A TECHNIQUE FOR ESTIMATING THREE-DIMENSIONAL VOLUME-OF-INTEREST USING EYE GAZE

A thesis presented to the faculty of the Graduate School of
Western Carolina University in partial fulfillment of the
requirements for the degree of Master of Science in Technology.

By

Carl Cole Drawdy IV

Director: Paul Yanik, Ph.D.
Assistant Professor
Department of Engineering and Technology

Committee Members:
Yanjun Yan, Ph.D., Department of Engineering and Technology
Peter Tay, Ph.D., Department of Engineering and Technology

March 2015

ACKNOWLEDGEMENTS

I would like to extend my gratitude to the Department of Engineering and Technology at Western Carolina University for facilitating the ideas and imagination of the students in the graduate program. An abundance of gratitude is extended to my committee members Dr. Yanjun Yan and Dr. Peter Tay for their guidance in several facets of this research. Special thanks is extended to the Department of Psychology at Western Carolina University and specifically to Dr. William Poynter for making their eye tracking resources available to me and my research. Finally, immeasurable respect and gratitude is given to my advisor Dr. Paul Yanik. His dedication to my research was only surpassed by my own. His guidance in preliminary research, method implementation, and writing was paramount to the timely completion and success of this thesis. Thank you so much for your time, help, supporting words, and technical guidance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A TECHNIQUE FOR ESTIMATING THREE-DIMENSIONAL VOLUME-OF-INTEREST USING EYE GAZE

Carl Cole Drawdy IV, M.S.T.

Western Carolina University (March 2015)

Director: Paul Yanik, Ph.D.

Assistive robotics promises to be of use to those who have limited mobility or dexterity. Moreover, those who have limited movement of limbs can benefit greatly from such assistive devices. However, to use such devices, one would need to give commands to an assistive agent, often in the form of speech, gesture, or text. The need for a more convenient method of Human-Robot Interaction (HRI) is prevalent, especially for impaired users because of severe mobility constraints.

For a socially responsive assistive device to be an effective aid, the device generally should understand the intention of the user. Also, to perform a task based on gesture, the assistive device requires the user's area of attention in three-dimensional (3D) space. Gaze tracking can be used as a method to determine a specific volume of interest (VOI). However, heretofore gaze tracking has been under-utilized as a means of interaction and control in 3D space.

The main objective of this research is to determine a practical VOI in which an individual's eyes are focused by combining existing methods. Achieving this objective sets a foundation for further use of vergence data as a useful discriminant to generate a proper directive technique for assistive robotics.

This research investigates the accuracy of the Vector Intersection (VI) model when applied to a usable workspace. A neural network is also applied to gaze data for use in tandem with the VI model to create a Combined Model. The output of the Combined

Model is a VOI that can be used to aid in a number of applications including robot path planning, entertainment, ubiquitous computing, and others. An alternative Search Region method is investigated as well.

CHAPTER 1: INTRODUCTION

1.1 Background and Motivation

Assistive robotics promises to be of use to those who have limited mobility. Moreover, those who have limited movement of limbs may benefit greatly from such assistive devices. However, to use such devices, one would need to issue commands to a robotic agent, often in the form of speech, gesture, or text. The need for a more convenient method of Human-Robot Interaction (HRI) is prevalent, especially for impaired users due to severe mobility restriction.

There are several types of assistive devices. Most assistive devices do not exhibit social intelligence or sensitivity to commands. Some are capable of taking direction from the user or of adapting to the user's needs and are socially intelligent [1]. For a user to direct such an assistive device, the device must know the intention of the user. Also, to perform a task in 3D space, the assistive device could contain predetermined programming or receive current information on the user's area of attention. The latter is investigated in this research.

There are many ways of representing a user's area of attention to an assistive device. The most common is the use of gestures. Motion sensors can be used to track an individual's movements in order to receive commands. Some of the common gestures are pointing or motioning with the arm, hand, or head. A more intuitive method of determining the attention of an individual is tracking their line of sight (LoS) or gaze vector to a point of gaze (PoG). Where an individual is looking is a clear indicator of where their attention is located [2]. A recent and established focus of research using gaze tracking as a tool has involved navigation of graphical user interfaces (GUI) [3].

Gaze tracking systems are used to find the PoG of an individual. The motivation for constructing such systems is, in general, to estimate an individual's gaze target. Two types of systems emerge when investigating gaze tracking: the 2-Dimensional system and 3-Dimensional system.

The 2-Dimensional gaze tracking system as referred to by this research does not refer to how the eye is physically modeled, but to what the system uses as a display to the user. In 2-Dimensional systems, the system estimates where on a particular plane, such as a video screen, the user is looking. That is, the system calibrates to a particular display at a certain distance from the user. These types of systems are used for GUI navigation which includes the use of the *eye-mouse* and *eye-typing*. Currently, there are several accurate systems that have been commercialized to navigate GUI interfaces [4], [5], [6].

Although 2-Dimensional gaze tracking systems are useful when working with 2D devices, they limit the ability of a user to interact with a device that moves or utilizes 3D space. The extension of gaze tracking from 2D to 3D has been the focus of research by several authors since the early 2000s [7], [8], [9], [10], [2], [11]. Unlike 2D gaze tracking systems, 3D systems aim to use a stereoscopic environment (virtual reality) or a natural environment (reality) as the display to an individual. However, 3-Dimensional systems introduce complications because estimation techniques become more difficult with the added dimension.

For an individual to interact with a device in 3D space, the gaze tracking system must know the fixation point or area of interest of the individual in 3D space. These systems differ from gesture recognition systems or systems that track the 3D coordinates of a stylus because there is no item to physically track. 3D gaze tracking systems must use features that characterize the eye or the physical appearance of eyes to estimate the PoG of an individual. Therefore, saccadic rhythms, vergence, and other eye characteristics must be investigated [12].

Fixation, saccades, and vergence are high level characteristics in eye movement that are important to consider when determining the PoG for an individual. Fixation is the action of keeping the eye in a fixed position. Saccades are quick rotations of the eye between points of fixation. Vergence refers to the amount that two eyes converge or diverge when focusing on objects at particular distances [13].

As stated earlier, eye tracking systems that are incorporated to navigate a GUI use a 2-dimensional format. These tracking devices use data collected from the characteristics of eye movements and classify these movements into commands. Generally, these systems include sensory hardware that collect a user's eye movement data, and software to correlate the movement data to a PoG on a monitor or 2D screen. Hardware and software of this type will be used for research in this study and extended to use in 3D gaze estimation.

In summary, the need for convenient methods of directing assistive robotics suggests the use of 3D gaze tracking. To utilize gaze tracking as a proper tool to navigate a robotic device, prior studies and methods for 3D gaze tracking are investigated. The findings and conclusions of these studies are used to formulate a logical method to determine the efficacy of using gaze data collected in a 2D setting for finding 3D PoG in the interest of directing an assitive device.

1.2 Purpose

The purpose of this research is to use eye tracking to extend the recent 2D advances in eye tracking into a 3D environment. Instead of collecting data only on eye movement characteristics that correspond to point of gaze (PoG) at one depth plane, vergence data can be collected to correspond to other depth planes. The data collected could be used to locate an individual's area of interest using gaze tracking only.

1.3 Objectives

The main objective of this research is to investigate vergence as *one* characteristic in *serveral* that should be included in a gaze directed assitive robot system. The technical objectives of this research are to first, apply a previously established non-contact, minimal calibration 3D gaze estimation model to determine a volume of interest in a usable workspace.

Secondly, the study will apply techniques that boost the performance of the model, setting a promising foundation for future research.

1.4 Significance of Study

Individuals may suffer from disease or spinal cord injury which impairs use of their limbs. For such individuals, eye tracking devices to interface with computers can be very advantageous. Applying this concept to a 3D application would give these users the option to interact with assistive robotics, in turn affording opportunities to interact with society with greater ease.

1.5 Limitations of the Study

The study will investigate how eye tracking data can be used to determine depth of gaze in real 3D space. Also, the study will use data collected to find a region of interest at varying depths in the user's field of vision. The study does not generate a generalized model for directing assistive robotics, and does not present a closed system that is interactive from the eye-tracking stage to the robotic movement stage. However, the study does lay proper foundation for future experiments on this topic.

CHAPTER 2: LITERATURE REVIEW

2.1 Eye Movement Characteristics

Toward detecting PoG, the classification of eye movements must be understood. In a recent study, eye movement characteristics are detected and classified according to their characteristic features. Blythe et al. [13] compare eye movements in three separate viewing scenarios: 2D representation, stereoscopic, and 3D representation (reality). The study aims to explain how saccadic rhythms and vergence act together in the viewing scenarios above.

Here, the researchers used several LEDs as stimuli to be viewed in all three scenarios. Results suggest what can and cannot be used as characteristics for detecting gaze depth in 3D situations. Important conclusions of the study show that in all viewing situations, the versional (strictly lateral) component of saccadic rhythms was the same. Also, it is apparent through the study that special attention must be given to saccadic rhythms not only during fixation, but also during lateral movement.

One conclusion that is revealed is that vergence magnitude is very similar in both stereoscopic settings and in real setting. This finding supports the motivation of this research. Also, the study notes that saccadic rhythms occur during fixation to continually receive sensory information. For the proposed research, their conclusions support that a filtering stage is preferable during data processing.

2.2 3-Dimensional PoG Estimation Techniques

Several methods have been developed to track PoG in both 2D applications and 3D applications. However, in both cases, the characteristics used are very much the same. For the purposes of this research, there are two prominent methods for tracking PoG in 3D space: the Pupil-Corneal Reflection technique and the model-based technique.

2.2.1 3D Gaze Tracking Models and Related Studies

Pupil-Corneal Reflection (PCR) utilizes the reflection created by light off the cornea and pupil to determine the orientation of the eye and direction of gaze. This method requires restriction of head movement due to its sensitivity to change of reflection off the eye. Also, the distance from the eye sensor must remain approximately static to achieve acceptable accuracy.

Model-based techniques can alleviate the need for the user to maintain constant head position. Shape-based models use a prior model of eye contours to track the movement of the eye. Appearance-based approaches build a model based on the appearance of the eye under analysis. Prediction templates are constructed for the eye in one position to predict the appearance of the eye under different movements. The predictions and actual images are then compared using a similarity measure to indicate the position and gaze direction of the eye [14].

Some model-based methods incorporate the PCR technique as a feature. Generally, methods that incorporate several techniques for eye tracking are referred to as *hybrid* models. Past studies have used such models to estimate PoG of an individual. Such tracking systems could be separated into different categories depending on (1) display type, (2) hardware requirements, and (3) the use of *a priori* data of the environment [2].

Duchowski *et al.* [7] developed a system to determine 3D PoG by using a head-mounted display. The display was stereoscopic, meaning each eye was presented images of slightly different point of view, simulating reality. An electromagnetic tracker was used to track head pose and a head mounted eye tracking system determined the PoG on each 2D screen presented to the respective eyes. Using both head pose information and two separate 2D data sets for each eye, the 3D PoG was estimated. The system required that users calibrate to the eye tracker, the distance between each eye, and the distance of the eyes to the head mounted screen.

Mitsugami *et al.* [8] developed a head mounted system to estimate 3D PoG from 2D PoG estimates using a view camera. The display was a real 3D setting. Intersection of *view lines* were used with known position of and orientation of the head to find 3D PoG. The authors took samples of view lines from different angles of head positions. This was simulated by laterally moving a fixation point across a viewing plane. A novel stochastic algorithm was used to improve the technique by using all viewing lines from the different angles. The viewing distances of the gaze targets were set at impractically far distances away (300 - 500 cm). This research is focused in using a similar idea to estimate 3D PoG, without the use of head gear and within a practical, but usable, volume.

