators, and external measurement devices. Furthermore, if we want to convert the game into a simulation or training tool, we can simply use Unity3D to re-target to another platform (*e.g.,* Android or iOS), with minimal modification.

[18]Unity Technologies, "Audio Listener", November 23, 2015, `http://docs.unity3d.com/Manual/class-AudioListener.html`

[19]Unity Technologies, "Audio Source", November 23, 2015, `http://docs.unity3d.com/Manual/class-AudioSource.html`

3.3.3.2 Environment Design

Using the game engine, we can generate very realistic 3D environments or scenes, in terms of both 3D structure and colors and texture mapping. This would cover the full spectrum of environment designs: from maze-like (*fictitious*) environments for more scientific studies, to *realistic* environments based on real floor plans for users' training. In general, the design should be open-minded. Figure 3.7 shows some examples of environment designs[20] [21] that are possible.

(a) Environment with a hallway[20]

(b) Environment with a stair[21]

Figure 3.7: Example of environment designs

In a VE, both static and dynamic objects can be included. VEs are especially useful when testing in the real world involves the possibility of physical danger to subjects. It is also useful when experiments require subjects to be stationary, but the task is navigation. To achieve the experiment objectives, a VE has to be designed in such a way that it conforms to task definitions and allows prescribed avatar behavior. If the task definition involves multiple floors, then the VE needs to include an elevator, escalator, staircase, or some combination of the three. Similarly, if the task definition involves collectibles, then the VE needs to have appropriately placed objects that the avatar can "collect." Sound cues may also be placed on or near the collectibles to alert users that the avatar is near a collectible. If the task

[20]NVYVE Inc, "Media — PAMELA GAME", December 8, 2015, `http://www.pamelagame.com/media/`

[21]Unity Technologies, "Press Downloads", December 8, 2015, `http://unity3d.com/public-relations/downloads`

definition involves obstacle avoidance, the VE needs to have colliders[22] on objects such that it can detect when the avatar bumps into them and takes appropriate actions to avoid them (*i.e.,* record the bump and play a sound to alert subjects to go around them).

The VE can also be embellished with visual textures. Although this may not be immediately useful for a VIP, it may be in the future; implants can enable them to see colors and texture in addition to ranges using audio and haptic. At the moment, it might be useful for other sighted users or observers to monitor a VIP's progress. Furthermore, if the experiment needs to provide GPS-like directions, then the environment needs to have a pathfinding[23] artificial intelligence (AI) included that can provide simple directions (such as forward, left, and right), to simulate the GPS. This Environment Design component is also responsible for instantiating virtual sensors and placing them appropriately on the avatar's body. The environment needs to enable objects to be detected by the virtual sensors. The environment also needs to enable the physics of objects that are interactable (such as those where an avatar can open a door, or slide a chair across the floor). Of course, sound cues may also be used to indicate such actions.

3.3.3.3 Controller Setup

Control commands coming in from a controller need to be processed so that they enable the defined tasks and avatar behavior. Normally, a game engine is compatible with standard input devices via plug-and-play, such as game pad controllers and keyboards. In this case, the controller setup involves capturing specific key action events (*e.g.,* key or button presses). This framework is also flexible enough to allow complex (albeit natural) interaction, so that controller devices such as Microsoft Kinect or an optical tracking system may be used. In such cases, a complex setup is involved where a customized input driver is needed to process data from the controller device and send control commands to the VE (Chapter 3.2.1).

[22]Unity Technologies, "Colliders", November 23, 2015, `http://docs.unity3d.com/Manual/CollidersOverview.html`

[23]Unity Technologies, "Navigation System in Unity", November 23, 2015,`http://docs.unity3d.com/Manual/nav-NavigationSystem.html`

Processing data involves detecting hand gestures from the Kinect, or detecting users walking motions using the tracking system. A good compromise between the plug-and-play and complex natural interaction devices is the long-awaited Virtuix Omni (Chapter 3.2.1), which is designed as a plug-and-play device but also allows users free range of walking motion in 360° virtual reality environments. Finally, this component also serves as a starting point of data collection procedure where it logs all user inputs (more details in the next section).

3.3.4 Data Collection

To effectively assess an AT for enhancing or substituting visual perception of an individual with limited or no sight, an experimenter needs to collect numerous data and evaluate the given tasks for performance. Data collection in the GIVE-ME Framework allows for various types of data to be collected; this includes but is not limited to, multimodal sensory data, brain or behavioral measurements, and control or action data. *Multimodal sensory data* such as range data generated by the virtual sensors, the output generated by the transducing component (*i.e.,* the data being sent to real stimulators), and any static sensory data (*i.e.,* ambient and action or reaction sounds). If connected, *brain or behavioral measurements* can be collected from the Measurement Device (*e.g.,* EEG). Such measurements are generally stored externally from what is being collected inside the VE, thus the VE has to send a trigger command to the Measurement Device to synchronize data collection on both ends. A trigger command serves as a timestamp in both the VE, where an action is performed, and the Measurement Device, where a stimulus is expected to occur and to be measured. Lastly, *control or action data* includes all user inputs, events happening in the VE (*i.e.,* bumping into an obstacle), and the game state (*i.e.,* an avatar's position and scoring).

3.3.4.1 Task Evaluation

The quantitative data mentioned above can also be used to supplement a qualitative analysis (*i.e.,* in addition to interviewing subjects and giving them a survey to fill out). While one

can employ any set of features they want for the analysis, we recommend the following set of metrics. This is partly adapted from Plos *et al.* [71], Dakopoulos and Bourbakis [22], Giudice and Legge [32], and partly inspired from interaction with and desired by VIP:

Acceptability: Design that is useful, reliable, robust, aesthetic, and has positive impact on quality of life of a user.

Compatibility: Design that is compatible with the lifestyle of a user and of other technologies.

Adaptability: Design that can be easily adjusted (*i.e.,* function, location).

Friendly: Low learning curve for the system; easy to use.

Performance: Overall performance.

Acceptability assesses whether an AT is providing a useful solution for a VIP, in addition to the AT's aesthetic features and functionalities. Since there is no single, universal AT, *compatibility* assesses how an AT and its components interact with other devices. Similarly, *adaptability* assesses how versatile an AT is in adapting to different situations. *Friendly* assesses the intuitiveness of an AT and how easy it is to use. The overall *performance* includes the complexity of an AT (proportional to information bandwidth), its strengths and weaknesses over various circumstances, *etc.* While these metrics are posed as open-ended questions, they can also be presented as rating surveys. Data to assess friendly and performance features can also be extrapolated from the quantitative data (*e.g.,* sensory and control or action data, and brain or behavioral measurements). For example:

- Time to completion,
- Number of errors (*e.g.,* bumping into obstacles, incorrect response),
- Game score,

- User's trajectory, and,
- Brain or behavioral measurements (*e.g.,* EEG, fMRI, *etc.*).

Assuming sufficient training time has elapsed, an effective AT should allow users to complete a task in a reasonable time and with fewer mistakes. Users' trajectory in VEs gives researchers an insight into how friendly an AT is. Ideally, a trajectory should be smooth, and not jagged, as if trying to compensate. For advanced users, gamification introduces challenging tasks (*e.g.,* exploring all rooms for collectibles, reaching waypoints before destination, *etc.*), in addition to simple navigation. The performance of these tasks can be encapsulated in a game score, where each completed task is awarded a certain amount of points. The game score can then be used as a metric to compare other ATs, in addition to keeping users engaged in an arduous experiment.

3.3.4.2 Establishing Ground Truth

It is a common practice when benchmarking an algorithm to compare its performance with ground truth data. In evaluating ATs, obtaining ground truth data in the real world is difficult due to hardware limitations (*i.e.,* noisy data) and lack of well-controlled environment. For this reason, this framework provides a virtual world that is noise-free (with the ability to add known Gaussian noise to simulate real scenarios), in terms of sensor data, and a well-controlled environment by managing each of the aforementioned components. Of course, consensus on what kinds of data need to be collected is needed in order for others to benchmark against it. The ground truth data obtained in the virtual world is transferable to the real world. For example, designers and developers can use the data to do a baseline calibration of their system and fine-tune it later with real world testing. It is expected that the real world performance will be poorer than the virtual ground truth data due to reasons mentioned before, but it should provide a relative performance rating (to ground truth) as to how well the AT is performing. If one AT's performance has a small deviation from ground

truth compared to another AT which has a larger deviation, then one might conclude that the former AT is performing better compared to the latter.

3.4 Intended Use

In summary, the GIVE-ME framework is meant to be deployed by AT developers and researchers to:

1. Test their systems,
2. Provide a simulation and training tool to users, and,
3. Establish communication between rehabilitation professionals and end-users with visual impairment.

GIVE-ME has several advantages in relation to simulation, stimulation, and evaluation. GIVE-ME enables rapid prototyping of sensors and stimulators before integrating into AT. Sensors are simulated in a virtual environment, even sensors that do not exist yet, but are highly desired. Stimulators are connected to the VE in an ad hoc manner for testing and evaluation. In addition to traditional evaluation, psychophysics evaluation can also be conducted while subjects are using an AT. Furthermore, this framework allows AT developers and researchers to benchmark their systems with other heterogeneous ATs. As a community, AT developers and researchers can agree on a set of metric features for task evaluations and benchmarks. It is not the goal of this dissertation, however, to determine a set of metric features. Lastly, this framework facilitates early stakeholder involvement in the design of AT [43, 63].

As more and more people utilize this framework, a dataset of hybrid data can be generated. Hybrid data from GIVE-ME can yield valuable information regarding users' perceptions, thoughts, and opinions, and quantifiable assessments of the ATs. If successful, psychophysics experiment data that was not possible before can be produced, specifically

navigational data. A similar experiment has already been conducted on lab mice[24]. Combined with functional and objective assessments of ATs, these data can help shape better and more human-centric ATs. Furthermore, simulation and training tools can be developed to better educate and train users in using ATs. One of the goals of this framework is to determine the minimally optimal set of sensors and stimulators that can be integrated into an easy-to-use AT which provides concise information about the users' surroundings in the simplest and most intuitive manner.

3.5 Software Implementation

In this section, we present the workflow of the GIVE-ME framework to evaluate a navigational AT and a software version of the conceptual framework.

3.5.1 Workflow

The workflow of applying GIVE-ME is as follows:

1. Identify the application and tasks (games) to virtualize;
2. Identify the multimodal sensor(s) and stimulator(s) needed;
3. Virtualize the sensors and establish connections from VEs to stimulators.

The application is generally related to navigation, but with some transformation, it can be easily adapted to other applications (*e.g.,* helping autistic people in focusing on a task). In the VE, the tasks as described in Chapter 3.3.2, can be as simple as navigation from origin to destination. The programming or design works here are to generate a VE (Chapter 3.3.3), program tasks (*e.g.,* collect objects, avoid obstacles, *etc.*) and allow interactions (*e.g.,* open doors, ducking, *etc.*), and data collection procedures (Chapter 3.3.4).

[24]MIT Technology Review, "How Brain Scientists Outsmart Their Lab Mice", November 15, 2015, `http://www.technologyreview.com/news/541891/how-brain-scientists-outsmart-their-lab-mice/`

Once the sensors and stimulators that one wants to use have been identified, one simulates the sensors based on its real specification (Chapter 3.3.1). This can be a creative task because the basic component in most game engines for sensor reading is raycasting (*i.e.,* casting a ray out from an origin point and capturing the distance as it hits an object). The completed virtual sensors are mounted on an avatar, such as on the arms and torso. Similarly, stimulators are physical devices that are placed appropriately on the target locations (*i.e.,* locations on a user's body). They also need to have a communication channel established with the VE, so that the VE can send commands to stimulators, which in turn transduces specific information to users (Chapter 3.2.2). For example, sending distance readings to vibrotactile-range stimulators will allow users to feel intensified vibrations as the avatar approaches an obstacle.

3.5.2 GIVE-ME Package

To implement the framework, we decided to use Unity3D for its ease of use, excellent documentation, and a large community base. Unity3D allows developers to export their project's source codes and settings as a *package* and then import it into another project. Normally, game development from scratch takes months to complete, especially with a one-person team. However, we made this process easier by packaging components we have and providing predefined interfaces for others to use. This way, other developers can just specify what they want and the framework gives them what they need; they do, however, need to write some code in Unity3D for novel functionalities that are not provided by existing versions of the framework. We believe the framework implementation significantly reduces development time.

The conceptual framework has seven components as discussed in Chapter 3. For the software implementation, we group them into four questions and the answers to these questions are our implementation. The four questions are:

1. **How to specify the sensors?** (Multimodal virtual sensors excluding transducing);

2. **How to specify the stimulators?** (Multimodal stimulators);

3. **How to specify the data to be collected?** (Measurement device and data collection); and,

4. **How to specify the environment to be navigated?** (Controllers, game mechanics, and VE toolbox).

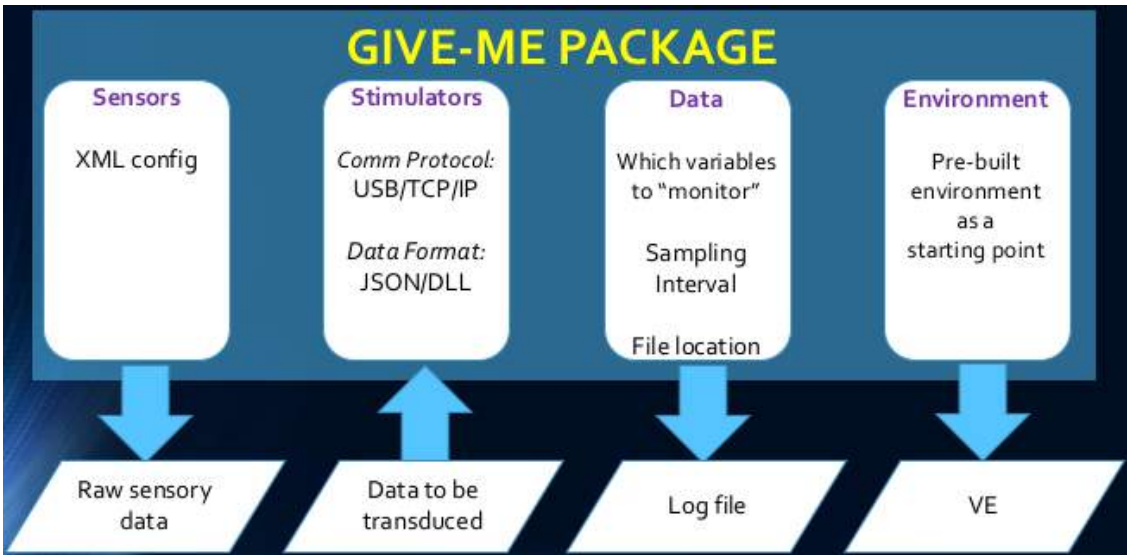


Figure 3.8: Package Overview

Figure 3.8 shows an overview of the software implementation as a Unity3D package. It is available at `http://ccvcl.org/~khoo/GIVE_ME.unitypackage`. By selecting options in the modules and specifying input parameters, the package generates a set of data and an environment for developers to evaluate their ATs. The following four sections discuss each module in detail.

3.5.2.1 Environment

After importing the package, a menu will appear in the menu bar, allowing developers to begin their project with one of the provided environments (Figure 3.9). This menu is customizable by modifying the included editor source code. The provided environments

include the layout and model of actual environments and a randomized hallway. Basic interactions (*i.e.,* interactive doors, sound cues embedded in objects, *etc.*) are also included. The spirit here is to use one of the environments as a starting point because chances are, developers will need to modify the environment for their experimental needs. Furthermore, building an environment from scratch is time-consuming as you need to model and build the layout, texture and map the surfaces, if needed, and program in the basic interactions, among other designs. To accelerate this process, developers can use the work done by Tang *et al.* [82] for automatic map generation using a floor plan. Their algorithm takes in a 2D floor plan image and render a simple 3D model that preserves rooms, doors, hallways, and walls. Of course, developers may want to use their own environment (*i.e.,* one that they designed), which is possible in this framework.

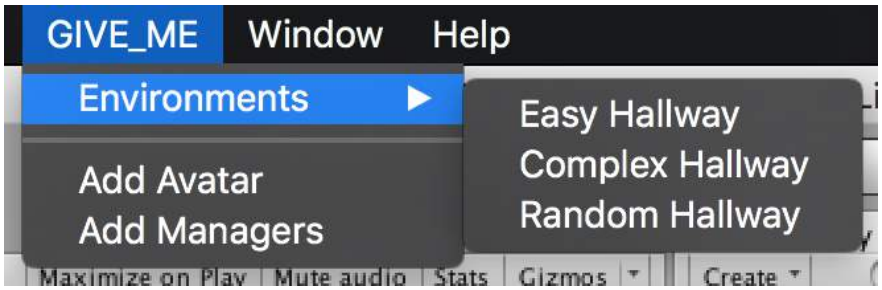


Figure 3.9: GIVE-ME Menu

3.5.2.2 Sensors

Developers can specify what virtual sensors they need for their project via an XML configuration file. The XML schema (*i.e.,* tags and values) contains information such as the type of sensors they want, the placement of the sensors on the avatar, its orientation, and specifications of the sensors (*e.g.,* maximum range and beam angle). Using this approach, developers can specify the sensors during runtime via a menu selection that reads in the XML file. Figure 3.10 shows the inspector of the sensor manager script. Note that in the current version of this package, virtual sensors include infrared and sonar. To simulate other

types of sensors, developers need to write a simulation script for each sensor, generate a sensor prefab[25], and include each in the prefab list in the sensor manager in order to use it.

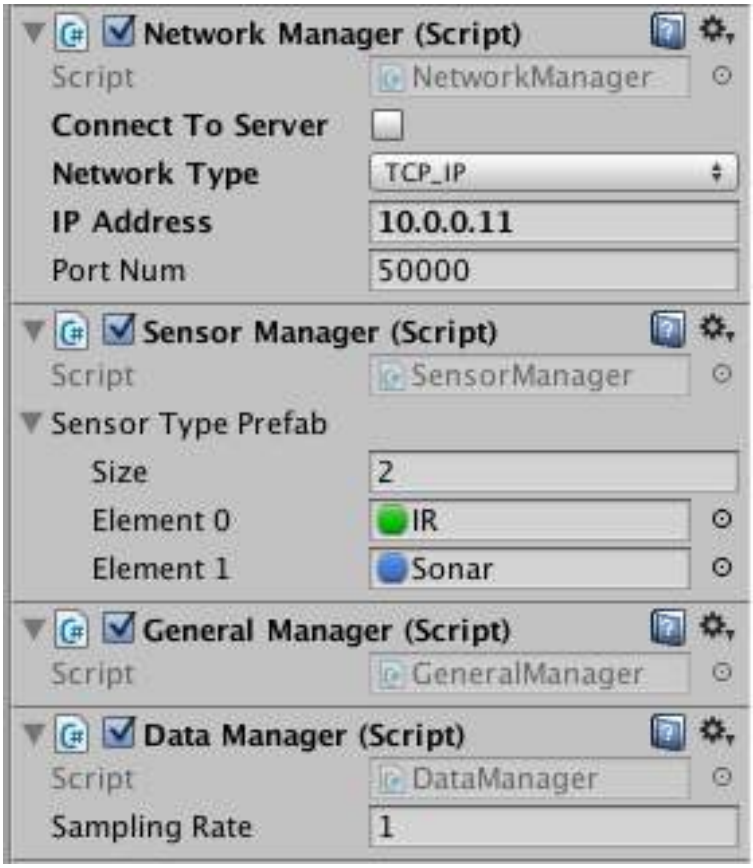


Figure 3.10: GIVE-ME Inspector

To simulate a new virtual sensor (script), developers have to inherit from an abstract class called *AbstractSensor* and implement the *Sensing* function in the derived class. The distance sensing simulation code is in the *Sensing* function. The function must assign the final sensed distance reading to the public float variable *Distance*. Updating this variable is important because this is the sensory data that the sensor manager will query and make available for transmission.

In the package, we only provide the virtual sensors that generate raw sensory data, such as a distance measure from a point sensor, a 2D array of color values from a color camera, and a 2D array of range data from a range sensor like Kinect. Since it is difficult to foresee

[25]Unity3D Manual, "Prefabs", `http://docs.unity3d.com/Manual/Prefabs.html`

how developers want to transduce their data and to what stimulators they are connected to, the transducing part has to be implemented by developers. The virtual sensors provide a common output that developers can pass on to another user-defined function that processes the data and transduces the results. This approach is especially useful if developers want to implement their own connection to a different stimulator than what is provided.

3.5.2.3 Stimulators

This component is closely coupled with transducing of the virtual sensors because a physical stimulator directly connected to the VE serves as a medium to the transduced sensory data. Whereas the transducing component has to be implemented by developers, we provided a general API for implementing stimulators. In general, we find that stimulators can be connected to a VE in one of two ways: one, direct to Unity3D via USB; and two, via an interoperation program. In the top part of Figure 3.10, a network manager is provided to facilitate the connection.

For the first method, a serial port connection is provided to connect to a stimulator and send data to it. Developers have to select "USB" and specify the COM port that the stimulator is connected to. The connection accepts a string or JSON data object (*i.e.,* sensory data to transduce) and prepares it to be sent to the stimulator. Depending on what kind of data format the stimulator is expecting, developers have to overload the output function of the serial port manager. If the stimulator, however, is built into the computer, such as audio speakers, a text-to-speech dynamic link library (DLL) can be used to speak a string.

For the second method, a simple server-client program is provided to extract data out of the transducing module in VEs and send it out to another program. Developers have to select "TCP_IP" and specify the IP address and port number that the external program or server is running and listening. Similar to the first method, this connection accepts a string

or JSON data object and prepares it to be sent to the stimulator. This method is preferred if the software or driver for the selected stimulator is not compatible with Unity3D.

3.5.2.4 Data

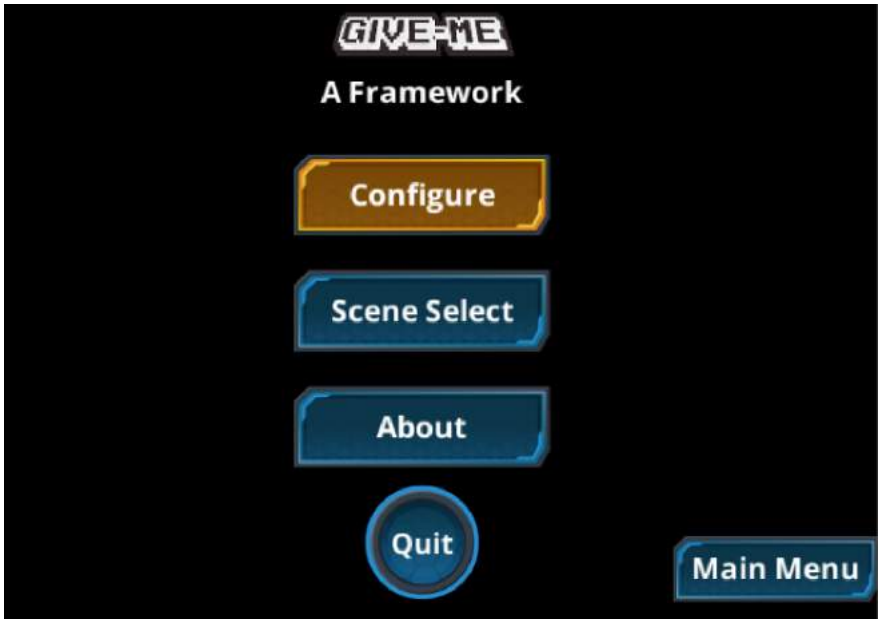


Figure 3.11: GIVE-ME Default GUI

There are many approaches to collect data. The naïve approach is to simply record every single piece of data (*e.g.,* variables, objects, function values, *etc.*), similar to a data dump. Developers, however, may not need most of this data, and may need to laboriously sift through the data for useful information. A more sensible approach is to allow developers to select what data to record. As such, a data manager is provided with the package. To record data for an object with a Unity script attached, the script needs to inherit the class *AbstractDataCollect* and implement the function *RecordData*. Inside the function, developers code in variables that they want to collect and output a comma-separated string. The data manager will take the output from this function and save it to a specified log file with timestamps. The log filename can be specified during runtime via the GIVE-ME default GUI (Figure 3.11). Developers also need to specify the sampling rate in second, in the inspector (bottom of Figure 3.10). This way all collected data are synchronized across all mediums

(*i.e.,* Measurement Device, software, and pen-and-paper method) for offline analysis. A default setting is provided where the manager will record both the avatar's position, and time passed since the VE startup, since these two data are the most common.

