POINT OF GAZE APPLICATIONS FOR ASSISTIVE INTERACTION

by

JONATHAN WALTER RICH

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON
May 2012

ACKNOWLEDGEMENTS

I would like to thank Dr. Fillia Makedon, my supervising professor, for the guidance and support provided in the Heracleia Human-Centered Computer Laboratory. The support I have received from this lab has helped me greatly in the success of my graduate research, and I want to express my gratitude to all the researchers who worked with me in the lab.

I would also like to thank Dr. Vassilis Athitsos and Dr. Vangelis Metsis for being a part of my committee. Dr. Athitsos steered me into the direction of computer vision after I took his artificial intelligence course as an undergraduate, and I have enjoyed the challenge and applications of the computer vision field. Dr. Metsis has provided great ideas to consider throughout my research, which has helped me tremendously.

Additionally, I would like to thank Christopher Dale McMurrough who brought me to work with the terrific researchers in the Heracleia Human-Centered Computer Laboratory. His support and contribution on our projects has been invaluable, without it this work would not have been possible.

Finally, I would like to thank my parents for all of the opportunities they have provided for me throughout my studies. My family's love and support helps push me to do my best in all aspects of my life.

April 19, 2012

ABSTRACT

POINT OF GAZE APPLICATIONS FOR ASSISTIVE INTERACTION

JONATHAN WALTER RICH, M.S.

The University of Texas at Arlington, 2012

Supervising Professor: Fillia Makedon

Eye tracking has many useful applications in human-machine interfaces and assistive technologies. Traditional input methods, such as the keyboard and mouse, are not practical in certain situations and can be completely ineffective for users with physical disabilities. Since many computing interfaces contain a strong visual component, it follows that knowledge of the user's *point-of-gaze* (PoG) can be extremely useful. Several approaches to PoG tracking have been described with various results and associated costs. Commercially available tracking devices have proven to be very useful for some disabled users, though often the systems can be cost prohibitive. Many computing applications could be enhanced via reliable yet inexpensive PoG tracking devices. Ordinary consumers as well as the disabled could greatly benefit from the combined technologies.

In the past there has been a void in a publicly available eye tracking dataset which combined the information of the head position with the eye tracking video and gaze points. Such a dataset was needed as a standard for comparing accuracy of methods. Many eye tracking devices did not account for head tracking and relied on users remaining still relative to the monitor. Systems which do not allow shift in head position work better in theory than in reality due to inevitable head movement. A low-cost head and eye tracking solution was needed to show such devices are practical without being cost prohibitive.

Presented in this thesis is a new publicly available dataset, and a low-cost head and eye tracking solution. To further environment interaction, a structured light sensor was added to the eye tracking solution. A structured light sensor allows for depth mapping of the environment. Combining the depth mapping with the user's PoG offers more efficient and reliable model segmentation. Ultimately the goal of such a system is to allow for natural human-machine interaction with assistive technologies.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Table Page

CHAPTER 1

INTRODUCTION

1.1 Background

Eye tracking has many useful applications in human-machine interfaces and assistive technologies. Traditional input methods, such as the keyboard and mouse, are not practical in certain situations and can be completely ineffective for users with physical disabilities. Since many computing interfaces contain a strong visual component, it follows that knowledge of the user's *point-of-gaze* (PoG) can be extremely useful.

A key requirement for estimating PoG is detection of the pupil. This is generally done by applying computer vision techniques, such as the "Starburst" algorithm [1], to captured video data. The video data can be obtained using either wearable or remotely mounted recording equipment. Commercial eye tracking equipment tends to be expensive, though low-cost solutions are presented in several works. One approach for building a mobile eye tracking headset is presented in [2]. Another approach integrated with the "Starburst" pupil tracking algorithm is presented in [3]. A system designed for ALS patients using readily available components is described in the "EyeWriter" project [4].

Such systems have been shown to be tremendously helpful as user input devices and can work in situations where no other device would be practical. Most commercially available eye tracking devices are cost prohibitive. However, using off the shelf sensors, low-cost eye tracking devices can be built with a reliable degree of accuracy.

As eye trackers become more accessible the interest in their applications will grow rapidly.

1.2 Purpose

In the past there has been a void in a publicly available eye tracking dataset which combined the information of the head position with the eye tracking video and gaze points. Such a dataset was needed as a standard for comparing accuracy of methods. Similarly, many eye tracking devices did not account for head tracking, and relied on users remaining motionless relative to the monitor. Systems which do not allow shift in head position work better in theory than in reality due to inevitability of head movement being inevitable. A low-cost head and eye tracking solution was needed to show such devices are practical without being cost prohibitive.

To further environment interaction, a structured light sensor was added to the eye tracking solution to provide depth mapping of the environment. Combining the depth mapping with the user's PoG allows for more efficient and reliable model segmentation. Providing segmented models to caretakers will allow for more natural interaction with an environment versus only providing the point in space where the user is focused.

1.3 Thesis Organization

This thesis is broken down into five remaining chapters: chapter 2 focuses on the hardware used throughout the thesis; chapter 3 presents the publicly available dataset for standardizing comparisons of head and eye tracking methods; chapter 4 presents a low-cost head tracking solution to improve point of gaze estimation; chapter 5 discusses a method for combining the PoG information for efficient model segmentation, and chapter 6 provides final thoughts and concluding remarks on the work.

The dataset presented in chapter 3 was submitted and accepted at ETRA 2012 (Eye Tracking Research and Applications) [5]. The low-cost head tracking solution presented in chapter 4 was submitted to PETRA 2012 (PErvasive Technologies Related to Assistive Environments) [6]. Christopher Dale McMurrough and Dr. Vangelis Metsis were authors on both of the submitted papers, and An Nguyen contributed on the low-cost head tracking project. Additionally, the work accomplished in chapter 5 was completed with Christopher Dale McMurrough. All work was done under the supervision of Dr. Fillia Makedon in the Heracleia Human-Centered Computing Laboratory.

CHAPTER 2

HARDWARE

2.1 Overview

This chapter discusses the hardware used throughout the experiments in this thesis. The Vicon Motion Capture System was used as a tool for measuring ground truth. The Applied Science Laboratory Mobile Eye XG captured the eye video data for the eye tracking dataset presented in chapter 3. The Sony PlayStation Eye camera was utilized in the low-cost head tracking hardware solutions presented in chapter 4 and was used as the camera for tracking the user's eye. In the head tracking solution presented in chapter 4, a Nintendo Wiimote stereo camera setup was implemented to track four infrared LEDs. In chapter 5 the Asus Xtion Pro replaced the Wiimote sensors, providing a depth mapping with its structured light sensor.

2.2 Vicon Motion Capture System

The motion capture system consists of 16 tracking cameras surrounding an area measuring roughly 10 x 10 meters. The system is able to track the position and orientation of multiple rigid structures equipped with reflective markers at a rate of 100 Hz with sub-millimeter accuracy. Using this information, we are able to reconstruct the homogeneous transformation matrix for each tracked structure at each time step. Figure 2.1 shows the Vicon setup that was used during the data collection process.

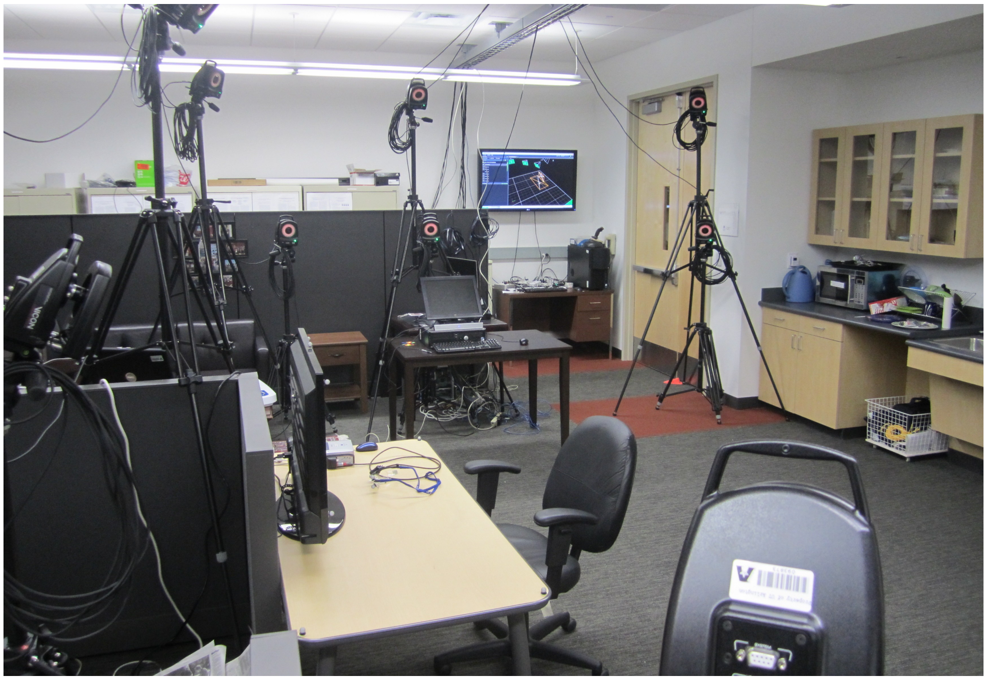


Figure 2.1. Vicon Motion Capture System Setup At Heracleia Laboratory.

2.3 Applied Science Laboratories Mobile Eye XG

A commercially available eye tracker was used to obtain eye video data for the dataset which is presented in chapter 3. The eye tracker used is an Applied Science Laboratories Mobile Eye XG[1]. The video data is recorded with a resolution of 768 x 480 pixels at a frame rate of 29.97 Hz. The Mobile Eye projects a triangular infrared glint pattern on the user's eye during recording, which can be used as an additional tracking feature.