Essig *et al.* [9] introduced a tracking system that utilized a neural network to estimate the 3D PoG based on 2D binocular data. A computer screen displayed random dot stereograms to users and data was collected on each eye's characteristics. The gaze tracker used was also head mounted. The system required that the user calibrate the gaze-tracking system to the monitor and that the neural network be trained.

Kwon *et al.* [10] displayed a stereoscopic setting using a 2D parallax barrier. A parallax barrier is split into several sections so to create the effect of 3D by displaying slightly different images to each eye without the use of headgear. This method required a static distance between the tracker and the user.

2.2.2 Recent Studies

Hennessey and Lawrence use a hybrid model to estimate the absolute (x,y,z) coordinates of an individual's gaze [2]. The authors' objective is to develop the first binocular system for estimating the absolute (x,y,z) gaze point coordinates of an individual when viewing a real world setting. Also, the authors claim that their model-based method is the first of its kind to detect 3D gaze points and that their method is the first to be noncontact. No head gear is required for their approach to calculate PoG in 3D space.

The study implements a three stage system. An image processing stage extracts image features. A model-fitting stage calculates corneal centers and the optical axes. The last stage uses a vergence algorithm for computing the 3D PoG.

The system configuration is composed of multiple infrared (IR) light sources, a high speed digital camera, and a set of 3D PoG markers. The IR sources are used during the image processing stage to extract the contour of the pupil in each eye. The high speed digital camera is used to record the corneal reflection off the eyes as a result of ON-axis and OFF-axis LEDs. The set of markers is used during testing. Participants fixate upon the markers and a comparison between estimated PoG coordinates and marker coordinates is performed. The workspace considered has dimensions of 30 × 23 × 25 cm. Seven test participants were used to calculate precision, accuracy, and robustness of the overall system.

The study also introduces a filtering technique to increase precision. The filtering technique is employed to reduce *jitter*, or what is formally known as saccadic movement [13]. This produces a latency of the system to a maximum of 1.5 s. Latency refers to the computational delay between the collection of data and the prediction by the system.

Results of the study show that the system has an average accuracy of 3.93 cm from the target. Accuracy error was calculated by the Euclidean distance error. The system allowed for a 3 × 9 × 14 cm head space volume which corresponds to the amount of movement participants' heads varied.

Specific drawbacks are not easily spotted in the Henessey and Lawrence study. The workspace of the Hennessey and Lawrence study proves too small for robotic movement, and it is one of the aims of this proposal to create a methodology to detect 3D gaze detection in a larger, more practical workspace. Studies such as Hanhela et al. [15], and Wang et al. [16], and Duchowski et al. [11], are examined.

Hanhela et al. [15] determines the link between number of observers and precision of gaze position estimation, and estimate the number of participants needed for sufficient estimation precision. The study focuses on extracting stereoscopic volumes of interest (SVOIs) from each participant, then intersecting these SVOIs to estimate the overall VOI. The experiment consisted of 13 participants. Each participant would view a single object (white ball) moving in 3D space on a stereoscopic display. The display was 30 cm from the participants, while the apparent depth of the ball ranged from 28.24 cm to 31.99 cm from the participant. The authors find that estimating the x and y ranges for the SVOI can be achieved with considerable accuracy with 4 participants. The z range estimation was more challenging. The reported minimum root mean squared error over all participants was 3.55 cm.

The methodology to estimate the SVOI for each participant was to build a 2D heat map of the left and right eye as the participant views the moving object. The heat maps are projected to each respective eye to create a cone-like structure which extends into the stereoscopic display. Intersecting these projections would result in a SVOI as the white ball moved around the display. Once the SVOIs were compiled for all participants, they were intersected to find the overall VOI to estimate where the ball was moving on the stereoscopic display.

Wang et al. [16] present an approach to 3D gaze estimation in stereoscopic displays comparable to proprietary methods. The authors incorporate an online filter and a depth calibration step and use a triangulation of gaze depth to produce depth estimates. Their method uses a derivation for the *disparity model.* This method is given in equation 2.1 below. To find the gaze depth z based on a Δx disparity.

$$\frac{-z}{D' - z} = \frac{\Delta x}{a} \rightarrow z = \frac{\Delta x D'}{\Delta x - a}, \tag{2.1}$$

where $D' = \sqrt{D^2 + y_e^2}$, $y_e = (y_l + y_r)/2$ and $a = 6.3$, the average separation of the eyes. Also, z is the estimated depth of gaze, Δx is the horizontal difference between gaze points, D' is the distance to the screen, which varies with height of the gaze points, and y_e is the average y coordinate in 3D space.

To triangulate the coordinate of gaze depth, the left and right eye 2D on-screen positions are measured. There is error associated with spatial triangulation, meaning both visual axis may not intersect in 3D space. However, the authors average the y-coordinate of the left and right eye gaze vectors to produce a usable y coordinate estimate.

To remove outliers, the authors replaced x and y coordinates that were two standard deviations away from the mean with the most recent valid reading. To calibrate the measured depth estimate in z and the actual depth in z, the authors introduce a linear system $S = ZB$, where S is the vector of known depth values, Z is the vector of measured depth values, and B is a vector of unknown coefficients.

Eight participants were used in the study. Each were asked to pass a preliminary depth test to be qualified for the study. Next, the participants underwent the 3D calibration step. Finally, in the testing phase, the participants were shown the sphere grid stimulus and were instructed to follow a sequence of highlighted spheres in a predetermined pattern on an 11 x 11 sphere grid space. Each row of the grid space was separated by 5 cm. The closest horizontal row of 11 spheres was located at 25 cm towards the viewer out of the screen. When the sequence was activated, the user would follow the highlighted sequence of spheres at five depths: 20, 10, 0, -10 and -20 cm. The value '0' is reserved as the "on-screen" position.

The study was completed on both a haploscope and a desktop active stereo system. The haploscope proved to be more consistent. Distance from screen to user of the haploscope setup was 86.36 cm. Images were rendered assuming an eye separation of 6.3 cm. The screen was 48 cm wide and 30 cm high. For the desktop setup, the screen was set at 50 cm from the user and the monitor was 47.5 cm x 29.2 cm.

2.2.3 The Vector Intersection Model

Duchowski et al. [11] build upon the Wang study by comparing depth estimation when viewing stereoscopic displays and viewing a similar physical scene. The scene was made of four Snellen charts, patterned after the stimulus used by Love et al. [17]. The stimuli for the stereoscopic display were placed at four apparent depths relative to the screen: -15, -5, 5, and 15 cm. The value ‘0’ is reserved again as the “on-screen” position. The charts were offset horizontally so that none would be directly behind the other. In the physical scene setup, the Snellen charts were printed out and placed on cardboard. They were placed at 40, 50, 60, and 70 cm from the user and were also horizontally placed so that no chart overlapped with the other.

For depth estimation in the stereoscopic scene, a Wheatstone system is implemented on a haploscope. This system uses the disparity model outlined by [16]. A prior step is needed for 2D calibration which is completed by proprietary software. For 3D calibration, methods of [16] are implemented for further filtering.

For depth estimation in the physical scene, the binocular Dikablis system is used. The hardware for this system is comprised of two individual cameras that track eye movements and produce monocular gaze estimates for each eye. The system also produces two scene camera videos representing the apparent view of each eye. The system requires a 2D calibration to a particular depth by using four points on a calibration plane. Since there is no physical screen to calibrate to, the authors use four calibration points on a wall, a certain distance from the user. Therefore, once the system is calibrated to the calibration plane and the 3D coordinates of the user’s eyes are known, gaze vectors can be produced from each eye to the calibration plane. Intersection of these vectors produces a gaze point that can be used to know where the user is looking. This spatial triangulation method is referred to as the Vector Intersection (VI) Model.

Data from 15 participants were used after the removal of datasets belonging to participants with procedural problems or outlier characteristics. The overall result of the study proves that for situations involving physical stimuli, and when a physical fixed accommodation screen is not present, the VI Model for computing depth is more accurate. The disparity model, while accurate for stereoscopic displays, is not an effective indicator for estimating gaze depth in physical environments. Therefore, the VI Model will be implemented in this research.

2.3 Neural Network

As stated earlier, work in [9] utilized a neural network to categorize gaze data when viewing autostereograms. This approach will be applied to the physical experimental setup proposed by this research. To validate the use of a multilayer neural network to 3D gaze estimation, an overview of neural network theory is provided using subject matter from [18].

2.3.1 Multilayer Neural Networks

A multilayer neural network is a method used primarily for statistical pattern recognition. It is an extension of simpler training classifiers which utilize linear discriminants and the gradient descent method to properly classify an input data set to an output data set. There are several cases in which these simpler methods effectively classify data and generate an effective hyperplane defined by linear functions. In a more complex circumstance in which a nonlinear hyperplane must be generated, multilayer neural networks are applicable.

The overall goal of any pattern recognition system is to clearly classify input data, or minimize classification error. Multilayer neural networks solve the problem of generating a set of nonlinear functions to define decision regions. By learning parameters that define the nonlinear functions while simultaneously learning parameters that define the linear discriminant, multilayer neural networks effectively define a proper nonlinear hyperplane *and* minimize classification error at the same time.

Generally, three layer and four layer neural networks are sufficient for complex classification. For the purposes of this research, given that there is a relatively low number of inputs, a three layer topology is utilized. While the four layer topology is based upon the same principles, it will not be investigated further [18].

2.3.2 Topology

The topology of a three layer neural network includes an input layer, hidden layer, and output layer. Each layer consists of a number of nodes or *units*. The number of input units is simply determined by *the dimensionality of what needs to be categorized*. The number of outputs is determined by *the number of classes*. More ambiguously, the number of hidden units cannot be specifically predetermined. Although, quantity thresholds can be generally established to produce effective classification results. These thresholds will be discussed in section 2.2.3.

2.3.3 Neural Network Training

Each input unit is "connected" once to every hidden unit, and each hidden unit is "connected" to every output unit once. Each hidden unit produces a weighted function of all inputs. Each output unit receives these functions from all respective hidden units, generating a function of all hidden units. Therefore, all input units affect the output at each unit through the hidden layer. The input to the hidden layer can be mathematically modeled in (2.2) and (2.3), taken from [18].

$$\begin{aligned} net_j &= \sum_{i=0}^{d} x_i w_{ji}, \text{ weighted sum of input units} && (2.2) \\ y_j &= f(net_j), \text{ nonlinear function produced by hidden unit } j, && (2.3) \end{aligned}$$

where, x_i is an i^{th} input and y_j is the output at the j^{th} output unit.

Note that i is the index for input units and j is the index for hidden units. The weight w_{ji} references the weight applied to the input at hidden unit j.

The hidden to output layer process can be modeled by (2.4), showing the weighted sum *net* for a given hidden unit k, and the output z from an output unit.

$$
\begin{aligned}
net_k &= \sum_{j=0}^{n_H} y_j w_{kj}, \text{ weighted sum of hidden units} && (2.4) \\
z_k &= f(net_k), \text{ nonlinear function produced by output unit } j && (2.5)
\end{aligned}
$$

Substituting (2.2) into (2.4) yields the output function for a given input $\mathbf{x}$ as given by (2.6).

$$
g_k(\mathbf{x}) = z_k = \sum_{j=0}^{n_H} f(\sum_{i=0}^{d} x_i w_{ji}) w_{kj} \tag{2.6}
$$

Equation (2.5) shows the forward feeding network topology. The network is trained in a supervised fashion as follows. To learn the input-hidden weights (w_{ji}) properly, the Backpropagation method must be applied. For a given input $\mathbf{x}$, backpropagation works by recursively calculating error between results of the output units and ideal results, or *target results* selected by the experimenter. Equation 2.7 gives the general definition for training error, $E(\mathbf{w})$:

$$
E(\mathbf{w}) = \frac{1}{2} \sum_{k=1}^{c} (t_k - z_k)^2, \tag{2.7}
$$

where t_k is the desired output, c is the number of output units, and $\mathbf{w}$ is a vector of input-hidden weights. The weights start as random values and change in direction using gradient descent to minimize $E(\mathbf{w})$. This change is generally expressed as

$$
\Delta \mathbf{w} \propto -\frac{\partial E}{\partial \mathbf{w}}. \tag{2.8}
$$

There are other characteristics that have an effect on the overall training of the network such as learning rate and sensitivity. This research is not concerned with improving the efficiency or speed of the established three layer neural network method. Descriptions of other characteristics are left to supplemental reading.