The default GUI in Figure 3.11 allows developers to configure their setup by specifying sensor XML file and data collection filenames. Then the developers can select a scene from the menu to start the experiment.

3.5.2.5 Development Cycle

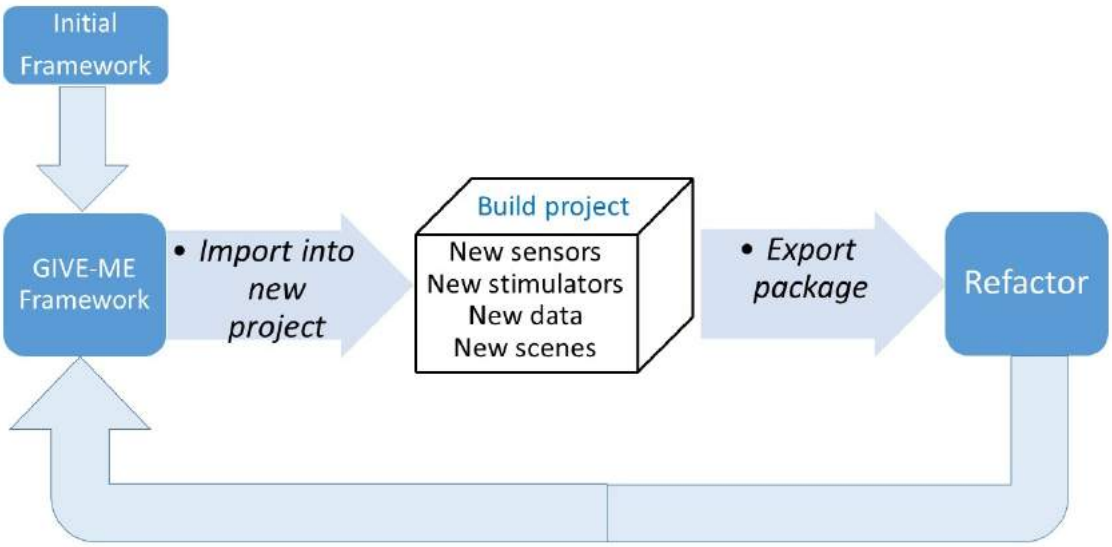


Figure 3.12: Framework development cycle

We envision the GIVE-ME Framework being used and improved as follows (Figure 3.12). The initial software framework includes technologies we used thus far and is available online at `http://ccvcl.org/~khoo/GIVE_ME_PACKAGE.unitypackage`. Developers then can download the package, import it into their project and build a new multimodal evaluation experiment. Here, we are interested in those who had to write custom codes for their project or experiment in order to simulate a new sensor, connect to a new stimulator, or create a new virtual environment. This can expand the framework with more functionalities and features that might be useful for more people. To do this, these developers will have to submit their

new changes to us. We then perform code integration and refactoring to incorporate the new changes by abstraction, while preserving the behavior and interfaces of GIVE-ME. This will produce a new and improved version of the GIVE-ME Framework, which we can publish to the same online location.

Chapter 4

Application 1 - BrainportNav

To start, we applied GIVE-ME to Brainport navigation with collaborators at CUNY [86]. Below, we first discuss the experiment, and then present how to use the framework to conduct the experiment.

4.1 Experiment

Brainport (Figure 4.1) from Wicab[11] is a tongue-based electrical stimulation device that conveys brightness contrast of a scene in front of users through a 20×20 electrode array pressed against the tongue. A camera is mounted on a pair of eyeglasses that captures videos. The video data is then sent to a base unit that processes the data. This involves converting it to a grayscale image and downsampling it to 20×20 pixels. The electrotactile stimulus data are then transmitted to the tongue display unit [7, 23, 46, 48, 59].

The goal of this experiment is to assess the effectiveness of the Brainport system in conveying visual information. The initial conclusion indicated that subjects have difficulty discriminating shapes, regardless of whether it is filled-in or outlined. The subjects, however, were able to distinguish three line orientations that differ by more than 45° angles [86]. Using this initial result, a simple maze is developed (Chapter 4.6) to simulate a navigation task that exploits the three line orientations (*i.e.,* (1) backslash, (2) vertical, and (3) forward

Figure 4.1: Brainport

slash) to indicate left, forward, and right turn, respectively, and to determine whether VIP may benefit more from the Brainport device for navigation. While [86] having focused on reporting the experimental results on identifying the three line directions by the same four human subjects, this application focuses on the framework aspect to enable the data collection and analysis.

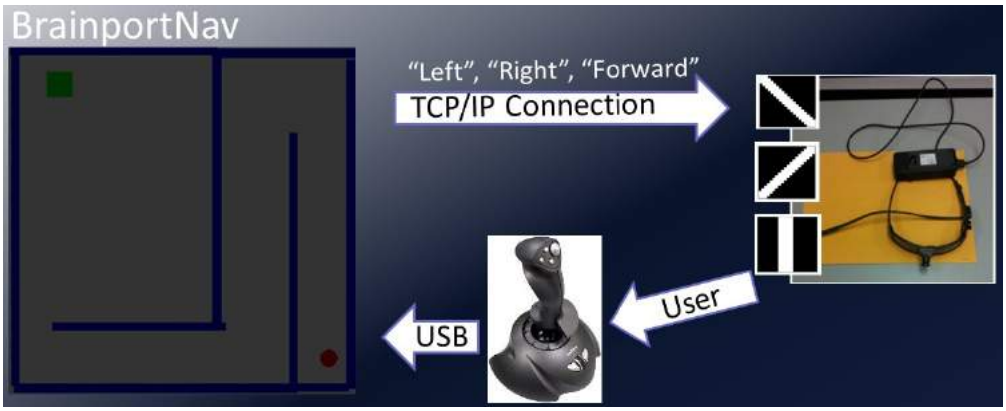


Figure 4.2: Brainport experiment setup

Figure 4.2 illustrates the experiment setup for BrainportNav. The maze is connected to the Brainport Workstation via TCP/IP connection and commands (*i.e.,* left, right, forward) are sent through it. After receiving the commands, a Brainport application on the Workstation converts and transduces them to the subjects via the 20×20 electrode array. They, in

turn, control the avatar via a joystick. The following subsections describe each part of the setup in detail.

4.2 Controller and Its Setup

Two controller options are provided to the VE: keyboard and joystick. Ultimately, the joystick is used because of its intuitiveness. The keyboard option works in similar fashion using only the four arrow keys. Since both keyboard and joystick are standard devices, they can be connected to the VE via plug-and-play. To allow for a controlled experiment (*i.e.,* to better capture a subject's action in response to a tongue stimulus), we decided to discretize the task by having a constant walking velocity and rotation angle. The velocity and rotation angle can be changed via a configuration file (the discretized rotation in our experiment is 45°). The above action (determining the direction to turn) is realized by having the subject control a joystick, pushing the stick left or right to indicate a corresponding turn, and then pushing the stick forward to move forward.

4.3 Stimulator

The Brainport uses a tongue-based electrical stimulation device to convey brightness contrast of a scene to the user. The SDK that came with the Brainport only allowed us to feed 20×20 pixel images to the electrode array. Originally, the Brainport system was designed to send the 20×20 images directly to the tongue of users to "percept" the scene. However, studies by us [86] and others [6, 74] have shown that it was very hard for users to read even the simplest shapes such as circles and squares, let alone to recognize complex real world scenes for navigation. Thus, we transduced the images into four directional outputs: left, right, forward, and stop (see more details in Chapter 4.4). A wrapper class was written, where it established a TCP/IP connection between the Brainport application and the VE. The VE sent a direction to the Brainport application via the connection, where it converted the

direction (of type string) to the 20×20 electrode array (the stimulator). Subjects, using their tongues, had to feel the electrode array and determine the direction to turn using the controller.

4.4 Virtual Sensor and Transducing

The Brainport's sensor is a camera, and thus, the virtual sensor is the camera in the VE (the game view). Instead of using image processing and computer vision to process the images to determine the navigation direction, we decided to use the Unity3D built-in pathfinding algorithm, NavMesh Agent[1]. Given start and finish locations in the VE, the pathfinding algorithm generated a series of waypoints for the path. As subjects moved the avatar, a direction (*i.e.,* left, right, and forward) to the next nearest waypoint in the series was determined (this is akin to sensing). The directions are generated in real-time at 30 Hz, even if subjects are idling. Thus, a direction is only sent to the Brainport's application if it differs from the previous direction (this is akin to processing). The direction is then encoded as a string and sent via the established TCP/IP connection. Once the Brainport's application received a direction, it converted the direction and fed it to the 20×20 electrode array (the stimulator).

4.5 Game Mechanics

The task definition for this experiment is very simple: navigate from start to finish without bumping into the wall, and use the joystick and tongue-based electrical stimulation device. The avatar behavior is also straightforward. Its forward motion is at a constant velocity, such that no matter how hard one pushes the joystick forward, the speed is the same. As mentioned before, to allow for a controlled experiment, the rotation angle is discretized. In

[1]Unity Technologies, "NavMesh Agent", November 23, 2015, `http://docs.unity3d.com/Manual/class-NavMeshAgent.html`

other words, instead of a smooth rotation of the avatar left or right, subjects had to push the joystick left or right and then release the joystick back to a neutral position, which caused the avatar to rotate left or right at a fixed angle (45°).

4.6 Environment Design

In light of the task definition, a simple maze was generated, where it had a single path from start to finish. The maze was a single floor and only contained one type of obstacle (*i.e.,* walls). With a starting location, we generated the maze with several hallways that included left and right turns. As the maze was generated, a path was computed and wasguaranteed to terminate at the border of the maze. The end of the path was designated as a finish location and was visually verified (by the experimenter) by a floating box at the location. The VE had nothing lavish; subjects guided the avatar down a hallway-like maze to a destination with walls on either side. The maze had a dimension of 20 feet by 20 feet (assuming 1 Unity unit is 10 feet). The path in the maze was also able to be visualized for debugging purposes.

4.7 Measurement Device & Data Collection

For this experiment, no measurement device was used. Four blind adult subjects were recruited, and each gave their written informed consent to voluntarily take part in the experiments. This study was approved by the CUNY IRB. Due to limited availability of Brainport (*i.e.,* a short and expensive leasing period), we were not able to conduct as many experiments as we had wished.

The features we collected for this experiment are as follows:

1. Time: From start to finish locations.

2. Position: 3D position of avatar.

3. Heading: 3D vector of avatar's forward heading.

4. Turn_Direction: Turn direction that was sent to Brainport.

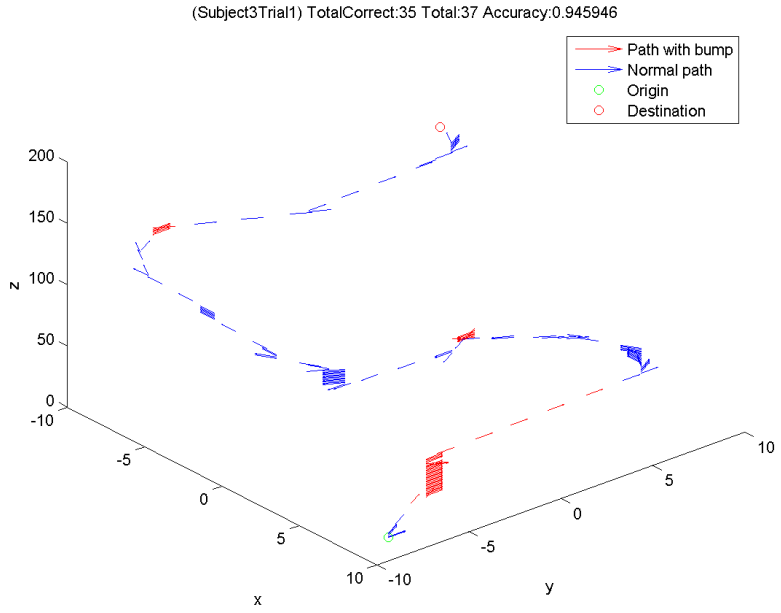
5. Bump: Whether the avatar is bumping into an obstacle.

6. UserInput: User's response.

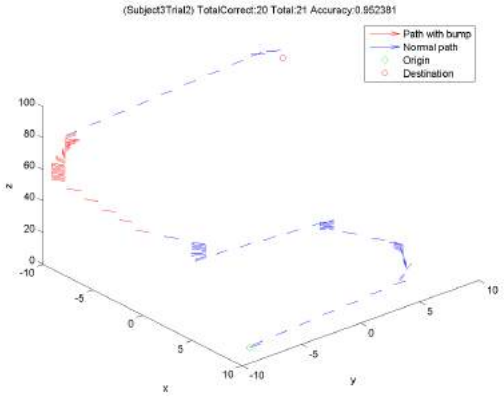
The data collection procedure has a 20 ms interval. In other words, every 20 ms, the above data were recorded to a file. *Time* serves as a timestamp for the other features and it is reset to 0 at the beginning of each maze. Thus, the last timestamp for each maze trial indicates the time (in seconds) it took the subject to guide the avatar from start to finish. Every 20 ms, the 3D *position* of the avatar and its forward *heading* (a unit vector) were recorded. The refresh rate of *turn direction* varied depending on whether the avatar needed corrective feedback to get back on track, or easily it reached the next waypoint. Regardless, every 20 ms, the current turn direction is recorded. Similarly, *bump* indicates whether the avatar was in a state of wall collision or not. Lastly, *UserInput* indicated the action the subject took; that is, pushing the joystick forward, left, right, or not pushing at all. In the last case, the text "NA" was recorded. All analysis and graph generation was done through a custom written Matlab script. The next section shows the results.

4.8 Results

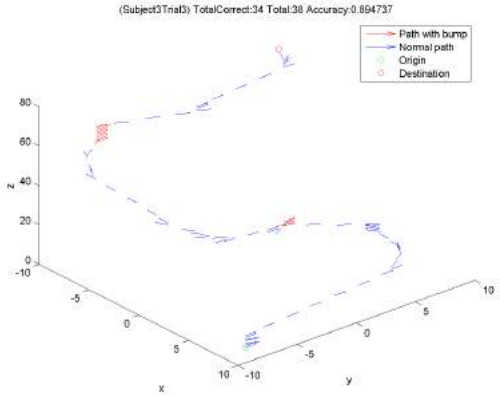
Depending on whether subjects made mistakes, corrective feedback may have had to be transduced to the subjects to guide them back on track. For this reason, there was an average of 267 corrective feedback or instructions for this maze. During the experiment, vertical and diagonal lines (a corrective feedback) were presented to direct the subjects through the maze. Line orientations were presented in accordance to the virtual sensor outputs. The left diagonal line was presented an average of 111 times, the right diagonal line an average of 116 times, and the vertical line an average of 40 times. During each corrective feedback, a line type was presented to the tongue continuously until the subjects responded.



(a) Run 1: Time=151s, Accuracy=0.95



(b) Run 2: Time=91s, Accuracy=0.95



(c) Run 3: Time=78s, Accuracy=0.89

Figure 4.3: Trajectories of Subject #3 over three runs of the maze in BrainportNav.

To see how well our subjects performed in virtual navigation using the Brainport, Figure 4.3 shows the trajectories of Subject #3 over three runs of the maze. The x-axis and y-axis show the 2D position of the avatar in a top down view (the height dimension is ignored since it didn't change). The z-axis shows the number of samples used to graph (*i.e.*, time-step). Since each run generated several thousand samples and given the sampling interval, most samples are redundant. The graphs were generated by downsampling the samples by half, if not more. The original sample size was 31,000. Each graph shows the origin (green

circle), destination (red circle), and the path taken by the avatar. The path also denotes where the avatar bumped into a wall (red line with arrows). Furthermore, it shows that out of all corrective feedback ($Total$), how many were responded to correctly by the subject ($TotalCorrect$), and the $Accuracy$ is $TotalCorrect/Total$. Looking across the three graphs, the subject's performance improved in each run, as indicated by fewer red arrows on the path (blue arrow indicates trajectory without any bumping). The subject also took less time to complete subsequent runs. While the accuracy for the first two runs were high (95%) for Subject #3, the third run took a dip in accuracy. The subject could have become overconfident, and wished to finish fast, to beat the previous times. Overall, this shows that Subject #3 completed the runs using Brainport in approximately 2 minutes, with good accuracy.

Table 4.1: Accuracy for four subjects

Subj_Run	L_Correct	Total_L	R_Correct	Total_R	S_Correct	Total_S	Accuracy
Sub1_R1	55	68	69	82	3	3	0.83
Sub1_R2	6	11	12	12	11	14	0.78
Sub1_R3	10	11	9	11	12	13	0.89
Sub2_R1	49	63	56	79	5	7	0.74
Sub2_R2	179	184	172	173	4	4	0.98
Sub2_R3	20	23	13	22	17	21	0.76
Sub3_R1	10	12	10	10	15	15	0.95
Sub3_R2	6	7	8	8	6	6	0.95
Sub3_R3	12	12	8	10	14	16	0.89
Sub4_R1	12	23	18	25	15	24	0.63
Sub4_R2	8	12	11	14	15	17	0.79
Sub4_R3	6	17	19	19	12	19	0.67

Table 4.1 shows the accuracy for all 4 subjects for each of the runs (average of 267 corrective feedback for each run). Each row shows a run for a subject. Each column starting from the second column, shows how many left turn corrective feedbacks ($L\_Correct$) were responded to correctly by the subject, out of the total ($Total\_L$). The next four columns are similar for right turns and straight forward (or vertical). The last column is the accuracy for that run. Accuracy is computed by comparing UserInput to Turn_Direction or ($L\_Correct+$

$R\_Correct + S\_Correct)/(Total\_L + Total\_R + Total\_S)$. Average accuracy for each subject can also be computed over all three runs. The average accuracy for each subject was 83.33%, 82.66%, 93%, and 69.66%, respectively. Overall, each subject's performance improved in each run and less time was needed to complete each run. Three out of four subjects achieved an average of at least 80% accuracy.

While the previous experiment's result showed poor performance in discriminating filled-in or outlined shapes, this result showed that Brainport can be better used to give simple directions to people for navigation. Real world navigation requires complex directions, however, which limits the capabilities of Brainport in this regard.

4.9 Discussion

We found that instead of using the built-in camera on Brainport to convey scene information to subjects for testing, which has been shown to be very hard for users, we can generate a VE and provide navigational directions to subjects. By doing so, we were able to send dynamic images (*i.e.,* directions) in real-time and measure subjects' responses. Furthermore, subjects were able to have a more engaging session; they were able to receive immediate feedback via sound, especially when they made a mistake (*e.g.,* bumped into wall). Without GIVE-ME, we would not have been able to obtain precise trajectory data to evaluate subjects' performance, because one would need a large room and an expensive motion tracking system installed, to track subjects' trajectory. Without GIVE-ME, conducting this experiment would have been even more challenging because measuring performance and psychometrics data while subjects are walking in the real world, is extremely difficult. Since the workstation version of Brainport (and its bulky converter unit) requires it to be connected to a laptop, any real world navigation experiment design would have been limited by the length of the power cord used; whereas, the VE design was limited only by the programmer's creativity and imagination. Even a multi-floor experiment design is possible, if so desired. We did not

attempt to do so this time due to time constraints. There are limitations on this experiment; we conducted an experiment using the Brainport in such a manner that was not intended in its original functionalities. Brainport captures visual scenes in front of users and downsamples it into 20×20 pixels for the users to feel, in real time. We, however, sent directions to the users via the electrode array. Training done in this case will not be transferable to the real world because the real world visual images involved complex shapes and objects beyond those in simple 2D lines.

Chapter 5

Application 2 - CrowdSourceNav

In this application, we proposed a smartphone-based crowd-sourced navigation solution with a focus on evaluation in a virtual environment. To get from point A to point B, VIPs used their smartphones to stream videos of the view in front of them to an online portal where a group of "crowd volunteers" directed them to their destination. Algorithm developments [66] by the members of the Visual Computing Lab included developing a smartphone app to enable such a service, a user-interface of the online portal, and an aggregation algorithm that transformed a set of volunteers' directions to a single response for each VIP. We have evaluated the crowd navigation system using an earlier version of the GIVE-ME Framework [47].

5.1 Experiment

Before we conducted real user testing, we proposed to use virtual environments to simulate VIP navigation while testing the crowd's ability in assisted guidance. Not only did virtual environments give us a more controlled study (with regards to personal safety and inherited device's limitations), but it also allowed us to establish ground truth data for future comparisons, fine-tuning the aggregation algorithm, and troubleshooting the online portal. Furthermore, it allowed us to verify the viability of the system.

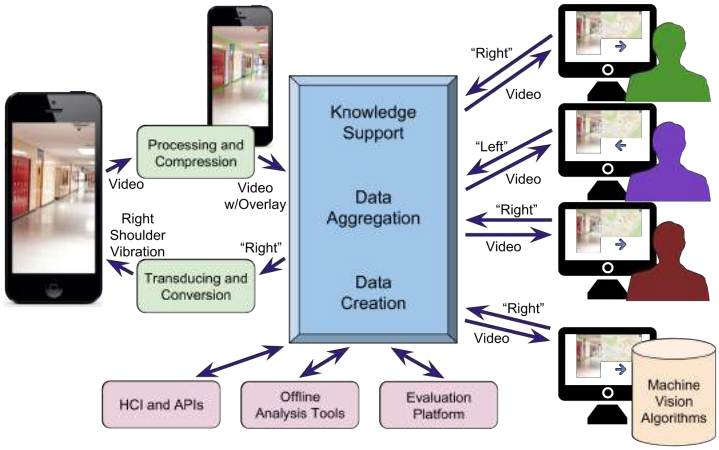


Figure 5.1: CrowdSource System Dataflow

To set up the stage for using a VE for evaluating the effectiveness of utilizing the information from the crowd for navigation, we first gave a brief overview of the crowd navigation system (see Figure 5.1) [47]. The system had to be developed which would allow the instructions from the crowd to be examined, aggregated, and fed back to users in a timely manner. Furthermore, the system had to be easy and interesting for the volunteers to use.

To accomplish this, a webapp[1] was developed using Google App Engine. Users were able to log into the webapp using a regular Google account. The visually impaired users then had the option to create a uniquely identified video stream which we refer to as a "room". When sighted volunteers logged onto the service, they entered any of the existing rooms and provided instructions. For any given room, all the instructions from all the crowd members were collected and aggregated via various methods.