2.4 Sony PlayStation Eye Camera

The Playstation Eye device was chosen based on the results presented in previous work on the EyeWriter project [4]. The device is capable of providing images with a resolution of 640x480 at a rate of 60 Hz. The interface to the host computer is

[1] http://www.asleyetracking.com/

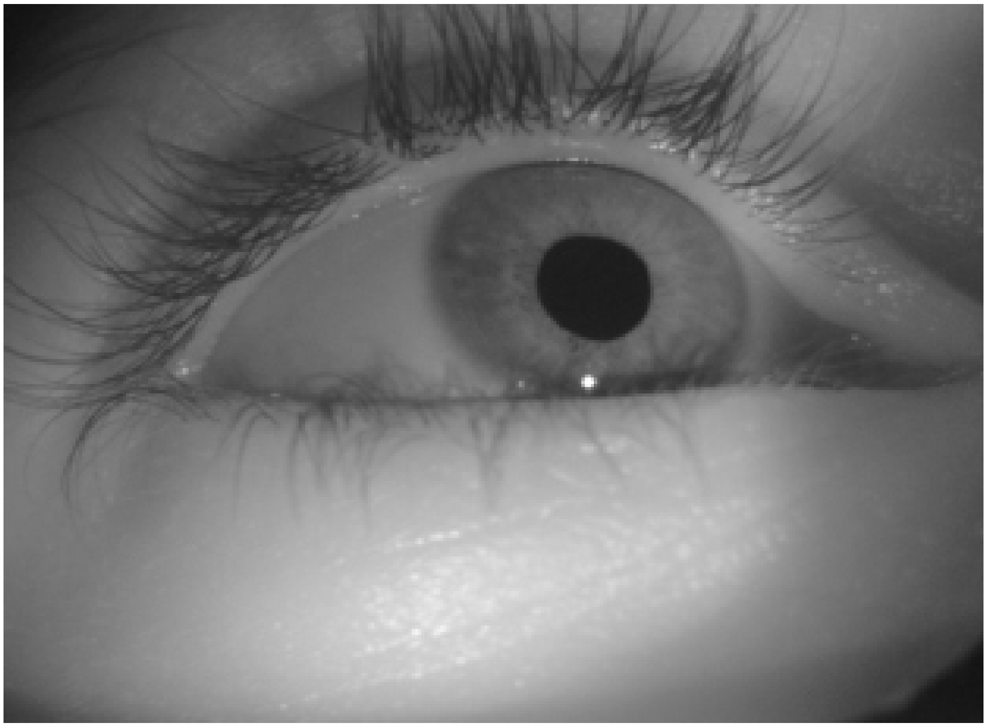


Figure 2.2. Eye Camera View.

provided using the USB 2.0 standard, which removes the need for a dedicated video capture interface for real-time streaming of images.

The Playstation Eye device comes equipped with an infrared filter and wide field of view lens. In order to provide close-up infrared video of the pupil, the filter and lens must be replaced. The modification process used in our approach is identical to the method documented by the EyeWriter project. After replacing the factory lens and image filter, two 850nm infrared LEDs were added to the camera in order to illuminate the user's eye outside of the visible light spectrum. This wavelength was chosen to minimze the possibility of interference with the Wiimote sensors, which are most sensitive to 940nm infrared light [7]. This resulting high-quality images are robust to changing conditions in ambient light. A captured image using the eye camera is shown in Figure 2.2.

2.5 Nintendo Wiimote Stereo Camera

These sensors consist of a CMOS camera with integrated vision processing, capable of tracking up to four infrared light sources at a rate of 100 Hz. These specifications allow for the creation of an effective stereo tracking solution for fixed sources of infrared light, such as LEDs.

2.5.1 Stereo Calibration

Before the stereo camera subsystem can be used to obtain 3D positions, the imaging sensors must undergo a calibration process. Through this process, an estimation of the intrinsic parameters of both imaging devices, as well as the extrinsic parameters of the stereo pair can be acquired. In our approach, the Camera Calibration Toolbox for Matlab [8] is used to obtain the calibration constants. These constants need only be computed once, assuming that the position and pose of the two imaging devices relative to each other are fixed.

To calibrate the stereo pair using the Camera Calibration Toolbox, a calibration grid must be created with a known distance between points. When calibrating a standard video camera, a checkerboard pattern with known dimensions is generally used. This will not work in our approach for two reasons: The Wiimote sensors can view only 4 points at a time, and the sensors are capable of detecting only light in the infrared spectrum. To address this problem, a 13 x 13 calibration grid , shown in Figure 2.3, was created using a perforated board with small holes drilled on a 1 inch grid. Data from the Wiimotes is collected until all 169 distinct grid points are detected as an infrared LED is moved along the back of the fixed board. Once enough points are collected, the data is saved in an image file and the stereo pair is moved to a different position and orientation. This process is repeated until enough calibration images are acquired (in our case, 12 different positions). The resulting

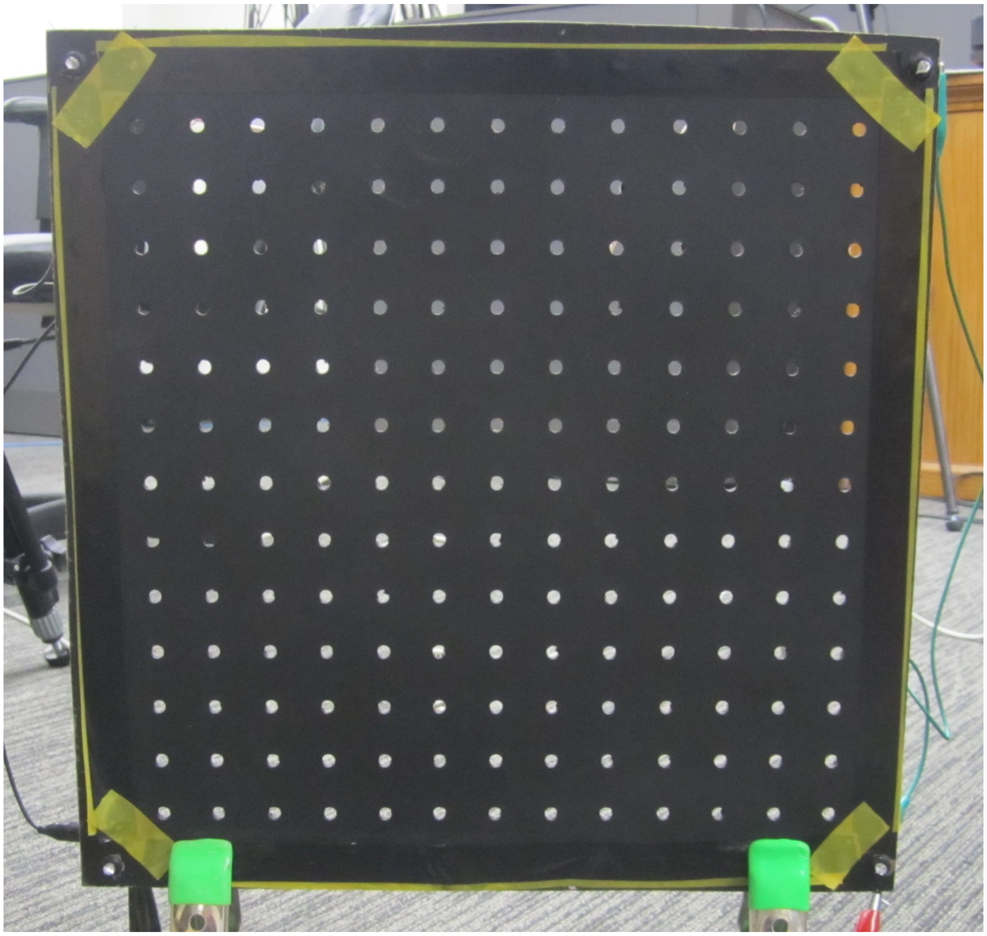


Figure 2.3. Calibration Grid.

image set is then used to estimate the intrinsic and extrinsic camera parameters using the calibration toolbox. The position of the grid in each of the calibration images relative to the stereo camera subsystem is shown in Figure 2.4.

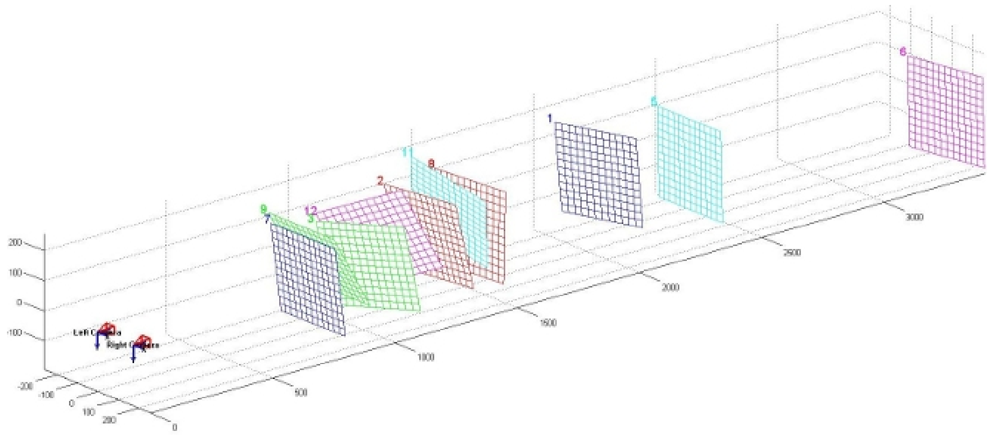


Figure 2.4. Stereo Camera Calibration Process.

2.6 Asus Xtion Pro

The Asus Xtion Pro is a device similar to the Microsoft Kinect, which has a structured light sensor for retrieving depth maps and a standard RGB camera. For the purpose of these applications the Xtion Pro was more ideal due to its smaller size and the fact that it does not require an AC power source. These factors allowed it to be more easily mounted to a pair of glasses.

2.6.1 Structured Light Sensor

The structured light sensor in the Xtion Pro provides a depth map with a resolution of 640x480 at a rate of 30 Hz. This depth map gives a point cloud which can be manipulated to further understand an environment more easily than images which contain only RGB data. Such a sensor will provide data in environments that have normal indoor lighting conditions to no lighting at all, but it will not function properly in an outdoor setting.

2.6.2 RGB Camera

The front facing RGB camera provides standard images at a resolution of 640x480 at a rate of 30 Hz. The data from the RGB camera is combined with the point cloud from the structured light sensor, but can be manipulated separately as well. This camera is important in the calibration of the system when combining with the eye tracking information, and from this camera we can then infer which point in the point cloud a user is gazing upon.

CHAPTER 3

EYE TRACKING DATASET FOR POINT OF GAZE DETECTION

3.1 Overview

Some of the approaches try to detect the PoG at each time point by detecting the pupil center or other features and some other approaches track the eye motion over time. In all cases, the inherent difficulty of tracking the eye itself, plus the fact that the head position has to be taken into consideration for the estimation of the final PoG, prevent most systems from giving very accurate results. In addition, direct comparison of the effectiveness of each of the proposed methods is not possible due to the different parameters of the datasets on which each method has been tested.