It is imperative that viable training set protocols are adhered to, such that the network is trained correctly. There are three main patterns that are used when training the network:

stochastic, *batch*, and *on-line* training patterns. Stochastic training patterns present random inputs from the training set, and the weights are adjusted accordingly after the complete input set is presented, or each *epoch.* In Batch training, patterns are presented to the network before any learning takes place. On-line training requires each pattern is presented only once to the network. This research will use a stochastic training pattern so that the system is not over-trained to any particular group of gaze data.

With any pattern classification system, it is important to have protocols that indicate when to halt training. The neural network uses a cross-validation to accomplish this task. The validation set consists of patterns not yet used in the gradient descent training. If n represents the total number of patterns in D (the input set), then the classifier is trained m times. The index m refers to how many combinations of patterns there are. Meaning, the total number of patterns n is divided by a chosen m, and error between sets of **used** and **unused** patterns is taken for each m. The mean of the m errors is taken after each epoch. Thus, for a given D, and a set network topology (a set number of input, hidden, and output units), there are n patterns present to train the data. An m is chosen to divide the patterns evenly, n/m, such that there are m-1 training pattern sets and one validation set for each training index m. For each epoch, the system is trained m times, and the error is recorded at each epoch as the mean of the m errors. Training halts when this error is minimized. This minimization essentially means that no other pattern set produces better classification.

Recall that the number of units to use in the hidden layer is still ambiguous, but there are heuristics for choosing this number. Indirectly, the minimum number of hidden units to use for any three layer implementation is proven by Kolmogorov [18] as $2n + 1$; where n in this case refers to number of input units. However, using a low number of hidden units proves to create high classification error because of the lack of network parameters. Too many hidden units proves to overtrain the data to the particular training set. As a general rule, the number of hidden units should be related to the number of total weights by $s/10$; where s refers to the number of training points. Thus, if the total number of input units are known and the network is fully connected, the total number of hidden units can be derived using (2.6).

$$\begin{aligned} W_{total} &= \frac{s}{10}, && (2.9) \\ n_H &= \frac{W_{total}}{i}, && (2.10) \\ & && (2.11) \end{aligned}$$

where W_{total} is the total number of weights in the network, n_H is the total number of hidden units, and i indexes the input units.

The target set is very important to how the Neural Network learns. The target set consists of what the output is supposed to be for a given input. As a rule, the target category should be defined using a +1, and non-target categories should be defined as -1. Thus, in a *5*-category decision, if the first category is the target, the output should be defined as $t_o = (+1, -1, -1, -1, -1)'$.

2.4 Proposal

There are limitations of the studies highlighted in section 2.2. The study in [2] is accurate for its application, but the workspace is too small for applications involving assistive robotics. To build upon this study, this proposal aims to expand the workspace, and use state-of-the-art software and more modular proprietary hardware to develop a similar non-contact system over a more usable volume.

Several of the studies mentioned above are conducted in explicitly stereoscopic settings [7], [9], [10], [16]. While these controlled settings are useful for expanding methodologies of 3D gaze tracking, [11] reports that the error under stereoscopic settings may not be an accurate reflection of what the error would be for similar experiments performed in physical settings.

Understanding that there is also error associated with the vector-intersection model for 3D gaze estimation, it is also of interest here to attempt to bound such error. Using the vector-intersection technique in tandem with the application of a neural network may prove to remedy some problems associated with the vector-intersection model by producing two 3D PoG estimates. Essig et al. [9] used a similar technique when tracking vergence movements in autostereograms.

Therefore, the research completes four tasks. The first is to collect 2D non-contact gaze data from several participants over a usable workspace. Next, the study will implement the vector-intersection model producing a calculated estimate of 3D PoG. Third, the study will use half of the participant pool to train a neural network to be tested on the other half. Using the combined model to produce a bounded vector-of-interest (BVecOI), a VOI can be ultimately estimated for use by the assitive robot in the workspace.

CHAPTER 3: METHODOLOGY

3.1 Identification of Hardware and Software

An eye tracker is used to collect gaze data for the experiment. The Tobii TX300 is a state-of-the-art eye tracker that measures several eye movement characteristics at very high precision on a 2D platform [5]. It operates at 300 Hz, meaning that one sample of gaze data is taken approximately every 3 ms. The eye tracker has an option for use without the screen attached known as the "Scene-camera Setup." This setup is used when conducting the experiment.

Software to interface with the TX300 in the experiment and to construct the estimation algorithm will be needed. Aside from the proprietary Tobii Studio software required to collect data from the TX300, all filtering and analysis is performed in the high-level language programming environment Matlab R2013a.

3.2 Experimental Setup and Description

The experiment setup is as follows. The eye-tracking device is set facing the user at approximately 60 cm away. In front of the eye-tracking device, several planes are presented to the user. The planes or windows increase proportionally in size and are placed at increasing distances from the user, creating a pyramidal frustum of view. The experiment's layout is given in Figure 3.1. Window 1 is the *initial plane* or calibration plane. Table 3.1 gives the number, dimension, and depth for each window.

Table 3.1: Window Dimensions

Window Number	Dimension (L x W) (cm)	Number of Dots	Depth (cm)
1	40.6 x 25.6	9	0
2	54.1 x 34.1	25	20
3	67.7 x 42.6	25	40
4	81.2 x 51.2	25	60
5	120.6 x 76.8	25	120

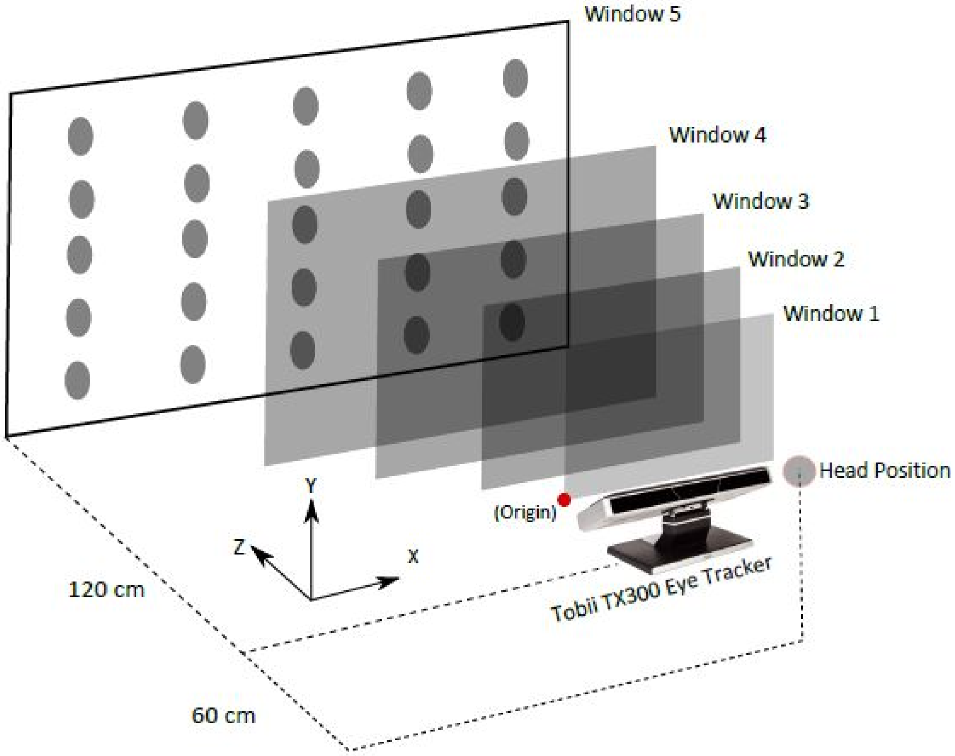

Figure 3.1: Experimental Setup

The eye-tracking device uses a monitor or 2D screen for calibration. The device is meant for tracking eye movements with respect to one plane of depth. That is, all data collected on eye movements is used to approximate the PoG of a user on the 2D screen, rather than in 3D space. Similar to the Dikablis system [11], each user calibrates by the eye-tracker to a 2D plane first, before any data is taken. Window 1 is used as the calibration window. Notice in Figure 3.1 that the universal coordinate system origin is set at the bottom left corner of the calibration plane (Window 1). This is congruent to the TX300 system's origin.

3.3 Experimental Novelty

Taking prior research into consideration, this study is novel in two ways. First, the workspace volume considered is a larger, more usable workspace than previous studies. Current literature shows that the most variance in stimuli depth in stereoscopic experiments is 40 cm [16] and in physical experiments 30 cm [11]. This study investigates a variance in depth of stimuli over a distance of 120 cm, at five distances: 60, 80, 100, 120, and 180 cm from the user. Also, the volume considered in this research is 5.351×10^5 cm^3. For comparison, [2] investigated a volume size of 17250 cm^3.

Secondly, the error associated with the vector-intersection model for 3D PoG estimation is predicted to be large given a workspace of the volume presented. There is a need to bound such error. To this end, a neural network is applied to the system. As in [9], a neural network is used to achieve a second model, with the data collected is from stimuli in a real setting, rather than an autostereogram setting. This difference, while seemingly minor, could produce significant differences in accuracy when applying the neural network.

CHAPTER 4: EXPERIMENTAL PROCEDURE

4.1 Gaze Data Collection

Eight participants volunteered for the experiment. A chin rest was used to insure that head movements were minimized and that both right and left eye position would be kept consistent. Each participant was instructed first to undergo a 2D calibration step, as required by the eye tracking software. The calibration was performed on the nine Dots of Window 1.

Next, the participant would be asked to fixate upon each Dot in order from Dot 1 to Dot 9 on Window 1, and from Dot 1 to Dot 25 on Windows 2 through 5. This process is shown in Figure 4.1. The participant was allowed to rest after each window session was completed. For each Dot, there was a recorded time period ranging from 1-3 seconds. Recalling that the eye tracker records at 300 Hz, this translates to 300 - 900 samples per Dot.

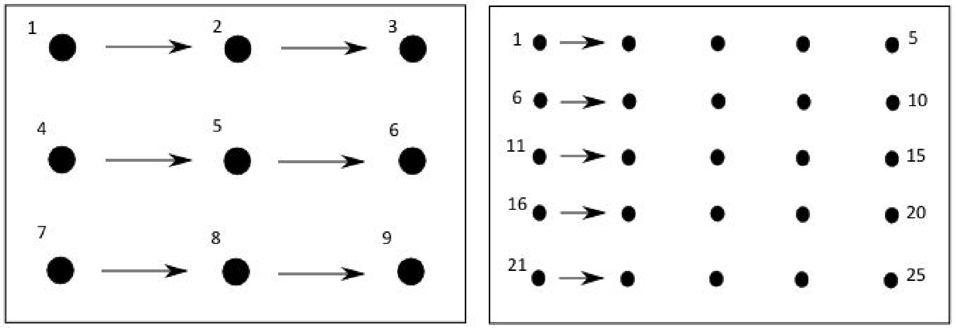

Figure 4.1: Fixation Direction: Window 1 (Left), Window 2-5 (Right), Not to scale.

