Real Time Hardware Vision Processing for a Bionic Eye

by

Horace Josh

Thesis

Submitted by Horace Josh

for fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Primary Supervisor: Prof. Lindsay Kleeman

Monash Vision Group
Department of Electrical and Computer Systems Engineering
Monash University, Australia

November, 2015

To James and Romaine

Contents

List of Tables

List of Figures

Real Time Hardware Vision Processing for a Bionic Eye

Horace Josh
[redacted]
Monash University, Australia, 2015

Supervisor: Prof. Lindsay Kleeman
[redacted]

Abstract

A recent objective in medical bionics research is to develop visual prostheses - devices that could potentially restore the sight of blind individuals. The Monash Vision Group is currently working towards implementing a fully autonomous direct-to-brain vision implant called the Gennaris. Although research in this field is progressing quickly, initial implementations of these devices will be quite naive, offering very basic levels of vision. The vision is anticipated to be binary - that is with black and white pixels - and a low resolution of several hundred pixels. Improving this dramatically is currently improbable, as it would require significant advancement in electrode stimulation technology and substantial research into the complexities of the visual cortex.

This PhD project aims to contribute to the development of the Gennaris and other bionic vision devices, in the hope of improving the quality of life for future patients. More specifically, the key goals of this work have been to develop a portable real time visual prosthesis simulator that is suitably representative of anticipated vision expected of the Gennaris; to investigate the potential capabilities of future patients under this limited vision, and possible image processing techniques that can be used to improve their performance; and to investigate the feasibility of integrating 3D depth sensing and advanced functionality that could aid navigation. An emphasis of this work has been high framerate, low latency and real time operation.

An immersive real time simulator system, based on a Field Programmable Gate Array architecture, has been developed called the Hatpack Simulator. This system improves upon limitations of platforms in existing research. The Hatpack is portable, weighing only 3 kilograms, operates at 60 frames per second with a constant low latency of 17 ms, and is low in power consumption, able to last up to 4 hours on a full charge. Five psychophysics trials have been carried out in order to evaluate the effectiveness of various 2D image processing functions implemented, and the ability of users to complete simple tasks that resemble everyday activities. Results of the psychophysics experiments show that the reduction in user capabilities due to binary and low resolution degrading of vision is significant. This motivates the use of 3D sensing to assist with image representations for bionic vision.

The integration of a second generation Microsoft Kinect depth sensor has been investigated and a new hardware plane fitting algorithm based on least squares has been achieved. An implementation of this has been applied to the detection and highlighting of tables and free floor space in a real time end-to-end system running at 60 frames per second.

Real Time Hardware Vision Processing for a Bionic Eye

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Horace Josh
November 26, 2015

Acknowledgements

First and foremost I would like to thank Lindsay Kleeman for his supervision and guidance over the course of my PhD candidature. He has been a constant source of advice, expertise, encouragement and support, and has been very generous with his time and patience. No matter how disappointed, discouraged, worried or unsure I was about my work or progress, a meeting with Lindsay would always provide me with the necessary reassurance and motivation to carry on and for that I am very grateful. The completion of this thesis would not be possible without him.

I would like to give special thanks to Benedict Yong, whose contributions while a research assistant at Monash Vision Group have been instrumental to the early work of this thesis involving the development of the Hatpack system and conduction of the first three psychophysics experiments. Thanks also to Nicholas Price for help with designing the third psychophysics experiment, and Nicola Andrews from Grey Innovation for assisting with the industrial design/layout of the Hatpack system.

Thanks to Collette Mann for helping to design, conduct and complete a statistical analysis of the fourth and fifth psychophysics experiments. Her expertise in psychophysics and statistical analysis has been very useful for this work and is much appreciated. Thanks also to Dennis Lui for his help with conducting the third experiment. I would also like to thank everyone who participated in the five experiments carried out as part of this work.

I would like to thank all of the members of MVG who have shared their advice, opinions and suggestions with me over the years, and have always been willing participants for the various testing that was performed with the Hatpack. Thanks to Jeanette and Arthur for allowing me to be part of the Monash Vision Group and for facilitating this work. Thank you to Wai Ho Li, for providing the updated arrangement of phosphene mapping. Thanks to fellow Monash Vision Group PhD students Emma and Titus for sharing their own PhD journeys alongside me. They have been a great support network and become great friends of mine.

Finally, I am very thankful for my family and friends. Thanks to my parents who have supported me at home for at least 4 years longer than they should have.

They have provided a stable environment for me to live and study, and have always encouraged me to strive for more in life. Thanks to all of my close friends, especially students from the Electrical and Computer Systems Engineering department and those from the Monash Photography Club and committee who have been a big part of my postgraduate life at Monash and have always been around to encourage me and provide necessary distractions. Special thanks to Sophie, Kaushika and Kevin for generously offering their time to proofread this thesis before submission.

This research has been financially supported by Monash University through an Engineering Research Living Allowance scholarship, and by the Monash Vision Group through a top up scholarship and funding for travel and conference attendance. Monash Vision Group has been funded through the Australian Research Council Research in Bionic Vision Science and Technology Initiative (SR1000006).

Horace Josh

Monash University, Australia
November 2015

Chapter 1

Introduction

Blindness is the condition of poor visual perception, the inability to see light. As defined by the World Health Organisation (WHO), blindness is vision of less than 20/500 or a visual field of less than 10 degrees [72]. The three leading causes of blindness are Cataract - the progressive increase of opacity of the lens in the eye; Glaucoma - the increase of pressure within the eyeball, which causes loss of vision; and Age-Related Macular Degeneration (AMD) - the deterioration of the Macula, the light-sensitive tissue that lines the back of the eye [91]. Worldwide, there are currently about 39 million people who are blind [117]. Since vision is the primary sense in humans, blindness is a major disadvantage to a person's life. Simple everyday tasks, such as walking around the house, having a shower, eating, sending an email, or making a phone call, all become much harder (or near impossible) to perform and can often lead to dangerous consequences. Many years of accustomisation are likely to follow the loss of sight. Fortunately, various treatments and visual aids have been developed to alleviate some of the problems faced by blind individuals.

Although there are many available and commonly used aids for the blind, such as walking canes [90], trained guide dogs [86], braille [18] and other tactile and auditory systems [113], there are still many things that just cannot be done without at least some sort of visual light perception. The Monash Bionic Eye Project is a revolutionary new research project, funded by the Australian Research Council (ARC), that aims to provide a solution to this problem. The project is being undertaken by a multidisciplinary team known as the Monash Vision Group. The solution proposed is a prosthetic implant (visual prosthesis) that would restore sight through electrical stimulation of the primary visual cortex in the brain.

The Monash Vision Group (MVG) was established in 2010, and comprises of Monash University researchers from departments of Engineering, Physiology, Computer Science and Mathematics, as well as clinical and industry partners. The

MVG's proposed device has been named Gennaris, and will consist of a brain implant made up of a number of tiles. The tiles will be powered and controlled wirelessly utilising in total, several hundred stimulating electrodes. The first in-human clinical trial is expected by the end of 2016.

Research into the development of visual prostheses has been carried out since the late 1960s with the seminal work of Brindley and Lewin [19], Dobelle et al. [37], and Bak et al. [11]. These studies confirmed that electrical stimulation of parts of the brain can evoke bright spots of light known as phosphenes in a patient's visual field, and can in fact restore sight. Subsequent studies by Humayun et al. [56] and Veraart et al. [114] have also shown that similar results can be achieved via electrical stimulation of the retina, and the optic nerve as well. The results of these early studies have provided a basis for widespread research into the development of a functional vision implant, that could potentially allow blind people to see again. Currently, there are many research groups working towards developing such devices with various different approaches, including retinal implants [1, 49, 54, 57, 59, 73, 82, 103, 105], optic nerve implants [80, 93, 115], and implants on the primary visual cortex of the brain [32, 43, 48, 70, 81, 96].