Although many of the proposed methods manage to roughly estimate the PoG on a computer display unit, none of them provides enough accuracy to allow convenient input through eye motion in a standard visual computer interface, especially when head motion is involved. Such fine grained detection of the PoG would be of great benefit to people, for example, that cannot use their hands to interact with a computer due to some type of disability.

This datasets aspires to provide a resource for the development of new, more accurate eye tracking methods with focus on point of gaze detection, as well as a benchmark for the comparison of such methods. The dataset includes a set of videos recording the eye motion of 20 human test subjects as they looked at predefined positions of a computer display or followed a target while it moved inside the display dimensions.

The ground truth of where the subject was looking at, at every time point, is known in advance, as the subjects were advised to keep their eyes on the target at all times. The subject's head position and orientation relative to the display is tracked in three dimensions using a Vicon Motion Capture System[1], which provides sub-millimeter and sub-degree accuracy for the translation and rotation of the tracked object respectively, in the 3D space. This guarantees a virtually perfect ground truth regarding the subject's head pose and location, which allows researches to only worry about dealing with the eye tracking accuracy when trying to determine the exact point of gaze. In the collected data, the motion of the right eye of each subject as they were staring at, or following, predefined targets on the display has been recorded using an Applied Science Laboratories Mobile Eye XG, head mounted, infrared monocular camera.

The synchronized and timestamped eye recordings and head tracking data have been made publicly available to be used for educational and research purposes.

3.2 Data Collection Methodology

A total of 20 subjects (2 women and 18 men) participated in our data collection sessions. The participants were graduate and and undergraduate students of the University of Texas at Arlington. Some of them had normal vision and some of them wore contact lenses or spectacles. The participants that normally wore spectacles did not use them during the data collection process. However, their vision level was still good enough to locate the displayed target on the monitor. The special characteristics of each subject are given as metadata together with the dataset.

Each subject participated in two video recording sessions in which they looked at (or followed) a target on a *Samsung LN32C350 (32 Inch, 1360x768 pixels)* display

[1] http://www.vicon.com/

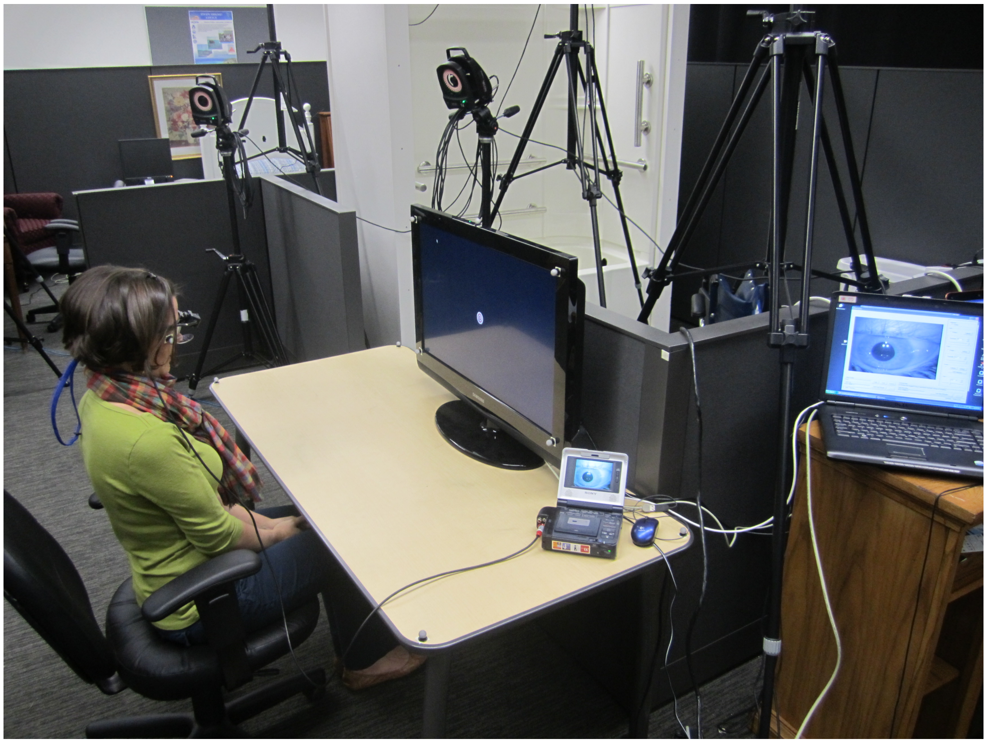


Figure 3.1. Subject Wearing Eye Video Recording Device.

unit. In the first session the subjects were allowed to move only their eyes while keeping their head still, whereas in the second session, they were allowed to freely move their heads and eyes at their convenience. Figure 3.1 shows a photo taken during the data collection process. In the photo, the reader can see the experimental setup used for the data collection.

In the first session the subjects were asked to keep their head still and their eye motion while looking at different patterns of targets appearing on the computer display was recorded. In the first pattern, a target appeared at 9 different positions of the display for a few seconds and the subjects were instructed to look at the target as soon as it appears and until it disappears. Note that since the human eye may require a few milliseconds from the moment the target appears on display until the moment the eye point of gaze moves to fall onto it, the target location and the point of gaze

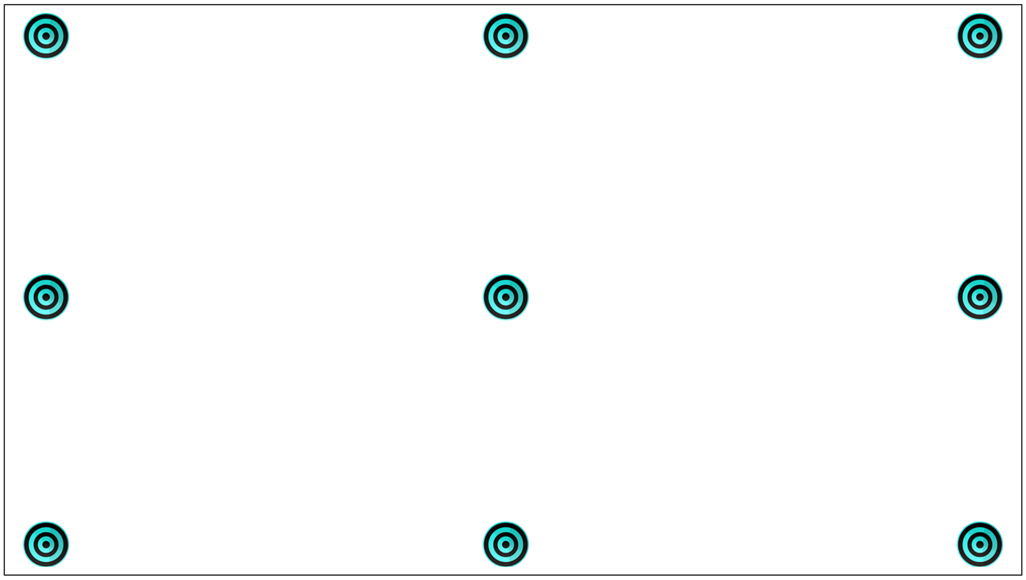


Figure 3.2. Example 9 Target Location Pattern.

may not align for a few milliseconds after the appearance of each target. However, they should be aligned soon after.

Similarly, in the second pattern, 16 targets appeared on the display and the subject had to repeat the same procedure. The number of targets and their location on the display was chosen so as to resemble common calibration patterns used by eye trackers. Figure 3.2 shows an example of targets appearing on different locations of the display. The third pattern, instead of static targets, involved a target moving inside the display and the subjects had to follow the target with their eyes at all times. This patterns is particularly useful for eye tracking methods which do not statically determine the point of gaze but follow the center of the pupil or other eye features over time.

Since in real life people do not move only their eyes but also their heads to look at different targets (or follow a target as it is moving), in the second session the subjects repeated the same process as in the fist session, but this time they did not have any constraints regarding their head motion. The exact position of the head

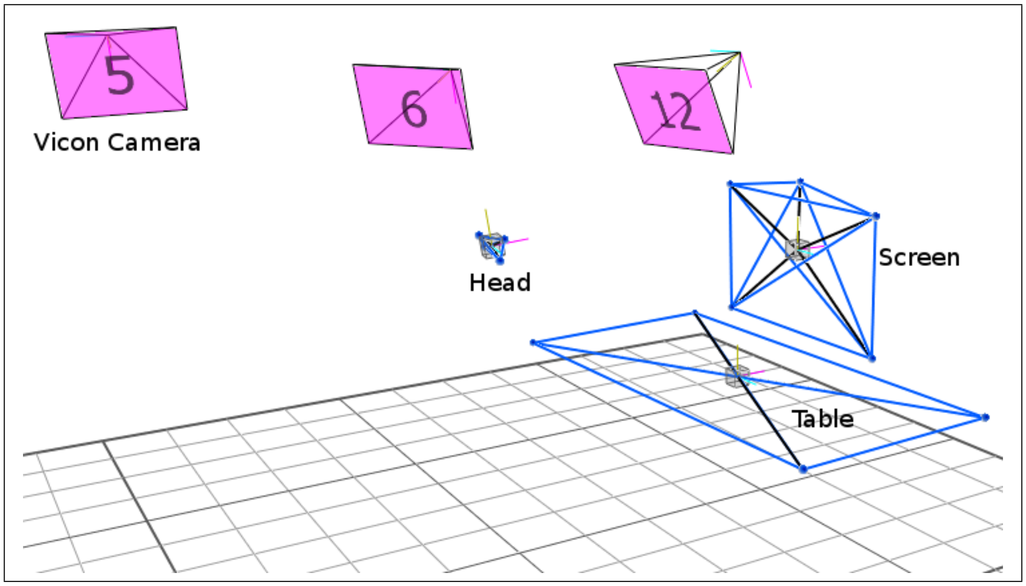


Figure 3.3. Vicon Motion Capture System Virtual Representation.

in the 3D space was tracked by the Vicon system using a set of markers attached to the head mounted eye video recorder. For convenience, four markers were also attached at the corners of the computer display. This allows us to determine the exact location of the head and the display monitor in the same coordinate system. Figure 3.3 visualizes the locations of the head and the computer display in the 3D space as captured by the Vicon System.

Requiring from the users to keep their head still while moving only their eyes is a common practice used by many eye tracking systems which incorporate some kind of calibration before using the eye input. The accuracy of such systems deteriorates over time even with subtle head movement and they need to be re-calibrated to become usable again. One of the reasons that we included the session of trying to hold the head still in our collected data was to consider such cases and provide the option for experiments that evaluate the accuracy loss due the head position drifting. The exact head pose and orientation is still tracked by the Vicon system and provided as ground truth.