Each of the users were then "graded" on their inputs and given points for doing a good job (or had points deducted for malicious behavior). These points were then used in later instances to give more or less weight to each user's feedback.

Members of the crowd were presented with an interface (Figure 5.2) designed to allow them to best assist the VIPs. A panel displayed the video being streamed from the VIP's

[1]CCNY Visual Computing Lab, "Crowd Assisted Navigation", November 5, 2015, `http://crowd-navigation.appspot.com/`

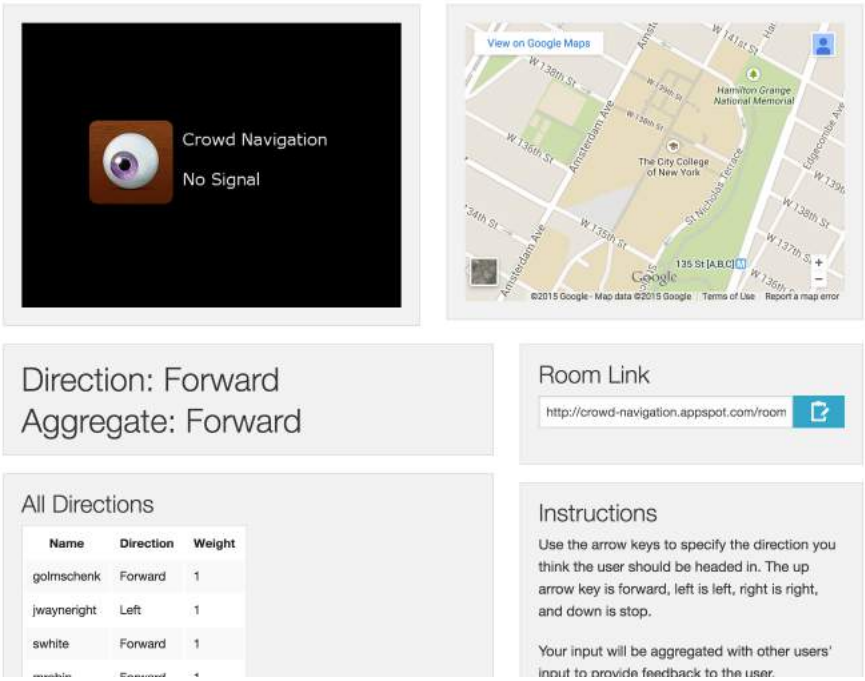


Figure 5.2: A screenshot of the webapp

phone, and location data from the phone was also used to give a GPS position for users on a map. The crowd member's feedback was displayed, along with all other crowd members' instructions and the aggregate, which was sent to the VIP.

In the current iteration of the webapp, the crowd members were only able to choose between four directions as feedback: forward, left, right, and stop. This limitation on the level of feedback was specifically chosen so that our testing would be simplified and variables were better controlled. Once the usefulness and limitations of this system are better comprehended, we intend to extend its features and capabilities.

One of the major concerns and areas of focus for this experiment is how the crowd feedback is aggregated. The naïve approach would have been to just simply relay every instruction given from the crowd directly back to the visually impaired users. This of course would have lead to an overwhelming amount of feedback, possibly causing conflicts. Many of the crowd members may have had different plans as to how users should have proceeded and the constant changing of the instructions would have been no help at all.

A more reasonable choice was to take the aggregation of the instructions given from the crowd and send that back to users. This way, only the primary opinion came through to users (*i.e.,* we computed the average over a given time interval relative to the user's request). However, even this raised issues of time length and delay.

Another alternative for the aggregation then was the use of a legion leader. Given all the instructions, the crowd member who most closely matched the overall opinion of the crowd was chosen as the "leader" for the next time interval. The leader was given complete control during that time interval, and only the leader's instructions were returned to the visually impaired user.

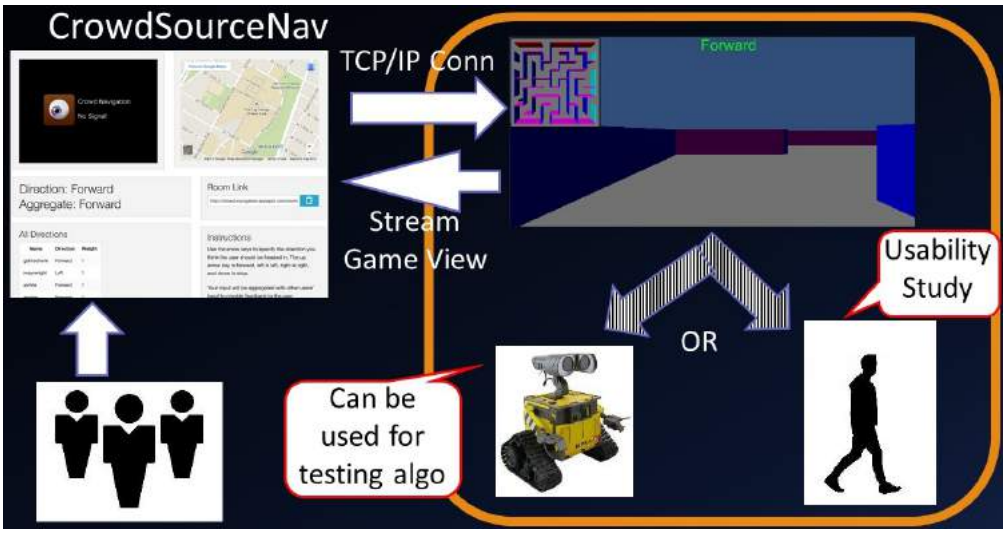


Figure 5.3: CrowdSourceNav experiment setup

Figure 5.3 illustrates the experiment setup for CrowdSourceNav. In VE evaluation, we simply replaced the VIP's phone camera with a video stream generated from the GIVE-ME Framework (orange bounding box in the figure). The VE-based mazes were connected to the webapp via screen capturing and TCP/IP connection, where commands (*i.e.,* left, right, forward) were retrieved. The commands were then displayed in the top central view as well as text-to-speech. Subjects, in turn, controlled the avatar via a joystick. In the following, we discuss the major components of the implementation.

5.2 Controller and Its Setup

Two controller options were provided to the VE: keyboard and joystick. Ultimately, the joystick was used because of its intuitiveness. The keyboard option worked in similar fashion using only the four arrow keys. Since both keyboard and joystick are standard devices, they are connected to the VE via plug-and-play. Similar to BrainportNav, CrowdSourceNav receives four commands (*i.e.,* forward, left, right, and stop). If a "Forward" command is received, subjects push the joystick forward or press the up arrow key to move the avatar forward until a new (different) command is received. If a turning command (*i.e.,* left or right) is received, subjects push the joystick left/right or press the left/right arrow key to rotate the avatar in place at a constant speed in the corresponding direction until a new command is received. If a "Stop" command is received, subjects just let go of the joystick or keyboard, which will stop the avatar from moving and rotating until a new command is received.

5.3 Stimulator

The stimulator for CrowdSourceNav is audio, specifically text-to-speech. The text-to-speech plugin is incorporated into the VE and speaks the command whenever a new one is received. Furthermore, working in conjunction with the controller, the current command is audibly repeated whenever users pull a trigger on the joystick or right click on a mouse. The plugin was obtained from Unity3D's community[2]. It is compatible with 32-bit and 64-bit Windows. Once imported into Unity3D (by dragging the DLL into Unity3D Editor), we call the exposed API with a string, which invoked the host Windows machine to annunciate the string.

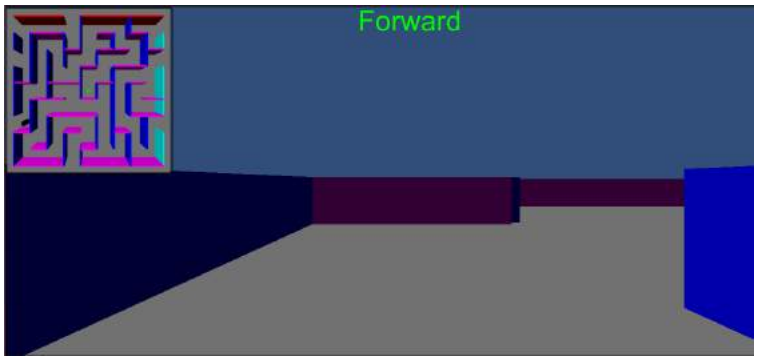


Figure 5.4: A screenshot of the game. This is streamed to the left panel of the webapp

5.4 Virtual Sensor and Transducing

In the real application, smartphone cameras were used to stream video to the online portal, where crowd members logged in to the webapp and were presented with a live stream video (left panel of Figure 5.2). Similarly, the virtual sensor in this experiment was a camera, specifically the game view (Figure 5.4). Streaming the game view to the online portal was done via a screen capturing software, ManyCam[3].

It may seem counterintuitive, but the transducing part in CrowdSourceNav is a group of online crowd members/people. Imagine the human vision system as the sensing, processing and transducing units that by looking at the streamed view, turn images into navigation directions. In essence, there are multiple sensors. Furthermore, this part can also be substituted with machine vision (as shown in Figure 5.1), where the stream view (from VE or smartphone) can be fed into the algorithm and compute a direction.

Based on a provided destination query, the crowd members guided subjects (or avatar) to the destination using the webapp and four directional commands. Not all crowd members' commands, however, were routed to the subjects (or avatar) directly. Their commands were, first aggregated on the webapp using one of the aforementioned aggregation methods. Then a custom written Python script residing on the same machine as the VE, queried the webapp for the command via standard HTTP GET. Finally, the Python script sent the command

[2]Unity Technologies, "Unity Community", November 5, 2015, `https://unity3d.com/community`
[3]Visicom Media Inc., "ManyCam", November 5, 2015, `https://manycam.com/`

to the VE via an established TCP/IP connection and displayed it in the top central view, as well as annunciated it in text-to-speech.

5.5 Game Mechanics

The task definition of this experiment was very simple: navigate the subjects or avatar from start to finish without bumping into a wall and for the crowd only, use the webapp to provide commands. The avatar behavior in this VE can be described as two navigation modes: 1) automatic; and 2) manual. In automatic mode, the avatar faithfully obeyed the aggregated command received. If a "Forward" command was received, the avatar moved forward until a new (different) command was received. If a turning command (*i.e.,* left or right) was received, the avatar rotated in place at a constant speed in the corresponding direction until a new command was received. A "Stop" command could have also been issued, which stopped the avatar from moving and rotating until a new command was received. In manual mode, the avatar was controlled by a subject, who may or may not have been a VIP, via a joystick. In either case, the subject was not be able to see the view and was only able to hear the command being spoken via text-to-speech. The subject could have the command repeated, if he or she missed it, by pulling the trigger on the joystick. Subjects could have taken a wrong turn, and in this case, the crowd would have had to correct the subject.

5.6 Environment Design

In light of the task definition, five mazes were procedurally generated, where each had a single path from start to finish. Having had the subjects go through five mazes not only produced enough data for analysis, but also prevented them from memorizing any path. All mazes were single floor and only contained one type of obstacle (*i.e.,* walls). To generate a maze, a starting location was specified by the experimenter and then the generation procedure started. As the maze was generated, a path was computed, and was guaranteed to terminate

at the border of the maze. The end of the path was designated as a finish location and could be visually verified (by experimenter) by a floating box at the location. Although the mazes were procedurally generated, we had fixed them by storing the seed value, for repeatability. The crowd members guided subjects (or avatar) down hallway-like mazes to destinations with walls on either side. The maze had dimension of 80 feet by 80 feet (assuming 1 Unity unit is 10 feet). When an avatar was being guided, we studied the effectiveness of the selected aggregation method and the crowd intelligence. When a subject was being guided, we studied the interaction between crowd members and the subject.

5.7 Measurement Device & Data Collection

For this experiment, the measurement device used was a survey conducted at the end of the experiment. The survey used a scale of $1 - 7$, where 1 was strongly disagree and 7 was strongly agree, to the following statements:

1. It is useful;
2. It is easy to use;
3. It is user friendly;
4. I learned to use it quickly; and,
5. I am satisfied with it.

The results of the survey are presented in the next section. The features we collected for this experiment are as follows:

1. Time: From start to finish locations.
2. Position: 3D position of avatar.
3. Heading: 3D vector of avatar's forward heading.

4. Turn_Direction: Turn direction that was received from the webapp and crowd.

5. Bump: Whether the avatar was bumping into an obstacle.

6. UserInput: User's response, if any.

The data collection procedure had a 20 ms interval. In other words, every 20 ms, the above data were recorded to a file. *Time* serves as a timestamp for the other features and it is reset to 0 at the beginning of each maze. Thus, the last timestamp indicated the time (in seconds) it took the crowd members to guide the avatar/subject from start to finish. Every 20 ms, the 3D *position* of the avatar and its forward *heading* (a unit vector) were recorded. The refresh rate of *turn direction* varies depending on whether subjects needed corrective feedback to get back on track or reached the next waypoint. Regardless, every 20 ms, the current turn direction was recorded. Similarly, *bump* indicates whether the avatar was in a state of wall collision or not. Lastly, *UserInput* indicates the action subjects took, if in manual navigation mode. The automatic avatar always responded to the command correctly.

Data is also recorded on the webapp side. Data such as each individual crowd member's response, the aggregation method (*i.e.,* simple sum or legion leader), the aggregated response that was sent to the VE, the "leader" (only applicable for the legion leader method), and timestamp. Table 5.1 summarizes the variables for which data were collected. All analysis and graph generation were done through a custom written Matlab script. The next section shows the results.

Independent	Aggregation method	Crowd size	
Dependent	Completion time	Number of errors	Shortest distance to destination, if timed-out

Table 5.1: Independent and dependent variables.

5.8 Results

A total of 27 undergraduate subjects was recruited and given written informed consent to take part in the experiments for partial fulfillment of a course requirement. This study was approved by the CUNY IRB.

We conducted two sets of experiment. In the first experiment, we generated five different mazes for testing. Sixteen crowd volunteers participated in the first experiment. For each maze, we recorded the completion time and the number of times the crowd-directed avatar came into contact with walls. The ground truth times were obtained by the experimenters navigating the mazes locally (*i.e.,* without going through the webapp) and without inputs from the crowd. Table 5.2 shows the crowd completion and ground truth times for all five mazes. The results showed that the crowd completion time was significantly different from the ground truth time (Two-sample t(8) = 5.03, p=0.001 at 5% significance level, N=10).

Table 5.2: Crowd vs ground truth times in first experiment.

Maze #	1	2	3	4	5
Crowd time (s)	513.94	345.47	325.00	258.87	505.94
Truth time (s)	131.82	114.88	156.22	124.29	124.04

Figure 5.5 shows the trajectories of the crowd in mazes 1 - 3, using a simple average method. The x-axis and y-axis show the 2D position of the avatar in a top down view (the height dimension is ignored since it didn't change). Since each trial generated several thousand samples, and given the sampling interval, most samples are redundant. The graphs are generated by downsampling the samples by half, if not more. The original sample size is 25,700. Each graph shows the origin (green circle), destination (red circle), the path taken by the avatar, and the ground truth path. The path also denotes where the avatar bumped into a wall (red line with arrows) and 30-second interval (gray squares). While the avatar in mazes 1 and 3 moved exactly to the crowd's response, maze 2's avatar was controlled by a VIP subject. The subject navigated the maze based on the crowd's feedback. The subject was able to complete the maze without bumping into any walls. During the first

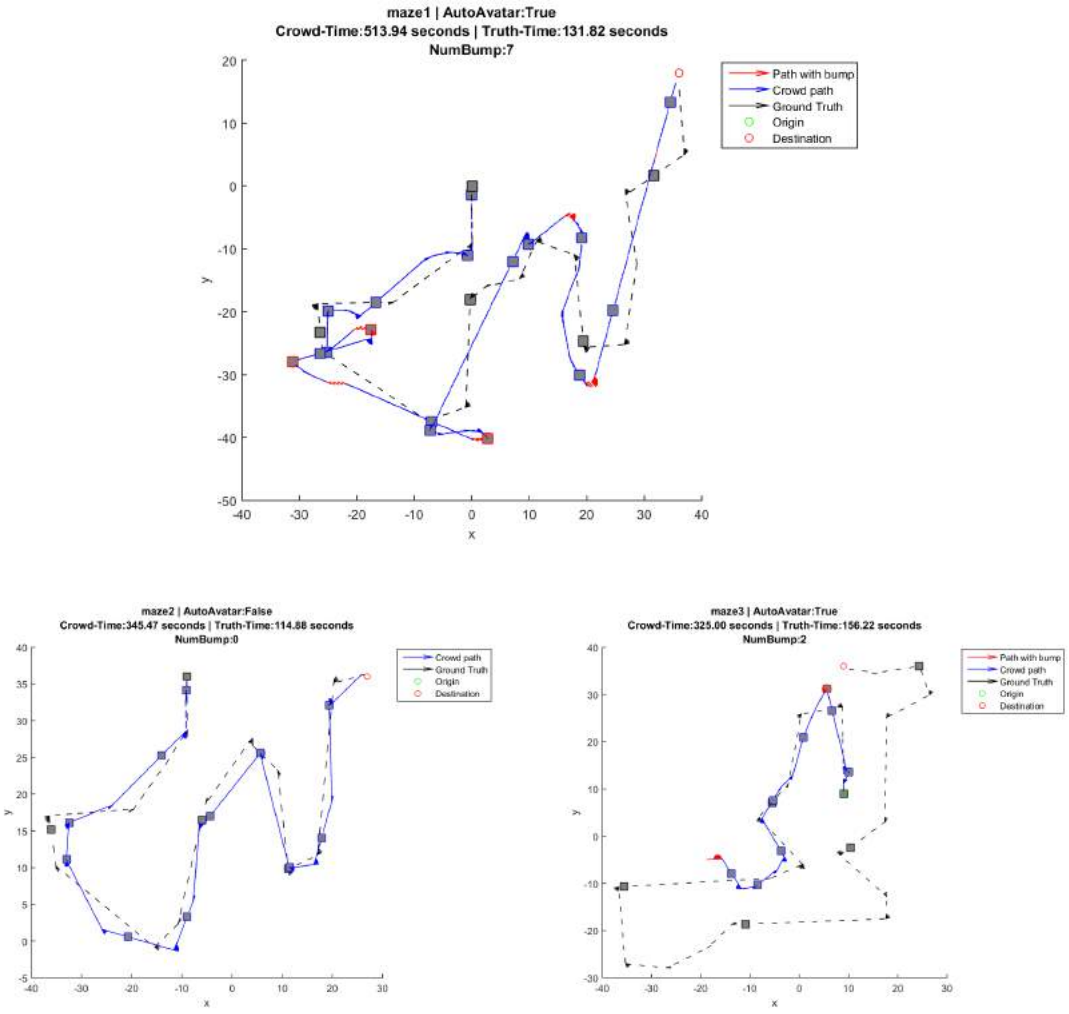


Figure 5.5: Top down view of mazes 1 - 3 in first experiment, showing crowd paths, paths with bump, ground truth paths, origin, destination, and 30-second interval (gray squares)

experiment, network issues resulted in a blurry and broken video streaming. With the entire group of crowd members being on the same network, the issue caused significant problems in the first experiment, particularly in the later mazes. For these mazes, the wall bumpings were attributed to slow or delayed network and low resolution of the video stream. The end-of-experiment survey also reflected this opinion.

For the second experiment, we constructed mazes with larger wall and objects such that even when streamed in low resolution, the view was easy to see. We also worked to ensure proper network connection by streaming the view from a high-bandwidth location, slowed

down the automatic avatar's speed, and increased the objects' size (*e.g.,* walls). Furthermore, the second experiment included the legion leader aggregation method. Eleven volunteers participated as crowd members for the second experiment. The number of volunteers in this experiment differs from the previous because we wanted to see how the number of volunteers affected the aggregation methods, and thus the performance of the subject or avatar navigation through the mazes.

Table 5.3 shows the crowd completion times for both simple sum and legion leader aggregation methods. Maze 4's avatar was controlled by a VIP subject. The subject navigated the maze based on the crowd's feedback. The subject was able to complete the maze faster using the legion leader aggregation method. Although the result showed that crowd completion time for either aggregation method was not significantly different (two-sample t(6) = -0.84, p=0.432 at 5% significance level, N= 4+4=8), the crowd time in Table 5.2 vs. simple average crowd time in Table 5.3 was significantly different (two-sample t(7) = 2.10, p=0.074 at 10% significance level, N= 4+5=9). The number of mazes in this experiment differed from the previous because a subject withdrew from the experiment.

Table 5.3: Crowd times in second experiment.

Maze #	1	2	3	4
Simple avg time (s)	221.64	180.27	292.86	322.79
Legion leader time (s)	219.65	182.41	263.50	228.89

Figure 5.6 shows the trajectories of the crowd in mazes 1 - 3, using a legion leader method. The x-axis and y-axis show the 2D position of the avatar in a top down view (the height dimension is ignored since it didn't change). Since each trial generated several thousand samples, and given the sampling interval, most samples were redundant. The graphs were generated by downsampling the samples by half, if not more. The original sample size was 11,000. Each graph shows the origin (green circle), destination (red circle), the path taken by the avatar, and the ground truth path. The path also denotes where the avatar bumped into a wall (red line with arrows) and 30-second interval (gray squares). The avatar in mazes

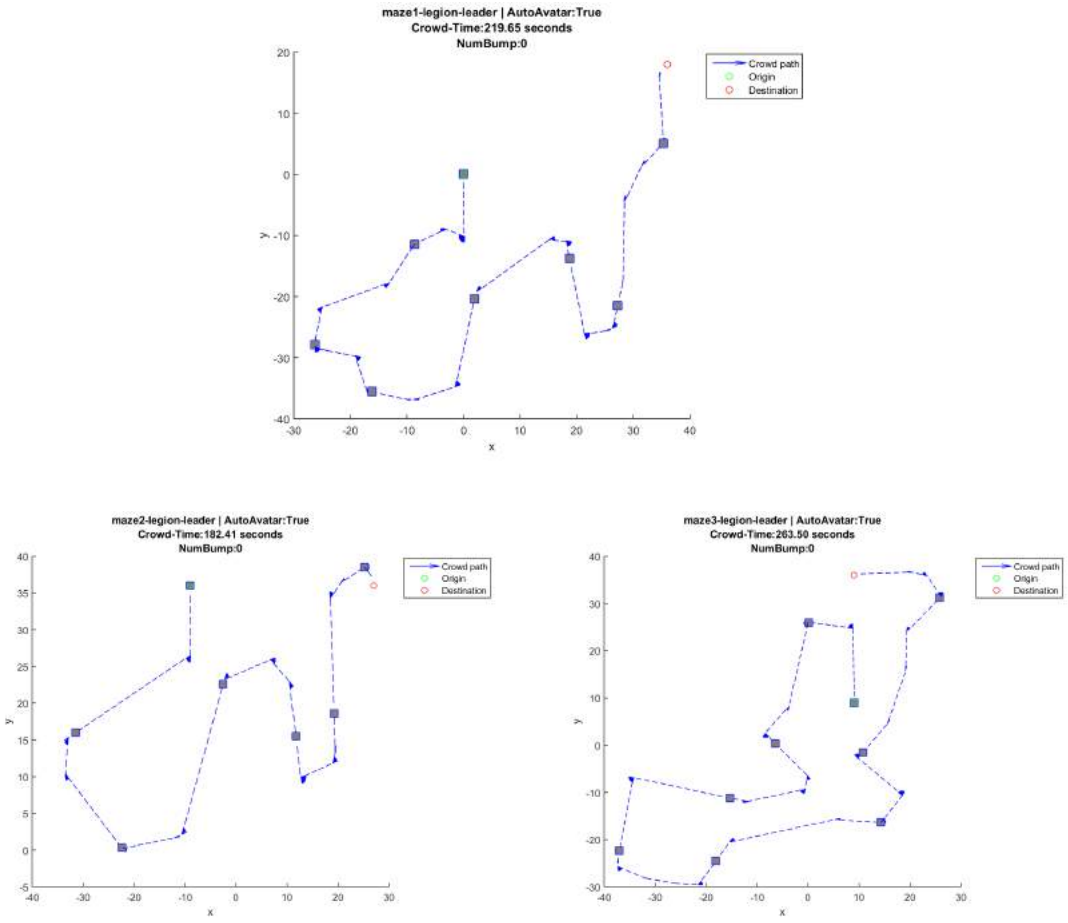


Figure 5.6: Top down view of mazes 1 - 3 in second experiment, showing crowd paths, paths with bump, ground truth paths, origin, destination, and 30-second interval (gray squares)

1 to 3 moved exactly to the crowd's response. The overall trajectories in this experiment were much smoother than those in Figure 5.5. The crowd was also able to navigate the avatar to its destination in a shorter time. The overall feedback from the crowd members was positive and much improved from the first experiment. The crowd was able to clearly see the mazes and navigate the avatar or subject without much network delay.