Despite substantial research in the field over the past 50 years, the level of achievable vision of implants will be limited at least initially in terms of spatial and temporal resolution. In addition to this, the ability to control the intensity of visual perceptions will also be quite limited as suggested by [37]. Therefore, there is a need to exploit available sensor data in order to best represent the scene viewed by the patient with a limited number of 'pixels'. In order to do this however, simulation platforms called bionic vision simulators are required and innovative processing algorithms need to be developed.

Bionic vision simulators provide normally sighted individuals the ability to view the environment in a form that mimics the low vision of a visual prosthesis. A typical simulator comprises a camera to capture the scene, processing hardware to convert the camera frames into a low resolution form, and a display device to convey the information to the user. A number of simulator systems and processing techniques have been implemented and tested over the past 10-15 years [24, 27, 34, 46, 51]. However, many of these make use of image processing libraries implemented on a computer, and may have significant limitations: such as large size and power consumption; inability to be worn or moved around easily; simplified representations of visual perceptions; frame rate or latency issues. Advancements in high density Field Programmable Gate Array (FPGA) technology may provide alternative solutions to these problems.

FPGAs are integrated circuits that are designed to be reconfigured by the end-user in practice. They contain large arrays of reconfigurable logic, memory and interconnect circuitry that allow the development of complicated application specific circuits. The reconfigurability of FPGAs allows much flexibility in designs. Complex logic functions can be implemented many times in a single device and performed simultaneously. This high level of parallelism often allows for significant speed-up of performance compared to serial implementations on a CPU. This is especially useful when implementing image processing algorithms [9] which usually involve many identical and independent operations that could be performed simultaneously. It is often possible to achieve the same functionality on an FPGA as on a CPU based system with much less hardware, as a result the power consumption of the FPGA design will be significantly lower. FPGAs also offer the ability to export a design to an Application Specific Integrated Circuit (ASIC), for even more speed and reduction in power consumption.

The progression of computer microprocessor technology in recent years has meant that more processing power is able to be achieved with less hardware and reduced power consumption. As a result, recently developed simulator systems have managed to address some of the limitations mentioned earlier through the use of high performance laptop computers. This has allowed for lightweight implementations that can be worn and achieve sufficient real time performance. One common limitation is still quite prevalent though - the assumption and use of purely visual sensing (with only a single camera). This may be a valid assumption as early bionic eye systems will only use a single camera as the sole input to the system. However, the addition of 3D sensing, such as Microsoft Kinect v2 sensor, as well as other non-visual sensing has the potential to provide a much richer collection of data. This would allow a variety of advanced functions such as object detection, path detection, plane tracking, skeleton tracking, mapping and localisation to be implemented and explored in an attempt to complement the low-level functionality of a bionic eye implant. This thesis will show that it is both desirable and that there is potential to enhance the poor phosphene images of a bionic eye with the use of 3D depth sensing.

Through the development of a fast hardware-based real time bionic vision simulator, the implementation of various processing functions, the evaluation of the system through psychophysics testing, and the integration of 3D sensing and plane detection, the research completed as part of this thesis aims to provide better quality of life for bionic vision patients.

1.1 Motivations, Goals and Challenges

Visual prostheses require the implantation of electrodes into some part of the visual pathway. Electrical current is injected via these electrodes to induce a visual response, usually a spot of light referred to as a phosphene. The retina in the back of eye and the visual cortex of the brain are the two most common targets of research groups that are currently working on prosthetic vision implants. The surface area available for device implantation in these locations is very limited, with the central part of the retina (fovea) having on average an area of 1 mm^2 and the equivalent region in the visual cortex on average an area of 500 mm^2 [70]. This means that electrodes would need to be very small and densely packed in order for a bionic vision device to achieve 'high resolution'. It has been shown however, that it is not the size of an electrode that limits the spatial resolution, rather how closely the electrodes can be spaced before the effects of a particular electrode are no longer distinguishable or interfere with another [14]. It is unlikely that the number of achievable foveal stimulation points will surpass several hundred within the near future. Although there have been reports of the elicitation of coloured phosphenes in some stimulation studies [11, 37, 54, 96], the control of colour has yet to have been achieved. Limited control of phosphene intensity or brightness has been reported by various groups [11, 19, 37, 96] however the number of achievable distinguishable levels has not been consistent. The phenomena of depth perception is part of a higher level of brain function [8] and is unlikely to be achieved through electrode stimulation of the retina or visual cortex. Given all of these research findings, it is anticipated that early implants will be:

- Highly limited in spatial and temporal resolution
- Restricted to phosphenes of uncontrolled colour
- Have limited control of intensity
- Unable to provide any information about depth

The Monash Vision Group has chosen the visual cortex approach for their prosthesis [70]. The visual cortex provides a number of benefits over other parts of the visual pathway. The surface area is a lot larger, therefore more electrodes can be implanted giving higher effective resolution. The visual cortex is further along the visual pathway and so does not require any of the preceding sections such as the retina or the optic nerve to be active. This means that a cortical prosthesis could provide a potential cure for up to 80% of people with blindness. An implant on the brain will not have to withstand as much movement as one in the eye. There are also caveats associated with cortical implants. As the implant is required to penetrate the surface of the brain, it is arguably more invasive than other methods. The size of the visual cortex can vary greatly for different people. Implants on the cortex will be subject to the effects of visuotopic mapping, a non-linear warping of visual elicitations, which is described in detail in Chapter 2.

A typical bionic vision system will make use of a digital camera to provide input to the stimulating implant. These days even the most basic digital camera is capable of capturing images at resolutions exceeding hundreds of thousands of pixels. This is in stark contrast to the expected capabilities of preliminary vision implants with several hundred electrodes. Suitable processing techniques are needed in order to most effectively convey the high level of information in the camera image at the very low resolution of the implant. Evaluation of such processing implementations would ideally be carried out through psychophysical testing of actual implanted patients. However, this is impractical with the current progress of research in the field. As an alternative, many research groups have turned to the use of simulated prosthetic vision for the purposes of design, testing and evaluation of potential vision processing functions. Many studies have been done in the field of simulated prosthetic vision. Most involve the use of some sort of computer simulation of expected prosthetic vision. As mentioned earlier though limitations exist in the systems and there is a need to solve these issues in order to better simulate prosthetic vision and explore a wider range of situations.

The goals of this work stem from the goals of the Monash Bionic Eye Project. As such, the main aim of this work has been to contribute the development of the Gennaris device, and potentially improve the quality of life for future patients. The work however can be used to guide the development of other devices as well and provide options for implementation in future revisions of these devices. The more specific goals of this work can broken down into the following three sections.

Development of a Real Time Simulation Platform

A simulator needed to be developed that would simulate the anticipated level of vision that the Monash Bionic Eye device would provide. In order to handle the expectations of the general public, the simulator needed to be suitably representative of the first in human device characteristics. It would need to provide a suitable platform for the implementation, testing and evaluation of potential image processing functions. Finally, the system should improve upon the systems developed and used by other research groups.

Psychophysics

A key goal of this work was to use the simulator system to carry out psychophysics tests to evaluate the implemented functions and potential capabilities of bionic vision patients. This would guide the development of further revisions of the implant and other implants, as well as provide a basis for more formal clinical trials and larger scale testing that could be carried out.