All eight participants completed the experiment. If errors occurred while recording data, the participant would be requested to repeat that portion of the experiment. Participant data was exported to Microsoft Excel spreadsheets using Tobii Studio software. These spreadsheets were then imported into Matlab for further analysis. Figure 4.2 shows the actual data collection fixture.

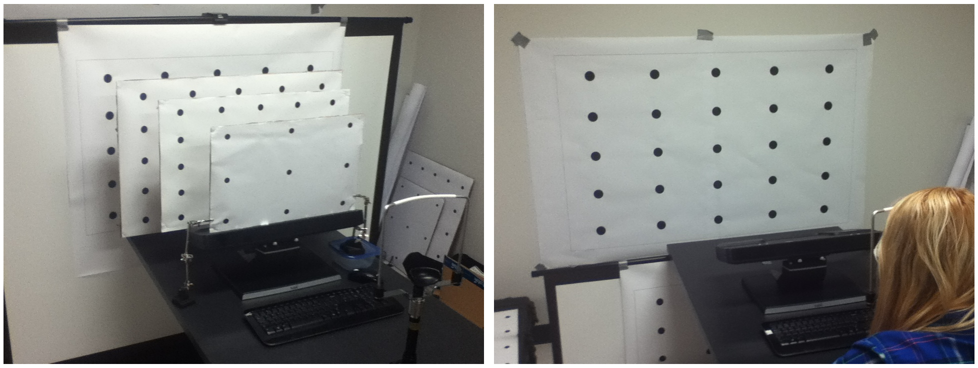

Figure 4.2: Data Collection Fixture (Left), Window 5 Data Collection (Right)

4.2 Data Processing

Participant gaze data was imported into a Matlab data *structure* so that data could be categorized by Participant, by Window, and by Dot number. The data imported for each Dot was a collection of samples. Each sample consisted of coordinates representing where the left and right eyes' line of sight intersected the calibration screen. Therefore, each four dimensional sample consisted of (X_{Left}, Y_{Left}) and (X_{Right}, Y_{Right}). These two points are where the eye tracker estimates each eye is focused on the initial Window, regardless of which Window a Participant is focused upon.

4.2.1 Filtering and Smoothing

Samples were removed if there were blank entries for the left eye, right eye, or both. This is justified given that the nature of this research is to study vergence, which requires data from both eyes. The samples were filtered and smoothed by dimension. Data more than two standard deviations from the mean (per dimension) were removed. A moving average was applied to each dimension. The Window size was 15 samples. This smoothing step is shown in Figure 4.3. These two steps would ensure that the data used would exclude sporadic variations in eye movement. Figure 4.2 shows how the data is filtered. The green and magenta squares represent prefiltered data of (X_{Left}, Y_{Left}) and (X_{Right}, Y_{Right}), respectively. Red and black circle markers represent data left over after removing outliers, while blue and cyan circle markers represent data remaining after applying the moving average filter.

The example data presented in Figure 4.4 shows 120 samples taken from Dot 3, Window 1, Participant 1.

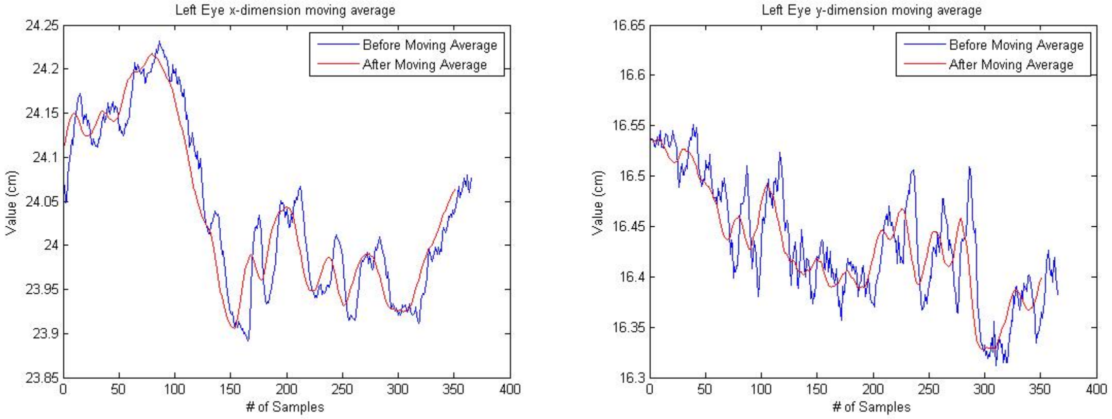

Figure 4.3: Smoothing in x,y - dimension (Left eye)

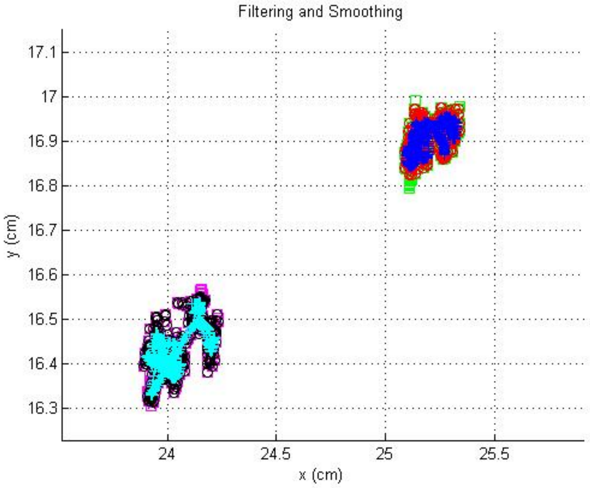

Figure 4.4: Filtering and Smoothing

4.2.2 The Vector of Interest

A fundamental component of estimating where an individual is looking in 3D space is determining which direction the eyes are pointing. This *vector of interest* (VecOI), gives a gaze path that can be used for a number of applications. However, it is the distance along such a vector that is the interest of this research. The models that are applied to gaze data in this study give clues to how far one is looking in 3D space. By knowing the VecOI and the estimated depth of focus (estimations of models), a proper PoG can be determined for each sample generated by the eye tracker. Collecting several of these samples over time on one particular stimulus of interest produces a cluster of data points. Finding the centroids of such clusters can be used to determine where an individual is focused. Since the gaze data used in this research can be used to find the VecOI, the models presented below will refer to the VecOI for a proper gaze path for each Participant for every Dot stimulus. The models are used to estimate how far on this vector the PoG lies or, on which section of this vector the bounded vector of interest (BVecOI) lies . Whichever the case, the VecOI creates an accurate gaze path as shown in Figure 4.5.

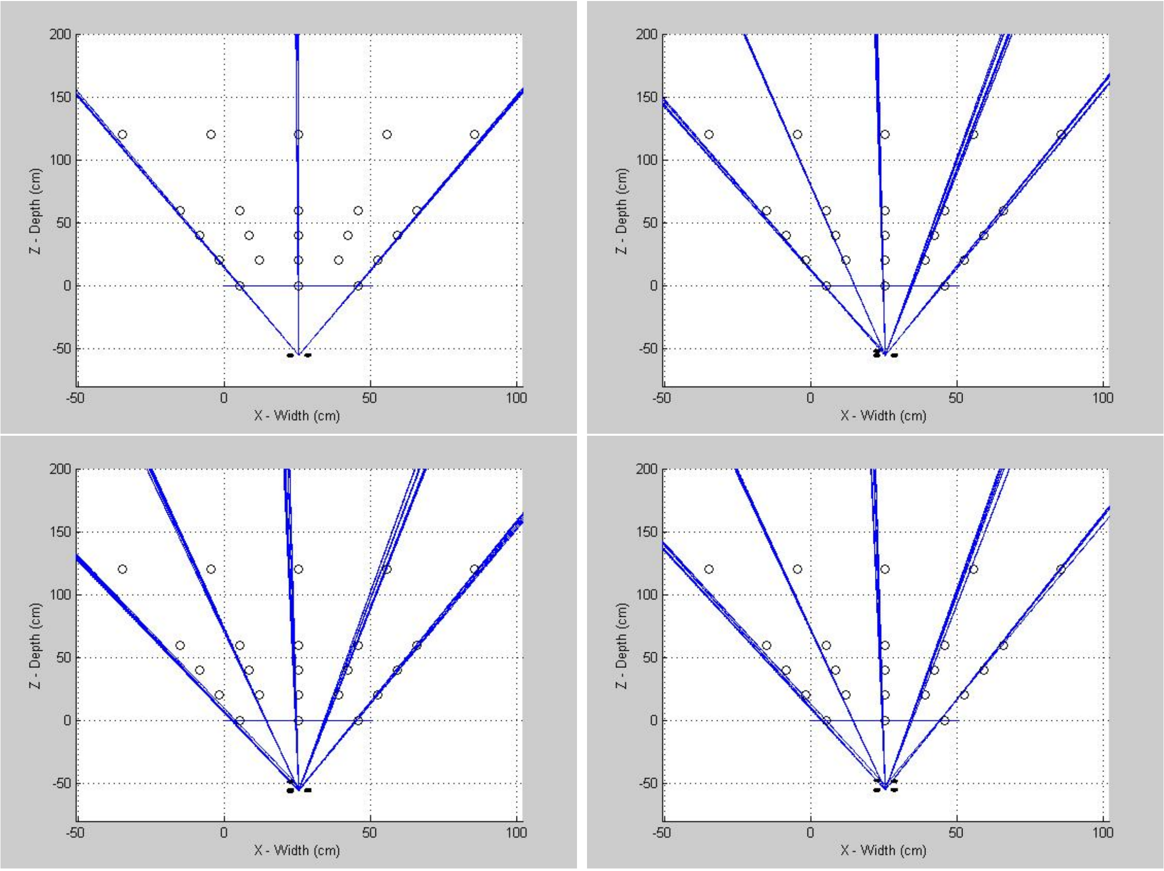

Figure 4.5: Overhead of Participant 1 VecOIs

Figure 4.5 displays an example of the VecOI of Participant 1 for the first four Windows. There are 120 VecOI's per Dot, creating the band of blue lines seen at each Dot. As the figure shows, the VecOI's are accurate, passing through or very close to their respective

Dots. All Participants were observed to follow this same pattern.

4.2.3 Vector Insection Approach

To estimate where in 3D space each participant was focused, the vector intersection (VI) model was applied. This model takes the (x_e, y_e, z_e) position of each eye and draws a vector to the respective gaze point on the calibration plane. Both vectors are extended through the calibration plane. Where the two vectors intersect produces a specific (x_s, y_s, z_s) estimation for each sample. Since there are several samples for each Dot per participant, a cluster will form for each Dot. The centroid of this cluster is computed and is assigned as the final estimated point for each particular Dot. Therefore, for each participant there are 109 centroids computed. Figure 4.7 visualizes this process for one particular Dot. The top image of Figure 4.7 shows the eye positions, the right and left gaze vectors, and the cluster of red estimation points. The bottom image shows a zoom on the cluster, showing the position of the real coordinate of the Dot, the red estimation cluster of points and the position of the centroid of the cluster. Error for the VI model is calculated by finding the Euclidean distance from the centroid to the real Dot coordinate.

To generate the vector from each eye, a set of parametric equations was used to generate x-coordinates, y-coordinates, and z-coordinates to define the vectors. Using the (x_e, y_e, z_e) for each eye and the (x_{iw}, y_{iw}, z_{iw}) on the initial window for each eye ($iw =$ initial window, two points can now be used to define an equation that intersects both points. This process is executed for both gaze vectors. The maximum of parameter t is specified such that there is enough length in each gaze vector for both to intersect. A value of t is found for each sample for which both gaze vectors intersect in the x and z dimensions. The average y-coordinate is found between the two vectors at that t value and the resultant (x_s, y_s, z_s) coordinate is used as the final estimation point for a particular sample. This averaging method is shown in Figure 4.6. The pseudocode below gives a summation of the process in Matlab. To find the centroid of each cluster, the *kmeans* function in the Matlab Statistics toolbox was utilized.