Table 5.4: Survey Results

Statement #	1	2	3	4	5
Average	5.27	6.13	5.93	6.47	5.20

Table 5.4 shows the averaged results of the survey responses (sample size of 15). To reiterate, the survey statements are 1) It is useful; 2) It is easy to use; 3) It is user friendly; 4) I learned to use it quickly; and 5) I am satisfied with it. Overall, most people agreed with the statements, scoring at least a 5 (out of 7). Most of the constructive criticism was to improve the map and improve connection bandwidth/streaming speed. Some of the positive comments included "Aggregate algorithm looks pretty refined. Good job!", "Good System", and "The application definitely have it's (sic) potential in becoming a very useful app for navigating a visually impaired [person]through an environment."

5.9 Discussion

We found that using GIVE-ME to evaluate crowd-sourced navigation alleviated our concerns for subjects' safety, especially given the unpredictable behavior of subjects and the unknown environment. We had suspected that network bandwidth might be an issue, both in real and virtual cases, and this experiment confirmed our suspicion because video was streamed at such a low resolution (due to dropped frames) that crowd volunteers had difficulty determining which direction to direct the avatar. These findings helped to inform us of what future development and experimentation with network bandwidth we needed to keep in mind. Without GIVE-ME conducting similar experiments in the real world would have been very difficult for two reasons. First, since the crowd volunteers were directing an avatar in a maze navigation, we did not have to worry about subjects' errors, as we would have had to in real world navigation experiments. We could have introduced such errors into the experiments by having users control the avatar based on directional feedback received from the online crowd volunteers. Second, GIVE-ME afforded rapid testing of the aggregation methods (simple average and legion leader) and various crowd sizes in a single session. Without GIVE-ME, if we had changed these parameters for experimentation, it would have been time consuming.

While eventual and extensive testing in the real world with crowd-sourced navigation would be required, the VE experiment provided training to the online crowd volunteers in directing users to navigate from one room to the next[66]. The online crowd volunteers learned that they needed to preemptively provide directions to turn before users actually reached the point of turning. This was probably due to network delay. Such knowledge was also transferred to the real world because the crowd volunteers used preemptive instructions in testing sessions. Most crowd volunteers also learned that if users made a mistake by making the wrong turn and continued walking, the volunteers issued a stop command instead of a corrective turn feedback, especially when turning corners in a narrow hallway. The crowd volunteers also brought this knowledge to real world testing; volunteers tended to give a direction (*e.g.,* forward, left, or right turn) followed by a stop command, to ensure users did not bump into an obstacle. The knowledge and skills acquired by both the volunteers and users in the virtual experiment were observed during the real world testing, which made the transition from the virtual experiment to the real world experiment smoother and less time consuming. This was because less time was needed on learning the user interfaces and the meaning of feedback.

Chapter 6

Application 3 - VibrotactileNav

In this application, we proposed a wearable VISTA (*Vibrotactile Intelligent System for Travelling Aid*) to enhance a person's awareness of their surroundings through the use of touch. This was achieved by placing range sensors, coupled with small vibrators, on their bodies. This allowed subjects to feel objects and obstacles in close proximity to them without having to physically touch them. Other research groups have sought to use haptic or vibratory feedback to provide navigational information to the blind, but they have concentrated on providing feedback that either conveys navigational information or enhances the use of another device, such as a white cane [16, 68, 85]. Our VISTA device design enhances a person's sense of physical awareness with their surroundings by providing feedback that directly corresponds to distance to nearby obstacles. To this end, we constructed an array of connected vibrators and range-sensors that are as small, modular, and reconfigurable as possible. We have also built small armband devices for the vibrators that are worn as close to the skin as possible. These connect wirelessly to range-sensing armbands that are worn on top of any clothing users might be wearing [61].

We specifically explored the development of small units that are worn around the body to provide direct feedback to wearers about obstacles in their immediate surroundings. Figure 6.1 shows one of the early prototypes that illustrates the basic principles of our approach.

In this example, seven pairs of range-vibrator sensors were sewed onto a shirt for upper-body range sensing. A person wearing our range sensors and vibrotactile stimulation devices experienced a tactile sensation that has been described as having a "range force field" around the wearer, causing alerts whenever a part of their body is near a wall or obstacle. By using parts of the body, we minimized potential interference to senses that could be used for other tasks, such as hearing.

Figure 6.1: Prototype device designed to be embedded in a shirt, showing the concept of whole-body wearable. The device (upper-body) consists of seven sensors and vibrators connected to a central control box.

6.1 Experiment

Previously, we tested sensitivities of various parts of the body where we assumed placement of the range-vibrotactile pairs would be useful. In particular, we tested the elbows, shoulders, and wrists, as shown in Figure 6.1. We presented subjects with sensations of more and more intensity of vibration, until they indicated that they felt the same. We found that, on average, subjects could discern about 3 - 4 levels of vibration (with voltage from 0 to 5 Volt) [46, 68].

Using the initial results, we performed experiments in virtual environments to investigate whether efficient navigation could be achieved using 3 - 4 levels of vibration. Experiments in the VE allowed for a rapid design and evaluation of the range-vibrotactile device, before we performed large-scale user testing in real environments. The experiments involved subjects

wearing the vibrotactile shirt (Figure 6.1) and guiding the avatar to a destination. The objective here was to reach the destination with a minimum number of collisions. Subjects, however, were not allowed to view the screen, but instead had to rely on audio and haptic feedback for completion. This experiment was also designed as a game called "Chicken Finder" because the destination had an audible "chicken chirping" sound, which subjects could rely on for orientation.

Figure 6.2: VibrotactileNav experiment setup

Figure 6.2 illustrates the experiment setup for VibrotactileNav. The VEs were connected to the Arduino via USB/serial port connection, where sensory data were transduced to subjects through Arduino to vibration motors. The subjects, in turn, controlled the avatar via a joystick and mouse to reach their destination. The following subsections describe each part of the setup in detail.

6.2 Controller and Its Setup

Initially, we used a joystick as an input controller. We noticed quickly that subjects had difficulty completing the experiment due to their inability to orient the avatar in the VE. For that reason, we replaced the joystick with a modified a mouse (Figure 6.3) to give subjects a better sense of direction. A regular mouse does not give one any feedback, visual or physical, when one rotates. Thus, one has no idea how many degrees one has turned in the virtual environment. To remedy this, we built a steering device by cutting a mechanical (roller ball)

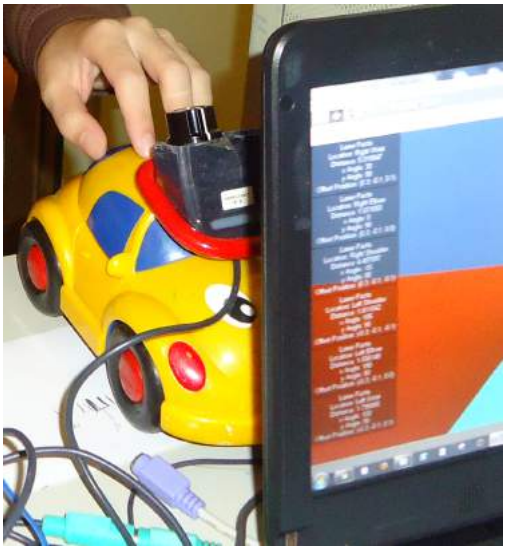


Figure 6.3: Modified mouse

mouse in half to expose one of the rollers. We then attached a rotary dial knob to the roller, by which subjects had use to steer. This fix ensured that when subjects rotated the knob 90°, the virtual avatar also rotated 90°. The knob also provides a visual (a white line) and tactile (a bump) cues of which orientation the avatar was facing. The joystick was still used, but only for translating the avatar. The modified mouse is used only for rotating the avatar. We mounted the modified mouse on top of a toy car, for stability and for presenting the knob at a 45° angle (relative to the table) to subjects. The joystick and modified mouse were then connected to the VE via the standard plug-and-play.

6.3 Multimodal Stimulators

We used two types of stimulators in this experiment; haptic and audio. For ***haptic*** feedback, we used the shirt shown in Figure 6.1, but with only six units for this experiment; three on each arm: one on the upper arm, one on the elbow, and one on the wrist. Each unit was connected to the central control box via wires. The sensor components on the shirt, however, were disabled as it simulated in VE, which we describe in the next section. The central control box is connected to a breadboard, which in turn, is connected to Arduino

UNO's digital pins. Lastly, the Arduino is connected to the VE via a USB. In short, the stimulators were controlled from the VE by writing voltage values to each vibrators, via the connected serial port. The higher the voltage was, the stronger the vibration was. A custom Arduino script was already loaded into Arduino that interpreted commands from the serial port and wrote the voltage values into the corresponding digital pins. For ***audio*** feedback, we used three sound cues. They were an "ouch" sound when the avatar bumped into an obstacle, the avatar's "footstep" sound, and a "chick chirping" sound (such sound clips loop forever). All sound clips used were royalty free, and directly imported into Unity3D and placed on the appropriate model.

6.4 Virtual Sensor and Transducing

Figure 6.4: Six IR range sensors and one sonar sensor configuration on an avatar.

Figure 6.4 shows a visualization of the infrared and sonar simulations. Two different sensor configurations (including types of sensors, numbers of sensors and their locations) are shown. There were three IR range sensors attached to each arm; one on the wrist, one on the elbow, and one on the upper arm (or shoulder, depending on the subject size), all pointing outward. This is illustrated in Figure 6.5. This was as if a person was walking with their arm raised in front of them, elbows bent. The sensors were mounted on their wrists, elbows, and

shoulders at 30°, 90°, 100° angles, respectively. There was also a sonar sensor on their chest facing forward. The white object in front of a person was a speaker icon which indicated that there was an audio clip attached to the person (This was only visible in Editor mode, allowing us to get a better screenshot). On the left side was a red wall. One could see some IR rays hitting the wall and a ray hitting the obstacle in front of it.

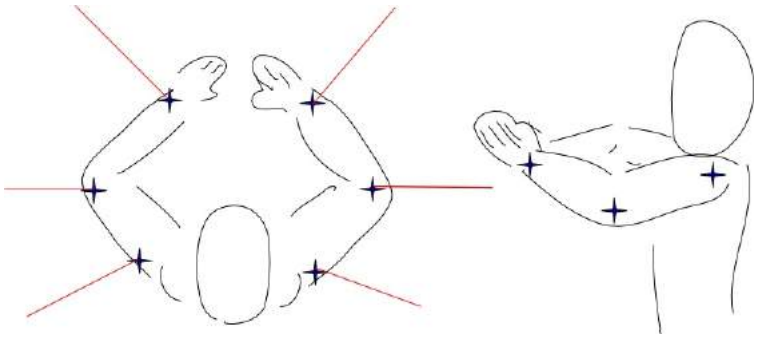


Figure 6.5: Placement of sensors in Unity3D for testing

Both types of sensors were simulated using Unity3D's raycast function as described in Chapter 3.3.1. Each virtual sensor was placed on the avatar in such a way that the raycast vector was perpendicular to the surface (*i.e.,* parallel to the surface's norm at the point of placement). For infrared, a single raycast was used with a maximum range of 1 meter. If the raycast were to hit an obstacle within 1 meter, it returned the distance to that obstacle. Otherwise, no result would have been returned. For sonar, we simulated a cone shaped profile, with the tip of the cone on the avatar's body. The cone was simulated with 10° field of view, both horizontally and vertically (*i.e.,* a cone with a radius of ≈ 0.6). For every angle in the x and y axes of the cone, a raycast was initiated with a maximum range of 8 meters. Of all the raycast that returned a hit within 8 meters, only the shortest distance was recorded (*i.e.,* the virtual sonar detected the closest obstacle within the maximum range). The sonar sensor, however, was not used in this experiment, but it showed that it was possible to use.

The VE polled all virtual sensors every 20 ms for distance readings, even though the game engine updated the sensor's output around 60 Hz. Then the readings were thresholded into three categories: low, medium, and high. Each category dictated the voltage value

(*e.g.,* high means high voltage value) which was written onto the serial port's buffer. For example, *low* covered distance readings interval $\left(\frac{4}{3}, 2\right]$ meters, for an obstacle that was far and thus resulted in a weaker vibration. *Medium* covered interval $\left(\frac{2}{3}, \frac{4}{3}\right]$ meters and *high* covered $\left[0, \frac{2}{3}\right]$ meter.

Finally, the category, and which motor was activated, was sent to the Arduino. The categorizations and motor selection were first encoded as a string and then sent to the Arduino via a serial port connection (*i.e.,* the COM port that the Arduino is connected to). The script within the Arduino then parsed the incoming string to determine which motor was to be activated and with what pre-fixed intensity value. Motor activation was achieved by writing to the corresponding pin with an intensity value.

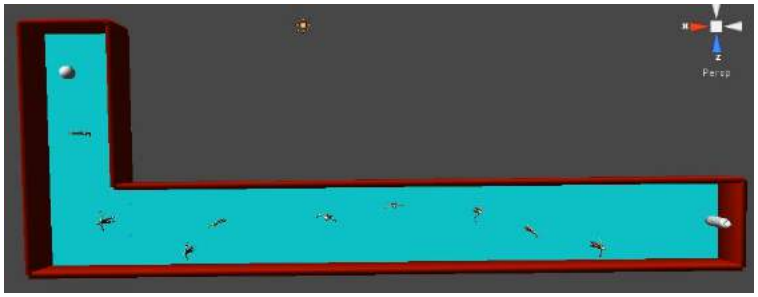
6.5 Game Mechanics

The task definition for this experiment was to navigate the avatar from start to finish without bumping into a wall or stationary obstacles, without looking at the computer screen, and relying only on audio and haptic feedback. The avatar behavior was very straightforward. Its forward motion was at a constant velocity, such that no matter how hard subjects pushed the joystick forward, the speed was the same. The avatar had unrestricted rotation, and subjects determined the avatar's orientation by using the modified mouse. Furthermore, while the avatar was in motion, a footstep sound was constantly playing. When the avatar bumped into a wall or stationary obstacle, an "ouch" sound was played once.

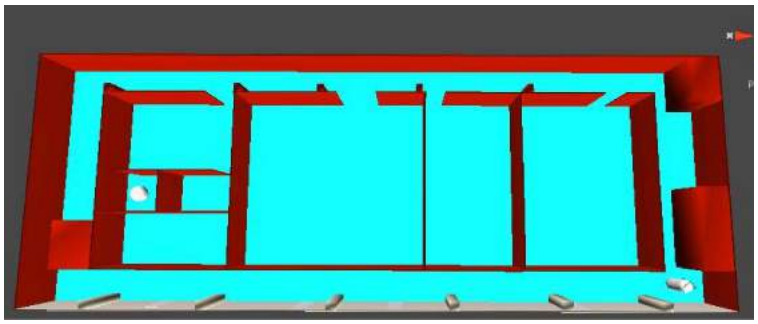
In summary, the game was set up as follows:

- Subjects have to find the source of the sound of a baby chick chirping without any visual information.
- The computer screen is faced away from the subject.
- Subjects have to navigate the virtual environment and avoid obstacles based on the varying intensity of sounds and vibrations.

6.6 Environment Design



(a) Easy Hallway



(b) Complex Hallway

Figure 6.6: Aerial view of two VEs

For this experiment, we generated two levels (of VE). Figure 6.6 shows the levels we created. The first level is an easy L-shaped hallway. It has a dimension of 60 feet by 20 feet (assuming 1 Unity unit is 10 feet). The avatar started in the right-end hallway and proceeded down the hall while avoiding the wall and stationary obstacles, and reached the white sphere at the end of the hallway (left-end). The white sphere was also the source of the chirping sound. The second level was a complex hallway based on an actual floor on the City College campus. It has a dimension of 65 feet by 25 feet (assuming 1 Unity unit is 10 feet). At this level, however, there were no stationary obstacles, just the layout. This level contained a perimeter hallway, three large-sized rooms, one medium room, and two small rooms. The white sphere was located in one of the small rooms. The task here was the same as before: an avatar started on the right side of the map and reach the destination on the left side.

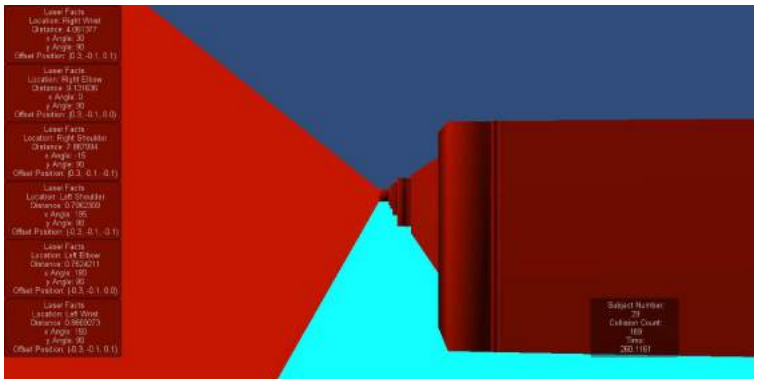


Figure 6.7: Screenshot and view of the screen (not seen by subject)

Figure 6.7 shows the first-person view in the complex hallway (Figure 6.6b). Note that this screen is not seen by subjects while the experiment is running. The experimenter, however, was able to see all of the information on the screen, such as distance information for each sensor, time, and the number of times subjects bumped into something.

6.7 Measurement Device & Data Collection

For this experiment, no measurement device was used. The features we collected for this experiment, however, were as follows:

1. Time: From start to finish locations.

2. Position: 3D position of avatar.

3. Heading: 3D vector of avatar's forward heading.

4. Bump: Whether the avatar was bumping into an obstacle.

The data collection procedure wrote the above data to a text file every 20 ms. It recorded the time that each vector of features was written to a file. The last vector also indicated the finish time. It also recorded the avatar's position and orientation in the VE, as well as a flag that marked whether the avatar was bumping into an object. Bumping was recorded

because it allowed us to evaluate how well subjects performed and where they bumped into something. The next section shows the results.

6.8 Results

Eighteen subjects (a range of 18-24 years in age) gave written informed consent and took part in the experiments either for monetary compensation or for partial fulfillment of a course requirement. This study was approved by the IRB of CUNY.

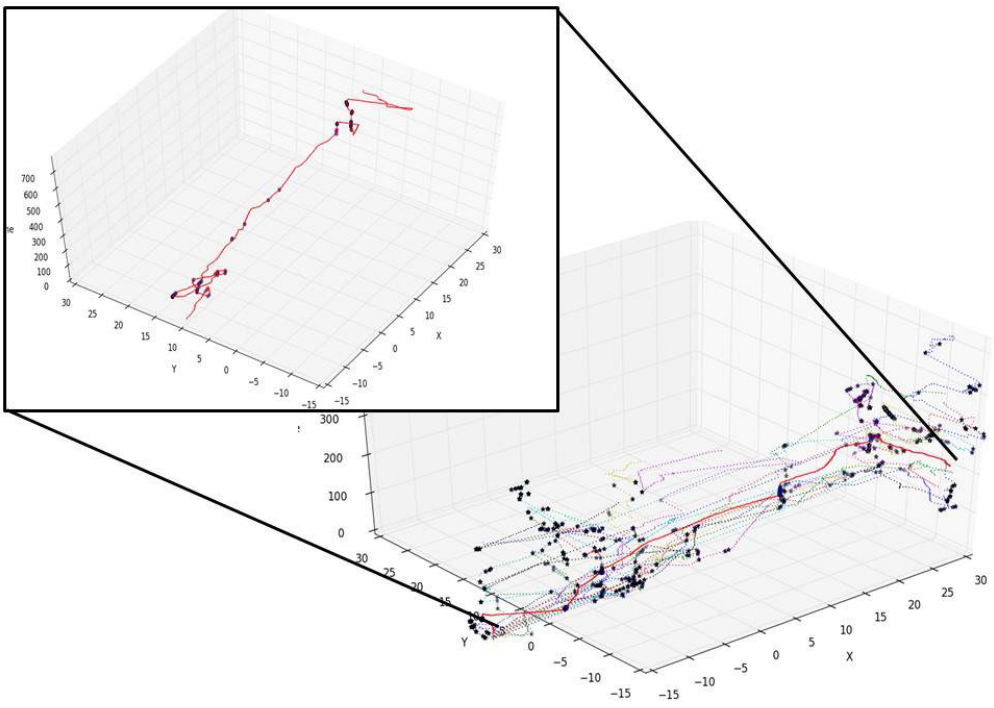


Figure 6.8: Trajectory plots of subjects in Easy Hallway

The data processing was done using a custom written Python script. Because data were sampled and recorded on every frame, this experiment generated a large amount of data. One way we visualized the result was to plot the subject's trajectory through the hallway, as shown in Figures 6.8 and 6.9. The plots also show where subjects have bumped into something, as indicated by a black cross in the figures. For illustration purposes, we superimposed a single trial of a subject onto all trials for both cases of easy and complex

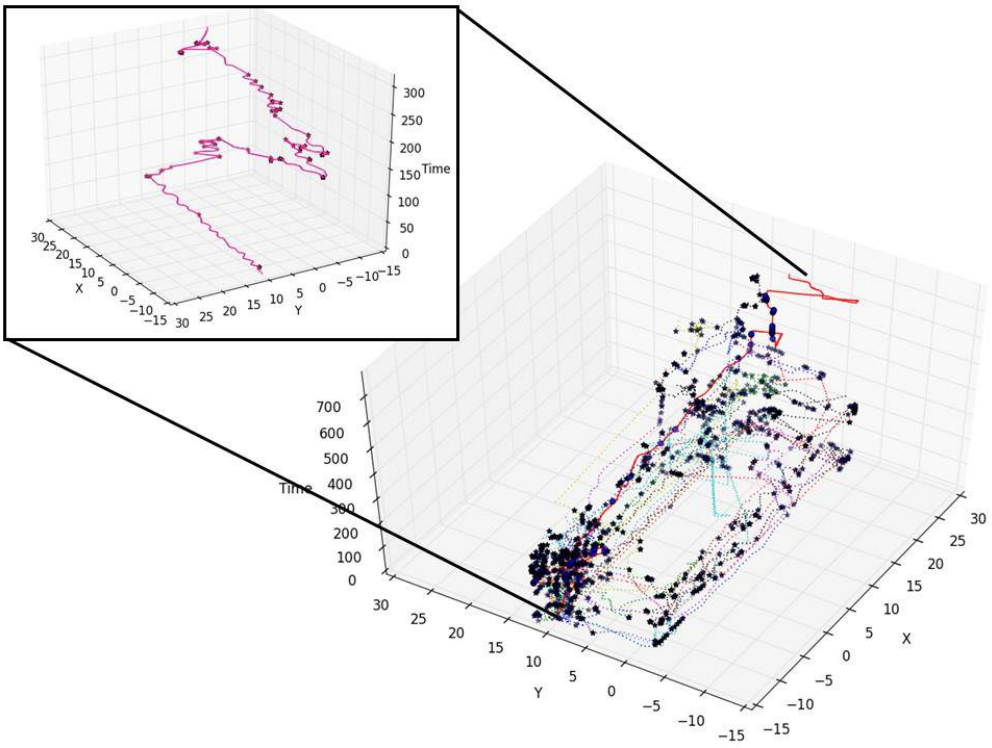


Figure 6.9: Trajectory plots of subjects in Complex Hallway

hallways. The x-axis and y-axis show the 2D position of the avatar in a top down view (the height dimension is ignored since it didn't change). The z-axis shows the number of samples used to graph (*i.e.,* time-step).