Advanced Sensing

A subsequent but important goal for this work is the exploration of more sophisticated sensing technology and more advanced functionality that could aid a user in navigation. This would allow for better use of the variety of information available in a scene.

1.2 Thesis Contributions

The work completed as part of this thesis has resulted in three main contributions which span the fields of bionic vision, simulated prosthetic vision, psychophysics, real time hardware based image processing and computer vision, and 3D sensing. The work has been presented in a number of publications including three conference papers, a book chapter and has been included in a submitted journal paper. The following outlines the contributions of this work and publications arising from each.

Hatpack Simulation System

A simulation and psychophysical test platform has been developed. The system builds upon the shortcomings of simulators in other published works at the time. It is wearable and portable, operates in real time at camera framerate with low latency, and is fully implemented in hardware using an FPGA development board. The Hatpack incorporates the Monopole model, which mimics the effect of visuotopic mapping. Various image processing functions have been implemented into the system, based on the expectations of the Monash Bionic Eye device at the time.

The work involving the development of the Hatpack System and the implemented functions was published in the following peer reviewed conference paper.

- H. Josh, B. Yong, L. Kleeman. A Real-time FPGA-based Vision System for a Bionic Eye. In *Proceedings of Australasian Conference on Robotics and Automation (ACRA)*, Melbourne, Australia, 2011.

Psychophysics Experiments with Hatpack

Five different psychophysics experiments have been carried out using the Hatpack with normally sighted individuals. The experiments tested the abilities of participants to complete everyday tasks such as navigation, object manipulation, shape discrimination and contrast discrimination under the simulated low vision restrictions of a bionic eye. The tests also allowed for the comparison and evaluation of various processing functions that have been implemented.

The results of the psychophysics experiments have been published in the papers listed below. The first paper is a peer reviewed conference paper and it won the prize of **Best Student Paper**. The second is a revised edition of this paper which was published as a book chapter. The final is a peer reviewed conference paper containing results of the fourth experiment.

- H. Josh, B. Yong, L. Kleeman. Mobile, Real-Time Simulator for a Cortical Visual Prosthesis. In *Proceedings of the International Conference on Biomedical Electronics and Devices*, Villamoura, Portugal, 2012.
- H. Josh, B. Yong, L. Kleeman. A Real-Time and Portable Bionic Eye Simulator. In *Biomedical Engineering Systems and Technologies* (pp. 51-67), 2013.
- H. Josh, C. Mann, L. Kleeman, D. Lui. Psychophysics Testing of Bionic Vision Image Processing Algorithms Using an FPGA Hatpack. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Melbourne, Australia, 2013.

Hardware Plane Fitting Implementation and Applications

A plane fitting algorithm has been developed, based on the least squares method. The algorithm has been reduced to a form containing simple arithmetic and implemented on an FPGA as part of a real time system. This system incorporates 3D sensing via a depth sensor and applies the plane fitting implementation to the detection of floor and tabletop planes. An error analysis of the sensor and plane fitting algorithm has been performed to specify logic precision and thresholds for the system.

This work and the results gained have been included in the following journal paper that has been recently submitted.

- H. Josh, L. Kleeman. A Novel Hardware Plane Fitting Implementation and Applications for Bionic Vision. **Submitted to** *Machine Vision and Applications*, 2015.

Declaration of Collaborative Contributions

Since the work in this thesis is part of a highly collaborative team-based project, it is only natural that there has been some work completed with the help of colleagues. The following is a list of contributions made by others to specific sections of the research.

- **Benedict Yong** – helped with the physical construction of the initial Hatpack system. He also contributed to the visuotopic mapping implementation and the conduction of tests and analysis of results for the first two conference papers.
- **Dr. Dennis Lui** – helped with the conduction of the psychophysics testing for the third conference paper.
- **Dr. Collette Mann** – helped with the conduction of the testing and analysis of results for the fourth and fifth psychophysics experiments.
- **Associate Professor Lindsay Kleeman** – as supervisor provided various supportive input throughout the course of this work and designed the hardware implementation of the Otsu's thresholding method that is outlined briefly in Chapter 2.

1.3 Organisation of Thesis

Chapter 2: The Hatpack Simulator

The introduction of this chapter reviews briefly the history of visual prosthesis research. An overview of the Monash Bionic Eye is given since the work presented in this thesis is part of this project and has been based on the proposed architecture of this device. The main content of this chapter involves a detailed overview of the components and physical construction of a real time simulator system that has been developed as part of this work - the Hatpack Simulator. It also covers the various functions and algorithms developed and how they have been implemented in hardware. This chapter also includes a detailed discussion of the phenomenon of visuotopic mapping which is important to consider given the cortical nature of the Monash Bionic Eye implant.

Chapter 3: Psychophysics Testing with Hatpack Simulator

Five different experiments have been carried out with the Hatpack system in the form of psychophysics tests using normally sighted participants. The tests comprise a navigational task, an object manipulation task, a dynamic scene interaction task, a shape and contrast discrimination task and a sock sorting task. This chapter gives a detailed description of the experiments, the experimental setup and the results obtained. An analysis of the results and how they guided subsequent research is also presented.

Chapter 4: Least Squares Plane Fitting and Depth Sensor Error Characterisation

Results and observations of the Hatpack work indicated a need for advanced sensing approaches. Extension of this work into the 3D sensing domain prompted the application of plane fitting to object detection. This chapter covers in detail the development of a simplified least squares plane fitting algorithm that is suitable for implementation in hardware on an FPGA system. An analysis of sensor error and error propagation is covered in order to specify the appropriate logic precision and thresholds required for the hardware implementations that are covered in Chapter 5.

Chapter 5: Hardware Implementation of Plane Fitting and Applications to Bionic Vision

Chapter 5 gives a detailed explanation of the hardware implementation of the plane fitting algorithm developed in Chapter 4. The integration and performance testing of a real time end-to-end system is covered. The application of the plane fitting implementation to bionic vision related problems such as free floor space detection and simple object recognition is also presented.

Appendices

Appendix A provides detailed derivations for the error model equations (4.38) to (4.41) that have been discussed in Chapter 4.

Appendix B provides a list of web links to example video footage of the Hatpack simulator and a video of the visual results of the end-to-end system from Chapter 5. The videos in this Appendix are also made available on the CD-ROM that is attached to this thesis.

Chapter 2

Hatpack Simulator System

The research field of visual prosthetics has become increasingly active in recent years, with a number of research groups around the world currently working towards the development of an implant that could restore some level of vision to blind individuals - a bionic eye. Bionic vision devices will allow patients to see spots of light that will be brought about through the electrical stimulation of some part of the human visual pathway, most commonly either the retina in the eye or the visual cortex of the brain. The Monash Vision Group is currently working on a device based on cortical stimulation, and human trials are planned in 2016. Early implants will be limited, both in spatial and temporal resolution. To most effectively convey information to a patient given these restrictions, innovative image processing techniques and exploitation of sensor data are required. Most research groups in this field are still in the preclinical or clinical trial stages though, and so a need has arisen for simulation platforms that would allow early development and testing of potential functions and capabilities of users.