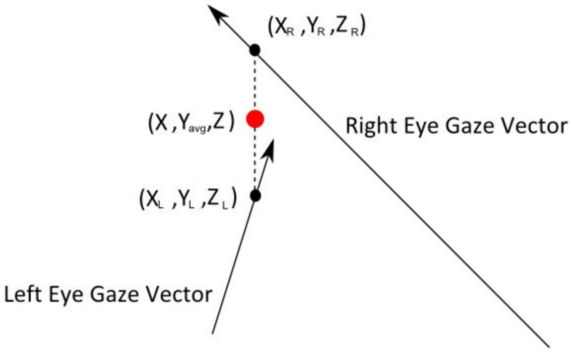

Figure 4.6: Averaging method to find y_s

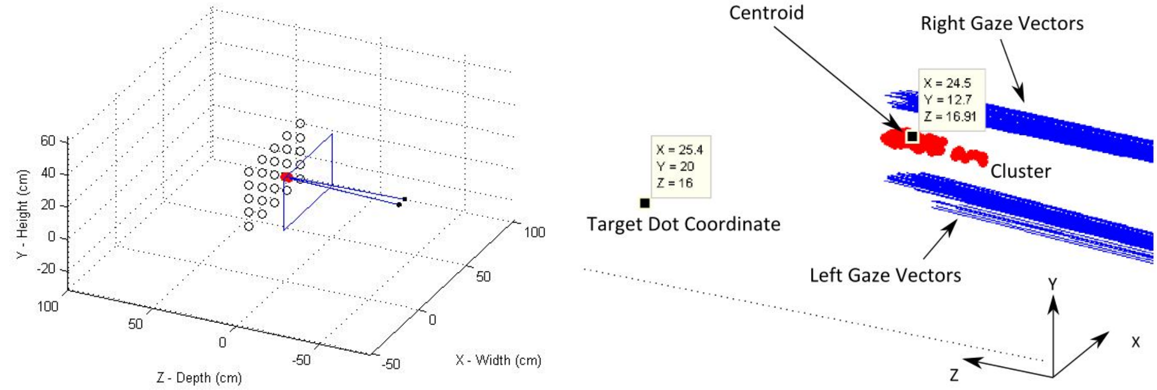

Figure 4.7: Vector Intersection Model Example

Algorithm 4.2.1: VECTOR INTERSECTION MODEL($Matlab$)

```
comment: Define Left and Right Eye Positions in 3D space
$LeftEyePosition = [X_L, Y_L, Z_L];$
$RightEyePosition = [X_R, Y_R, Z_R];$
comment: Generate the Left and Right Vectors originating from Eyes
for $i = 1 : length(GazeInputSampleSize)$
    $LeftEyeGazePoint = [LeftGazePointX, LeftGazePointY];$
    $RightEyeGazePoint = [RightGazePointX, RightGazePointY];$
    comment: Set up Parametric Equation
    $VectorLeft = LeftEyePosition - LeftEyeGazePoint;$
    $VectorRight = RightEyePosition - RightEyeGazePoint;$
    comment: Defining parameter 't'
    $t = -5 : .01 : 0$ Be sure that $t$ has a large enough range and high resolution.
    comment: Define LoS vectors for each dimension per eye
    $LoS_{Lx} = LeftEyePosition(1) + VectorLeft(1) * t;$
    $LoS_{Ly} = LeftEyePosition(2) + VectorLeft(2) * t;$
    $LoS_{Lz} = LeftEyePosition(3) + VectorLeft(3) * t;$
    $LoS_{Rx} = RightEyePosition(1) + VectorRight(1) * t;$
    $LoS_{Ry} = RightEyePosition(2) + VectorRight(2) * t;$
    $LoS_{Rz} = RightEyePosition(3) + VectorRight(3) * t;$
    $find_t = find($t when $LoS_{Lx} = LoS_{Rx}$ and $LoS_{Lz} = LoS_{Rz})$
    comment: Search $LoS_{Ly}$ and $LoS_{Ry}$ for y @ t and average them
    $Final_y = mean(LoS_{Ry}(find_t), LoS_{Ly}(find_t));$
    comment: Use either $LoS_{Rx}$ or $LoS_{Lx}$ to find X,Z @ t
    $Final_x = LoS_{Lx}(find_t);$
    $Final_z = LoS_{Lz}(find_t);$
    $FinalEstimation = [Final_x(x_s), Final_y(y_s), Final_z(z_s)];$
```

4.2.4 Neural Network Approach

As stated earlier, to somehow characterize the data to a particular distance, a neural network is implemented. The four inputs to the neural network were the (X_{Left}, Y_{Left}) and (X_{Right}, Y_{Right}) points. The system was trained with target outputs in discrete form. Figure 4.8 shows the neural network overall structure and how each target output was defined. Equation (4.1) displays which window each target output corresponds to. Note that in 4.1, each sample will take on one *row* as the target output, not the entire matrix presented.

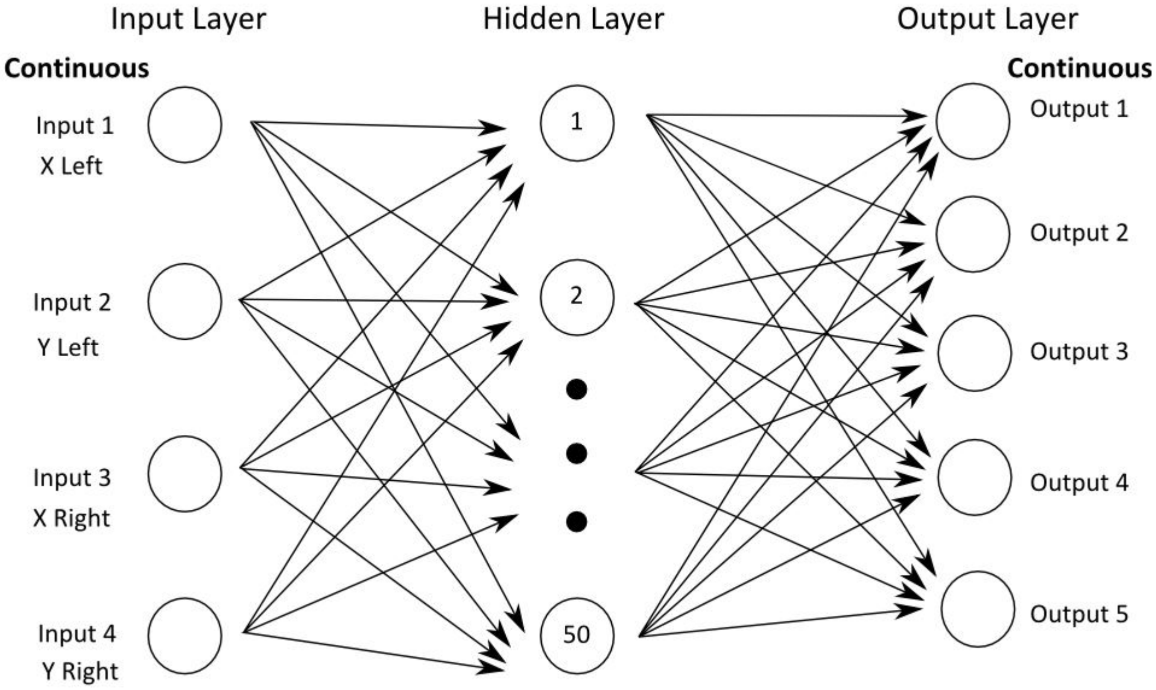

Figure 4.8: Applied Neural Network Topology

$$Target\ Output = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} = \begin{vmatrix} Window\ 1 & \rightarrow & 0cm \\ Window\ 2 & \rightarrow & 20cm \\ Window\ 3 & \rightarrow & 40cm \\ Window\ 4 & \rightarrow & 60cm \\ Window\ 5 & \rightarrow & 120cm \end{vmatrix} \tag{4.1}$$

The training set was compiled using four participants. To exhaust all possible combinations of training sets, an $n\ choose\ k$, $\begin{pmatrix} n \\ k \end{pmatrix} = \frac{n!}{k!(n-k)!}$ matrix is generated. For this experiment, $n = 8$ and $k = 4$. This results in seventy different training and test sets.

Once the neural network was trained to the training dataset, sample sets from the test participant set are used to find the accuracy of the network. The outputs of the network from the test set were not discrete values, rather each of the five values varied between -1 to 1; $-1 \leq O_n \leq 1$. The maximum of these outputs were found for each test sample input, so that each sample is assigned to a particular discrete depth at the window distances. For example, the network could be trained using sample sets from participants 1-4. The trained network is then tested for accuracy on participants 5-8. The network's output for a particular sample is $N_s = [O_1, O_2, O_3, O_4, O_5]$. If the maximum output at a particular sample occurs at O_2, then the output array for that particular sample would be $N_s = [0, 1, 0, 0, 0]$. Since it is known that each output O_n corresponds to a window, then each sample can be assigned a distance, or a z-coordinate.

To summarize in detail, the Neural Network Model is implemented in a few major steps. In the training step, data from four participants is used to to train a neural net. Each sample used to train the network is assigned a target output group of five discrete outputs or one of the five rows in 4.1. The training data used is comprised of 120 samples for each of the 109 Dots per participant. Therefore, a total of (4 × 109 × 120) 52320 samples were used in training.

Next, the trained net is applied to the other four participants' data, again comprised of 52320 samples, divided into each of the 109 Dots. The output group for each sample consists of five numbers, varying from -1 to 1. The maximum output from each output group is found and set to one, while all other outputs are set to zero. This produces a discrete output for each sample, effectively assigning it to a window and in turn, a z value. E.g., $N_s = [0, 1, 0, 0, 0] \rightarrow$ Window 2 → 20cm.

For each sample, an average vector can be found, indicating a vector of interest (VecOI). Since each sample is assigned a z value, a corresponding x,y value can be found on the respective VecOI. This produces a (x_s, y_s, z_s) estimate for each sample in the test data set. Recall that the samples are grouped by Dots, therefore there are 109 clusters per Dot, comprised of the 120 samples. A centroid for each of the clusters is found. The Euclidean error is then calculated using the known real coordinate of the Dots and the estimated centroids.

The Neural Network toolbox in Matlab allows for a simple and effective way to train networks and save each network used. The following steps give an overview of how the Neural Network Model was implemented in the Matlab environment to output data used for accuracy analysis.

1. Compile the training data set into one matrix. Interweave the data so that the network does not train to one Dot first.

2. Create a target matrix. That is, assign each sample in the training data set a vector of five elements, e.g. [1 -1 -1 -1 -1] (Window 1).

3. Set the seed value by using the *setdemorandstream()* function.

4. Use the *[net, tr] = train(net,B,C)* function to output a network (net) and training data (tr). The argument *net* is the network (which changes as it is trained), A is the training data set, and B is the target data set.

5. Save the network. For this studies' purposes, there were 70 saved networks, corresponding to the number of participant combinations.

6. Compile the testing data set in one matrix. The samples can be in any order (unless it is desired to have them categorized for further analysis).

7. Use the saved network (net) to classify the test data set.

8. Find the highest output value for each sample and assign the z value.

9. Compile the VecOIs for each sample. Find the corresponding x,y coordinate for each estimated Z coordinate.

10. Group the samples into Dots (if this has not already been accomplished in step 6). Find the centroid of each Dot cluster.