Most subjects were able to find the goal object in the Easy Hallway (Figure 6.6a). There were 18 subjects that tried the hallway and nine were able to find the goal. Table 6.1a shows the time to completion and the number of bumps for subjects who experimented in the Easy Hallway. The subject numbers between Tables 6.1a and 6.1b may not correspond to the same subject, since some subjects participated in one hallway and not the other.

The average time is 280.10 seconds and the average bumping is 17.3 for those who succeeded. For those who failed, the average time is 288.65 seconds and the average bumping is 22.1. We speculate that the similar numbers are due to each individual's learning curve, and the fact that while some people are tenacious in finding the goal, others gave up easily.

Only three subjects out of 18 were able to find the goal object in the Complex Hallway shown in Figure 6.6b, and Table 6.1b shows the completion time and number of bumps

Table 6.1: Chicken Finder data: Time and Bumping

(a) For Easy Hallway

	Time (s)	Bumping	Result
S1	257.02	13	Failed
S2	246.12	18	Failed
S3	252.54	12	Failed
S4	339.16	26	Failed
S5	316.76	5	Failed
S6	286.54	17	Succeeded
S7	266.70	32	Failed
S8	145.34	21	Succeeded
S9	185.62	16	Succeeded
S10	150.56	4	Succeeded
S11	292.30	26	Succeeded
S12	325.18	65	Failed
S13	210.34	20	Succeeded
S14	305.74	6	Failed
S15	230.38	15	Succeeded
S16	527.36	17	Succeeded
S17	389.52	9	Succeeded
S18	383.08	28	Failed

(b) For Complex Hallway

	Time (s)	Bumping	Result
S1	58.32	13	Succeeded
S2	102.12	6	Succeeded
S3	200.30	19	Succeeded
S4	351.02	17	Failed
S5	412.08	3	Failed
S6	373.30	22	Failed
S7	602.32	27	Failed
S8	325.40	25	Failed
S9	241.86	60	Failed
S10	272.74	48	Failed
S11	311.78	34	Failed
S12	316.66	31	Failed
S13	472.40	60	Failed
S14	311.54	108	Failed
S15	315.04	66	Failed
S16	307.32	42	Failed
S17	306.22	36	Failed
S18	385.32	62	Failed

of subjects who participated in this scenario. The average time is 120.25 seconds and the average bumping is 12.7 for those who succeeded. For those who failed, the average time is 353.67 seconds and the average bumping is 42.7. One would assume that the Complex Hallway would have taken more time to complete; however, our average time to goal is 120.25 seconds, much less than the Easy Hallway. This could be attributed to subjects who had participated in the Easy Hallway scenario previously and were familiar with it, and so navigated it with relative ease. Note that some subjects participated in the Easy Hallway and not in the Complex Hallway, and vice versa. These differences cannot be identified from Table 6.1.

More people succeeded in the Easy Hallway than in the complex one for several reasons. In the Easy Hallway, the goal was just down a hall and around one corner, while the Complex Hallway required subjects to navigate through rooms to find the goal. While the vibrotactile

array was informative about objects and walls nearby, the environment feedback could have been confusing the subjects. For example, in the Complex Hallway, when the goal was very near, the goal could have actually been in another room while subjects were outside of it in the hallway. Openings such as doorways, were hard to detect, and thus reaching the goal became difficult.

6.9 Discussion

We found that subjects were very engaged in this experiment, to the point of showing competitiveness. Though some subjects were very tenacious in reaching the goal, others gave up easily. Some subjects even commented that the process was more game-like than an actual experiment session. We also found that, as a side effect, the experiment tested the subjects' ability to determine direction using 3D sound cues (*i.e.,* following the chicken chirping sound to reach one's destination). Without GIVE-ME, experimenting with range-vibrotactile pairs would be difficult for two reasons. First, the prototype was wired to a microcontroller, which had to be connected to a power source. A real world navigation experiment would have been dependent on the availability and range of a power source. A battery pack may have been possible, but none were available for this experiment. Second, precise data (trajectory and action) would not have been available without GIVE-ME. Without a large room and motion tracking system, we could not have reliably recorded each subject's position. Despite all of this, there were still limitations with GIVE-ME in this experiment. The experiment setup was designed in such a way that the six range-vibrotactile pairs were fixed in space; the pairs were placed on arms that were locked in place while walking. In reality, our arms have six degrees-of-freedom while walking, and the infrared sensors would have detected different distances and obstacles as our arms swung back and forth.

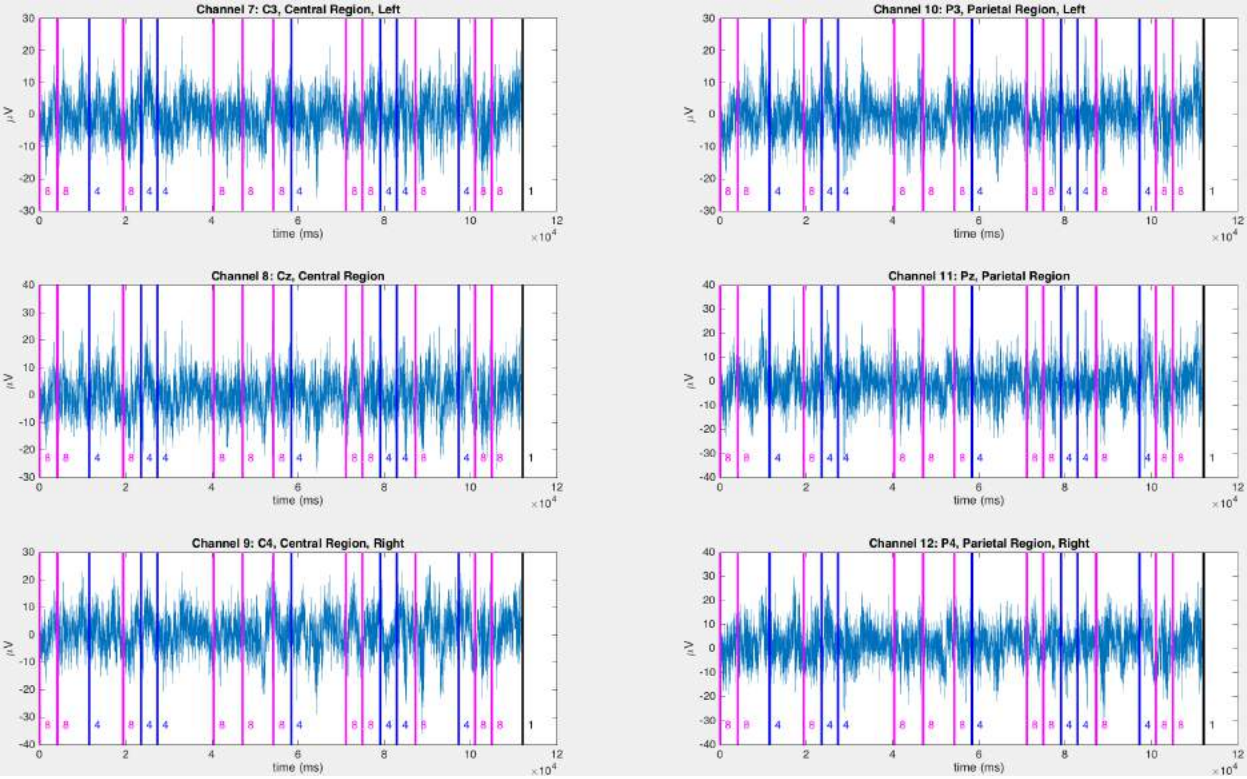


Figure 6.10: EEG data of a subject. Electrodes #7-12 with trigger values 8 (left turn), 4 (right turn), 2 (bump), and 1 (stop)

6.9.1 EEG Data Collection

To demonstrate the feasibility of connecting an EEG machine to GIVE-ME, we recruited six subjects for a pilot experiment. Unlike the settings in VibrotactileNav, this experiment involved only two stimulators: one vibrating unit on each wrist, to indicate a left or right turn. The virtual sensor is a pathfinding algorithm and functions similarly to that in Chapter 4.4. To signal the subject to make a left (or right) turn, the left (or right) vibrating unit was activated. A randomized maze layout similar to that in Chapter 4.6 was used.

The purpose of this experiment was to collect EEG data via a VE of subjects performing navigation tasks. It was not, however, to provide in-depth analysis of the EEG data. As mentioned in Chapter 3.2.3, the EEG device needed to be synchronized with the VE such that when a stimulus was presented from the VE, we could have expected the EEG measurements from that time forward to be a response to that stimulus, until a new stimulus was presented.

Synchronicity was achieved by using triggers (8-bit unsigned integers). The following trigger values were used in this mini-experiment: 1 (binary: 00000001) for stop; 2 (binary: 00000010) for a bump; 4 (binary: 00000100) for a right turn; and 8 (binary: 00001000) for a left turn. The trigger was sent by calling the parallel port plugin's *DlPortWritePortUshort* [13].

Figure 6.10 shows partial EEG data of a subject doing a navigation task in a maze. Each subgraph shows signal values which came from an electrode channel. The x-axis is the time series; the y-axis is the measured voltage in microvolt (μV). The left column of subgraphs shows electrodes on C3, Cz, and C4, which were placed on the central region of the scalp. From top to bottom, C3 was on the left side of the central region, Cz was on the middle, and C4 was on the right side. The right column of subgraphs shows electrodes on P3, Pz, and P4, which were placed on the parietal region of the scalp. From top to bottom, P3 was on the left side of the parietal region, Pz was on the middle, and P4 was on the right side. Each subgraph also had an overlay of trigger values marked as vertical lines in various colors. Magenta (8) signifies a left turn, blue (4) signifies a right turn, and black (1) signifies a stop. This subject did not make any bump, therefore, no trigger value of 2 (which would be a red vertical line) is shown.

The next step in this EEG study was to better understand what happens in the brain when a left or right turn instruction is given or when the subject made a mistake (*i.e.,* bumped into an obstacle). Using the triggers, we defined a time window for each of these events for analysis. For example, we averaged the EEG activity for all of the left and right turn instructions to show what happened in the brain for a task. We also averaged the EEG activity before and after the response. Such study provided an interesting set of insights on how visually impaired people (and blindfolded subjects) processed and retrieved spatial information for navigation. It was also interesting to see how they dealt with an unknown environment and how their brain activity changed when making a turn, or in anticipation of making a turn, or when walking down a long hallway.

Chapter 7

Application 4 - VistaNav

Figure 7.1: Wearable prototypes. (a) Prototype I (b) Prototype II (c) components of Prototype III (d) A blind person using prototype I while another blind person "watches" [61]

Since 2013, Vista Wearable, Inc. a spin-off of the City College Visual Computing Lab with the VISTA technology described in Chapter 6, has iterated several prototype versions and has conducted limited user testing. Figure 7.1 shows the various minimally functional prototypes. We were interested in evaluating Prototype III (we will call it the "Vista device" from now on). Each Vista device contained a microcontroller with Bluetooth Low Energy (BLE) communication[1], which controlled a Sharp IR sensor (with a 1 meter range) and an

[1]Texas Instrument, "DRV2605", November 15, 2015, `http://www.ti.com/product/drv2605`

eccentric rotating mass (ERM) vibration motor. The devices are Lithium-Polymer (LiPo) battery powered and USB rechargeable. Figure 7.2 shows a 3D printed version of the device.

Figure 7.2: A 3D printed version of the VISTA device (Prototype III)

7.1 Experiment

From the limited testing that Vista Wearable Inc. did, they were able to gather some insights about how users used the device. Most users were able to play-test one or two devices and they mostly placed them on their wrists to "scan" the environment. We wanted to conduct a usability test for this version of devices to determine whether placing more devices on arms, torso, or legs would improve their navigation performance. The major goals were (1) to study the impact of sensor configurations (including the numbers and locations) to navigation performance; (2) to evaluate if VE training is useful for real world navigation; and further, to identify gaps from VR to real.

To achieve this, we conducted two experiments: 1) a configuration experiment; and 2) a training experiment. For the configuration experiment, we wanted to identify what configuration of VISTA devices was the most efficient for navigation. This experiment was conducted in a virtual hallway. Each configuration defined how many devices would be used, where they would be placed on the body (and correspondingly on an avatar's body), and its orientation (whether facing front or side). We used three to six VISTA devices: on arms, torso, and legs, in this experiment. Figure 7.3 shows the four configurations that were tested in this experiment. For each configuration, subjects went through three L-shaped hallways with the first virtual hallway being used as "training."

Given the best configuration, we wanted to see how well they could actually perform the job in a real hallway with and without a VR training session. For this experiment, half of the subjects went through a 10-minute virtual reality training by allowing them to freely explore a virtual environment that was based on the hallway they would physically navigate. After the training session, subjects were placed in the real hallway for the testing session. The hallway was a U-shaped corridor with a dimension of 71 feet by 52 feet. Subjects were asked to walk from one end of the hallway to the other. Two VISTA devices were secured to their dorsal wrists. Subjects were allowed to freely move their arm to detect obstacles, but they were instructed not to touch the walls and stationary obstacles. Sighted and low-vision subjects were blind-folded, while totally blind subjects were not. At the end, a questionnaire was conducted.

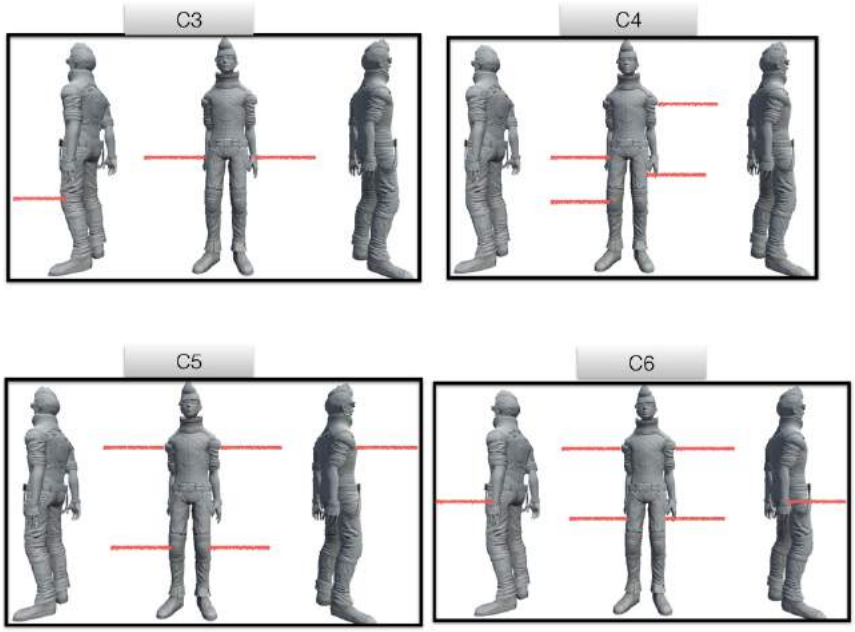


Figure 7.3: Vista device configurations. Starting from top-left, clockwise: 3 devices (C3); 4 devices (C4); 6 devices (C6); 5 devices (C5)

Ideally, we wanted to conduct the configuration experiment first, and then the training experiment second, using the best configuration we found in the first step. This required two independent groups of subjects, which imposed time and resource (*i.e.,* ability to recruit

large group of subjects) constraints. Thus, for practical considerations, we conducted the training experiment first and then the configuration experiment, using the same group of subjects for both experiments. This reversal of experiment order was needed to preserve the training condition of the training experiment.

The four configurations in the configuration experiment were chosen for the following reasons. C3 (three devices) was designed to cover subjects' sides and front. The two side devices were placed on subjects' wrists. Note that this experiment was conducted in a VE and the avatar's arms only had 1-degree of freedom when walking (*i.e.,* only swung back and forth; the swinging is part of the avatar's animation and users had no control over it). The front-facing device was placed just above the left knee (below the knee was fine as well). For C4, all devices were side-facing because we were curious about whether the lack of front-facing device would have affected the performance. We placed one device on upper left arm, one on left upper thigh, one on right wrist, and one on right knee. The rationale for C5 was similar for C3, except with two additional devices; two on upper arms, two on knees, and one on frontal chest. We moved the front sensor up to the chest to avoid confusion with the two knee sensors pointing to the side. For C6, we designed it to provide as much coverage as possible; two on upper arms, two on upper thighs, and two on wrists (front-facing). We did not design two or fewer devices because Vista Wearable Inc. had already conducted tests using one and two devices. We also did not test more than six because the Bluetooth protocol used could only handle up to eight connections (see Chapter 7.3 for more details) and we believed that seven or eight devices were overwhelming (*i.e.,* would have caused sensory overload).

We hypothesized that C6 would have performed the best because it had the most coverage (including both sides and front). C4 and C5 would have had an average performance, since they had similar coverage as C6. For C3, however, we had hypothesized that it would have been the easiest to use with an average performance. We speculated that C4 might not have performed as well as others, due to the lack of a front-facing device. However, the only way

subjects could have detected whether there are obstacles in front of them was when they bumped into them or they rotated 90° to either side.

For the training experiment, we used two VISTA devices (one on each dorsal wrist) in the real hallway because they could be "simulated" in various configurations. In VE, all the virtual sensors had at most one degree of freedom and thus needed more sensors to cover different angles. In the real world, however, subjects' arms have six degrees of freedom and so subjects could have moved the devices around to detect obstacles on the sides or front at various heights.

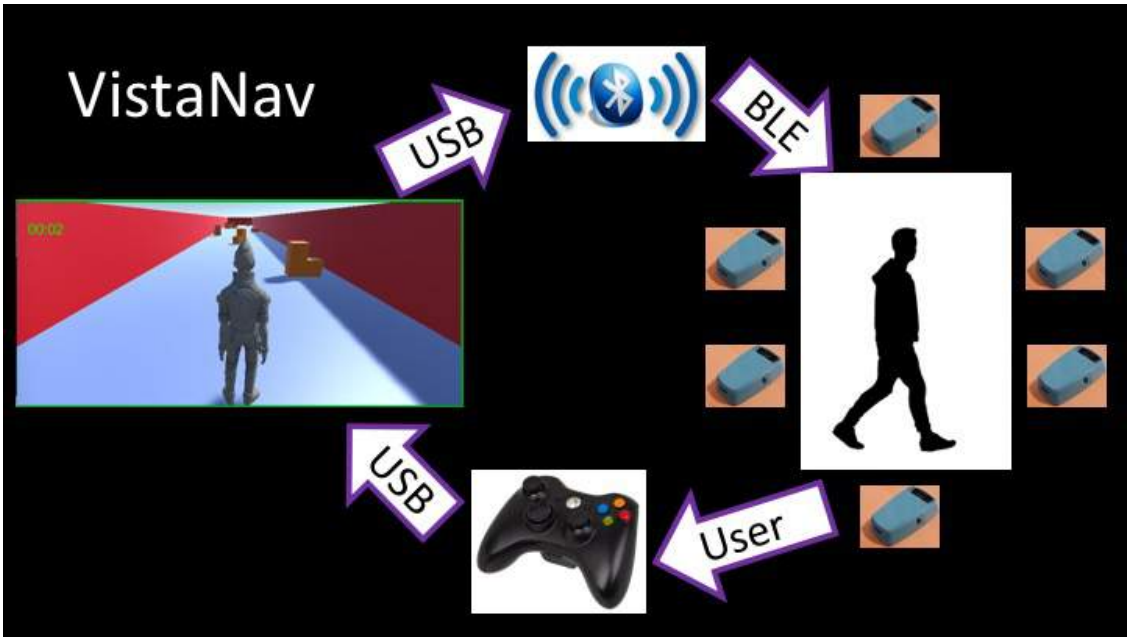


Figure 7.4: VistaNav experiment setup

Figure 7.4 illustrates the experiment setup for VistaNav. This setup only pertains to the configuration experiment and the training portion of the training experiment. The VEs were connected to the Vista devices via Bluetooth low energy connection, where sensor data were transduced to the subjects. They, in turn, controlled the avatar via a game pad to reach a destination. The following subsections detail each part of the setup.

7.2 Controller and Its Setup

For these experiments, we used an Xbox game pad controller (Figure 3.3c) as the input controller. The controller is plug-and-play, therefore, no special driver was needed. Once plugged into the computer, Unity3D registered it as a standard controller and it could be accessed via Unity3D APIs. The game pad controller provided three specific controls. The left joystick allowed users to move the avatar forward at varying speeds. The controls, however, would not have allowed the avatar to walk backward. To do that, the right joystick had to be used to turn the avatar around. The right joystick allowed discretized left and right turns. Each turn is 45°. The last control was the 'A' button. When pressed, the VE told the users what direction the avatar was facing (*i.e.,* north, east, south, west, *etc.*).

7.3 Multimodal Stimulators

The stimulators for these experiments were the Vista devices and sound cues. Instead of having used the full functionality of the Vista device, we turned off its IR sensing and used only its haptic control, since in VE experiments, range information is generated by the virtual sensors. The device was able to receive commands via BLE communication. In networking-speak, Vista devices are BLE clients. As such, the master node needed to be in a BLE capable platform. We used Windows as a host platform. While Windows is Bluetooth capable, it can be cumbersome for running experiments because Windows Bluetooth requires manual pairing and connecting. For that reason, we used a USB Bluetooth smart dongle[2], which could be connected to the VE via a serial port. Lastly, a BLE master could handle up to eight client connections, thus, we only had to use up to six Vista devices for this experiment, in configurations similar to Figure 7.3. There is a custom program that provides interoperability between Unity3D and USB BLE smart dongle. The custom program facilitates communication between the VE and Vista devices.

[2]Bluegiga Technologies, "BLED112", November 15, 2015, `https://www.bluegiga.com/en-US/products/bled112-bluetooth-smart-dongle/`

For ***audio*** feedback, three sound cues were used. They were an "ouch" sound when the avatar bumped into an obstacle, the avatar's "footstep" sound, and a "chick chirping" sound (which denoted destination). The chick chirping sound became louder as the avatar got closer to it. All sound clips used were royalty free, and could be directly imported into Unity3D and placed on the appropriate model. We used a wireless Parrot Zik 2.0 headphone[3] to transduce the audio.