In this chapter the development of a real time simulation and psychophysical test platform is presented. The simulator has been designed in line with the development of the Monash Vision Group's Gennaris device and seeks to address the limitations of other simulators in literature. The development of this system has been based loosely on a simplified version that was completed as a Bachelor of Engineering final year project at Monash University, in conjunction with Benedict Yong. A complete redesign of the system achieved in that work was needed for the work of this thesis. The whole physical structure was redesigned including the use of different camera, FPGA and display hardware, and a new mapping and sampling technique has been employed in place of a simple square lattice grid used in the final year project. The functions implemented in this work are all newly developed, however basic versions of luminance thresholding, and Gaussian phosphene modelling were implemented in the undergraduate work and have been improved for this work. Finally the overall performance achieved is significantly higher and the simulator itself is much more sturdy and comfortable. Part of the work presented in this chapter has been completed in collaboration with Benedict Yong who was employed as a research assistant for the Monash Vision Group at the time. He contributed to the physical construction and layout of the Hatpack and also helped with the visuotopic mapping work.

2.1 Introduction

Early experiments by Foerster [45] in 1929 showed that electrical stimulation of the visual cortex of the brain can cause the sensation of light. The sensation experienced by his patient was a spot of light that was directly in front and motionless. This is one the first documentations of visual percepts being elicited by electrical stimulation of the brain and similar results were shown by Krause and Schum in 1931 [63], Penfield et al. in 1952 and 1954 [84, 85], and Button and Putnam in 1962 [22]. Encouraged by this seminal work, Brindley and Lewin conducted their own study in 1968 [19], that has become widely regarded as the beginning of visual prosthesis research. Their work involved the implantation of a prototype vision prosthesis, comprising non-penetrating electrodes onto the surface of a blind woman's brain. They found that sending a uniform train of electrical pulses through the electrodes, elicited small bright spots of white light ("phosphenes") in the visual field of the patient. The most common types of phosphenes were described by the patient as "like a star in the sky" or "the size of a grain of sago at arm's length" [19]. Phosphenes were found to be gaze-contingent, meaning that during voluntary eye movements, the phosphenes would move along with the eyes. Supporting results were later found by Dobelle et al. in 1974 and 1976 [37, 38]. In 1990 Bak et al. investigated the use of intracortical penetrating electrodes and found that stimulation thresholds were 10-100 times lower than those used in the work of Brindley and Lewin, or Dobelle [11]. It was also found that spacing of electrodes could be reduced to 700 μm and that phosphenes evoked from microstimulation were much more stable and predictable. Further work completed by Humayun et al. and Veraart et al. [56, 114] showed that stimulation of the visual cortex was not the only way to elicit phosphenes, and that stimulating the retina or the optic nerve would also give the same effect. In fact it was found that electrical stimulation of any part of the human visual pathway would result in phosphene elicitation. These early studies provided a basis for widespread research into the development of functional visual prostheses.

A visual prosthesis, also often referred to as a 'bionic eye', is an implantable biomedical device that aims to restore vision to the blind. The core component of these devices is an array of electrodes, driven by specialised electronics. The electrodes inject electrical current into a particular section of the patient's visual pathway in order to generate an ‘image’ in the visual field. The term visual pathway refers to the path that signals take from the retina in the eye where they are generated, to the primary visual cortex at the back of the brain. Light that is incident on photoreceptors in the retina, a layer of cells at the back of the eye, results in the generation of signals. These signals are passed through the optic nerve and Lateral Geniculate Nucleus (LGN) before arriving at the primary visual cortex (V1), which is at the back of the brain. From V1, signals diverge to subsequent levels of visual cortex where higher level processing takes place. In a blind individual, parts of the visual pathway may not function. Therefore, visual signals do not reach the visual cortex. A successful prosthesis would bypass these inoperative sections in order to deliver signals to V1.

In 2009, the Australian Research Council (ARC) funded a new collaborative research initiative to develop a functional visual prosthesis. Two proposals were accepted for this initiative, one of which was by a Monash University led team of researchers, now known as the Monash Vision Group (MVG) [77]. Established in 2010, the MVG comprises Monash University researchers from the fields of Engineering, Physiology, Computer Science and Mathematics, as well as clinical and industry partners such as Grey Innovation, MiniFab and the Alfred Hospital in Melbourne. The group aims to develop a prosthetic vision device centred on a cortical implant, making use of several hundred electrodes. The device, named ‘Gennaris’ [70], is in the pre-clinical stage and human trials are expected within a year. MVG’s vision system design includes a miniature digital video camera, which will be worn by the user on a custom designed headgear. The high-resolution images captured by the camera are processed by a small vision processor unit, about the size, shape and processing power of a modern smart-phone (see Figure 2.1). The most useful information from the camera images will be extracted and converted to a lower resolution form suitable for transmission to the brain. These signals are then transmitted via wireless interface to a number of stimulation tiles that have been surgically implanted in the primary visual cortex of the brain. Each tile comprises 43 microelectrodes, electrode driver circuitry and a wireless receiver coil. For the first-in-human trial, up to 11 tiles may be implanted, allowing the elicitation of up to 473 phosphenes [78].

As research grows in this new area of bionics, there is a great need for simulation or visualisation of the possible results of such an implant. Bionic eye simulators serve as good platforms for researchers to investigate the effectiveness of implemented algorithms, tune parameters, and realise the importance of certain parameters prior to actual clinical trials. The simulators would be used most in psychophysical trials trials involving normally sighted individuals attempting to complete tasks with the limited vision provided by a simulator. However, the simulators would also be of use to the general public for educational purposes and to handle the expectations of

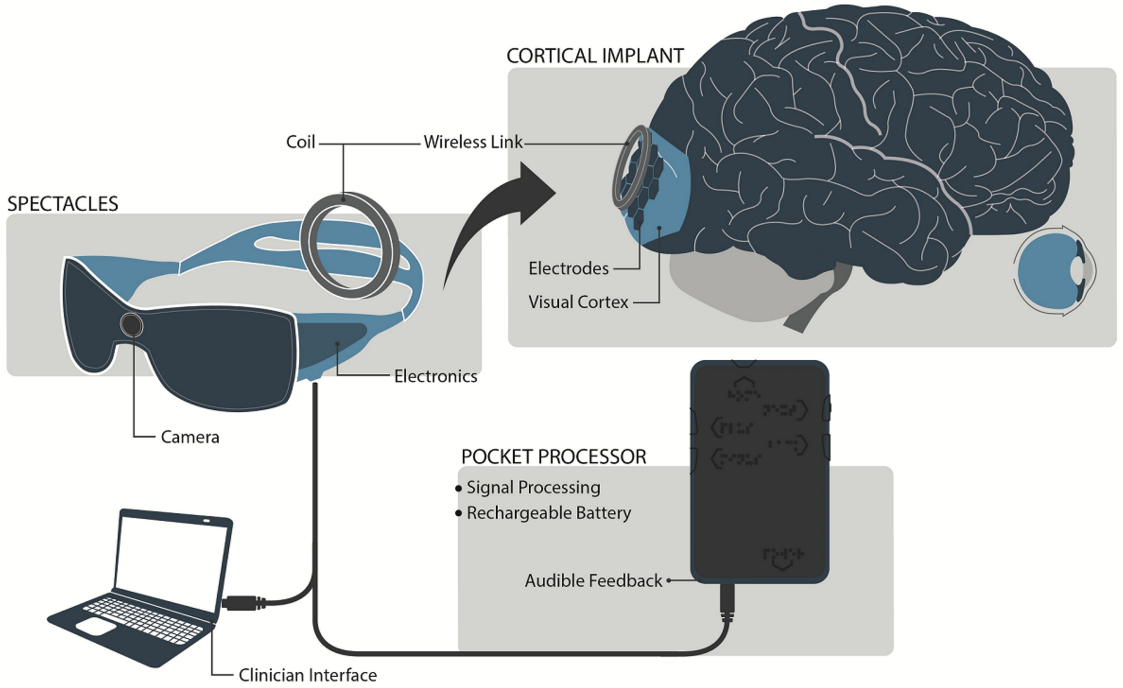

Figure 2.1: System overview of the Gennaris vision device. Image courtesy of Monash Vision Group. Used with permission.

families and friends of potential patients. Input to the system is in the form of an image or image stream. This image data goes through processing that transforms it into a representation, that attempts to mimic the elicitation of phosphenes through electrode stimulation. The processed image data is then stored and/or displayed on a screen for viewing by the user.