11. Calculate the Euclidean distance from centroid estimation coordinate to real Dot coordinate for each cluster.

Once each sample is assigned a discrete z value, this value is found on the VecOI, which is generated by a parametric equation. Since the VecOI is known for each sample, and now the z coordinate is known on that vector, the corresponding values for x and y are found on that particular vector. Figure 4.9 illustrates this process and again, uses data from Participant 1, Window 2, Dot 3.

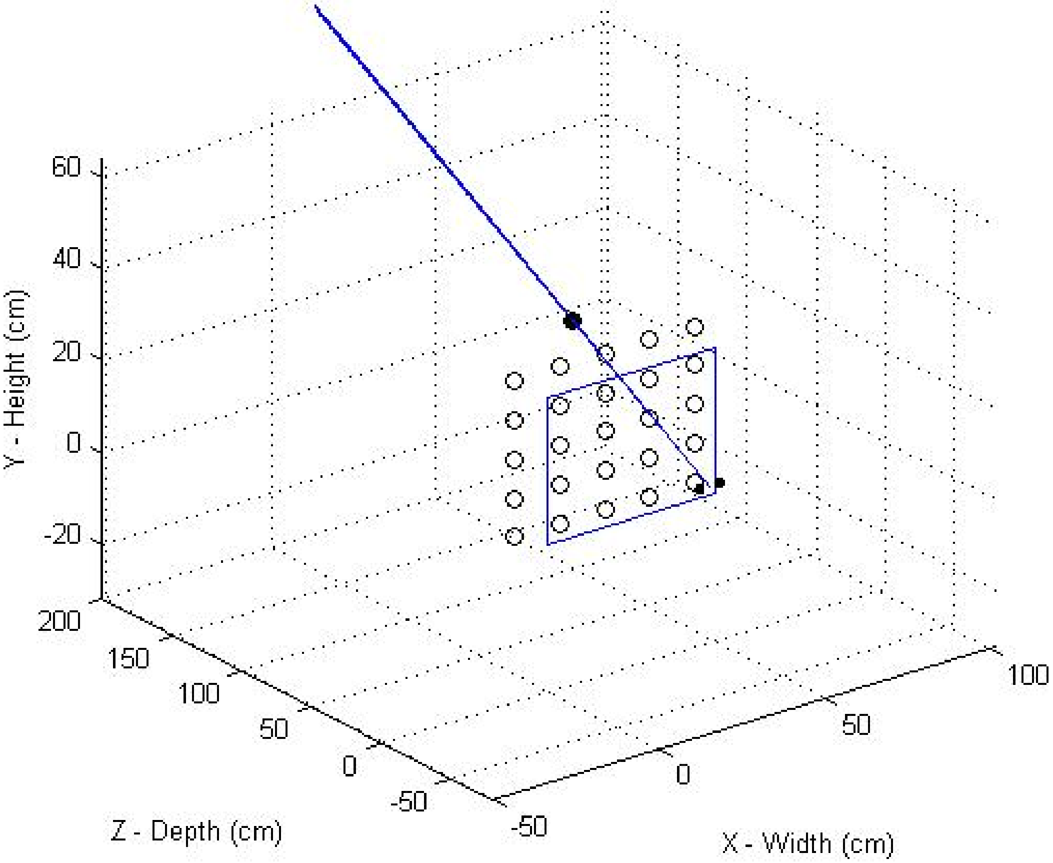

Figure 4.9: Neural Network Estimation using Average Gaze Vector

Each Dot will have a cluster of points and, unlike the vector intersection model, this cluster is restricted to discrete z values. The centroid of each cluster is found for each Dot, representing the estimated PoG for each of the 109 Dots per participant. The centroid in Figure 4.9 is shown by the black Dot. There is indeed one average gaze vector (blue lines) corresponding to each sample, thereby producing 120 average gaze vectors. Notice that this particular centroid estimation is incorrect by one window, estimating that the centroid is located on Window 3 instead of Window 2.

4.2.5 Combined Model Approach

To generate a bounded Vector of Interest (BVecOI), centroid estimates from the VI and Neural Network models are used. The centroids are connected by a vector and a radius is extended from each point on that vector, creating a search volume. Thus, for each participant, a VOI is generated per Dot stimulus, yielding 109 VOIs to be compared to the real Dot locations. Accuracy is found by determining if the estimated VOI encompasses the coordinates of its respective Dot. Figure 4.10 shows the BVecOI for an example group of Dots. The BVecOI is created with a parametric equation, as noted above. The parameter t is bounded from -1 to 0 to ensure that the vector would not extend beyond the two centroid estimate coordinates.

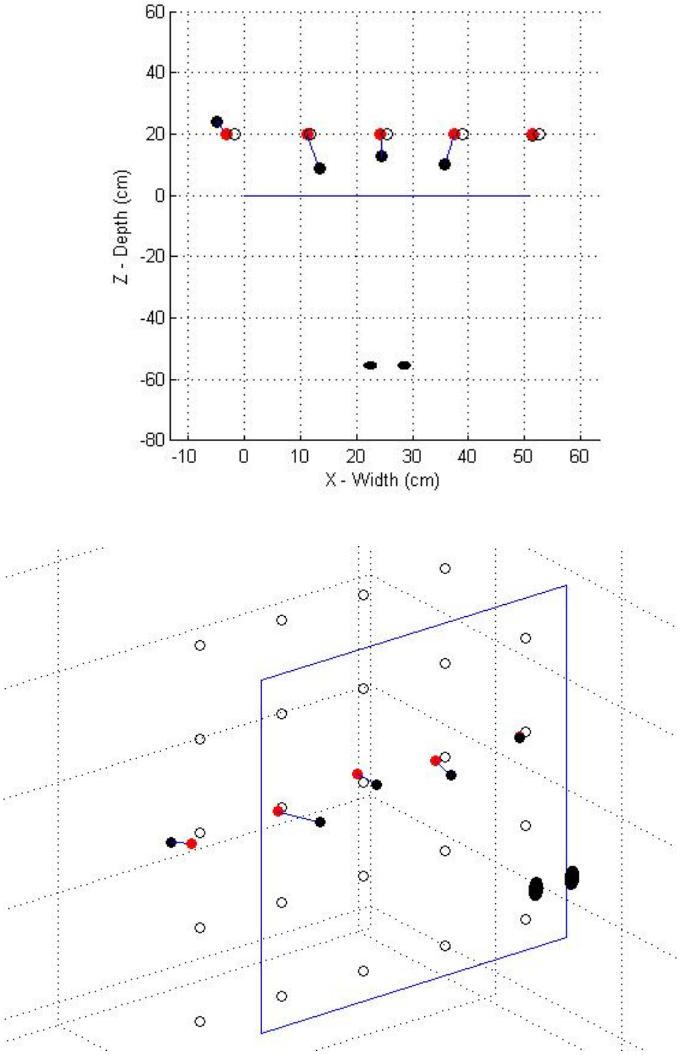

Figure 4.10: Combined Model Overhead View (top), Zoomed (bottom)

Note the red filled markers are the Neural Network Model centroids and the black filled markers are VI Model centroids. As in earlier figures, the position of the eyes and the initial window outline are shown. The non-filled circles represent the real Dot coordinates. The BVecOI is shown by the blue lines connecting their respective centroids.

4.2.6 Search Region Approach

The VOI in the Combined Model is determined by two centroids that may or may not be accurate under high resolution. Understanding that vergence, and in general the human eye, is prone to have variations depending on focus, human mechanics, or simply fatigue, it is warranted that larger Search Regions (SR) should be defined for an assitive robot to use to shorten its searching path. Therefore, the frustum of view will be divided into three volumes, which will be called regions. The regions are displayed below in Equation(4.2) and Figure 4.11.

$$Bounds = \begin{Bmatrix} -10cm \leq Bounds \leq & 30cm, & Region\ 1 \\ 30cm \leq Bounds \leq & 70cm, & Region\ 2 \\ 70cm \leq Bounds \leq & 130cm, & Region\ 3 \end{Bmatrix} \tag{4.2}$$

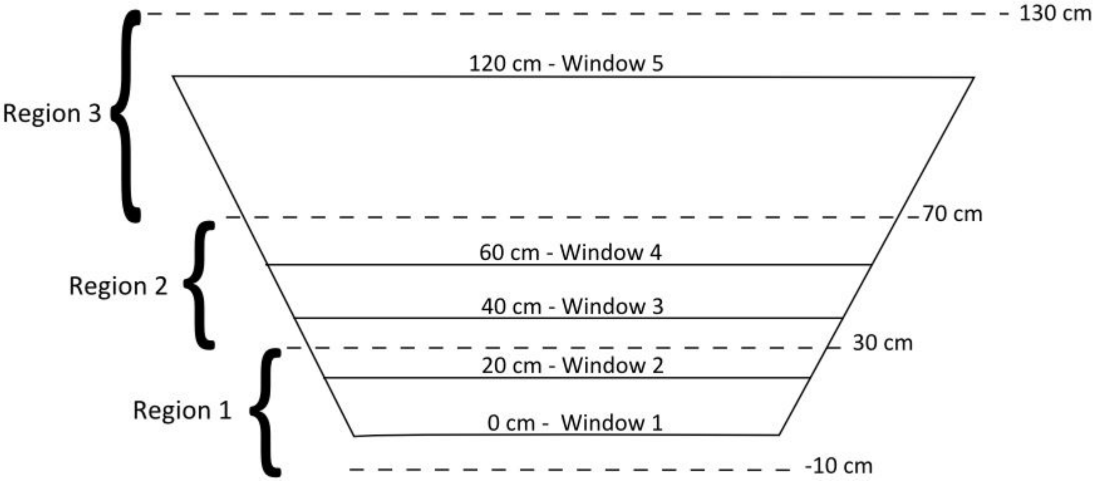

Figure 4.11: Search Regions

Both the VI Model and the Neural Network Model have centroid estimates for every Dot per Participant. These centroids fall within one of the above regions. Using these regions the VecOI can be bounded, but unlike the Combined Model, the bounds are fixed, and the models are treated separately. Based on this paradigm, the Search Region Method is now proposed.

The Searching Region Method is found by establishing the searching regions, generating an average VecOI, which is an average vector of all VecOIs for a given Dot, and bounding the average VecOI by the searching region decided by the z value of each centroid estimate (essentially defining the SBVecOI). Accuracy is found by calculating the Euclidean error between each point coordinate on the SBVecOI and the respective Dot coordinate. Since Windows 1 and 2 fall within Region 1, Windows 3 and 4 within Region 2, and Window 5 within Region 3, different threshold tolerances are applied when calculating accuracy. For example, if the estimation is for Dot 1 on Window 2, then a lower tolerance is used than if the estimation is for Dot 1 on Window 3. The tolerances used when calculating accuracy are the minimum of the two window tolerances in a particular Region, except for Region 3 which includes one window.

CHAPTER 5: RESULTS AND DISCUSSION

5.1 Accuracy of the Vector Intersection Model

As stated above, accuracy for the VI Model was determined by finding the Euclidean distance from the estimated centroid to the respective Dot under scrutiny. For each window a strict maximum tolerance is defined, given the dimensions between each Dot. Table 5.1 gives the distances between Dots in the x and z direction for each Window. Notice that for Window 1 there are only 9 Dots. The space between Dots shown for this Window will be given as though there were 25 Dots, to retain uniform calculation of accuracy. Maximum tolerance is the minimum of the two dimensional tolerances. Table 5.1 notes these tolerances.