7.4 Virtual Sensor and Transducing

Simulating the IR sensor was the same as those in VibrotactileNav (Chapter 6). The virtual IR sensors were placed on an avatar based on one of the four configurations loaded (Figure 7.3). As mentioned before, the game engine updated the sensor output at approximately 60 Hz. Sending data that fast to all Vista devices simultaneously would have been overwhelming. To address this, we introduced a co-routine that was executed at specific time intervals (*e.g.,* one second). While one second may seem slow, our experience indicated that subjects took at least a second to respond via a controller and sometimes were even idle before taking an action. The co-routine polled the virtual sensors for range data, normalized and thresholded it, and sent a command to the appropriate Vista device. The range data were normalized to $[0, 1]$. Thresholding was done in two parts. The first part was frequency, which was done by dividing the normalized value into 5 intervals. The second part was vibration effect (pattern), which consisted of "near" and "far". Near was for normalized range value < 0.5, and far for otherwise. The command was a string of text that encoded which device to activate, with what frequency and vibration effect. Then the string was sent via an established serial port connection (*i.e.,* the COM port that the USB Bluetooth smart dongle was connected to).

[3]Parrot, "Parrot Zik 2.0", Jun 9, 2016, `http://www.parrot.com/products/zik2/`

7.5 Game Mechanics

The task definition for the configuration experiment was to navigate the avatar from start to finish without bumping into the wall and stationary obstacles, without looking at the computer screen, and by having had to rely only on audio and haptic feedback. Since there were four configurations and each configuration had three hallways, each subject had to repeat this process a total of 12 times. The avatar behavior was very straightforward. Its forward motion was at a variable velocity, such that depending on how hard subjects pushed the left joystick forward, the speed changed. The avatar could only make a left or right turn at a 45° angle each time, and the subjects could determine the avatar's orientation by pressing the "A" button on the game pad. Pressing the button told the subjects whether they're facing east, south, north, *etc.*Furthermore, while the avatar was in motion, a footstep sound was constantly playing. When the avatar bumped into a wall or stationary obstacle, an "ouch" sound was played once. For the training portion of the training experiment, the task definition was to simply explore the virtual hallway for 10 minutes with the aforementioned behavior and control.

In summary, the game was set up as follows:

- Subjects had to find the source of the sound of a baby chick chirping without any visual information (only for the configuration experiment).
- The computer screen was faced away from subjects.
- Subjects had to navigate the virtual environment and avoid obstacles based on the varying intensity of sounds and vibrations.

7.6 Environment Design

For the configuration experiment, we generated three L-shaped hallways, similar to Figure 6.6a. Hallway 1 had a dimension of 21 feet by 60 feet. Hallway 2 had a dimension of 16

feet by 30 feet. Hallway 3 had a dimension of 21 feet by 46 feet. The width of the hallway varies from five to ten feet. For all three hallways there were stationary obstacles placed randomly throughout the hallway. The goal for each was to go from one end of the hallway (starting point) to the other end of the hallway (destination). The first virtual hallway was used as a training hallway for each configuration, to get subjects acclimated navigating the hallway with a particular configuration. This also provided an opportunity for those who needed to familiarize themselves with the equipment. For the remaining two (testing) hallways, we imposed a time-out of 5 minutes for each hallway. However, if subjects made it close to the destination (more than half way through), we extended the time-out to 10 minutes, to give them some time to reach the destination. In the training hallway, some guidance was provided to assist them in reaching the destination; for example, telling them which direction to turn. A time-out was needed due to time constraints; each subject had to complete 12 (four configurations $\times$ three hallways) runs in 1.5 hours.

7.7 Measurement Device & Data Collection

Data was collected from the two experiments. For the training experiment, recorded video was analyzed to determine each subject's completion time and the number of times each bumped into obstacles. At the end of the hallway, each subject was given a questionnaire to assess the usability of the Vista devices. For details and questions asked in the questionnaire, please refer to Appendix B.

For the configuration experiment, the following data were collected for each hallway and configuration:

1. 3D positions of starting location and destination.

2. Sensor locations and its ID.

3. Time: From start to finish locations, unless timed-out.

4. Position: 3D position of avatar.

5. Heading: 3D vector of avatar's forward heading.

6. Bump: Whether the avatar is bumping into an obstacle.

7. Sensor values: A pair of sensor location and reading value for each device.

The data collection procedure wrote the above data to a text file every second with a timestamp. We had tried a sampling rate of less than one second, but it produced redundant data and introduced minimal lagging in the system, especially with sending data to the Vista devices. The last row of the data file also indicated the finish time or time-out. It also recorded the avatar's position and orientation in the VE as well as a flag that marked whether the avatar was bumping into an obstacle. Bumping was recorded because it showed us where the subject bumped into something and for how long. The next section shows the results.

7.8 Results

The results for both experiments are discussed here. For consistency, we present the results in the ideal experiment order (configuration experiment first, and then the training experiment), instead of the order we actually conducted the experiments. Table 7.1 shows a descriptive statistics of all subjects recruited for both experiments. Tables 7.2 and 7.3 show a breakdown of all subjects by VIP and sighted only, respectively. All subjects gave written informed consent and took part in the experiment for monetary compensation. This study was approved by the IRB of CUNY.

Table 7.1: Subjects descriptive statistics for all

	Total	Male	Female	Mean Age (SD)
Configurations Experiment	18	12	6	34.0 (19.0)
Training Experiment	21	14	7	36.2 (18.9)

Table 7.2: Subjects descriptive statistics for VIP

	Total	Male	Female	Mean Age (SD)	Mean Year Impaired (SD)
Configurations Experiment	7	3	4	50.6 (21.1)	48.6 (19.2)
Training Experiment	10	5	5	50.3 (18.4)	44.5 (17.6)

Table 7.3: Subjects descriptive statistics for sighted (for both experiments)

Total	Male	Female	Mean Age (SD)
11	9	2	23.5 (5.6)

Configuration Experiment

In this experiment, there were 18 subjects, where testing of VISTA device configurations is conducted in virtual environments. The configurations we tested are illustrated in Figure 7.3. For each configuration, subjects had to navigate through three hallways, with the first hallway used as training. The goal here was to determine which of the four configurations performed best, in terms of completion time.

To provide an overview of what happened in the virtual environments, Figure 7.5 shows intensity maps for each of the three hallways, over all subjects and four configurations. It shows how frequently a particular location was visited, from dark blue (less frequent) to dark red (very frequent). The starting position is denoted as "start" and dark red, since every one began at the same location. The maps are shown in a top-down view, with the x- and y-axis being the coordinate system in the virtual environment. In the training hallway (H1), most subjects did not explore the lower right side of the hallway much; they were able to turn the corner when detected. For the other two hallways, they seemed to have explored most of the layout. There were some sporadic white squares throughout the hallway, which indicated the location of obstacles.

To break this down even further, Figure 7.6 shows intensity maps for each hallway in the first configuration (C3), for all subjects. Unlike the previous figure, the intensity maps here show more white squares. This means that in addition to obstacles, subjects were able to navigate more quickly and with ease, thus spending less time at some locations. Furthermore,

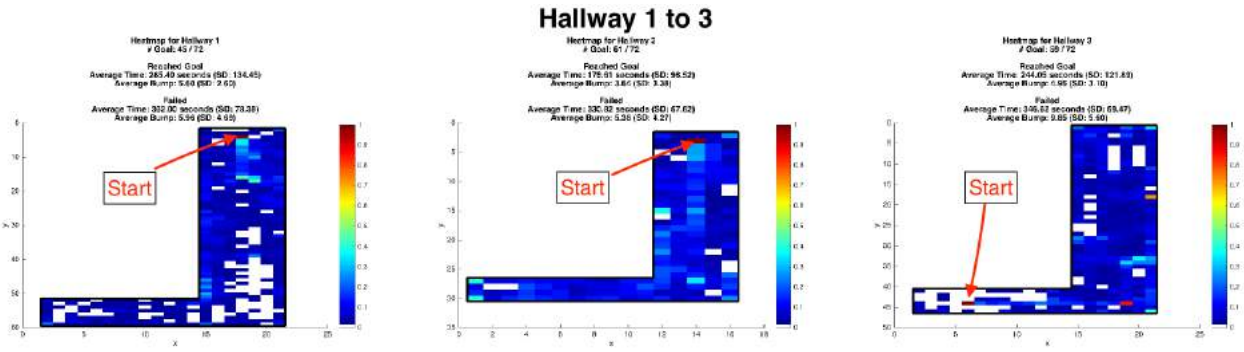


Figure 7.5: Intensity maps for each hallway, for all subjects and configurations

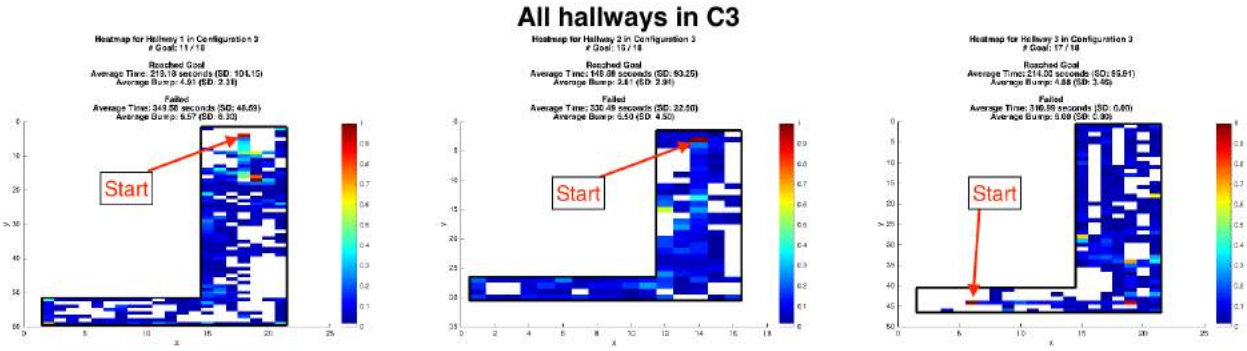


Figure 7.6: Intensity maps for each hallway in the first configuration (C3), for all subjects

the average completion time was lower compared to the overall average in Figure 7.5. For example, in Hallway 2 (middle column in figure), the average completion time for those who reached the destination in C3 was 149.88 seconds, compared to the overall average in Hallway 2, which was 178.61 seconds.

Another illustrative breakdown is Figure 7.7, where intensity maps for each configuration in Hallway 2, for all subjects, are shown. Here, we can see that subjects had an easier time navigating using the first (C3) and second (C4) configurations, as indicated by more white squares and darker blue squares (lighter colors toward red mean higher frequency), faster completion time, and more subjects reaching the destination. For example, 16 out of 18 people reached the destination using C3 and C4, compared to 14 and 15 for C5 and C6, respectively.

A repeated measures two-way ANOVA was conducted in Matlab with hallway (two levels) and configuration (four levels) as the two within-subject factors and completion time as the

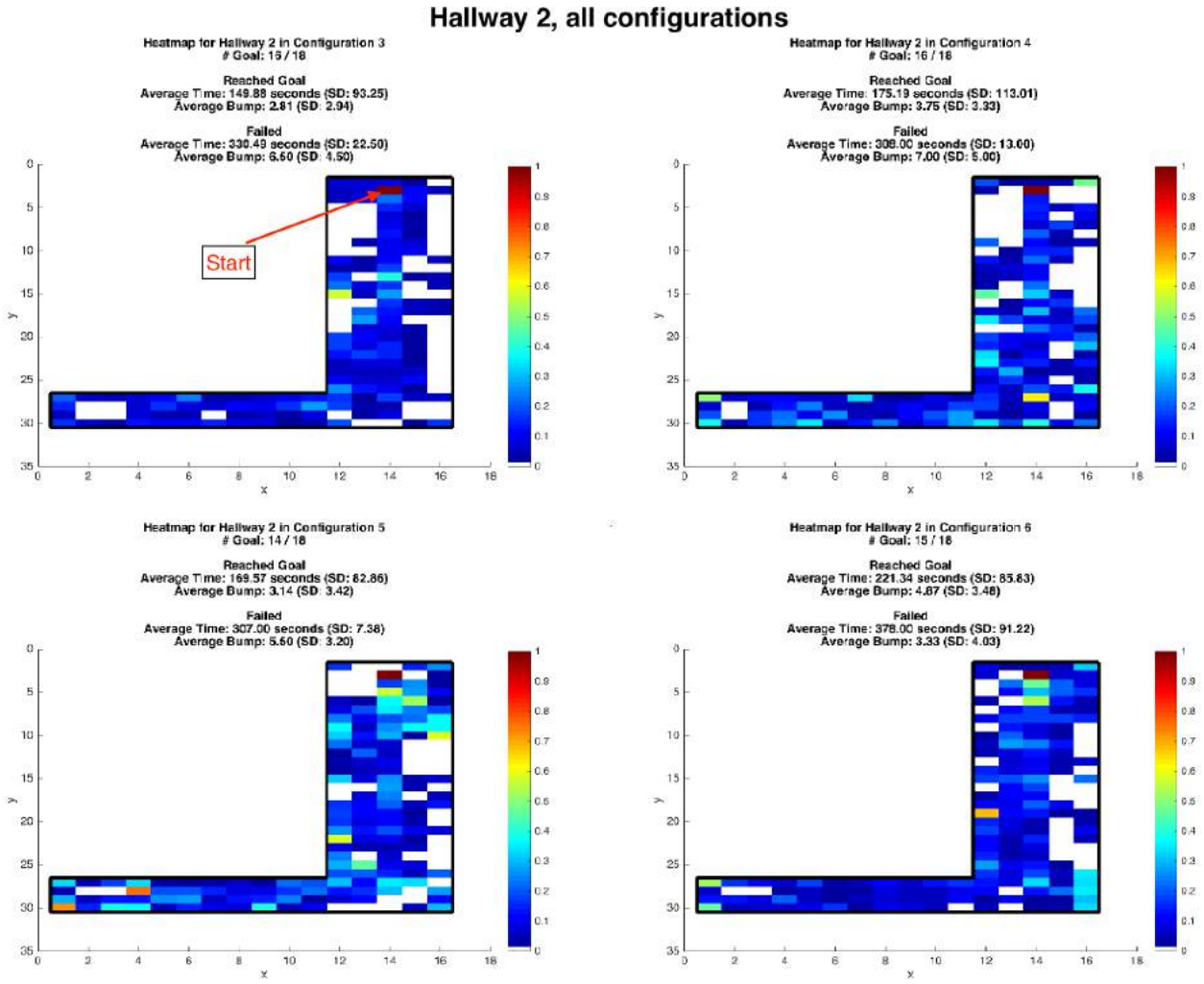


Figure 7.7: Intensity maps for each configuration in Hallway 2, for all subjects

dependent variable. Although we collected the number of bumps for each subject, we decided not to include it in the analysis because most subjects bumped on purpose. Furthermore, for those who did not reach their destination, their completion times were replaced by the average of those who succeeded in that particular hallway and configuration. If sphericity assumption is violated as determined by the Mauchly's test, adjusted Greenhouse Geisser's p-value, p_{gg}, is reported. The analysis is broken down into three groups: all subjects, VIP only, and sighted only.

For all subjects, there were statistically significant differences between configurations as determined by two-way repeated measures ANOVA[4] ($F(3, 51) = 10.54$, $p = 0.00$). A multiple comparisons test (multcompare in Matlab) with Bonferroni correction revealed that the completion time was statistically significantly lower for configuration C3 ($p = 0.00$) compared to C4 and C6. There were, however, no statistically significant differences between C3 and C5 configurations ($p = 0.28$), C4 and C5 configurations ($p = 0.55$), C4 and C6 configurations ($p = 0.58$), and C5 and C6 configurations ($p = 0.07$). Each configuration's mean and standard deviation for all subjects are shown in Table 7.4. Among the VIP only group (7 subjects), however, there was no statistically significant difference among configurations ($F(3, 18) = 2.02$, $p_{gg} = 0.18$). Each configuration's mean and standard deviation for VIP only are shown in Table 7.5.

For sighted only, there were statistically significant differences between configurations ($F(3, 30) = 8.13$, $p = 0.00$). A multiple comparisons test with Bonferroni correction revealed that the completion time was statistically significantly lower for configuration C3 compared to C4 ($p = 0.02$) and C6 ($p = 0.01$). There were, however, no statistically significant differences between C3 and C5 configurations ($p = 0.90$), C4 and C5 configurations ($p = 0.77$), C4 and C6 configurations ($p = 1$), and C5 and C6 configurations ($p = 0.09$). Each configuration's mean and standard deviation for sighted only are shown in Table 7.6.

Table 7.4: Group means for all configurations and subjects.

	Mean (in sec)	SD (in sec)
C3	159.73	82.59
C4	228.00	100.10
C5	198.26	97.69
C6	261.25	121.17

We conclude that C3 is the best configuration among all four; it has the lowest average completion time for all configurations and all groups. Feedback from subjects suggests that C3 is much more intuitive to use and easy to navigate compared to other configurations.

[4] $DOF_1 = (n-1) \times (k-1)$, where n is number of configurations and k is number of hallways
$DOF_2 = (s-1) \times (n-1)$, where s is number of subjects

Table 7.5: Group means for all configurations, for VIP only

	Mean (in sec)	SD (in sec)
C3	172.63	59.08
C4	225.59	106.04
C5	209.56	90.74
C6	249.35	149.39

Table 7.6: Group means for all configurations, for sighted only

	Mean (in sec)	SD (in sec)
C3	154.80	96.24
C4	235.35	100.14
C5	195.05	104.78
C6	266.63	103.45

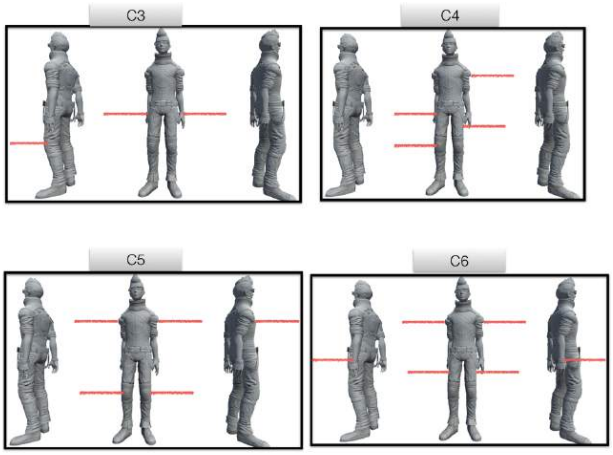


Figure 7.8: Figure 7.3 is reproduced here for your reference.

This is because C3 provided a minimally optimum coverage for obstacle detection in a simple navigation; both sides and front are covered. Statistics show that C3 and C5 are not significantly different, and C5 has the second lowest completion time. Thus, we can conclude that C5 is the second best to C3 because it has similar coverage as C3, albeit having redundancy on the sides. C4 did not perform as well probably because it lacked a front sensor, which made it difficult for subjects to detect obstacles without bumping into them. Also, some subjects had said that they moved a bit slower in order to avoid bumping into obstacles. They also rotated in-place, so as to use the side sensors to detect obstacles in front of them. C6 has the worst performance, despite having redundant coverage on all

sides. Furthermore, subjects said that it was overwhelming when all units were vibrating at the same time (this happened when they were facing a corner). They had to pause to interpret the haptic feedback to determine what was around them in the virtual environment, and to make a decision about whether to turn or move forward. The lack of significance result among the VIP only group suggests that the VIPs were very flexible so we did not have to spend time in configurations because they could change them based on situational conditions.

Training Experiment

For this experiment, there were 21 subjects and it took place in a real hallway. The real hallway was a U-shaped hallway (71 feet $\times$ 52 feet). Half of the subjects received training in a virtual environment where they were allowed to freely explore the virtual counterpart of the floor where the hallway was located. They were not allowed to view the screen - instead, they had to rely on haptic and audio feedback to navigate around. The haptic devices used in training were the Vista devices; one on each dorsal wrist and one on frontal chest. Audio was 3D sound cues conveyed via a wireless headphone. The training session lasted 10 minutes.

For all subjects, the testing began at one end of the U-shaped hallway and the goal was to reach the other end of the hallway without bumping into any obstacles or walls. Sighted and low vision subjects were blindfolded in this part of the experiment. Unlike the training session, we only used two Vista devices in this part; one on each dorsal wrist. Previous results concluded that C3 is the best configuration, however, we only had two devices that had infrared and vibrator coupled together that could operate standalone for navigation. Furthermore, because the virtual avatar's arms only have one degree of freedom when walking compared to a real user (who has six degrees of freedom), two VISTA devices had to be used to "simulate" other configurations, such as C3. We video recorded the sessions of each subject for offline analysis. Analysis included determining the completion times and

number of times subjects bumped into an obstacle. In retrospect, the number of bumps was not meaningful because some subjects used it as a navigation strategy, or because they felt insecure and had to touch the wall. Thus, the analysis below only includes completion times.

Only 17 out of 21 subjects are included in this analysis. Two outliers are trimmed from one end of each group (training *vs.* no training) due to experimental errors. For example, one subject's long sleeved sweater blocked the sensor causing false vibration and was not caught until later. Overall, the virtual training significantly improved the performance in real hallway navigation ($t(15) = -1.91$, $p = 0.04$, two-sample, one-tailed) with those who received training ($M = 249.00s$, $SD = 92.45s$, $N = 8$) completed the hallway faster than those who did not receive training ($M = 333.78s$, $SD = 90.76s$, $N = 9$). Breaking it down into two groups (VIP and sighted), there is significance difference for the VIP only group ($t(5) = -2.07$, $p = 0.05$, two-sample, one-tailed). VIPs who received training ($M = 171.25s$, $SD = 43.36s$, $N = 4$) completed the hallway faster than the VIPs who did not receive training ($M = 309.00s$, $SD = 127.45s$, $N = 3$). Non-VIPs only, however, showed no significance difference in completion times ($t(8) = -0.45$, $p = 0.33$, two-sample, one-tailed) between those who received training ($M = 326.75s$, $SD = 44.06s$, $N = 4$) and those who did not ($M = 346.17s$, $SD = 78.29$, $N = 6$).

Prior to this experiment, subjects were naïve to the VISTA devices. Results show that where VE training was provided, even for 10 minutes, it improved their navigation performance compared to those who did not receive any training. The insignificance result for the non-VIP only group is due to the fact that sighted subjects who normally walk with vision had to be blindfolded while walking, which is a totally new experience and may have caused exhibited hesitation or caution. Note that those in training sessions were not blindfolded, and they received no visual stimuli from the VE. Furthermore, based on observations of all subjects, we noticed that subjects tended to place their hands at their sides (with VISTA devices facing their sides) or in front of them (with the devices facing forward), or they alternated between side and front, which effectively "simulated" the C3 configuration. There

were also a few subjects who "scanned" the real hallway at various heights, which was similar in configuration to C5.

After completing the hallway navigation, subjects were asked to complete a questionnaire to assess the usability of the VISTA devices. We employed the System Usability Scale (SUS) developed by John Brooke [15]. It consists of ten questions with five Likert-scale responses, which ranges from Strongly Disagree to Strongly Agree. There are five positive statements and five negative statements, which alternate. For the specific statements asked and more details, please check Appendix B. We chose to use SUS because research has shown that it enables a measure of the usability of a technology with a small sample (*e.g.,* 8-12 users) and allows us to be fairly confident of the assessment [84]. This allows us to reach a conclusion quickly, accurately, and consistently.