A number of visual prosthesis simulators have been developed prior to the beginning of this work. The earliest of which was used in the work of Cha et al. from the University of Utah [24–26]. Their simulator was made up of a small video camera and miniature monochrome CRT monitor that was mounted in a pair of ski goggles, powered by a battery belt worn by the user. A number of perforated masks were used in front of the screens to give the effect of phosphenated vision. Researchers from John Hopkins University School of Medicine used digital PC video cameras, head mounted displays and desktop computers for their own simulation studies [34, 35, 51, 111]. Variations of components were used for each of the studies, however the underlying system involved a camera to capture images or video, a computer that would process the camera input into a low resolution form and a display device to present it to the user. Only one of these systems was portable via use of a laptop computer in a backpack [33, 34]. Similar and more recent work by other research groups can be found in [27, 42, 46, 101, 112, 120].

Despite the number of simulators that have been used in prior work, there are some significant limitations that arise in their implementations. The majority of these simulators perform their image processing on a computer using image processing libraries, and so are often limited to use within an area close to a stationary computer. Depending on the complexity of processing and the available processing power of the equipment in use, these systems may suffer from latency and frame rate issues. Simulations for visual prostheses have often been generalised and as a result visuotopic mapping has been overlooked or simplified models have been used. Visuotopic mapping is the mapping of electrode placement on the visual cortex to elicitation of phosphenes in the visual field and is an important consideration for cortical implants. A number of challenges stand in the way of producing a simulator system that could provide a solution to these limitations. Developing a system that is very portable requires lighter hardware that is able to be powered from a battery. Usually this would come at the cost of reduced computing performance. This however, would hinder the ability to achieve a high framerate and low latency operation which is another limitation that needs solving. This work takes an alternative to conventional computer-based processing. The solution adopted here is to use a Field Programmable Gate Array (FPGA). The simulator used in the work by Srivastava et al. [101] makes use of FPGA technology and this simulator is the closest implementation to that of the work presented in this chapter. Their simulator uses a digital camera to capture images, an FPGA development board to implement an image processing filter that incorporates an expected cortical phosphene mapping, and computer to incorporate eye-tracking data and display the final output. The main limitation of this system however, is the fact that it is a large, fixed system. It is not wearable, cannot be moved around easily, and the use of a computer means that power consumption will be relatively higher. As will be discussed later in this chapter, these limitations are not present in the Hatpack simulator system. Earlier work by Srivastava et al. [102] involving a prototype bionic vision stimulation device achieved a more compact FPGA based system incorporating visuotopic mapping. However this work did not have as much image processing functionality as the simulator in their later work and was used for actual electrode stimulation studies with non human primates as opposed to simulations for psychophysical studies.

Field Programmable Gate Array (FPGA) systems are an alternative architecture that have gained popularity with image processing applications in recent years [9]. An FPGA is an integrated circuit that can be reconfigured after it is manufactured. In fact it is designed to be configured and reconfigured by a customer or designer while in use out in the field. FPGAs are usually made up of a vast array of programmable logic blocks, reconfigurable routing channels and I/O pads. The logic blocks comprise a number of smaller elements called logic cells which can be configured to perform a number of different combinational logic functions or act as simple logic gates (AND, OR, XOR, etc.). Logic blocks typically also contain storage or memory elements such as flip-flops. Multiple logic blocks can be connected together via routing channels which are reconfigurable interconnects. The channels can connect to I/O pads in order to interface on-chip logic implementations with external devices and communication buses. FPGAs often also contain hard or non-configurable logic blocks for specific functions such as multiplier logic, digital signal processing functions, communication interfaces and many more.

The reconfigurability of FPGAs allows for a lot of flexibility in designs. Furthermore, parallelism of operations can be exploited since complex logic functions can be implemented many multiples of times in a single device and performed simultaneously. This is especially advantageous for image processing and computer vision applications which involve performing identical operations on a large number of independent data points. Since FPGA designs are able to be tailored to the specific task at hand, implementations can often be run at much lower clock speeds and with much less hardware than the equivalent computer based implementation. This in turn has favourable implications for power consumption. There are a number of challenges presented with FPGA design however. There is a limited amount of logic, memory and routing resources available on an FPGA device, and this needs to be carefully managed by the designer, while still satisfying timing constraints and requirements. Since there are often multiple ways to solve a task with an FPGA design, finding the method that provides best performance can sometimes be difficult and result in a longer development cycle. Furthermore, as designs utilise higher percentages of available resources, the job of the compiler becomes much harder and therefore compilation times can increase to many hours.

In the sections that follow, the development of a simulator system based on FPGA technology is presented.

2.2 System Overview

In order to develop a bionic eye simulator, a base system has been constructed to provide a solid platform for the implementation of various functions that attempt to mimic the level of vision provided by a cortical visual prosthesis. The system needs to be wearable, mobile, and operate in real time.

It was decided that an FPGA-based architecture will be used. This is due to the possibility of highly parallel implementations offering very low latency functionality, ability to connect multiple sensors and peripherals, low power consumption, and small form factor for maximum mobility. An Altera DE2-115 FPGA development board from Terasic has been chosen (shown in Figure 2.2). This board is ideal for our implementation as it is relatively inexpensive, has a low power Altera Cyclone

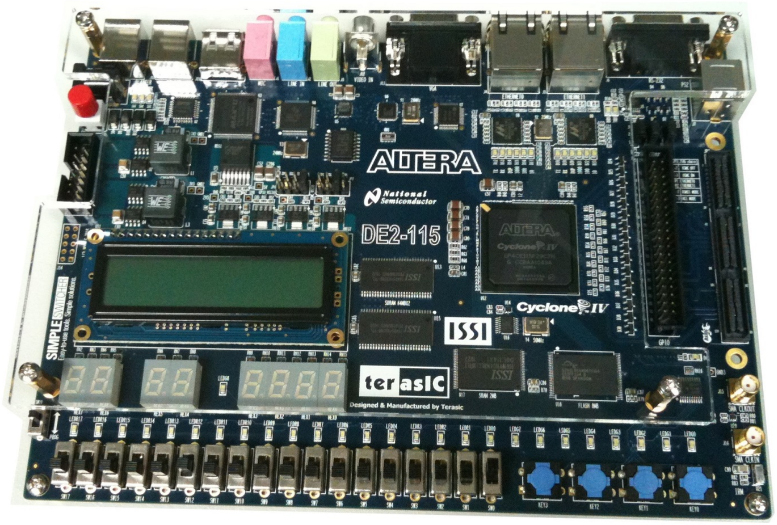

Figure 2.2: DE2-115 FPGA development board

Figure 2.3: Sparkfun Electronics CM-26N/P CMOS camera

IV FPGA chip with a sufficiently large number of available logic elements and on-chip memory, and many external peripherals and I/O pins that allow for connection to devices such as cameras and displays.