3.3 Hardware

This section describes the hardware setup used during data collection. In each recording session, the user is instructed to visually track target points on a fixed video display while wearing an eye recording device. This device collects video data of the subject's eye while being tracked in 3D within a motion capture system. The data streams from the motion capture system are synchronized with the video frames provided by the eye recording device, as well as the pixel coordinates of the video display target points.

The Mobile Eye is used to obtain the eye video data, and is worn on each subject's head during data collection, and is positioned such that the user's right eye is centered in the video frame. The user head position and pose was measured during the data collection process using a Vicon motion capture environment. The tracked structures of interest in our case were the display unit and the subject's head.

3.4 Coordinate System

In this section we describe the coordinate systems (CSs) used in the dataset. We have the following CSs:

1. $\{W; \mathbf{x}_w, \mathbf{y}_y, \mathbf{z}_w\}$ The world reference frame
2. $\{M; \mathbf{x}_m, \mathbf{y}_m, \mathbf{z}_m\}$ attached to the center the display monitor
3. $\{E; \mathbf{x}_e, \mathbf{y}_e, \mathbf{z}_e\}$ attached to the eye tracker

Each CS is defined using the right-hand notation. The origin of CS {W} is located on the ground behind the test subject, with $\mathbf{z}_w$ pointing up and $\mathbf{x}_w$ pointing

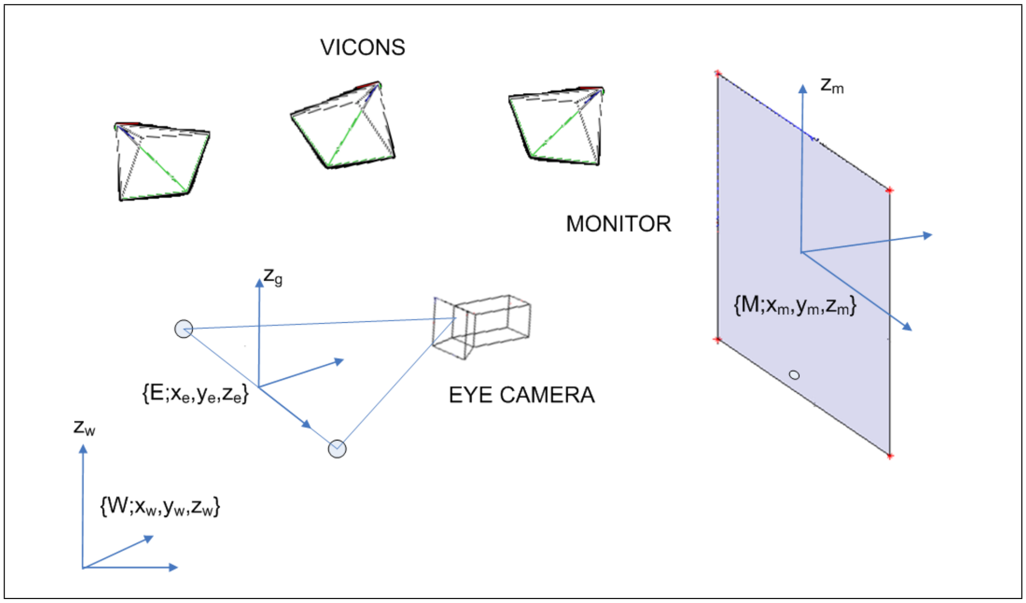


Figure 3.4. Coordinate Systems: World (W), Eye (E) and Monitor (M).

to the right. CS {M} is attached to the center of the monitor, with $\mathbf{z}_w$ pointing up and $\mathbf{x}_w$ pointing to the right. CS {E} is attached to the center of the MobileEye device, with $\mathbf{z}_w$ pointing up and $\mathbf{x}_w$ pointing to the right. The transformation matrices contained in the dataset are from CS {W} to CS {M} and from CS {W} to CS {E}. Figure 3.4 shows the relative assignment of CSs used in the dataset.

3.5 Dataset Structure and Format Details

For every subject, the dataset includes six videos in avi format and corresponding metadata. File sets 00001, 00002, 00003, come from the sessions in which the subject was to remain as still as possible, and file sets 00004, 00005, 00006, come from the session in which the subject was freely allowed to move their head. File sets 00001 and 00004 contain the videos of the first pattern of each session (a target appearing in 9 different locations of the display unit), file sets 00002 and 00005 contain the second pattern (16 target locations), and file sets 00003 and 00006 contain the videos coming from third pattern (target moving within the display). The metadata includes two homography matrices, H from {W} to {M} and H from {W} to {E}, and it also includes the (i, j) pixel location of the target location for each video frame. This data is packaged as a MATLAB (.mat) data file and as a CSV file. The .mat file contains a structure which includes the two homography matrices and the pixel locations. The data is written into the CSV file with each column corresponding to a frame in the video, rows 1:16 represent the flattened homography matrix from {W} to {E}, rows 17:32 represent the flattened homography matrix from W to M, and rows 33:34 represent the flattened pixel locations. The homography matrices are unflattened by their (i, j) value being the $(((i - 1) \times 4) + j)$ of their respective columnar data. The pixel locations are unflattened by their (i, j) value being the first and second values respectively.

3.6 Future Work

A resource still missing from the eye tracking community is a dataset where objects located or moving in the 3D space, instead of a 2D computer display, are followed by the human eye. Such a resource would be of great interest to robotics applications where the communication of humans with robots involves interaction in the 3D space. Future work will be to publish another dataset that covers this gap.

CHAPTER 4

HEAD POSITION TRACKING FOR GAZE POINT ESTIMATION

4.1 Overview

A key requirement for estimating PoG is detection of the pupil. This is generally done by applying computer vision techniques, such as the "Starburst" algorithm [1], to captured video data. The video data can be obtained using either wearable or remotely mounted recording equipment. Commercial eye tracking equipment tends to be expensive, though low-cost solutions are presented in several works. One approach for building a mobile eye tracking headset is presented in [2]. Another approach which is integrated with the "Starburst" pupil tracking algorithm is presented in [3]. A system designed for ALS patients created using readily available components is described in the "EyeWriter" project [4].

Another key requirement for PoG estimation is knowledge of the head position. In order for a vector to be traced from the pupil of the subject to a 3D location in the world, the position of the head in the world reference frame must be known. While this can be ignored when the head maintains a fixed position and orientation, most real-world applications require at least some degree of head movement, and thus tracking becomes necessary. Head tracking using remote cameras has been previously accomplished with computer vision techniques such as using intensity gradients and color histograms [9] or known texture-mapped 3D models [10]. Position estimation by tracking features in a visual scene from a head-mounted camera is described in [11].

Head position and orientation tracking using standard camera images tend to include some drawbacks. Issues such as lighting conditions and image quality can make visual features harder to detect, resulting in reduced performance. Head tracking using infrared cameras and LEDs has been successfully demonstrated in [12]. Infrared tracking tends to be very robust to ambient lighting conditions, and can be found in inexpensive commercial devices such as the Nintendo Wiimote. One approach for using the Wiimote sensor to track the head pose is presented in [13]. Tracking of 3D marker positions has been shown feasible when two Wiimote devices are used as a stereo camera system in [14]. In [15], a similar approach is used to track a person in a virtual reality application.

Presented in this section is a low-cost solution (roughly 120 USD) for real-time tracking of a human user's head position and eye gaze with respect to a video display source. The display source is equipped with 4 infrared LEDs, which are tracked in 3D using a Wiimote stereo camera system. We describe the hardware integration of stereo Wiimote head tracking and pupil tracking by building upon a pair of EyeWriter glasses [4], in addition to providing mathematical methods for integration of gaze estimation and head position.

4.2 Hardware

The head and eye tracking solution consists of four key subsystems: the stereo camera assembly, eye tracking camera, video display with infrared markers, and the host computer. The stereo camera and eye tracking components are both attached to the headset worn by the user, while the video display and host computer are located remotely. The headset structure is provided by a pair of modified sport sunglasses for comfort. Figure 4.1 shows the assembled headset.

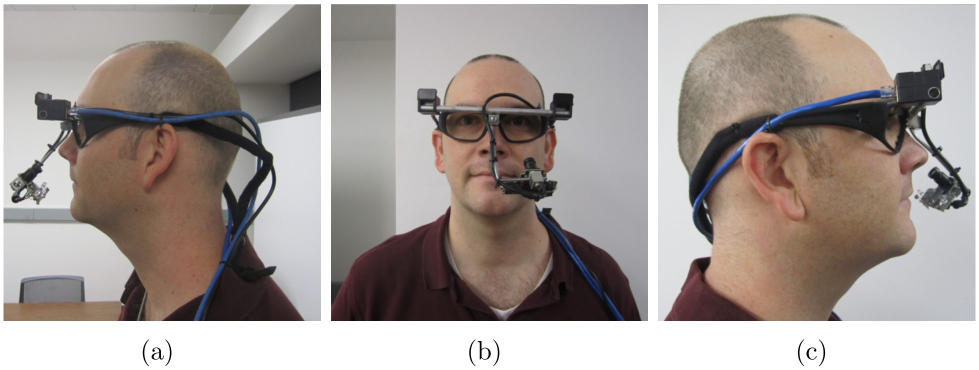

(a) (b) (c)

Figure 4.1. Headset: a) Left View, b) Front View, c) Right View.

The eye tracking subsystem, consisting of a single camera, is used to extract the pupil position during eye movements using computer vision techniques. In our system, a Playstation Eye device is positioned in front of the left eye such that the pupil image is as large as possible in the camera frame while minimizing the visual obstruction to the user. The Playstation Eye device was chosen based on the results presented in previous work on the EyeWriter project [4].

The stereo camera subsystem is used to obtain the 3D positions of the display markers relative to the headset. These measurements, obtained using stereo triangulation, are then used to estimate the position and orientation of the headset relative to the monitor. Implementation of this subsystem was achieved using a pair of Pixart imaging sensors extracted from Nintendo Wiimote controllers.

The Wiimote imaging sensors were mounted on the left and right sides of the headset frame in such as way that they do not obstruct the view of the user. To save and reduce the weight of the headset, the sensors were removed from the Wiimote circuit board and mounted in a smaller plastic housing. A standard 8-pin networking cable was used to bridge the connection between each of the sensor contacts and their appropriate place on the circuit board. The length of cable used for each sensor was approximately 4 feet, which allows the Wiimotes to be carried in a comfortable location away from the users head. The baseline distance between the Wiimote sensors when mounted on the headset frame is approximately 6.5 inches.