SUS score is computed using the following equation:

$$SUS = [\sum(odd\_number\_score - 1) + \sum(5 - even\_number\_score)] \times 2.5 \tag{7.1}$$

The equation converts the ratings to a score of 0 to 100. Note that these scores are not percentages, but rather a relative ranking [14]. To add more meaningful information to this ranking, we added an adjective rating (*e.g.,* Good, Excellent, *etc.*). This provided us with a measure of a subject's subjective perceptions of the usability of the VISTA devices. Table 7.7 shows the descriptive statistics of SUS scores for all 21 subjects. It shows how many subjects voted for an adjective, and for those who voted for that adjective, the average SUS score for it. For example, five subjects selected adjective rating "Good", and their average score is 75.50.

It is fair to say that, for a SUS score above 75.5, it is considered "good" (above 81.35, it is "excellent", *etc.*). However, there is not enough information to derive a threshold for the other less desirable adjectives. To resolve this, we used a "grade-on-a-curve" method to assign a value range for each adjective based on overall mean (80.48) and its standard

Table 7.7: SUS Descriptive Statistics

Adjective	Count	Mean SUS (SD)
Worst Imaginable	0	00.00 (0.00)
Awful	0	00.00 (0.00)
Poor	0	00.00 (0.00)
OK	0	00.00 (0.00)
Good	5	75.50 (9.91)
Excellent	13	81.35 (16.00)
Best Imaginable	3	85.00 (5.00)

deviation (13.62). For example, SUS scores between 66.86 and 80.48 are "good" and scores between 80.49 and 94.11 are "excellent". Thus, if we want to have a "passing grade", we want scores above 53.24 (*i.e.,* better than "Poor"). Using this scale, we adjusted the original adjective to be more normally distributed. Table 7.8 shows the original adjective, SUS score, and adjusted adjective for each subject. Overall, it shows that the VISTA devices are excellent.

Looking at the adjusted adjectives, we have 11 out of 20 that are excellent. With more responses, the "grading scale" can be refined to be more robust such that a SUS score alone can provide meaningful quantitative information (*e.g.,* a grading score) without soliciting subjective adjectives.

7.9 Discussion

Overall, we found that the C3 configuration is the best because it offers coverage to either sides of the subjects and the front. This allowed them to detect obstacles in front of them, and also to determine whether they were deviating from the path and getting closer to a wall on the side, while the number of sensors did not get them confused and overwhelmed. We found that subjects enjoyed using the Vista devices both in the real hallway and in the virtual. Some subjects did report that the VE training provided prior to the real hallway test was helpful in getting used to and interpreting the haptic feedback. Some subjects also commented that the virtual environments testing in the Configuration Experiment was fun

Table 7.8: SUS results

ID	Original Adjective	SUS Score	Adjusted Adj
1	Best Imaginable	80	Good
2	Good	72.5	Good
3	Excellent	50	Poor
4	Good	77.5	Good
5	Excellent	90	Excellent
6	Best Imaginable	90	Excellent
7	Excellent	77.5	Good
8	Excellent	85	Excellent
9	Excellent	82.5	Excellent
10	Excellent	62.5	OK
11	Excellent	55	OK
12	Good	85	Excellent
13	Good	60	OK
14	Excellent	97.5	Best Imaginable
15	Excellent	90	Excellent
16	Best Imaginable	85	Excellent
17	Excellent	82.5	Excellent
18	Good	82.5	Excellent
19	Excellent	100	Best Imaginable
20	Excellent	92.5	Excellent
21	Excellent	92.5	Excellent

and engaging. Some even showed signs of competitiveness in trying to reach the destination without triggering the time-out. Furthermore, for the VIP group, we found that some VIP subjects had a difficult time interacting in the VE. Often times we had to remind them how to use the input controller and what their goal was. Interpreting haptic feedback was also a challenge, especially with 6 devices; subjects said it was overwhelming (*i.e.,* sensory overload). Most, however, felt that it was a novel experience and that they could use more time to get used to it, especially with more than three VISTA devices.

Not only did the VE training improve performance time in the real hallway testing (Training Experiment), but without using GIVE-ME in the Configuration Experiment, it would have been difficult to conduct a test to determine which configuration is better for navigation. GIVE-ME enabled us to rapidly test multiple configurations in a smooth and efficient manner, and replicated the exact testing conditions multiple times. Conducting such

tests in the real world only would have taken a much longer time and exact conditions cannot be guaranteed between sessions. Also, data collection would have been difficult because the experimenter would also have to have been in the hallway to collect internal state data (*e.g.,* distance reading and thresholded vibration frequency) from each Vista device (in VE, these data are generated), to have ensured each subject's personal safety, video-recording, and time-keeping.

As evident in the results, there are some limitations in using GIVE-ME. If we had used Virtuix Omni[4] (which is a platform that allows users to "walk" in any direction without really walking around) instead of an Xbox 360 controller, the interaction would have been more intuitive and realistic (*i.e.,* mimic actual testing condition). This is especially true for VIP subjects because most of them were not familiar with a Xbox 360 controller. Furthermore, for sighted subjects, the learning transfer barrier was higher because they had to navigate a hallway blindfolded (which they normally do not), and thus they would have to had "trusted" the devices, and may have been hesitant at the beginning, despite VE training. Note that in VE training, trust was not an issue because subjects knew they would not have been able to hurt themselves, unlike facing real walls and obstacles. We believe that trust is another factor why the VIP-only group shows significant results, while the non-VIP only group did not. On average, the VIP subjects in the Training Experiment have been visually impaired for 44.5 years (Table 7.2), and thus were more comfortable navigating in a hallway non-visually.

Chapter 8

Conclusion and Discussion

8.1 GIVE-ME Framework

In this dissertation, we developed a framework named *Gamification In Virtual Environment for Multimodal Evaluation* (GIVE-ME), to evaluate navigational assistive technologies for the visually impaired. We first showed the conceptual model of the framework, and then discussed the framework's user interfaces and components. We then showed a software implementation of the framework in Unity3D and four applications of the framework (BrainportNav, CrowdSourceNav, VibrotactileNav, and VistaNav). In summary, we have the following findings and observations, along with our four major contributions:

Unified evaluation framework. We developed an evaluation approach for multimodal visual ATs that can handle a variety of sensors and stimulators, such as infrared, sonar, haptic, and audio. We tested it in four different applications with a variety of sensors and stimulators. This framework works with a camera sensor and an electrode tongue display unit (BrainportNav in Chapter 4), camera sensor and text-to-speech feedback (CrowdSourceNav in Chapter 5), and infrared or sonar, and haptic or audio feedback (VibrotactileNav and VistaNav in Chapters 6 and 7, respectively). New simulation of sensors can be added, such as magnetic field, gravity, pressure, and thermal. Connect-

ing the framework to new stimulators is also possible, such as electrical stimulation array, olfactory emitter, and gustatory system. In general, the framework allows researchers and developers to test their multimodal ATs with different types of inputs (sensors) or outputs (stimulators). By using GIVE-ME, various input modalities and output modalities can be compared with some other parameters fixed (the same parameter or equivalent). For example, combine human sensors (crowd in Chapter 5) and IR distance sensors (Vista devices in Chapter 7), or vibrotactile (vibrators in Chapter 6) and tongue stimulators (Brainport in Chapter 4). In addition to comparing various sensors and stimulators, and AT systems, this framework could also use the comparison results to establish a benchmark. As more and more people utilize this framework, a dataset of hybrid data can be generated. Hybrid data from GIVE-ME can yield valuable information regarding users' perceptions, thoughts, and opinions, and quantifiable assessments of the ATs.

Sustainable evaluation. We employed gamification techniques to create fun virtual environments to engage users, while collecting experimental data for analysis. In particular, in the experiment in Chapter 7, subjects were very engaged in the VE session, to the point of showing competitiveness. They were committed to reaching the destination within the time-out limit (5 minutes), and constantly asked how much time they had left. Similar enthusiasm had been expressed in other experiments as well. In addition to laboratory experiment sessions, a designed virtual environment can be packaged as a game and distributed online, as in our crowd navigation reported in Chapter 5. In this mode, data can be collected in a distributed manner and the game can also be used as a simulation or training tool to educate users or for users to practice using a new AT.

Novel psychophysics evaluation. We designed a framework that enables researchers to conduct psychophysics evaluation involving navigation tasks. The framework not only

allows researchers to collect data on subjects' performance and their subjective assessment of the technology, but also allows brain activity measurement such as EEG (Chapter 6.9.1). Most important of all, the framework timestamps all data collected, and thus all collection mediums are synchronized. While brain study is not the focus of this dissertation, this framework can also be used with fMRI. Since subjects only need to provide input via a controller such as a Xbox 360 controller, movement is minimized within the scanning machine and subjects are able to receive feedback from the VE via audio or haptic cues.

Novel collaborative environment. We created a framework that not only enables developers to rapid prototype and test their navigational assistive technology, but also enables an early stakeholder involvement that fosters communication among researchers, developers, and most importantly, users. Since part of the AT is virtually simulated, developers can try out different configurations and solicit feedback from users to refine their prototype in the next iteration. Such collaborative environment results in navigational ATs that are human-centric. Although we are not the developers of Brainport (Chapter 4), subjects had told us that they wished the system was not that awkward to use. GIVE-ME would certainly enable insightful comments during the development process of Brainport. For the crowd-source navigation app (Chapter 5), the developers are also our collaborators. During the experiment sessions, they were able to receive instant feedback on the usability of the system. They also discovered that participants give preemptive instructions to guide blind users to a destination, in order to deal with a slow and delayed network. For VibrotactileNav (Chapter 6), subjects expressed the novel experience of using haptic feedback for navigation as fascinating and suggested various alternative uses of the VISTA system, such as mounting the units on shoulders to detect overhangs, or on legs and ankles to detect potholes and stairs. This inspired the modularity design of the system, which resulted in VistaNav (Chapter 7). In this experiment, blind subjects were anxious to find out when the products would be re-

leased to the consumer market, and one VIP indicated that he was willing to pay $50 per unit.

The GIVE-ME framework is meant to be deployed by ATs developers to (1) better test their systems, (2) provide a simulation and training tool to users, and (3) establish communication to rehabilitation professionals and end-users with visual impairment. Hybrid data from GIVE-ME can yield valuable information regarding users' perceptions, thoughts, opinions, and quantifiable assessment, on the ATs developed. If successful, psychophysics experiment data that was not possible before, can be produced, specifically navigational data. Combined with functional and objective assessment of ATs, these data can help shape for better and human-centric ATs. Furthermore, simulation and training tools can be developed to better educate and train users in using ATs.

8.2 Discussion

Due to the niche nature of ATs, there is a lack of metrics for benchmarking and the cost is very high. VIPs who want to try out ATs have to spend a lot of money to acquire them and test them out. In most cases, they end up using one or two products. There is, therefore, a need for a comparative method that can inform users whether an AT satisfies their criteria and enables them to do comparison shopping. The GIVE-ME framework solves this need with added benefits. One benefit GIVE-ME provides is a platform that allows VIPs to try out an AT, perhaps even a prototype, without making a purchase. Conversely, VIPs are able to provide valuable feedback early on in the design process. Yet another benefit GIVE-ME provides is the ability to package the VE into a training tool for those who want practice with their new ATs.

The design and implementation of the initial framework was on a case-by-case basis; virtual environment was designed specifically for an experiment. This was necessary since we were building the framework from scratch; having to implement simulation scripts for each

sensor, network modules that enable connection to each stimulator, and parsing of command data appropriate for each stimulator. Further, each experiment has specific evaluation criteria, thus, a virtual environment had to be designed to accommodate them. But towards the end of the dissertation work, we were able to abstract-out the common sensor simulations, stimulator interfaces, environment design, and data collection procedures, leading to the unified GIVE-ME framework, such that it can be easily applied to another (new) experiments and parameters with minimal changes (in coding and design).

There is still a design and framework appliation limitations with regard to VR and the real world. More data is needed to make a convincing argument of the validity of virtual results and applying the results in a real world setting. We can begin to address this by simulating a realistis environment and comparing performance in a virtual with real world tasks. We will be able to find what VR can and cannot do for enabling real world navigation. And the added values of the VR for evaluating ATs for their uses in real environments.

The first application of GIVE-ME is Brainport (Chapter 4). At the time of the study, Brainport was still in clinical trial, thus, only a handful of people in the country had access to, and we were able to lease a developer version of it for a short period of time. Initial experiments involved shape discriminate tasks. Shapes were simple and included both filled-in and outline versions. The results were not very satisfactory because the discrimination rate is about 50%; subjects were guessing what they felt on their tongues. The subjects, however, were able to distinguish diagonal lines from vertical. At this point, we were curious if diagonal lines could be used to give directions and conduct another experiment with two weeks of lease remaining. Due to the advantages (rapid and modular design, and simulation of pathfinding) of GIVE-ME, we were able to finish the experiment in two weeks.

The second application of GIVE-ME is crowd-sourced navigation (Chapter 5). The main purpose of this experiment was to evaluate two aggregation methods (simple average and legion leader) and user interface of the feedback mechanism. Only these two aggregation methods were tested because simple average is a common and popular approach, and le-

gion leader has two advantages: 1) feedback to users is immediate; and 2) no conflicting instructions since there is only one leader. The main concern, however, was the safety of subjects and having a controlled environment for testing. Like any untested technologies, we were not certain how users (both online crowd volunteers and VIPs) would behave. Using GIVE-ME solved this problem and achieved the main purpose. We did discover during the experiment, that there were indeed, malicious members in the online crowd volunteers; they were providing wrong directions on purpose. Luckily, the victim is only a virtual avatar bumping into walls. This gives rise to an insight for future work in this project, which is to incorporate a weighted trust factor to each online crowd volunteer, in order to weed out malicious responses.

The third and fourth applications of GIVE-ME are vibrotactile navigation (Chapters 6 and 7). Both experiments are related, VistaNav is the newest iterative design of VibrotactileNav. Initially, for VibrotactileNav, we were determining the absolute and difference thresholds of perception on different parts of the body as the foundation of locations and discriminative levels of the vibrotactile stimulation. Then we simulated the prototype in the VE and configured it in such a way to mimic a real world setting. Some of the valuable insights about designing range-vibrotactile that we gained are:

- It is difficult to navigate in a VE without any feedback in terms of orientation; thus we created a steering device (for VibrotactileNav) and speech feedback (for VistaNav) that provides users with information regarding the extent of their turns.

- Both versions of vibrotactile devices work for stationary obstacle avoidance, but it becomes a challenge when obstacles become dynamic. In theory, the vibrotactile devices react in real time, but further work needs to be done to evaluate the performance in avoiding moving obstacles.

- Virtual reality gave us a positive outlook on real world testing; however, virtual testing is no substitute for real world testing. Virtual training can help to improve performance in real world task, as evident in Chapter 7.8.

In conducting VistaNav, we were determining the optimal configuration (placement and number) of the range-vibrotactile devices that enabled efficient navigation. In this version, the devices can work in standalone mode for real world navigation. To control these devices, however, bluetooth low energy protocol is needed to send data to them. We used Bluegiga's Bluetooth low energy USB dongle to communicate to the devices. It is a platform-independent device and allowed us to connect up to eight devices simultaneously.

After conducting these four experiments, the most intuitive and user-friendly AT is the Vista device (successor of the vibrotactile array). Compared to using Brainport and crowd-sourced navigation, Vista's learning curve is much lower. Subjects were able to understand the meaning of haptic feedback without much training (*i.e.,* vibration intensifies the closer one gets to an obstacle). Brainport's users, however, had a difficult time determining simple shapes using their tongue, the accuracy is not that much higher than guessing. While crowd-sourced navigation has some successes, some subjects were confused with the concept of "crowd." This is understandable because the "crowd" is intangible to them and some feedback instructions were contradictory (*e.g.,* a left turn is immediately followed by a right and then a left turn). Contradictory directions happen when there are malicious volunteers in the crowd or when a majority decision is lacking.

8.3 Future Work

The focus of this dissertation work has been on formulating and implementing the GIVE-ME framework for visual navigational assistive technologies. While we have demonstrated the applications of GIVE-ME, there is additional work left for the future. The following are topics we will continue to investigate:

- *Improvement on and validation of framework implementation.* The current version of the implementation includes sensors simulation, stimulators connection, data collection module, and environments designed for the four aforementioned applications. There are many more sensors, stimulators, and environments that can be created and incorporated to have a more comprehensive framework. We plan to integrate them into the framework implementation as we come across them. We also plan to scientifically validate the framework, focusing on comparisons between virtual and real tasks.

- *Determine metrics to evaluate ATs.* The current version of the framework supports arbitrary metrics to evaluate ATs. We want to determine a set of evaluating metrics that is useful for VIPs. We will achieve this by surveying the population and experts for what metrics they think are important for them in deciding what ATs to use. The responses will vary, but we will use the most commonly suggested metrics. Lastly, we will refine the framework implementation in order to collect data to support evaluating ATs based on the selected metrics.

- *Brain Data Collection.* Chapter 6 demonstrated the capability of using EEG to measure brain activity while subjects perform virtual navigation tasks. We want to scale this up to collect more brain data to better understand, not only our brain, but also how visually impaired persons perform visual tasks such as navigation. This will involve large-scale testing with both sighted and visually impaired subjects. We also plan to experiment with a portable and wireless EEG technology called EMOTIV, along with GIVE-ME.

- *Comprehensive benchmarking.* With the selected metrics, we will be able to build a comprehensive benchmark dataset for visual navigational ATs for VIPs. This will be a challenging and daunting task because most state-of-the-art AT systems or algorithms are either dependent on specific hardware or not fully described. Hardware can be very expensive to acquire. Thus, an alternative is to use an approximation of the

prototype for evaluation, since we would need to simulate the sensor component of the AT anyway.

Appendix A

Visual Function and Disorders

There are three levels of visual function: normal vision, low vision, and total blindness. Low vision is considered as having a visual acuity of 20/40 or worse in the better-seeing eye. Total blindness or legally blind, is considered as having a visual acuity of 20/200 or worse [29, 67]. Visual acuity measures the clearness of our vision. Loosely speaking, visual acuity measures the minimum distance between two points moving closer and closer together until the eye can no longer distinguish it as two points [87]. For example, if one's vision is 20/200, it means that at 20ft from an eye chart, that person can read what a healthy person can read at 200ft.

The most common eye disorders are:

Refractive errors Inability of the eyes to clearly focus and thus results in blurred vision. This includes myopia (near-sightedness), hyperopia (far-sightedness), astigmatism (distorted vision in all distances due to irregularly curved cornea), and presbyopia (inability to focus up-close).

Age-related macular degeneration (AMD) Abnormal aging of the retina that damages the central vision (see Figure A.1a).

Cataract Clouding of the eye's lens (see Figure A.1b).

Diabetic retinopathy A complication of diabetes that progressively damages retina's blood vessels (see Figure A.1c).

Glaucoma A group of diseases that damages the optic nerves due to increased fluid pressure in the eye (see Figure A.1d).

(a) Age-related macular degeneration

(b) Cataracts

(c) Diabetic retinopathy

(d) Glaucoma

Figure A.1: Common Eye Disorders [29]

While all these disorders deserve every attention to find cures/treatments and preventions, we will focus on visual impairment that is caused by AMD and other outer retinal degenerative diseases, such as retinitis pigmentosa (RP). RP is a group of genetic defects that damage photoreceptors. Although AMD is more common, RP is more severe and has no cure [29, 67, 87].

Appendix B

System Usability Scale (SUS)

Developed by John Brooke, the system usability scale (SUS) provides a quick way for measuring usability [15]. It consists of ten questions with five Likert-scale responses, ranges from Strongly Disagree to Strongly Agree. There are five positive statements and five negative statements, which alternate. Below is a sample questionnaire we used for VistaNav experiment (Chapter 7).

Based on your interactions with the product thus far, please use the following scale to rate the product: from 1 (Strongly DISAGREE) to 5 (Strongly AGREE).

Please check the box that reflects your immediate response to each statement. Don't think too long about each statement. If you don't know how to respond, simply check "3."

1. I think that I would like to use this product frequently.

2. I found the product unnecessarily complex.

3. I thought the product was easy to use.

4. I think that I would need the support of a technical person to be able to use this product.

5. I found the various functions in the product were well integrated.

6. I thought there was too much inconsistency in this product.

7. I imagine that most people would learn to use this product very quickly.

8. I found the product very awkward to use.

9. I felt very confident using the product.

10. I needed to learn a lot of things before I could get going with this product.

SUS score is computed using the following equation:

$$SUS = [\sum(odd\_number\_score - 1) + \sum(5 - even\_number\_score)] \times 2.5 \qquad \text{(B.1)}$$

The equation converts the ratings to a score of 0 to 100. Note, however, that these scores are not percentages, but rather a relative ranking. To add more meaningful information to this ranking, we added the following adjective question:

1. Overall, I would rate the user-friendliness of this product as:

    - Worst Imaginable
    - Awful
    - Poor
    - OK
    - Good
    - Excellent
    - Best Imaginable

Appendix C

Candidate's Publications

Book Chapters

1. E. Molina, **W. L. Khoo**, H. Tang and Z. Zhu (2017). Registration of Video Images. In A. A. Goshtasby (Ed.), *Theory and Application of Image Registration*. (*Invited*) Hoboken, NJ: Wiley Press.

Peer-reviewed Journals

1. M. Vincent, H. Tang, **W. L. Khoo**, Z. Zhu, T. Ro. Shape Discrimination using the Tongue: Implications for a Visual-to-Tactile Sensory Substitution Device. *Multisensory Research* (*Pending a final decision after a minor revision*)

2. **W. L. Khoo**, G. Olmschenk, Z. Zhu, H. Tong, W. H. Seiple, and T. Ro. Development and Evaluation of Mobile Crowd Assisted Navigation for the Visually Impaired. *IEEE Transactions on Services Computing* (*Under review; GO and WK equal contribution*).

3. **W. L. Khoo** and Z. Zhu. Multimodal and Alternative Perception for the Visually Impaired: A Survey. *Journal of Assistive Technologies* 10 (1). pp. 11-26. 2016.

4. **W. L. Khoo**, J. Knapp, F. Palmer, T. Ro, and Z. Zhu. (2013). Designing and Testing Wearable Range-Vibrotactile Devices. *Journal of Assistive Technologies, 7(2).*

Conference Proceedings

1. Z. Zhu, **W. L. Khoo**, C. Santistevan, Y. Gosser, E. Molina, H. Tang, T. Ro, and Y. Tian. EFRI-REM at CCNY: Research Experience and Mentoring in Multimodal and Alternative Perception for Visually Impaired People. *6th IEEE Integrated STEM Education Conference (ISEC '16)*, March 5, 2016, Princeton, NJ.

2. E. Molina, **W. L. Khoo**, F. Palmer, L. Ai, T. Ro and Z. Zhu. Vista Wearable: Seeing through Whole-Body Touch without Contact. *IEEE 12th International Conference on Ubiquitous Intelligence and Computing*, August 10-14, 2015, Beijing, China.