Visual input to the system is provided by a video camera. A suitable camera would have low power consumption, be relatively inexpensive and offer a moderate to high spatial resolution. The chosen camera is a Sparkfun Electronics CM-26N/P CMOS camera (Figure 2.3). This camera has a capture resolution of 640 by 480 pixels, frame rate of 59.94Hz, a viewing angle of 70°, is small in size, and has a simple 3-wire data/power connection.

For output of the processed video stream, an immersive form is required in order to give the best simulation. It was decided that a suitable device would be a head-mounted display (HMD) also commonly referred to as virtual reality goggles. These goggles need to be of comparable resolution to the video camera input, have an interface compatible with the FPGA board, and be lightweight, low cost, and comfortable to wear. A pair of Vuzix iWear VR920 (Figure 2.4) goggles has been chosen as a suitable solution. These goggles are USB powered and take a VGA

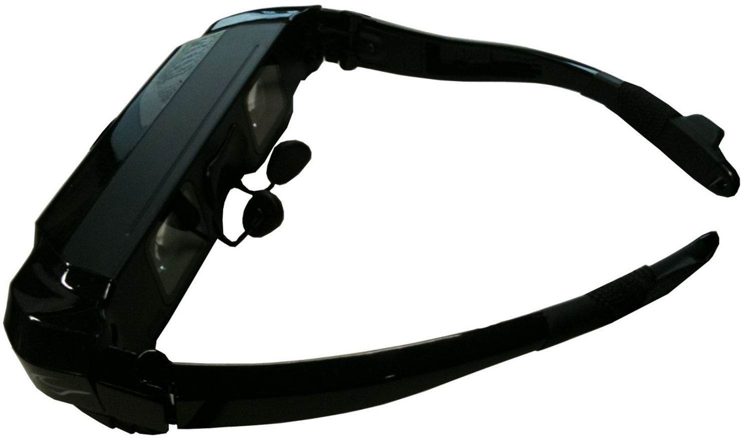

Figure 2.4: Vuzix iWear VR920 virtual reality goggles

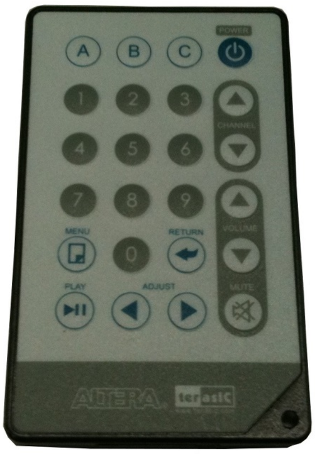

Figure 2.5: Terasic IR remote control

signal as input, meaning they can be connected directly to the VGA output of the FPGA board.

Since there would most likely be a wide range of functions implemented each with its own various parameters, there needs to be a convenient way of controlling/adjusting the system. This control was preferably to be wireless, easily accessible and simple enough for user control. The DE2-115 board has an IR Receiver module and comes with an IR remote control (Figure 2.5) which proved more than adequate for this purpose.

After all the major components of the system were sorted out, a neat, comfortable wearable package needed to be produced. This industrial design/layout work was aided by Nicola Andrews from Grey Innovation. It was proposed that the bulk of the components (processing hardware and 12V lithium ion battery power supply) be mounted within a protective casing that would in turn be placed in some sort of backpack. Cables would then need to run from the components in the backpack to the camera and HMD which would be worn by the user. With a focus on comfort and neatness, a slim-fit neoprene laptop bag was chosen to contain a hard plastic laptop casing that would house the components. Port connectors have been embedded into the side of the laptop casing so that power and data cables could connect externally to the camera and goggles, and that an IR Receiver extension cable could be used.

Initially, the video camera module was fastened to the HMD which was then worn by the user. However, it was soon realised that this set-up was neither stable nor comfortable on the user's head. It was also found that a lot of ambient light was leaking through as the goggles did not fully immerse the users' sight. The solution to this problem was to mount the HMD within the frame of a ski mask and to make the lens of the ski mask opaque. The camera would then be mounted on the front of the HMD/Mask combination. This setup proved to be stable and comfortable. Figure 2.6 shows the system and its main components.

2.3 System Implementation

The Hatpack system is implemented using Verilog hardware description language. Figure 2.7 shows a block diagram of the complete implementation of the system. Each block in the diagram represents one or more Verilog modules within the system. Image data is received from the camera as an NTSC analogue signal. This is converted to a YUV 4:2:2 signal and stored in an SDRAM provided on the FPGA development board. The stored data is retrieved from the SDRAM and converted to RGB data which is passed to the processing modules, that have been created as part of this thesis, one pixel at a time. After leaving the processing modules, the data is passed to the VGA controller which handles the signalling for the VGA output port. The IR Receiver block interprets the remote control signals from the IR receiver sensor and generates a function code which is sent to a control logic block that interfaces with all the processing modules in order to toggle modes or configure parameters. In Figure 2.7, the green blocks have been developed as part of this work, other blocks refer to reference designs provided as part of the DE2-115 FPGA development board package.

2.3.1 Visuotopic Mapping

Early physiological research [97, 116] showed that 'points' in the visual field correspond to specific locations on the visual cortex, inferring a 'map' or transfer function between visual field points and the visual cortex. Furthermore, that map is mostly continuous in that neighbouring points in the visual field correspond with neighbouring points on the visual cortex. The map or transfer function which describes