Table 5.1: Defining Accuracy Tolerances for VI Model

Window Number	Distance between Dots (x-dim,y-dim) in cm	Tolerance (x-dim,y-dim) in cm	Maximum Tolerance Min(x-dim,y-dim)
1	20.3, 12.8	5.08, 3.2	3.2
2	13.5, 8.5	6.7, 4.2	4.2
3	16.9, 10.6	8.4, 5.3	5.3
4	20.3, 12.8	10.1, 6.4	6.4
5	30.1,19.2	15.0, 9.6	9.6

Thus, when accuracy is computed, the maximum tolerance is used as a radius, forming a sphere from each estimated centroid. The model is deemed as accurate if the real coordinate Dot lies within this tolerance sphere. Table 5.2 shows the accuracy of the VI Model for each participant.

The accuracy of the VI model is poor, given that most participant accuracy did not reach twenty-five percent. To better understand the reason behind the results, error in the x, y, and z dimension is analyzed.

The VecOI that is not the main issue, but where on that vector one is focused. In turn, if an accurate z value is estimated, then the corresponding x,y values will be accurate as well. However, the fact still remains that the estimated centroids for each participant

Table 5.2: Accuracy of VI Model

Participant	Total Accuracy (%)
1	18.3
2	34.8
3	9.1
4	3.6
5	9.1
6	23.8
7	16.5
8	20.1

using the VI model are highly inaccurate. These inaccuracies may be explained by biological factors, meaning that, when an individual fixates on an object in 3D space, vergence supplies ocular focus *in tandem* with lensing focus. Also, image processing in the brain plays a large role in focus as well.

Inaccuracies could occur also due to what was focused upon, meaning that the black round Dots used as stimuli provided little interesting information to the user. Being near-2 dimensional and one solid color, it may be that focus was not easily maintained on such a stimulus. Given these drawbacks of the VI model, the motivation of using a neural network is warranted.

5.2 Accuracy of the Neural Network Model

As stated in section 4.2.4 the neural network was trained using seventy different combinations of training data and test data. Results from all combinations are not shown. However, the combination producing the most accurate results, the combination producing the median results, and the combination producing the least accurate results, are given in Table.

The training set that produced the best, median, and worst accuracy on their respective test set are analyzed in the table below. Recall that for neural networks, minimal mean squared error (MSE) for a training set does not necessarily mean better classification for an independent test set. Table 5.3, 5.4, and 5.5 show the combination number, the participants used for training data, the MSE for each trained combination, the participants used for test data, and the accuracy of the network on the participants. The accuracy shown in these tables was calculated in the same manner as accuracy for the VI model. A radius is drawn from the each estimated centroid, creating a sphere. If the corresponding Dot coordinate fell within that sphere, then the estimate was noted as completely accurate. Results are shown for three different cases in which the number of hidden units is 25, 50, and 100. In the final calculation of accuracy, the 50 unit case was used.

Table 5.3: Neural Network Model Results: 25 Hidden Units

NN Index	Training Participant Order	MSE	Test Participants	Accuracy per Test Participant (%)
41	2,3,5,7	0.2692	1,4,6,8	68.8, 46.8, 61.4, 75.2
8	1,2,4,7	0.2625	3,5,6,8	59.6, 45.8, 64.2, 69.7
14	1,2,6,8	0.1701	3,4,5,7	62.3, 39.4, 39.4, 55.0

Table 5.4: Neural Network Model Results: 50 Hidden Units

NN Index	Training Participant Order	MSE	Test Participants	Accuracy per Test Participant (%)
6	1,2,4,5	0.3169	3,6,7,8	65.1, 60.5,59.6, 70.6
54	2,5,7,8	0.4765	1,3,4,6	44.9, 49.5, 39.4, 44.9
29	1,4,6,7	0.7385	2,3,5,8	10.0, 9.1, 10.0, 9.2

Table 5.5: Neural Network Model Results: 100 Hidden Units

NN Index	Training Participant Order	MSE	Test Participants	Accuracy per Test Participant (%)
64	3,5,7,8	0.1294	1,2,4,6	63.3, 66.0, 48.6, 60.0
63	3,5,6,8	0.1262	1,2,4,7	71.5, 65.1, 48.6, 53.2
14	1,2,6,8	0.0885	3,4,5,7	60.5, 37.6, 42.2, 57.8

The percentage accuracy is much higher than that of the VI model, given that the neural network was trained to the Windows. Also, recall that each output of the neural network using the test data set was set to particular discrete value, effectively “snapping” to one of the Windows. It is not surprising that the overall accuracy is reflective of this fact.

Taking into consideration the poor accuracy of both models to accurately predict where one is focused in 3D space, it is advantageous to investigate a *Combined Model* approach to 3D gaze estimation. Using the estimates from both the VI Model and the Neural Network Model, a larger volume of interest can be generated.

5.3 Combined Model Accuracy

The Combined Model approach uses the previous models mentioned to produce a BVecOI based on estimated centroids. The centroids are connected by a vector and a radius is extended from each point on the vector. If the real Dot coordinate lies within any of the spheres generated on the vector, or more generally, inside the VOI, then the model is noted as completely accurate for that Dot. Table 5.6, 5.7, and 5.8 below shows the accuracy of this model per participant. To show the variation of accuracy between training combinations, the best performing network, the median performing network, and the worst performing network are included.

Table 5.6: Accuracy of Combined Model: 25 Hidden Units

NN Index	Test Participants	Total Accuracy per Test Participant (%)
41	1,4,6,8	80.8, 55.5, 75.7, 85.6
8	3,5,6,8	68.9, 57.6, 80.1, 85.6
14	3,4,5,7	69.6, 45.1, 48.8, 61.9

Table 5.7: Accuracy of Combined Model: 50 Hidden Units

NN Index	Test Participants	Total Accuracy per Test Participant (%)
6	3,6,7,8	73.7, 80.3, 69.9, 88.8
54	1,3,4,6	78.4, 74.4, 55.5, 66.1
29	2,3,5,8	59.2, 34.4, 36.0, 55.2

Table 5.8: Accuracy of Combined Model: 100 Hidden Units

NN Index	Test Participants	Total Accuracy per Test Participant (%)
64	1,2,4,6	80.0, 86.5, 57.9, 74.7
63	1,2,4,7	87.2, 90.4, 57.1, 65.3
14	3,4,5,7	68.1, 44.3, 53.6, 62.1

Although there is greater accuracy when using the Combined Model, there are metrics that must be investigated. Recall that the Combined model uses a BVecOI to generate the

VOI. The length of this vector varies. If the objective of this research is to somehow limit the VOI to a practical region, it is important to note the mean μ, standard deviation σ, maximum, and minimum lengths of the collection of BVecOI's for a given window. Table 5.9, 5.10, and 5.11 display these metrics for the 50 hidden unit case. Note that the best, median, and worst Neural Network indicies are shown in the tables below.

Table 5.9: BVecOI Metrics by Window - Index 6

Participant Number	Window	μ (cm)	σ (cm)	Max, Min (cm)
3	1	4.8	2.7	7.1, 0.1
	2	14.0	5.9	26.2, 3.9
	3	19.4	17.0	95.1, 0.7
	4	26.3	16.3	66.9, 3.0
	5	109.1	53.8	235.6, 35.4
6	1	7.0	6.3	17.8, 0.2
	2	23.8	11.2	59.6, 7.4
	3	41.5	48.2	236.1, 2.3
	4	51.6	30.3	150.0, 5.4
	5	132.8	48.6	232.7, 60.2
7	1	8.2	7.6	23.7, 0.2
	2	25.8	16.5	86.8, 3.6
	3	40.8	22.0	91.6, 14.0
	4	52.6	39.0	212.9, 5.7
	5	110.1	53.3	231.3, 8.5
8	1	3.5	2.0	6.3, 0.1
	2	12.7	8.1	35.6, 0.8
	3	30.0	21.3	107.1, 1.0
	4	51.1	29.9	150.0, 1.8
	5	136.9	47.1	236.4, 44.9

These tables give clues to how the Combined Model reacts as stimuli increase in distance from the user. As the tables show, an increase in accuracy does not always guarantee a short BVecOI. Conversely, a short BVecOI does not guarantee that the Dot coordinate is intercepted at all. However, it can be stated that the length of the BVecOI is dependent more so on the nature of the VI Model. If the VI Model is generally inaccurate, then the Combined Model will usually generate a BVecOI that is longer, unless the Neural Network Model produces a similarly inaccurate estimate. Thus, the VI Model is called into question.

The justification for this conclusion lies in the nature of the eyes and the validity

Table 5.10: BVecOI Metrics by Window - Index 54

Participant Number	Window	μ (cm)	σ (cm)	Max,Min (cm)
1	1	6.7	11.8	38.1, 1.2
	2	24.5	7.2	38.1, 8.7
	3	17.7	10.7	49.3, 4.4
	4	20.1	25.0	113.6, 0.1
	5	53.7	50.6	189.3, 0.5
3	1	9.3	11.9	40.7, 1.3
	2	24.8	8.0	34.0, 3.9
	3	22.4	9.8	51.4, 0.7
	4	17.8	15.6	65.4, 2.7
	5	40.7	28.2	105.9, 7.3
4	1	6.7	6.4	22.7, 0.1
	2	19.4	14.7	51.0, 0.8
	3	29.7	25.5	134.7, 2.3
	4	21.9	13.1	51.6, 1.5
	5	49.2	45.2	188.7, 0.5
6	1	20.5	19.4	54.1, 0.2
	2	21.7	16.3	90.2, 4.1
	3	21.0	23.3	106.2, 0.9
	4	21.6	18.1	66.7, 0.5
	5	49.6	46.6	189.8, 4.2

of the VI model. In short, the VI Model is a straightforward procedure: take two vectors that originate at each eye and intersect them in space to find the PoG. The error for this technique increases exponentially as the stimuli move further from an individual. The further an object of interest is away from an individual the less and less the eyes diverge, making a small change in vergence relate to a large change in estimated depth. The question arises: why use the VI Model at all?

The VI model judges where one's *eyes* are focused; which is not necessarily the focus of the individual. This does not mean that the VI Model is *inaccurate* in a broad sense, but only for applications that rely on highly resolved estimations. Keeping the VI model gives one strong advantage that the Neural Network Model can not: to make continuous estimations. It is imperative that the VI Model is not discarded so that window biasing does not occur.

Table 5.11: BVecOI Metrics by Window - Index 29

Participant Number	Window	μ (cm)	σ (cm)	Max,Min (cm)
2	1	3.0	1.7	5.6, 0.2
	2	19.9	5.6	31.1, 9.0
	3	29.2	13.2	52.2, 0.3
	4	50.8	20.4	88.0, 9.8
	5	106.3	37.5	217.4, 51.5
3	1	4.8	2.7	7.1, 0.1
	2	14.0	5.9	26.2, 3.9
	3	19.4	17.0	95.1, 0.7
	4	26.3	16.3	66.9, 3.0
	5	109.1	53.8	235.6, 35.4
5	1	4.6	3.5	9.4, 0.1
	2	16.2	18.3	91.7, 0.2
	3	20.4	14.9	68.8, 0.6
	4	40.6	44.7	169.3, 3.5
	5	84.3	52.5	230.8, 26.8
8	1	3.5	2.0	6.3, 0.1
	2	12.7	8.1	35.6, 0.8
	3	30.0	21.3	107.1, 1.0
	4	51.1	29.9	150.0, 1.8
	5	136.9	47.1	236.4, 44.9

5.4 Search Region Results

Unlike the Combined Model, the Search Region Method is advantageous in the fact that it does not have ambiguity in length when generating a BVecOI based on the estimates from the VI Model and the Neural Network Model. The SBVecOI is bounded by 40 or 70 cm in the z dimension. Table 5.12 and 5.13 show the total accuracy per participant when searching regions are applied to each model.

The results shown indicate a much higher accuracy than those of the independent VI Model in Figure 5.12. The disadvantage of this method is that, while some participants (1,2,6,7,8) reach 70 percent accuracy, accuracy of other participants is substantially lower and unusable for practical applications. The result is that decreasing resolution may not lead to meaningful accuracy for some individuals.