4.3 Gaze Point Estimation

In this section we describe the coordinate systems (CSs) of the proposed setup. We have the following CSs:

1. $\{M; \mathbf{x}_m, \mathbf{y}_m, \mathbf{z}_m\}$ attached to the upper left corner of the monitor, and coincides with the world CS.
2. $\{CL; \mathbf{x}_{cl}, \mathbf{y}_{cl}, \mathbf{z}_{cl}\}$ attached to the left camera
3. $\{CR; \mathbf{x}_{cr}, \mathbf{y}_{cr}, \mathbf{z}_{cr}\}$ attached to the right camera
4. $\{E; \mathbf{x}_{ce}, \mathbf{y}_{ce}, \mathbf{z}_{ce}\}$ attached to the camera that monitors the eye

We briefly denote as {A}, {B} the CS that we defined with origins the points A, B; as ${}^{A}\mathbf{T}_{B}$ the homogeneous transformation matrix that gives the transformation from CS {A} to CS {B} and as ${}^{A}\mathbf{x}$ the position vector in CS {A}. The left/right cameras make a stereo pair and their calibration was performed using the matlab stereo calibration toolbox. The calibration calculated the intrinsic parameters of the cameras and the ${}^{CL}\mathbf{T}_{CR}$ from the left to the right camera.

On the four corners of the monitor we fixed four LEDs (their positions related to {M} are known), which can be recognized in the images of our stereo cameras after embedded processing. The next step is stereo triangulation of the identified points. The matching of the points is simply done by matching the respective corners of the projected rectangle in both images, given that the distance between the left/right camera is small and that they move together. Then we extract the transformation matrices ${}^{CL}\mathbf{T}_{M}$, ${}^{CR}\mathbf{T}_{M}$ from {CL}, {CR} to the monitor respectively using the method described in [16].

The center of the iris circle in the {M} CS is given by:

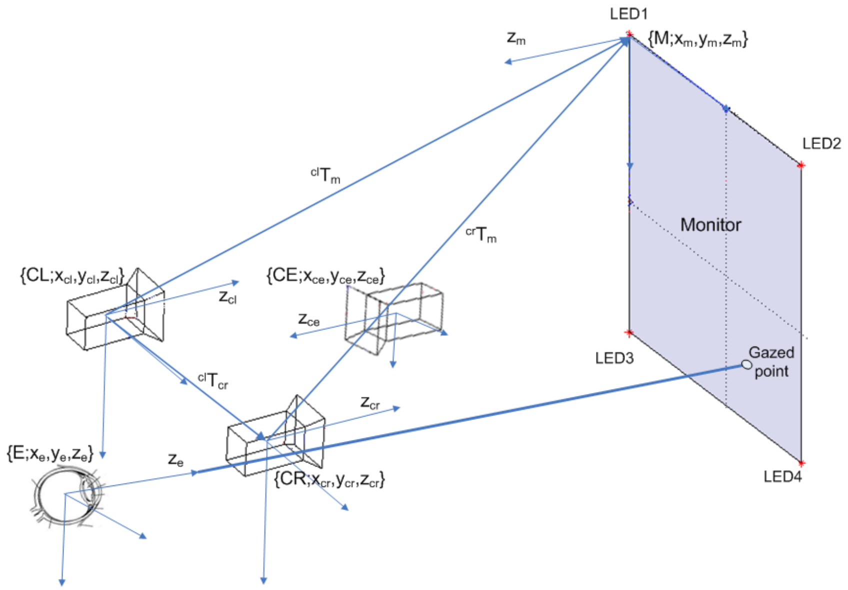


Figure 4.2. Coordinate Systems $\{CL\}$, $\{CR\}$, $\{E\}$ and $\{M\}$.

$$^{M}\mathbf{x}_{iris} = {}^{M}\mathbf{T}_{CL} \cdot {}^{CL}\mathbf{T}_{CE} \cdot {}^{CE}\mathbf{x}_{iris} \tag{4.1}$$

where the position of the iris in the {CE} $^{CE}\mathbf{x}_{iris}$ can be estimated along with the supporting plane with an algorithm like the ”one circle” [17]. Knowing the supporting plane and by taking the normal vector that passes through the image center it is possible to find the intersection with the monitor plane, which gives the actually gazed point. The transformation matrix $^{CL}\mathbf{T}_{CE}$ can be estimated either by mechanical means or by solving (1) for several points.

4.4 Experimentation

In this section, we describe the testing and evaluation of the head tracking capabilities of the stereo camera subsystem.

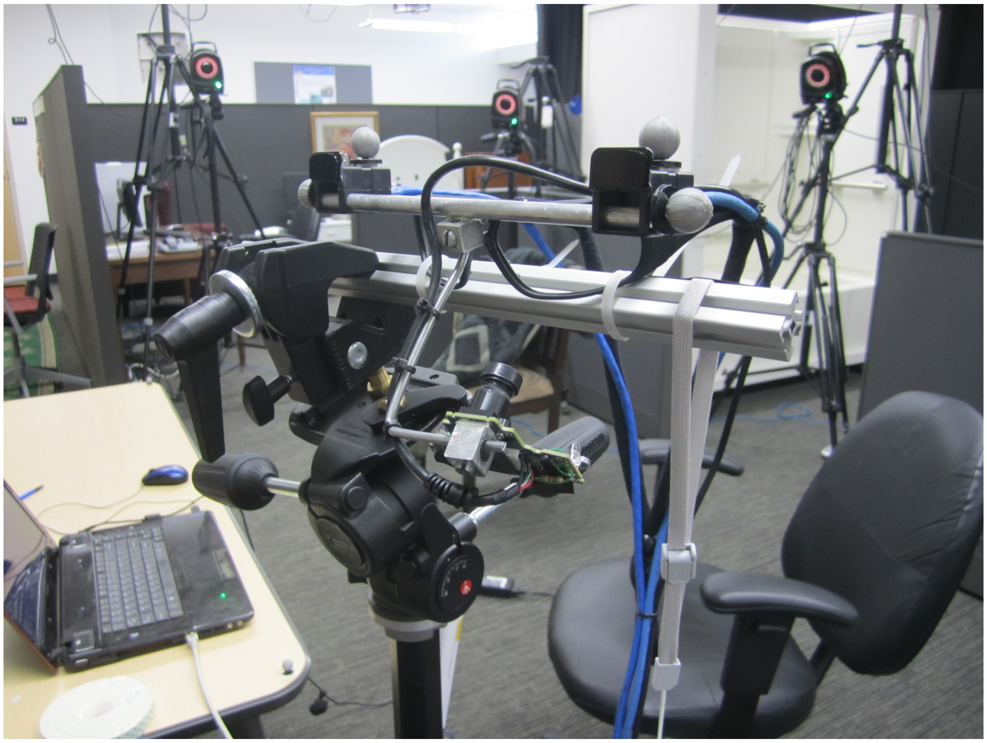


Figure 4.3. Setup With Reflective Markers.

4.4.1 Experimental Setup

Testing and evaluation of the headset tracking accuracy was performed within the Vicon Motion Capture System in the Heracleia Lab at the University of Texas at Arlington. The Vicon system was described in section 2.2. This provides a ground-truth comparison for our solution. Transformation matrices for the marked headset and monitor positions are saved along with the estimates provided by stereo triangulation at each time-step. This data is synchronized such that the tracking error of the triangulation estimate relative to the motion capture data can be computed offline.

4.4.2 Results

In order to compare the tracking estimate of the headset to with that of the motion capture system, the individual roll, pitch, and yaw angles were extracted from the transformation matrices along with the corresponding position components. The tracked position values by both the headset and motion capture system are shown in in Figure 4.4, while the rotational components are shown in Figure 4.5. The solid blue lines correspond to the headset tracking values, while the dashed red lines correspond to the motion capture system.

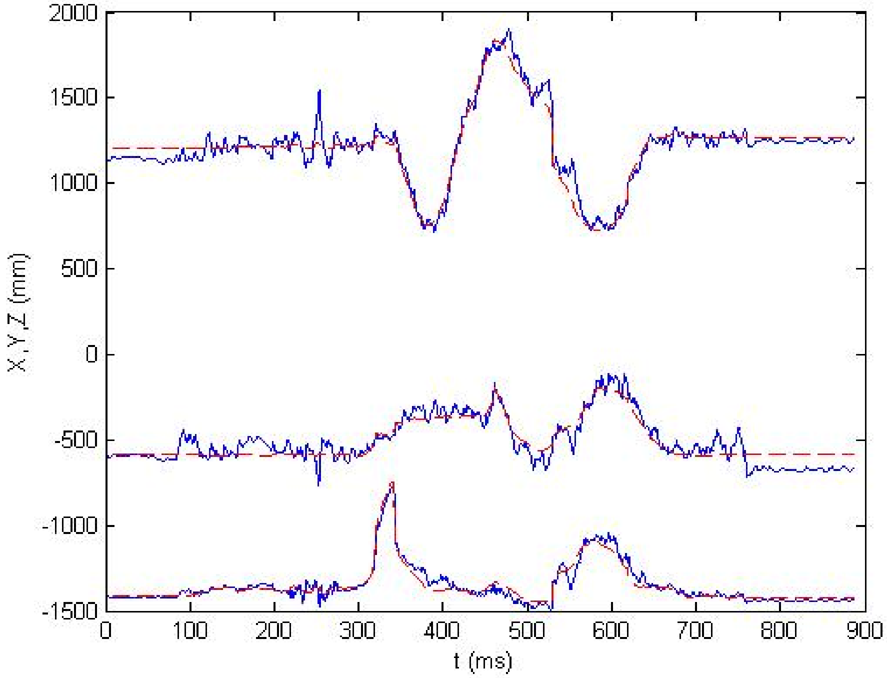


Figure 4.4. Position Comparisons of Estimated and Actual.

The root-mean-square errors (RMSE) were computed individually for each position and orientation component and are shown in Table 4.1. The measurements from the motion capture system and headset were not filtered prior to analysis, which contributes to much of the error found in the results. A simple low-pass filter would significantly reduce the error, as the Wiimote sensors readings to contain some noise.