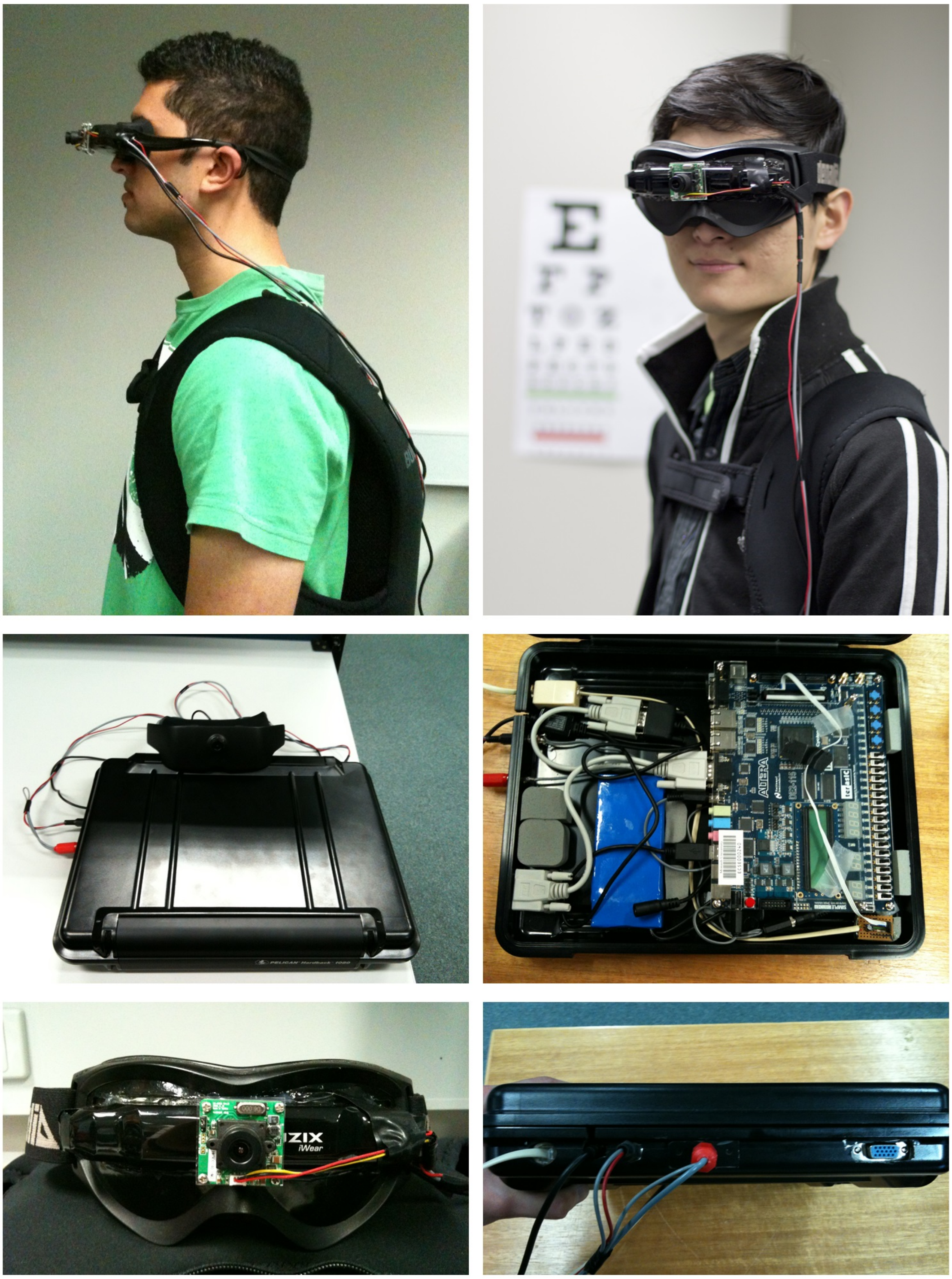

Figure 2.6: Hatpack System

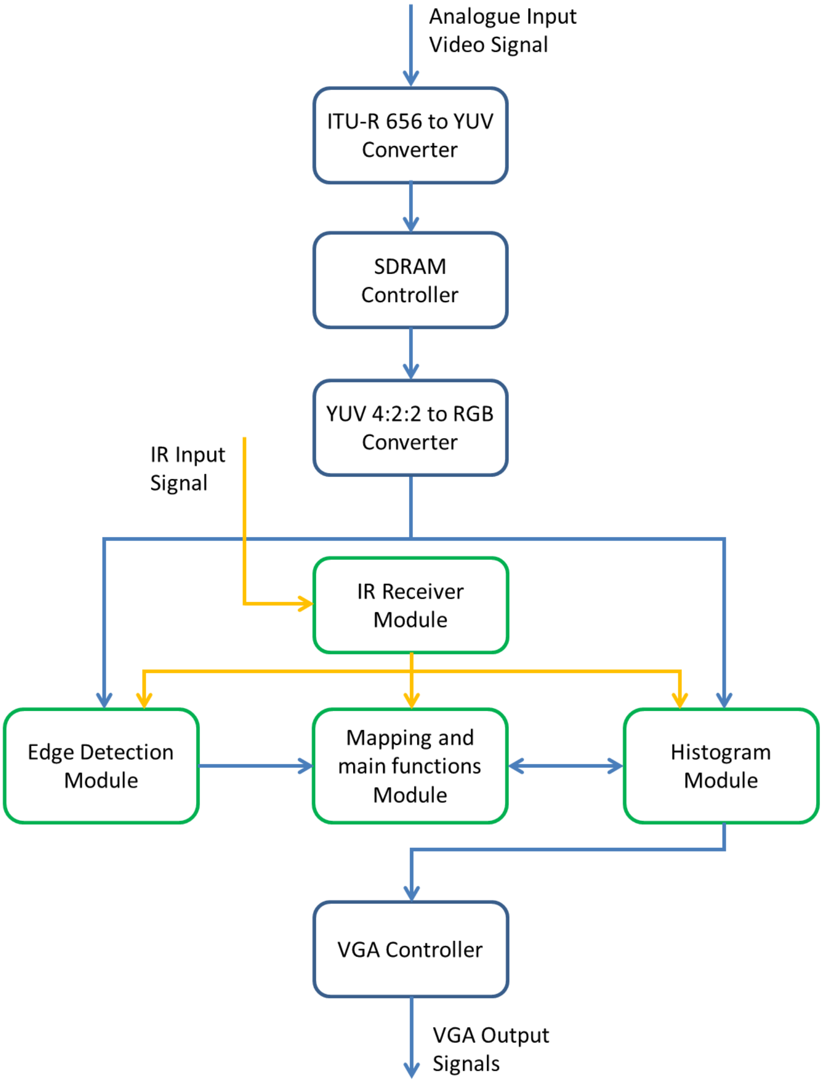

Figure 2.7: Block diagram of Hatpack implementation

the translation of points between the visual cortex to its corresponding points on the visual field is known as the visuotopic map. Due to the physiological non-linear properties of the visual cortex, the visuotopic map is also non-linear and 'distorted'. In humans, the phenomenon known as cortical magnification describes how a small region at the centre of the visual field, known as the fovea, corresponds with a much larger area of the visual cortex [40, 55]. Early work by Schwartz [97] indicated an approximation to the mapping by a 'log-polar' representation, where linear points on the visual cortex correspond to eccentrically logarithmic and angularly linear points in the visual field. The foveal region is represented this way as a dense packing of points in the centre of the visual field which corresponds to a disproportionately larger region on the visual cortex. Also important to note is that the visual cortex is spread over both halves of the brain with the left visual cortex corresponding with the right visual hemifield and vice versa, due to cross-over of the optic nerves [13].

Mathematical models that came from this include the Monopole model (defined from the ‘log-polar’ observations) [88, 94, 97], the Wedge-Dipole model (adds a second parameter to Monopole model to account for curvature in the periphery region of the visual cortex) [12, 88] and more recently the Double-Sech model (adds a shear function to the Wedge-Dipole model to account for changing local isotrophy as well as increasing accuracy of mapping at higher levels of visual cortex V2, V3) [94, 95].

At the time of development of the Hatpack system, the Gennaris design was still in early stages and not much was confirmed about the arrangement of electrodes, or where they would be placed on the brain. The implant was anticipated to consist of a linear array of electrodes, the resulting phosphene pattern would not be linear but rather follow this log-polar mapping. It would be useful and more accurate to model the output visualisation based on a mathematical model of the visuotopic mapping. Since the implant is expected to be placed in the primary visual cortex V1, and closer to the foveal side of the visual cortex, the Monopole model was chosen to model the output visualisation as it was mathematically simpler and still provides reasonable accuracy.

The Monopole equation (2.1) describes the left visual cortex w as a complex function of the right visual hemifield Z_w. $\mathbb{C}$ is the set of complex numbers, and k is a dilation factor constant.

$$w = k\log(Z_w + a) \in \mathbb{C} \tag{2.1}$$

Visual field Z_w can be represented as a complex exponential where r represents the eccentricity and θ represents polar angle.

$$Z_w = re^{i\theta} \in \mathbb{C} \tag{2.2}$$

Rearranging the Monopole equation, describes visual field Z_w as a function of the visual cortex w.

$$Z_w = e^{\left(\frac{w}{k}\right)} - a \in \mathbb{C} \tag{2.3}$$

The electrode array of the implant was assumed to be a linear array placed on the visual cortex closer to the foveal region. The visuotopic map was created using MATLAB and ported over to the FPGA for use as a large lookup table. Approximate values have been used for the Monopole equation parameters, which are reasonably consistent with the various values used in the literature: $k = 15$, $a = 0.7$ [42, 88, 95]. The exact dimensions and intended locations of the implant were still not known at the time, and so the eccentricity and polar angle were limited to an 18 by 18 linear array on the visual cortex that cover the following values on: $r = [10, 40]$,