3. **W. L. Khoo**, G. Olmschenk, Z. Zhu, and T. Ro. Evaluating crowd sourced navigation for the visually impaired in a virtual environment. In *IEEE 4th International Conference on Mobile Services*, pp. 431-437. 2015

4. **W. L. Khoo**, E. L. Seidel, and Z. Zhu. Designing a Virtual Environment to Evaluate Multimodal Sensors for Assisting the Visually Impaired. 13^{th} *International Conference on Computers Helping People with Special Needs (ICCHP)*, 7383, Springer Berlin Heidelberg, July 11-13, 2012, Linz, Austria, 573-580

5. A. Khan, J. Lopez, F. Moideen, **W. L. Khoo**, and Z. Zhu. KinDetect: Kinect Detecting Objects. 13^{th} *International Conference on Computers Helping People with Special Needs (ICCHP)*, 7383, Springer Berlin Heidelberg, July 11-13, 2012, Linz, Austria, 588-595

6. Y. Qu, **W. Khoo**, E. Molina, and Z. Zhu. Multimodal 3D Panoramic Imaging Using a Precise Rotating Platform. 2010 *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, July $6^{th} - 9^{th}$, 2010, 260-265

7. **W. Khoo**, T. Jordan, D. Stork, and Z. Zhu. Reconstruction of a Three-Dimensional Tableau from a Single Realist Painting, 15^{th} *International Conference on Virtual Systems and Multimedia*, September 9-12, 2009, 9-14

8. T. Jordan, D. Stork, **W. Khoo**, and Z. Zhu. Finding Intrinsic and Extrinsic Viewing Parameters from a Single Realist Painting, 13^{th} *International Conference on Computer Analysis of Images and Patterns*, 5702, Springer Berlin Heidelberg, September 2-4, 2009, 293-300

Technical Report

1. **W. Khoo** and Z. Zhu. 3D Measurements and Visualization of Gamma-Ray Cargo Inspection, *City College Grove School of Engineering Journal of Student Research*, May 2009

Patents Pending and Provisional

1. Z. Zhu, T. Ro, L. Ai, **W. L. Khoo**, E. Molina, F. Palmer. Wearable Navigation Assistance for the Vision-impaired, December 27, 2013. US Patent App. 14/141,742 (pending)

Bibliography

[1] Ziad O Abu-Faraj, Elie Jabbour, Paul Ibrahim, and Anthony Ghaoui. Design and development of a prototype rehabilitative shoes and spectacles for the blind. In *Biomedical Engineering and Informatics (BMEI), 2012 5th International Conference on*, pages 795–799. IEEE, 2012.

[2] AK Ahuja, JD Dorn, A Caspi, MJ McMahon, G Dagnelie, P Stanga, MS Humayun, RJ Greenberg, et al. Blind subjects implanted with the argus ii retinal prosthesis are able to improve performance in a spatial-motor task. *British Journal of Ophthalmology*, 95(4):539–543, 2011.

[3] Ashish Kishore Ahuja and Matthew R. Behrend. The Argus$^{\text{TM}}$II retinal prosthesis: Factors affecting patient selection for implantation. *Progress in Retinal and Eye Research*, 36(0):1 – 23, 2013.

[4] Lei Ai and Tony Ro. The phase of prestimulus alpha oscillations affects tactile perception. *Journal of neurophysiology*, 111(6):1300–1307, 2014.

[5] Aries Arditi and YingLi Tian. User interface preferences in the design of a camera-based navigation and wayfinding aid. *Journal of Visual Impairment & Blindness*, 2013.

[6] Paul Bach-y Rita, Kurt A Kaczmarek, Mitchell E Tyler, and Jorge Garcia-Lara. Form perception with a 49-point electrotactile stimulus array on the tongue: a technical note. *Journal of rehabilitation research and development*, 35(4):427, 1998.

[7] Paul Bach-y Rita, Mitchell E Tyler, and Kurt A Kaczmarek. Seeing with the brain. *International journal of human-computer interaction*, 15(2):285–295, 2003.

[8] Michael Banf and Volker Blanz. Sonification of images for the visually impaired using a multi-level approach. In *Proceedings of the 4th Augmented Human International Conference*, pages 162–169. ACM, 2013.

[9] Gabriel Barata, Sandra Gama, Joaquim Jorge, and Daniel Gonçalves. Improving participation and learning with gamification. In *Proceedings of the First International Conference on Gameful Design, Research, and Applications*, pages 10–17. ACM, 2013.

[10] Stefano Basso, Guglielmo Frigo, and Giada Giorgi. A smartphone-based indoor localization system for visually impaired people. In *Medical Measurements and Applications (MeMeA), 2015 IEEE International Symposium on*, pages 543–548. IEEE, 2015.

[11] Michael S Beauchamp, Nafi E Yasar, Richard E Frye, and Tony Ro. Touch, sound and vision in human superior temporal sulcus. *Neuroimage*, 41(3):1011–1020, 2008.

[12] Clodis Boscarioli, Marcio Seiji Oyamada, Jorge Bidarra, and Marcelo Fudo Rech. Evaluating the interaction of users with low vision in a multimodal environment. In *ACHI 2013, The Sixth International Conference on Advances in Computer-Human Interactions*, pages 256–262, 2013.

[13] Robin Brewer, Lisa Anthony, Quincy Brown, Germaine Irwin, Jaye Nias, and Berthel Tate. Using gamification to motivate children to complete empirical studies in lab environments. In *Proceedings of the 12th International Conference on Interaction Design and Children*, pages 388–391. ACM, 2013.

[14] John Brooke. Sus: a retrospective. *Journal of usability studies*, 8(2):29–40, 2013.

[15] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[16] Leandro Cancar, Alex Díaz, Antonio Barrientos, David Travieso, and David Jacobs. Tactile-sight: A sensory substitution device based on distance-related vibrotactile flow. *interface*, 14:21, 2013.

[17] Tilanka Chandrasekera, So-Yeon Yoon, and Newton DSouza. Virtual environments with soundscapes: a study on immersion and effects of spatial abilities. *Environment and Planning B: Planning and Design*, 42:000–000, 2015.

[18] Daniel-Robert Chebat, Shachar Maidenbaum, and Amir Amedi. Navigation using sensory substitution in real and virtual mazes. *PloS one*, 10(6), 2015.

[19] EC Connors, LA Yazzolino, J Sánchez, and LB Merabet. Development of an audio-based virtual gaming environment to assist with navigation skills in the blind. *Journal of visualized experiments: JoVE*, (73), 2012.

[20] Jos Arnaldo Shiomi da Cruz, Sabrina Thalita dos Reis, Rodrigo Marcus Cunha Frati, Ricardo Jordo Duarte, Hiep Nguyen, Miguel Srougi, and Carlo Camargo Passerotti. Does warm-up training in a virtual reality simulator improve surgical performance? a prospective randomized analysis. *Journal of Surgical Education*, 2016.

[21] L da Cruz, B Coley, P Christopher, F Merlini, V Wuyyuru, JA Sahel, P Stanga, E Filley, G Dagnelie, Argus II Study Group, et al. Patients blinded by outer retinal dystrophies are able to identify letters using the argustm ii retinal prosthesis system. *Invest. Ophthalmol. Vis. Sci*, 51:2023, 2010.

[22] D. Dakopoulos and N.G. Bourbakis. Wearable obstacle avoidance electronic travel aids for blind: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(1):25–35, 2010.

[23] Yuri Danilov and Mitchell Tyler. Brainport: an alternative input to the brain. *Journal of integrative neuroscience*, 4(04):537–550, 2005.

[24] N. Degara, T. Kuppanda, F. Nagel, and A. Wolfsmantel. The walking game: A framework for evaluating sonification methods in blind navigation. In *4th Interactive Sonification Workshop (ISon 2013)*, volume 12, 2013.

[25] Norberto Degara, Frederik Nagel, and Thomas Hermann. SonEX: an evaluation exchange framework for reproducible sonification. In *Proceedings of the 19th International Conference on Auditory Displays*, 2013.

[26] K. Dergousoff and R. L. Mandryk. Mobile gamification for crowdsourcing data collection: Leveraging the freemium model. In *33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1065–1074. ACM, 2015.

[27] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O'Hara, and Dan Dixon. Gamification. using game-design elements in non-gaming contexts. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 2425–2428. ACM, 2011.

[28] David Feng, Janine Walker, Nick Barnes, and Chris McCarthy. A bi-modal visual representation can enhance orientation and mobility performance with less than 20 phosphenes. *Investigative Ophthalmology & Visual Science*, 55(13):1799–1799, 2014.

[29] Center for Disease Control and Prevention. Vision health initiative: Common eye disorders. `http://www.cdc.gov/visionhealth/basics/ced/index.html`. Accessed: 2016-09-07.

[30] Cheryl D Fryar, Qiuping Gu, and Cynthia L Ogden. Anthropometric reference data for children and adults: United states, 2007-2010. *Vital and health statistics. Series 11, Data from the national health survey*, (252):1–48, 2012. Available: `http://www.cdc.gov/nchs/data/series/sr_11/sr11_252.pdf`.

[31] Neveen I Ghali, Omar Soluiman, Nashwa El-Bendary, Tamer M Nassef, Sara A Ahmed, Yomna M Elbarawy, and Aboul Ella Hassanien. Virtual reality technology for blind and visual impaired people: Reviews and recent advances. In *Advances in Robotics and Virtual Reality*, pages 363–385. Springer, 2012.

[32] Nicholas A Giudice and Gordon E Legge. Blind navigation and the role of technology. *Engineering handbook of smart technology for aging, disability, and independence*, pages 479–500, 2008.

[33] Jennifer Goldman, Glenn Stebbins, Doral Fredericks, and Mike Upchurch. Experiencing parkinsons disease psychosis via virtual reality simulation: A novel and effective educational tool (p1.011). *Neurology*, 86(16 Supplement), 2016.

[34] Hong Guo, Yuan Yang, Guan Gu, Yisheng Zhu, and Yihong Qiu. Phosphene object perception employs holistic processing during early visual processing stage. *Artificial organs*, 2013.

[35] J. Hamari, J. Koivisto, and H Sarsa. Does gamification work?–a literature review of empirical studies on gamification. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*, pages 3025–3034. IEEE, January 2014.

[36] Masayuki Hara, Solaiman Shokur, Akio Yamamoto, Toshiro Higuchi, Roger Gassert, and Hannes Bleuler. Virtual environment to evaluate multimodal feedback strategies for augmented navigation of the visually impaired. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 975–978. IEEE, 2010.

[37] Feng Hu, Zhigang Zhu, and Jianting Zhang. Mobile panoramic vision for assisting blind via indexing and localization. In *2014 European Conference on Computer Vision(ECCV) Workshops*, 2014.

[38] Ying Ying Huang. *Design and evaluation of 3D multimodal virtual environments for visually impaired people*. PhD thesis, KTH, 2010.

[39] Ying Ying Huang. Exploration in 3d virtual worlds with haptic-audio support for nonvisual spatial recognition. *Human-Computer Interaction*, 332:269–272, 2010.

[40] Yingying Huang. Exploration on interface usability in a haptic 3d virtual labyrinth for visually impaired users. *proceeding of IADIS Interfaces and Human Computer Interaction (IHCI). ALGALVE, Portugal*, 2009.

[41] YINGYING HUANG, Jonas Moll, Eva-Lotta Sallnäs, and Yngve Sundblad. Integrating audio and haptic feedback in a collaborative virtual environment. In *proceeding of HCI International Conference*, 2007.

[42] Horace Josh, Benedict Yong, and Lindsay Kleeman. A real-time and portable bionic eye simulator. In Joaquim Gabriel, Jan Schier, Sabine Huffel, Emmanuel Conchon, Carlos Correia, Ana Fred, and Hugo Gamboa, editors, *Biomedical Engineering Systems and Technologies*, volume 357 of *Communications in Computer and Information Science*, pages 51–67. Springer Berlin Heidelberg, 2013.

[43] Simeon Keates, P John Clarkson, Lee-Anne Harrison, and Peter Robinson. Towards a practical inclusive design approach. In *Proceedings on the 2000 conference on Universal Usability*, pages 45–52. ACM, 2000.

[44] Shawn K Kelly, Douglas B Shire, Jinghua Chen, Marcus D Gingerich, Stuart F Cogan, William A Drohan, William Ellersick, Ashwati Krishnan, Sonny Behan, John L Wyatt, et al. Developments on the boston 256-channel retinal implant. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–6. IEEE, 2013.

[45] Atif Khan, Febin Moideen, Juan Lopez, Wai L. Khoo, and Zhigang Zhu. Kindectect: Kinect detecting objects. In Klaus Miesenberger, Arthur Karshmer, Petr Penaz, and Wolfgang Zagler, editors, *Computers Helping People with Special Needs*, volume 7383 of *Lecture Notes in Computer Science*, pages 588–595. Springer Berlin Heidelberg, 2012.

[46] Wai L. Khoo, Joey Knapp, Franklin Palmer, Tony Ro, and Zhigang Zhu. Designing and testing wearable range-vibrotactile devices. *Journal of Assistive Technologies*, 7(2):102–117, 2013.

[47] Wai L. Khoo, Greg Olmschenk, Zhigang Zhu, and Tony Ro. Evaluating crowd sourced navigation for the visually impaired in a virtual environment. In *IEEE 4th International Conference on Mobile Services*, pages 431–437. IEEE, 2015.

[48] Wai L. Khoo, Eric L. Seidel, and Zhigang Zhu. Designing a virtual environment to evaluate multimodal sensors for assisting the visually impaired. In *Proceedings of the 13th international conference on Computers Helping People with Special Needs - Volume Part II*, ICCHP'12, pages 573–580, Berlin, Heidelberg, 2012. Springer-Verlag.

[49] Wai L. Khoo and Zhigang Zhu. Multimodal and alternative perception for the visually impaired: A survey. *Journal of Assistive Technologies*, in press. Submitted and accepted in 2015.

[50] Orly Lahav and David Mioduser. Haptic-feedback support for cognitive mapping of unknown spaces by people who are blind. *International Journal of Human-Computer Studies*, 66(1):23–35, 2008.

[51] Orly Lahav, David Schloerb, Siddarth Kumar, and Mandyam Srinivasan. A virtual environment for people who are blind–a usability study. *Journal of Assistive Technologies*, 6(1):38–52, 2012.

[52] Gregoire S. Larue, Christian Wullems, and Anjum Naweed. Evaluation of a new level crossing warning concept to improve safety of level crossings in remote locations. In *11th World Congress on Railway Research*, 2016.

[53] Wai Ho Li. Wearable computer vision systems for a cortical visual prosthesis. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*, pages 428–435, Dec 2013.

[54] Wei Li, Farnaz Abtahi, and Zhigang Zhu. A deep feature based multi-kernel learning approach for video emotion recognition. In *17th ACM International Conference on Multimodal Interaction*. ACM, November in press, 2015.

[55] Silvia Malatini and Alessandro Bogliolo. Gamification in mobile applications usability evaluation: A new approach. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*, pages 897–899. ACM, 2015.

[56] Stefano Mattoccia and Paolo Macri. 3d glasses as mobility aid for visually impaired people. In *Computer Vision-ECCV 2014 Workshops*, pages 539–554. Springer, 2014.

[57] Chris McCarthy and Nick Barnes. Time-to-contact maps for navigation with a low resolution visual prosthesis. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 2780–2783. IEEE, 2012.

[58] Anita Meier, Denys JC Matthies, Bodo Urban, and Reto Wettach. Exploring vibrotactile feedback on the body and foot for the purpose of pedestrian navigation. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction*, page 11. ACM, 2015.

[59] Peter B. L. Meijer. An experimental system for auditory image representations. *Biomedical Engineering, IEEE Transactions on*, 39(2):112–121, 1992. Available: `http://www.seeingwithsound.com/`.

[60] M Alex Meredith, James Kryklywy, Amee J McMillan, Shveta Malhotra, Ryan Lum-Tai, and Stephen G Lomber. Crossmodal reorganization in the early deaf switches sensory, but not behavioral roles of auditory cortex. *Proceedings of the National Academy of Sciences*, 108(21):8856–8861, 2011.

[61] Edgardo Molina, Wai L. Khoo, Franklin Palmer, Lei Ai, Tony Ro, and Zhigang Zhu. Vista wearable: Seeing through whole-body touch without contact. In *IEEE 12th International Conference on Ubiquitous Intelligence and Computing*, To appear, 2015.

[62] Jonas Moll, Yingying Huang, and Eva-Lotta Sallnäs. Audio makes a difference in haptic collaborative virtual environments. *Interacting with Computers*, 22(6):544–555, 2010.

[63] Austin M Mulloy, Cindy Gevarter, Megan Hopkins, Kevin S Sutherland, and Sathiyaprakash T Ramdoss. Assistive technology for students with visual impairments and blindness. In *Assistive Technologies for People with Diverse Abilities*, pages 113–156. Springer, 2014.

[64] H Naganuma, K Kiyoyama, and T Tanaka. A 37× 37 pixels artificial retina chip with edge enhancement function for 3-d stacked fully implantable retinal prosthesis. In *Biomedical Circuits and Systems Conference (BioCAS), 2012 IEEE*, pages 212–215. IEEE, 2012.

[65] Annika Neidhardt, Bernhard Fiedler, and Tobias Heinl. Auditory perception of the listening position in virtual rooms using static and dynamic binaural synthesis. In *Audio Engineering Society Convention 140*, May 2016.

[66] Greg Olmschenk, Christopher Yang, Zhigang Zhu, Hanghang Tong, and William H. Seiple. Mobile crowd assisted navigation for the visually impaired. In *Ubiquitous Intelligence and Computing (UIC), 2015 IEEE 12th Intl Conf on*, page In press. IEEE, 2015.

[67] World Health Organization. Visual impairement and blindness. fact sheet n*282. june 2012. `http://www.who.int/mediacentre/factsheets/fs282/en/index.html`. Accessed: 2013-08-26.

[68] Frank G. Palmer, Zhigang Zhu, and Tony Ro. Wearable range-vibrotactile field: Design and evaluation. In Klaus Miesenberger, Arthur Karshmer, Petr Penaz, and Wolfgang Zagler, editors, *Computers Helping People with Special Needs*, volume 7383 of *Lecture Notes in Computer Science*, pages 125–132. Springer Berlin Heidelberg, 2012.

[69] Jason Park, Helen MacRae, Laura J. Musselman, Peter Rossos, Stanley J. Hamstra, Stephen Wolman, and Richard K. Reznick. Randomized controlled trial of virtual reality simulator training: transfer to live patients. *The American Journal of Surgery*, 194(2):205–211, 2007.

[70] S. Parsons and P. Mitchell. The potential of virtual reality in social skills training for people with autistic spectrum disorders. *Journal of Intellectual Disability Research*, 46(5):430–443, 2002.

[71] Ornella Plos, Stphanie Buisine, Amziane Aoussat, Fabrice Mantelet, and Claude Dumas. A universalist strategy for the design of assistive technology. *International Journal of Industrial Ergonomics*, 42(6):533 – 541, 2012.

[72] Ondřej Poláček, Thomas Grill, and Manfred Tscheligi. Towards a navigation system for blind people: a wizard of oz study. *SIGACCESS Access. Comput.*, (104):12–29, September 2012.

[73] Michael J. Proulx, David J. Brown, Achille Pasqualotto, and Peter Meijer. Multisensory perceptual learning and sensory substitution. *Neuroscience & Biobehavioral Reviews*, 2012.

[74] Maurice Ptito, Isabelle Matteau, Arthur Zhi Wang, Olaf B Paulson, Hartwig R Siebner, and Ron Kupers. Crossmodal recruitment of the ventral visual stream in congenital blindness. *Neural plasticity*, 2012, 2012.

[75] Francis Quek and Francisco Oliveira. Enabling the blind to see gestures. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 20(1):4, 2013.

[76] Alejandro R Garcia Ramirez, Renato Fonseca Livramento da Silva, Milton Jose Cinelli, and Alejandro Durán Carrillo de Albornoz. Evaluation of electronic haptic device for blind and visually impaired people: A case study. *Journal of Medical and Biological Engineering*, 32(6):423–427, 2012.

[77] Emiliano Ricciardi and Pietro Pietrini. New light from the dark: what blindness can teach us about brain function. *Current opinion in neurology*, 24(4):357–363, 2011.

[78] Chad Richards, Craig W Thompson, and Nicholas Graham. Beyond designing for motivation: the importance of context in gamification. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 217–226. ACM, 2014.

[79] Jaime Sánchez, Mauricio Sáenz, and Jose Miguel Garrido. Usability of a multimodal video game to improve navigation skills for blind children. *ACM Transactions on Accessible Computing (TACCESS)*, 3(2):7, 2010.

[80] Maria T. Schultheis and Albert A. Rizzo. The application of virtual reality technology in rehabilitation. *Rehabilitation Psychology*, 46(3):296–311, 2001.

[81] Mahadevan Subramaniam, Parvathi Chundi, Abhilash Muthuraj, Eyal Margalit, and Sylvie Sim. Simulating prosthetic vision with disortions for retinal prosthesis design. In *Proceedings of the 2012 international workshop on Smart health and wellbeing*, pages 57–64. ACM, 2012.

[82] Hao Tang, Norbu Tsering, and Feng Hu. Automatic pre-journey indoor map generation using autocad floor plan. In *31st Annual International Technology and Persons with Disabilities Conference*, 2015.

[83] MA Torres-Gil, O Casanova-Gonzalez, and JL Gonzalez-Mora. Applications of virtual reality for visually impaired people. *WSEAS Transactions on Computers*, 9(2):184–193, 2010.

[84] Thomas S Tullis and Jacqueline N Stetson. A comparison of questionnaires for assessing website usability. In *Usability Professional Association Conference*, pages 1–12. Citeseer, 2004.

[85] Iwan Ulrich and Johann Borenstein. The guidecane-applying mobile robot technologies to assist the visually impaired. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(2):131–136, 2001.

[86] Margaret Vincent, Hao Tang, Wai L. Khoo, Zhigang Zhu, and Tony Ro. Shape discrimination using the tongue: Implications for a visual-to-tactile sensory substitution device. *Multisensory Research*, 2016.

[87] James Weiland and Mark Humayun. Retinal prosthesis. In Bin He, editor, *Neural Engineering*, pages 635–655. Springer US, 2013.

[88] James D. Weiland, Alice K. Cho, and Mark S. Humayun. Retinal prostheses: Current clinical results and future needs. *Ophthalmology*, 118(11):2227 – 2237, 2011.

[89] James D Weiland, Neha Parikh, Vivek Pradeep, and Gerard Medioni. Smart image processing system for retinal prosthesis. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 300–303. IEEE, 2012.

[90] Bob G. Witmer, John H. Bailey, Bruce W. Knerr, and Kimberly C. Parsons. Virtual spaces and real world places: transfer of route knowledge. *International Journal of Human-Computer Studies*, 45(4):413–428, 1996.

[91] Rayoung Yang, Sangmi Park, Sonali R Mishra, Zhenan Hong, Clint Newsom, Hyeon Joo, Erik Hofer, and Mark W Newman. Supporting spatial awareness and independent wayfinding for pedestrians with visual impairments. In *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*, pages 27–34. ACM, 2011.

[92] Koji Yatani, Nikola Banovic, and Khai Truong. Spacesense: representing geographical information to visually impaired people using spatial tactile feedback. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 415–424. ACM, 2012.

[93] Marc P Zapf, Paul B Matteucci, Nigel H Lovell, Shilie Zheng, and Gregg J Suaning. Towards photorealistic and immersive virtual-reality environments for simulated prosthetic vision: Integrating recent breakthroughs in consumer hardware and software. In

Engineering in Medicine and Biology Society (EMBC), 2014 36th Annual International Conference of the IEEE, pages 2597–2600. IEEE, 2014.

[94] Marc Patrick H Zapf, Mei-Ying Boon, Paul B Matteucci, Nigel H Lovell, and Gregg J Suaning. Towards an assistive peripheral visual prosthesis for long-term treatment of retinitis pigmentosa: evaluating mobility performance in immersive simulations. *Journal of neural engineering*, 12(3):036001, 2015.