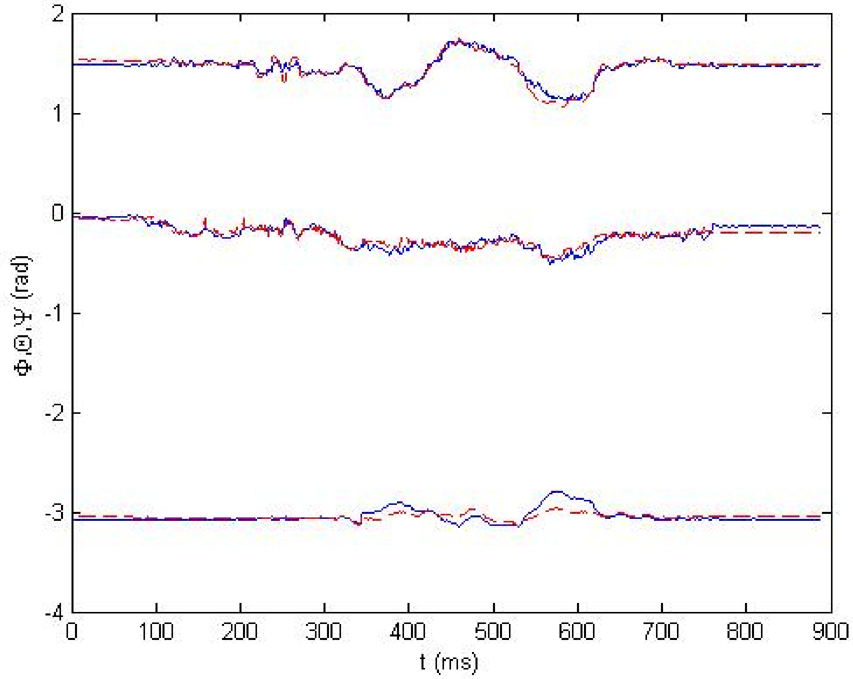


Figure 4.5. Angle Comparisons of Estimated and Actual.

Still, even with the amount of error present in the experiment data, the approach is accurate enough to provide a useful degree of tracking capabilities.

Table 4.1. RMSE Of Position And Orientation Components

	X (mm)	Y (mm)	Z (mm)	ϕ (rad)	θ (rad)	ψ (rad)
RMSE	31.465	57.829	55.516	0.045	0.038	0.0517

4.5 Future Work

In future work, we will complete the gaze vector estimation aspect of the system. Once this task is completed, the point of regard on the monitor can be computed using the approach presented in this paper. The addition of inertial sensors to the headset, such as accelerometers and gyroscopes, is also being investigated in order to improve the tracking accuracy.

Additionally, work will be done to include an RGB camera into the headset, in order to combine the estimate of the 3D gaze point and the utilization of a SIFT based algorithm to detect what real-world objects are present within the field of view. With such a setup, the current infrared sensors and LED beacons could potentially be removed in favor of passive visual markers which are located via feature extraction in the RGB camera image.

CHAPTER 5

MODEL SEGMENTATION WITH GAZE POINT INTEGRATION

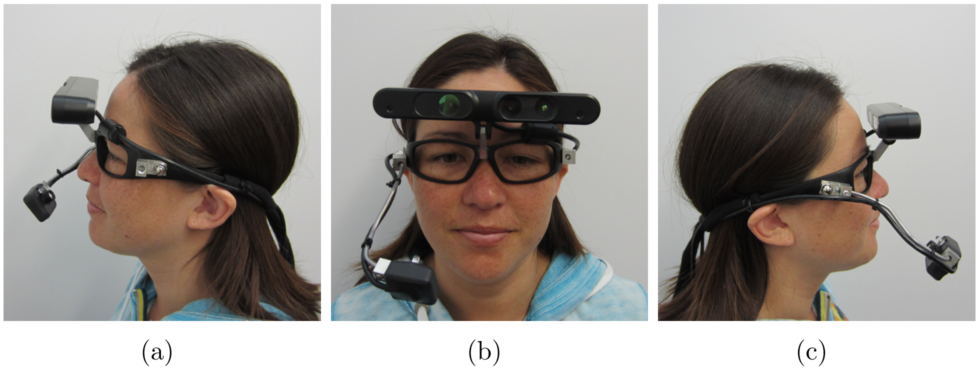
5.1 Overview

While PoG detection for interaction with 2D surfaces, such as monitors, can provide an invaluable user interface for those who are unable to communicate via traditional means, interaction within a 3D environment can provide a natural interface for user interaction. Combing point clouds collected from depth mapping sensors with the PoG information, can allow for interaction with caretakers or assistive robots.

When interacting with caretakers, providing a visual of the object being gazed upon rather than simply a location, will give the information necessary for the caretaker to properly aid the user. Objects of interest will need to be defined in such a way that they can be readily segmenting from a point cloud. For the purposes of this chapter cylinder models were segmented as they are defined with seven values, enough to illustrate such models could be segmented quickly enough to provide the information in a timely manner.

5.2 Hardware

The headset resembles the solution presented in section 4.2 with two main alterations: the stereo camera is replaced with the Asus Xtion Pro, and the video display with infrared markers is no longer necessary. The Xtion Pro camera and eye tracking components are both attached to the headset worn by the user, while the host computer is located remotely. The headset structure is provided by a pair of modified sport sunglasses for comfort. Figure 5.1 shows the assembled headset.



(a) (b) (c)

Figure 5.1. Headset: a) Left View, b) Front View, c) Right View.

The Xtion Pro is used to obtain both the depth mapping of the environment, as well as using its front facing RGB camera for calibration and PoG mapping purposes. The depth map will serve as the raw point cloud data to be used in the segmentation process. The RGB camera allows the ”Starbust” algorithm to be run to track the user’s pupil location, and provide an (i, j) location of the PoG in the RGB image which correlates to a point in the depth map.

5.3 Integration of Gaze Estimation and Model Segmentation

Utilizing the PoG of the user allows for more efficient model segmentation, the general flow is illustrated in figure 5.3. Taking a cube of space from the point cloud which represents the area being gazed upon by the user allows for less down sampling to be required, and removes false matches that are outside the scope of consideration. Subsection 5.3.1 presents the algorithmic flow.

The PoG is found using a the open source implementation of the ”Starburst” algorithm presented by [1, 3]. Using the front facing RGB camera, calibration points can be manually selected, and the algorithm with use a linear mapping to determine the gaze point when looking within the field of view. A future implementation will

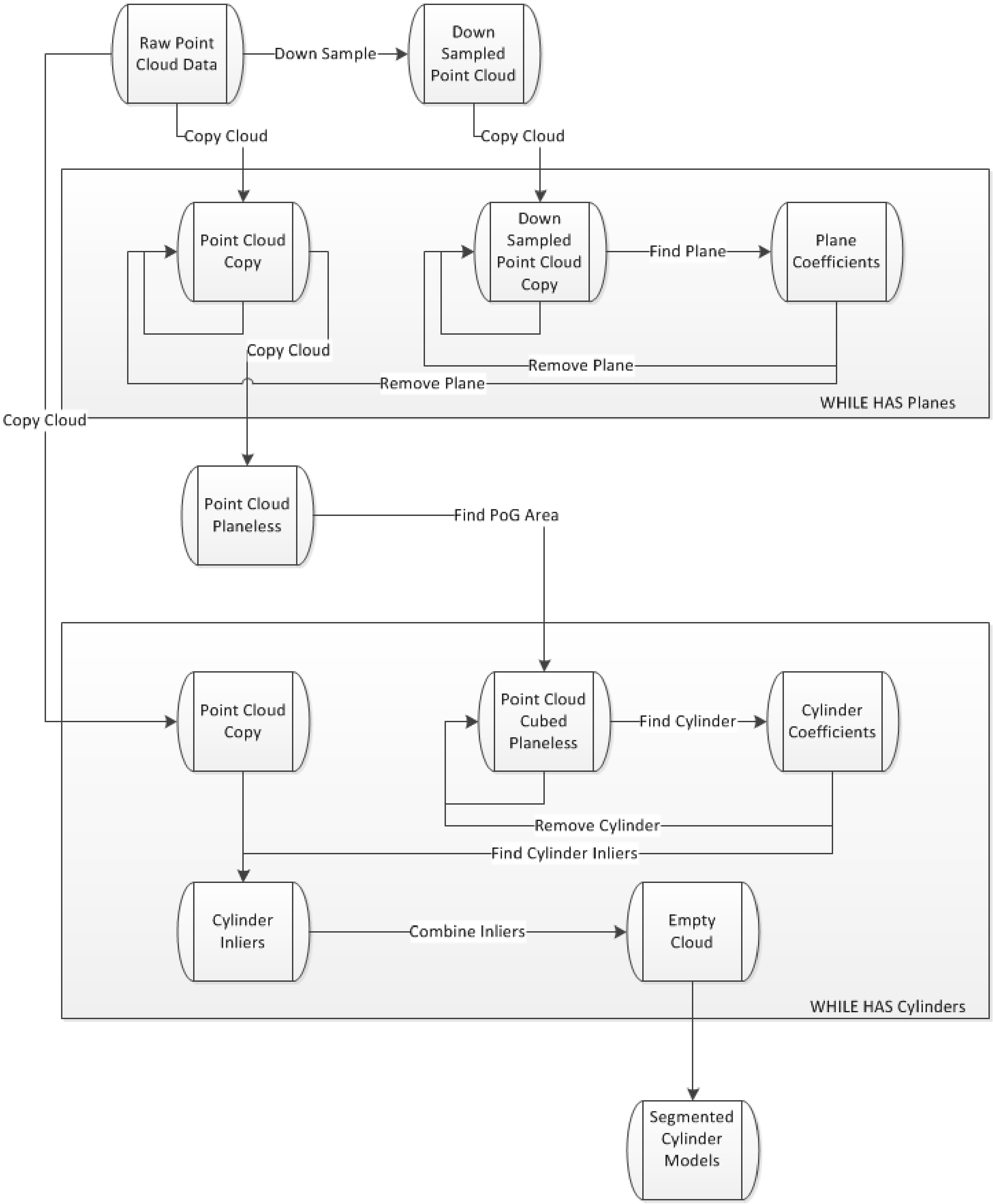


Figure 5.2. Data Flow For Model Segmentation.

possibly allow the user to select such points via blinking or another more automated method.

5.3.1 Model Segmentation

Since the objects of interest will rarely follow the plane model, and planes make up a considerable amount of the point cloud data, the first step will be to remove planes from the original point cloud. Planes which will be present in the point cloud include the walls, floor and ceiling. The removal process is described in algorithm 2. For object segmentation and general point cloud manipulation, the open source point cloud library (PCL)[1] was used [18]. The model segmentation in found in the PCL implementation includes is performed by a RANSAC (RANdom SAmpling Consensus) method to determine the number of inliers for a given model. Once the planes are removed, the point cloud must be filtered to only contain points which are within reasonable distance to the user's PoG. The filtering process is shown in algorithm 3. After finding the relevant points, a final extraction is made on the points which represent cylinder models, as shown in algorithm 4. The entire process flow is written in algorithm 1.