It can be seen that for Index 6 in Figure 5.13, the accuracy is considerably better than

Table 5.12: Accuracy of VI Model - Search Region Method

Participant	Total Accuracy(%)
1	71.4
2	83.4
3	59.6
4	42.2
5	50.4
6	73.3
7	69.7
8	77.9

Table 5.13: Accuracy of Neural Network Model - Searching Region Method

NN Index	Test Participants	Total Accuracy(%)
6	3,6,7,8	76.1, 70.6, 68.8, 81.6
54	1,3,4,6	68.8, 71.5, 67.8, 63.3
29	2,3,5,8	35.7, 32.1, 33.0, 34.8

using the independent Neural Network Model. Again, the resolution has been reduced, but the SBVecOI is predicable, unlike the Combined Model. A comparison of models is given in section 5.5 to show the advantages and limitations of each method.

5.5 Comparison of Models

There are advantages and disadvantages of using each of the approaches considered in this chapter; which are shown concisely in Table 5.14.

Table 5.14: Comparison of Models

Model	Advantages	Disadvantages
VI Model	• Generates continuous output data. • No window biasing. • No training needed besides initial calibration.	• Very poor accuracy over highly resolved intervals. • Favors individuals whose vergence varies greatly with depth.
NN Model	• Substantially higher accuracy than VI model. • With the correct training set, can be applied to independent test sets producing acceptable accuracy.	• Relies on a training set. Trained pattern may not be appropriate for independent test group. • Does not give continuous outputs.
Combined Model	• Can produce considerably higher accuracy. • Utilizes a larger volume based on a BVecOI.	• Produces unknown lengths of BVecOIs. • Low predictability on how BVecOI lengths vary with accuracy.
Search Region Method - VI	• Generates a predictable BVecOI (SBVecOI). • Considerably higher accuracy than standalone VI due to division by regions.	• Some individuals may still have unusable accuracy ranges. • Large regions cause resolution to decrease.
Search Region Method - NN	• Generates a predictable BVecOI (SBVecOI). • Comparable accuracy to standalone NN if proper training set is used.	• Patterns learned by training set must applicable to test set.

CHAPTER 6: CONCLUSION AND FUTURE WORK

In this research, the Vector Intersection approach, the Neural Network approach, a Combined approach, and the Search Region approach were investigated for gaze estimation in three dimensional space. The objective of this research were to apply these models to extend 2D data collection techniques to generate a VOI in a practical and usable workspace that can be used for 3D applications, especially to aid in assistive robotic path planning. These objectives were accomplished by constructing an experimental apparatus used to collect gaze data at five different distances spanning from 60 cm to 180 cm in front of each participant. The gaze data was collected over eight participants. A filtering, smoothing, and truncating step was taken for analysis purposes. The VI and Neural Network models were independently applied to each participant and accuracy was calculated accordingly. A Combined Model was presented, using both estimates of the VI and Neural Network Model to produce a VOI governed by a BVecOI. The Combined Model resulted in higer accuracy, but low predictability in terms of BVecOI length. Next, the Search Region method is applied to the VI and Neural Network Models' independently to produce a BVecOI that is more predictable in physical length. The accuracy of the Searching Region method is comparable to that of the Combined Model, making it the preferable approach out of those investigated.

6.1 Conclusions

Results show that the VI model does not produce accurate estimates over the workspace investigated. As stated earlier, the stimuli used may not have any interesting information, but the actual cause of inaccuracy lies with the nonlinear relationship between vergence and distance of stimuli from an individual. Vergence changes are smaller with increase in stimuli depth. This implies that a small change in vergence can represent a large change in estimated depth. This is a well-known problem with the VI model. Although the VI model has this disadvantage, the model does follow a pattern that can be measured. Generally, divergence increases with depth, but the magnitude of this change is not consistent. This phenomenon translates to the vector intersection model, making it highly inaccurate.

The Neural Network model results show that there indeed are general patterns in gaze behavior that can be used to discern the gaze depth of individuals with moderate accuracy. The effectiveness of this kind of model in this research shows that the need for a proper training set is vital in producing moderately accurate results in an independent test set. Results show that the best combination of training and test set renders around 60-70 percent accuracy across participants. This translates to 65-76 correct estimations out of 109, which is not acceptable for every application, but is promising for future work.

The Combined Model utilizes estimations of both models. Justification for using the estimations of both models to produce a larger VOI for each Dot rests in the fact that the VI model produces continuous estimates, and the Neural Network model produces discrete estimates. As stated previously, the VI model measures the movements of eyes alone; no depth calibration is needed for its estimations. The VI model also measures the physical mechanics of the eyes. Thus, the model may indeed accurately measure PoG according to the vergence of the individual, but cannot possibly take into account factors beyond that, such as focusing by the eye's lens or image processing in the brain. Therefore, it is not justifiable to discard the estimation of this model, especially for the number of participants investigated. Since the Neural Network model does train to particular windows, it learns patterns that uncover the relation between eye characteristics and depth of gaze. But the estimates of this model are discrete thus introducing certain window biasing. Biasing could become a problem if future investigations collect gaze data on a different set of individuals at different distances in a similar workspace. For example, if the Neural Network model is used to estimate the VOI of an individual gazing upon a stimulus with a z value of 45 cm in the investigated workspace, then it will estimate the individual's gaze to originate at some (x,y,z) coordinate on one of the windows. This is a set up for an inherently incorrect estimation. However, window biasing is alleviated with the use of the VI model, which is a continuous estimation.

A closer analysis of the BVecOI generated by the Combined Model shows that the average physical length of this vector is not consistent, meaning that the model will produce unpredictable BVecOI's for each sample set. This is not ideal. The Search Region Method solves this problem by reducing the resolution by dividing the workspace into regions. Although resolution is decreased drastically, the z distance searched is predictable: 40 or 70 cm. This reduces searching the z distance by 66 percent (80cm / 120cm) or by 42 percent (50 cm / 120 cm) inside the workspace.

6.2 Future Work

Future work can be divided into two categories. The first is to change certain items of the experiment to measure the affects on accuracy of each model to improve results. The second is to use the models investigated in this research in various applications utilizing 3D space. The list below gives suggestions on items to investigate and the suggestive direction of future work according to the author of this research.

1. Number of Participants

   This research investigates the gaze characteristics of 8 participants. To gain a broader understanding of the accuracy of the models investigated, more participants should be used in future work.

2. Hardware

   The experimental setup uses a Tobii TX300 eye tracker. The eye tracker itself is expensive and has a high sampling frequency. Use of a less expensive eye tracker with a lower sampling frequency should be investigated.

3. Stimulus

   In this research, the Dots on each Window used as stimuli are relatively close together and are all visible while the participant is instructed to fixate upon each one. Future experiments could utilize a moving monitor that presents each Dot individually. This improvement could minimize distraction of the participant and decrease noise in the gaze data.

4. Vector Intersection Model

   Hennessey and Lawrence investigate a modified VI model. This modified VI model could be implemented and compared with the VI model used in this research.

5. Neural Network Model

   The neural network was trained using several participants as a training set in an attempt to generalize gaze data patterns. Training *per participant* could reveal higher accuracy.

6. Voluminous Stimuli

   In future experiments, a voluminous stimuli could be used instead of flat or point placed stimuli. For proper analysis, this would require models to have increased accuracy. However, using these models or improved versions of these models could give clues to how individuals collect visual information on the objects around them.

7. Robotic Path Planning Applications

   The Search Region method is predictable and has moderate accuracy. If this accuracy can be improved, then the SBVecOI could be very effective in aiding with path planning for assistive robots by substantially decreasing a search path.

BIBLIOGRAPHY

[1] A. Tapus, M. Maja, and B. Scassellatti, "The grand challenges in socially assistive robotics," *IEEE Robotics and Automation Magizine*, vol. 14, July 2007.

[2] C. Hennessey and P. Lawrence, "Noncontact binocular eye-gaze tracking for point-of-gaze estimation in three dimensions," *IEEE Transaction on Biomedical Engineering*, vol. 56, no. 3, pp. 790–799, March 2009.

[3] Y. Zhang, "Improvements to the accuracy of eye tracking data based on probable fixations," *Department of CIS Technical Report 2010-04, University of Oregon*, 2010.

[4] SensoMotoric Instruments, "Eye tracking technology by sensomotoric instruments," *http://www.smivision.com/oem-eye-tracking/index.htm*, April 2014.

[5] Tobii Technology, "Tobii eye tracking research," *http://www.tobii.com/en/eye-tracking-research/global/*, April 2014.

[6] iMotions, "Remote eye trackers," *http://imotionsglobal.com/hardware/remote-eye-trackers/*, April 2014.

[7] A. Duchowski, V. Shivashankaraiah, T. Rawls, A.K. Gramopadhye, B.J. Melloy, and B. Kanki, "Binocular eye tracking in virtual reality for inspection training," in *2000 Symposium Eye Tracking Resolution Applications Proceedings*, New York, New York, U.S.A, October 2000, pp. 89–96.

[8] I. Mitsugami, N. Ukita, and M. Kidode, "Estimation of 3-d gazed position using view lines," in *12th International Conference on Image Analysis Processes Proceedings*, New York, New York, U.S.A, September 2003, pp. 466–471.

[9] K. Essigi, M. Pomplun, and H. Ritter, "Application of a novel neural approach to 3-d gaze tracking: Vergence eye-movements in autostereograms," in *Proceedings of the 26th Annual Meeting of the Cognitive Sciences*, 2004, pp. 357–362.

[10] Y. Kwon and K. Jeon, "Gaze computer interaction on stereo display," in *2006 ACM SIGCHI International Conference of Advanced Computer Entertainment Technology Proceedings*, pp. 526–531.

[11] A.T. Duchowski, D.H. House, J. Gestring, R. Congdon, L. Swirski, N.A. Dodgson, K. Krejtz, and I. Krejtz., "Comparing estimated gaze depth in virtual and physical environments," in *Proceedings of the Symposium on Eye Tracking Research and Applications*, Safety Harbor, Florida, U.S.A., March 26-28 2014, pp. 103–110.

[12] Tobii Technology, "Accuracy and precision test method for remote eye trackers," *Test Specification Version: 2.1.1*, pp. 6–7, February 2011.

[13] H.I. Blythe, N.S. Holliman, L.W. Tbaily S. Jainta, and S.P. Liversedge, "Binocular coordination in response to two-dimensional, three-dimensional and stereoscopic visual stimuli," *The Journal of the College of Optometrists*, vol. 32, pp. 397–411, May 2012.

[14] D.W. Hansen and Q. Ji, "In the eye of the beholder: A survey of models for eyes and gaze," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 478–500, March 2010.

[15] M. Hanhela, A. Boev, A. Gotchev, and M. Hannuteela, "Fusion of eye-tracking data from multiple observers for increased 3d gaze tracking precision," in *20th European Signal Processing Conference*, Bucharest, Romania, August 2012, pp. 420–424.

[16] R. Wang, B. Pelfrey, A.T. Duchowski, and D.H. House, "Online gaze disparity via binocular eye tracking on stereoscopic displays," in *IEEE 2012 Second Joint 3DIM/3DPVT Conference*, pp. 184–187.

[17] G.D. Love, D.M. Hoffman, P.J.W. Hands, J. Gao, A.K Kirby, and M.S. Banks, "High-speed switchable lens enables the development of a volumetric stereoscopic display," *Optics Express*, vol. 17, no. 18, pp. 15716–15725, 2009.

[18] R.O. Duda, P. E. Hart, and D.G. Stork, *Pattern Classification*, John Wiley & Sons, Inc., New York, New York, U.S.A., 2$^{\text{nd}}$ edition, 1996.