Algorithm 1 Find-Cylinder($cloud, point$)

```
  $cloudDownsampled \leftarrow$ Down-Sample($cloud$)
  $cloudPlaneless \leftarrow$ Remove-Planes($cloud, cloudDownsampled$)
  $cloudCubed \leftarrow$ Find-Cube($cloud, point, minPoints, maxPoints$)
  $cloudCylinders \leftarrow$ Extract-Cylinders($cloud, cloudCubded$)
```

[1] http://www.pointclouds.org/

Algorithm 2 Remove-Planes($cloud, cloudDownsampled$)

```
repeat
  $coefficients \leftarrow$ Find-Plane($cloudDownsampled$)
  $inliers \leftarrow$ Find-Inliers($cloud, coefficients$, PLANE)
  if Size($inliers$) $\leq$ THRESHOLD then
    return
  end if
  Remove-Inliers($cloud, coefficients$, PLANE)
  $inliers_{D}S \leftarrow$ Find-Inliers($cloudDownsampled, coefficients$, PLANE)
  Remove-Inliers($cloudDownsampled, coefficients$, PLANE)
until false
```

Algorithm 3 Find-Cube($cloud, point, minPoints, maxPoints$)

```
direction ← neutral
distance ← 1
repeat
    cubedCloud ← Filter-Distance(cloud, point, distance)
    if Point-Count(cubedCloud) ≥ maxPoints then
        maxDistance ← distance
        if direction IS neutral then
            direction ← shrink
            distance ← distance/2
        else
            distance ← (minDistance + maxDistance)/2
        end if
    else if Point-Count(cubedCloud) ≤ minPoints then
        minDistance ← distance
        if direction IS neutral then
            direction ← grow
            distance ← distance * 2
        else
            distance ← (minDistance + maxDistance)/2
        end if
    else
        return cubedCloud
    end if
until false
```

Algorithm 4 Extract-Cylinders($cloud, cloudCubed$)

```
$clyinders \leftarrow$ EMPTY
repeat
    $coefficients \leftarrow$ Find-Cylinder($cloudCubed$)
    $inliers \leftarrow$ Find-Inliers($cloud, coefficients$, CYLINDER)
    if Size($inliers$) $\leq$ THRESHOLD then
        return
    end if
    $cylinders \leftarrow cylinders \bigcup$ Extract-Inliers($cloud, inliers$, CYLINDER)
    $inliers_cubed \leftarrow$ Find-Inliers($cloudCubed, coefficients$, PLANE)
    Remove-Inliers($cloudCubed, inliers_cubed$, CYLINDER)
until false
```

5.4 Experimentation

This section describes the testing and evaluation of the segmentation performance when combined with the PoG information.

5.4.1 Experimental Setup

Comparisons to existing methods would be difficult, as PoG has not been commonly combined with point cloud manipulation, and segmenting models from unfiltered point clouds runs far too slowly to be feasible. When attempting to compare the method presented in this chapter to different variations, it was difficult to alleviate false positives, due to models matching a random set of points within thresholds allowed, and when thresholds were lowered the models did not contain enough points to be classified correctly.

Speeding up the process by allowing the appropriate down sampling combined with inlier removal would yield similar results to those presented in the next section, under the circumstances that the model was one of the only objects present within the environment. As additional objects were added and the environment became less planar, the PoG information began to show significant performance increases over methods without such information. Primarily in the speed at which models could be segmented, and the removal of most false positives. The results section illustrates the example environments which where assessed during the experimentation.

5.4.2 Results

Figure 5.3 shows the example of the scene in the environment without additional models. There were ten scenes similar to this one assessed, and they all yielded similar results; knowing the PoG did not significantly improve performance.

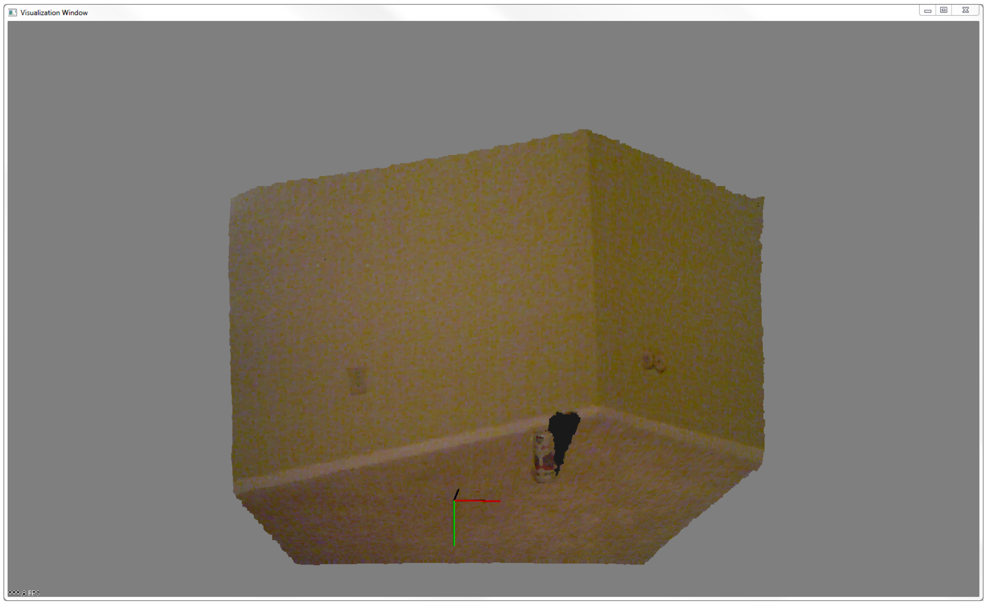


Figure 5.3. Empty Room Scene.

Again, this is due to the fact the majority of points will be removed during the plane filtering process, and both methods are using the same process for plane filtering. Although, due to noise in the sensor certain planar points remain, so it still allowed slight performance increase. Figure 5.4 illustrates the scene once planes have been removed, and figure 5.5 illustrates the scene once the model has been extracted entirely from the scene.

Figure 5.6 shows the example of the scene in a more cluttered environment. There were again ten scenes similar to the one shown. However, due to the fact a large portion of the point cloud remains unfiltered after the plane removal process, the PoG method ran quicker and had less false positives. Figure 5.7 illustrates the scene once planes have been removed.

Due to the random factor of the RANSAC method, each test case was ran a total of ten times and averaged out. Table 5.1 and table 5.2 shows the findings from performing the segmentation with and without the PoG information. Instead of

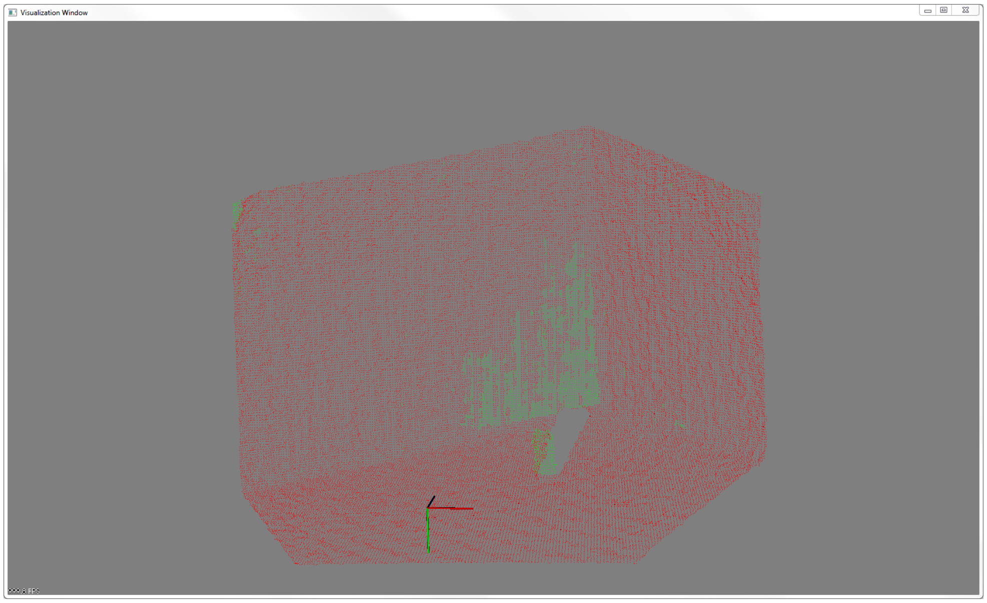


Figure 5.4. Empty Room Planes Filtered.

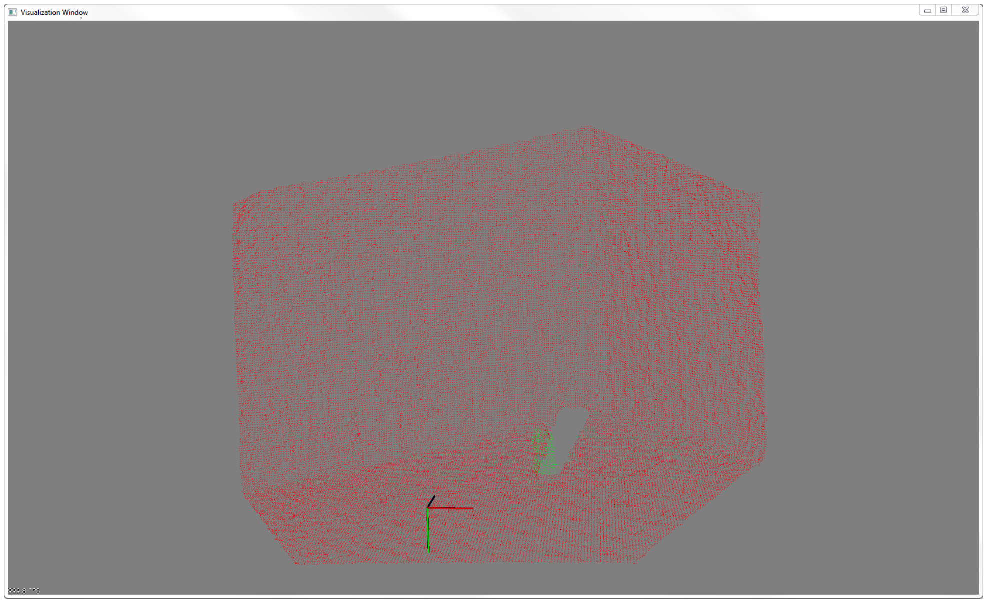


Figure 5.5. Empty Room Model Extracted.

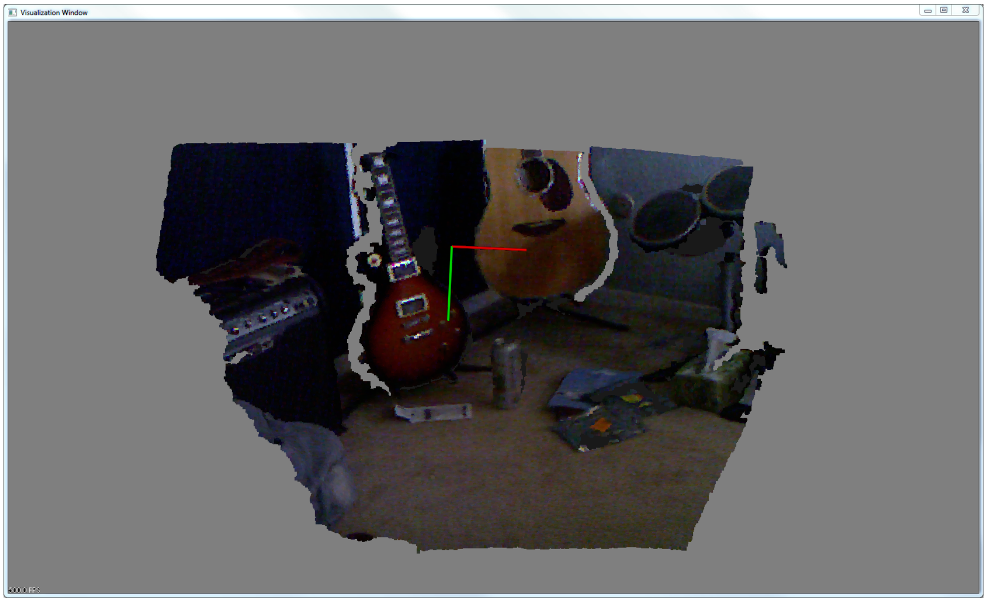


Figure 5.6. Cluttered Room Scene.

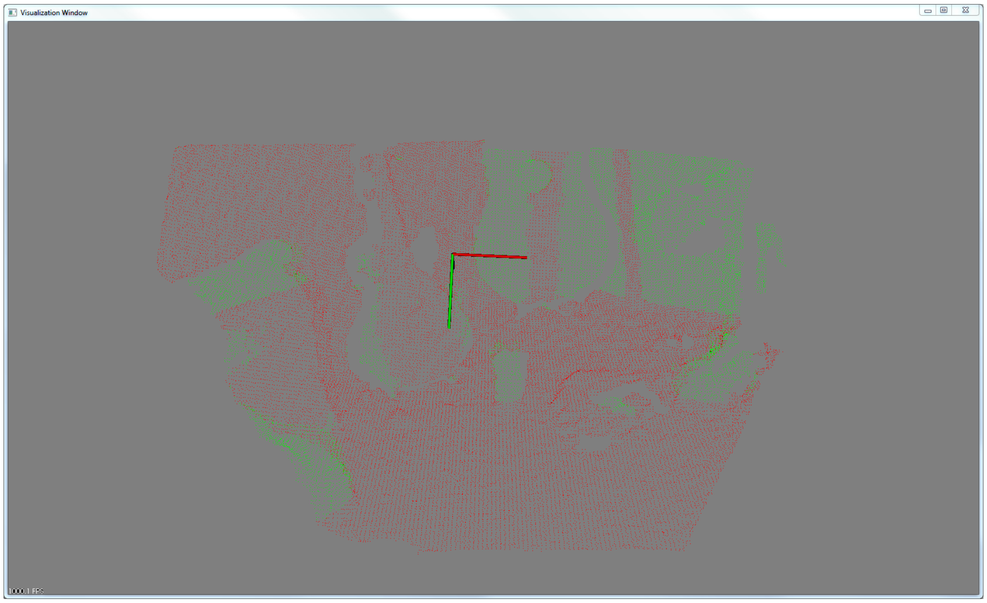


Figure 5.7. Cluttered Room Planes Filtered.

Table 5.1. Comparisons Of Methods For Empty Room

	TIME (Seconds)	TRUE POSITIVE (Average Per Set)	FALSE POSITIVE (Average Per Set)
With PoG	1	0.98	0.00
Without PoG	2	0.95	0.20

Table 5.2. Comparisons Of Methods For Cluttered Room

	TIME (Seconds)	TRUE POSITIVE (Average Per Set)	FALSE POSITIVE (Average Per Set)
With PoG	1	0.90	0.05
Without PoG	10	0.65	1.85

providing the number of points which were a part of the cylinder as the true positives, the model returned must simply match the cylinder model of interest. Similarly for false positives, the models were considered as a whole, instead of separate points. Each dataset had one cylinder model to be segmented, as such the true and false negatives can be calculated from the tables. Segmentation times are linearly related to the number of points in the point cloud, so the main performance gains is the reduction of false positives and increase in true positives.

5.5 Future Work

In future work, the RGB camera will be utilized for SIFT object recognition. Combining the efficient model segmentation presented in this chapter with SIFT object recognition, further classification of the models can be made. With such a method, accurate model segmentation becomes even more important to be able to quickly compare the models of interest with their stored key points for recognition.

CHAPTER 6

CONCLUSION

6.1 Final Thoughts

Presented was a publicly available dataset for standardizing comparisons of head and eye tracking algorithms. During the data collection process, a Vicon Motion Capture System was used for the ground truth of headset positioning. Also presented was a low-cost head and eye tracking solution which improves on eye tracking devices that lack the additional information of head tracking. Such a device allows for more reliable tracking over time, since a user's head position will inevitably shift. Additionally, the work was extended to incorporate a structured light sensor for depth mapping. This allows for models to be efficiently segmented and provides more natural interaction with an environment.

6.2 Future Work

Combining RGB data to perform SIFT object recognition into the projects is currently the priority on the future work. With object recognition, interaction with assistive robots or caretakers will be taken to the next step in the process. Demonstrating eye tracking as a viable user input for controlling of an assistive robot will open up more possibilities of environment interaction. Related to this work, a new 3D dataset needs to be collected containing a user's head position, eye tracking video, and the position in 3D space they are gazing upon.

REFERENCES

[1] D. Li, D. Winfield, and D. J. Parkhurst, “Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches,” in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, June 2005, p. 79.

[2] J. S. Babcock and J. B. Pelz, “Building a lightweight eyetracking headgear,” in *Proceedings of the 2004 symposium on Eye tracking research & applications*, ser. ETRA ’04. New York, NY, USA: ACM, 2004, pp. 109–114. [Online]. Available: http://doi.acm.org/10.1145/968363.968386

[3] D. Li, J. Babcock, and D. J. Parkhurst, “openEyes: a low-cost head-mounted eye-tracking solution,” in *Proceedings of the 2006 symposium on Eye tracking research & applications*, ser. ETRA ’06. New York, NY, USA: ACM, 2006, pp. 95–100. [Online]. Available: http://doi.acm.org/10.1145/1117309.1117350

[4] EyeWriter Initiative, “The EyeWriter,” 2009. [Online]. Available: http://www.eyewriter.org/

[5] C. D. McMurrough, V. Metsis, J. Rich, and F. Makedon, “An eye tracking dataset for point of gaze detection,” in *Proceedings of the Symposium on Eye Tracking Research and Applications*, ser. ETRA ’12. New York, NY, USA: ACM, 2012, pp. 305–308. [Online]. Available: http://doi.acm.org/10.1145/2168556.2168622

[6] C. D. McMurrough, J. Rich, V. Metsis, A. Nguyen, and F. Makedon, “Low-cost head position tracking for gaze point estimation,” in *Proceedings of the 5th*

International Conference on PErvasive Technologies Related to Assistive Environments, ser. PETRA '12, 2012.

[7] Z. Eveland, "Wii Remote IR sensitivity," 2009. [Online]. Available: http://wiicanetouchgraphic.blogspot.com/

[8] J.-Y. Bouguet, "Camera Calibration Toolbox for Matlab," 2007. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib

[9] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, June 1998, pp. 232–237.

[10] M. Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3D models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 4, pp. 322–336, Apr. 2000.

[11] S. M. Munn and J. B. Pelz, "3D point-of-regard, position and head orientation from a portable monocular video-based eye tracker," in *Proceedings of the 2008 symposium on Eye tracking research &#38; applications*, ser. ETRA '08. New York, NY, USA: ACM, 2008, pp. 181–188. [Online]. Available: http://doi.acm.org/10.1145/1344471.1344517

[12] S. Kohlbecher, K. Bartl, S. Bardins, and E. Schneider, "Low-latency combined eye and head tracking system for teleoperating a robotic head in real-time," in *Proceedings of the 2010 Symposium on Eye-Tracking Research &#38; Applications*, ser. ETRA '10. New York, NY, USA: ACM, 2010, pp. 117–120. [Online]. Available: http://doi.acm.org/10.1145/1743666.1743695

[13] J. C. Lee, "Hacking the Nintendo Wii Remote," *IEEE Pervasive Computing*, vol. 7, no. 3, pp. 39–45, July 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1449377.1449423

[14] D. Scherfgen and R. Herpers, “3D tracking using multiple Nintendo Wii Remotes: a simple consumer hardware tracking approach,” in *Proceedings of the 2009 Conference on Future Play on @ GDC Canada*, ser. Future Play ’09. New York, NY, USA: ACM, 2009, pp. 31–32. [Online]. Available: http://doi.acm.org/10.1145/1639601.1639620

[15] A. Boyali and M. Kavakli, “3D and 6 DOF user input platform for computer vision applications and virtual reality,” in *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on*, June 2011, pp. 258–263.

[16] K. S. Arun, T. S. Huang, and S. D. Blostein, “Least-Squares Fitting of Two 3-D Point Sets,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[17] J.-G. Wang and E. Sung, “Study on eye gaze estimation,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 32, no. 3, pp. 332–350, June 2002.

[18] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

BIOGRAPHICAL STATEMENT

Jonathan Walter Rich received his Bachelor of Science in Computer Science in 2010 and his Master of Science in Computer Science in 2012 from the University of Texas at Arlington. He has worked in the Heracleia Human-Centered Computing Laboratory under the guidance of Dr. Fillia Makedon. His research interests include computer vision, algorithms, and software engineering.