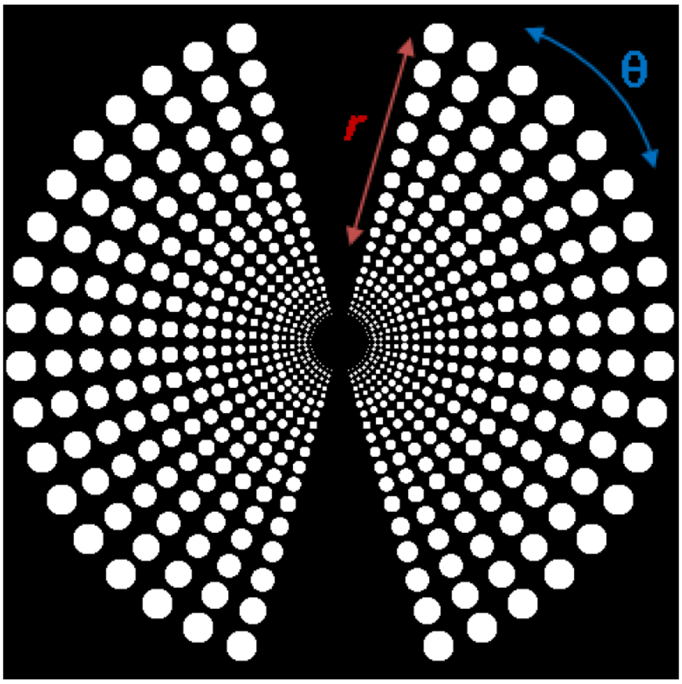

Figure 2.8: Resultant visual field of implemented visuotopic map

$\theta = [-0.8(\frac{\pi}{2}), 0.8(\frac{\pi}{2})]$. This only represents the left visual cortex, corresponding with the right visual hemifield. The 18 by 18 array was duplicated for the right visual cortex, creating another array on the left visual hemifield. This produces a total electrode count of 648. These assumptions were taken to make better use of the limited screen resolution of the head-mounted display while remaining realistic to the 'log-polar' mapping of the visual cortex. However, new maps can be simply regenerated on MATLAB to accommodate any changes to this and implemented into the system, as has been done for the revised mapping covered in Section 2.4.1. The resultant visual field of the implemented map, with all possible phosphenes, is shown in Figure 2.8.

As mentioned earlier, the implementation of the described mapping involves the use of a number of lookup tables (LUTs) and innovative sequential logic that utilises the information provided by these lookup tables. In order to transform the input frame data into the low resolution visuotopically mapped frame, the incoming frame pixels need to be sampled, averaged and stored, ready for display during the next output frame. Figure 2.9 shows a block diagram layout of the mapping implementation (excluding intensity transformations, Gaussian profile phosphene modelling, and dead electrode simulation).

The most resource-expensive part of this implementation (and of the whole Hatpack system) is the v-map lookup table. This LUT stores a 10-bit number for each of its 230400 elements. Each of these elements corresponds to one pixel within the

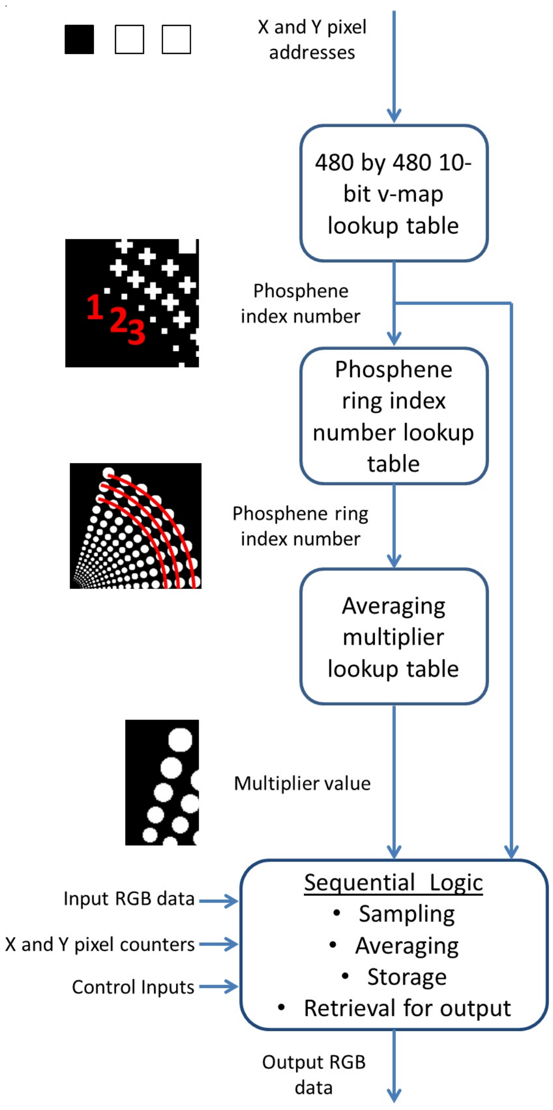

Figure 2.9: Mapping implementation (excluding intensity transformations, Gaussian phosphene modelling and dead electrode simulation)

center 480 by 480 square of pixels in the input frame. The numbers stored in the elements correspond to the index numbers of phosphene that a particular pixel belongs to, ranging from phosphene 1 to phosphene 648. A value of zero is given to elements that correspond to background pixels (pixels that do not belong to a phosphene and will appear black in the output frame). MATLAB scripts have been written and used to generate the arrangement of phosphenes and subsequently produce a memory initialisation file for the v-map lookup table. This innovative method of defining the mapping can be extended to almost any low resolution mapping with a simple redefinition of the LUT using MATLAB. This implementation could ideally be utilised by a system that requires low latency vision mapping.

With the knowledge of the phosphene number of the current pixel, sampling is made possible. The sampling method that is employed in the Hatpack system is an averaging sampler. For each phosphene, the values of all of its pixels are summed in a storage register that corresponds to that phosphene. After all phosphenes have been sampled in this way, the stored sums are averaged. Since division logic is quite expensive in hardware, a multiply and shift implementation is used to approximate the divide. This involves predefining a fraction that has a divisor which is a power of base 2. The numerator of the fraction will be the rounded result of dividing the number of pixels by the base 2 power. Averaging is then achieved by multiplying the stored sum by the numerator and bit shifting to the left by an amount equal to the power of the divisor. Since not all phosphenes are the same size, a lookup table is required to store the multiplier and divisor amounts for each phosphene. The implementation takes advantage of the fact that phosphenes within the same ring are all the same size. This allows for a cheaper implementation, albeit with two separate LUTs (ring number and averaging multiplier). It has also been chosen to use the same divisor for all phosphenes to further simplify the implementation.

After averaging the sampled phosphene pixels, the result is stored in one of 648 10-bit storage registers. Each register corresponds to one phosphene and its contents will be accessed and displayed in the next output frame.

Display pixel data is provided by simply retrieving the value stored in the register that corresponds to the current pixel's phosphene number (provided by the v-map LUT). As discussed in later sections, a slightly more complex method is required when implementing Gaussian phosphene modelling, intensity transformations, and dead electrode simulation. The display pixel data is passed on to a subsequent VGA controller module which handles the interface to the physical VGA port.

2.3.2 Averaging Sampler

Figure 2.10 outlines the averaging sampler implementation. After NTSC decoding, the image stream from the camera is made available one pixel at a time in a sequential fashion. As each pixel arrives at the sampling section of the system, its X and Y pixel count values are compared against the mapping lookup table. This lookup table stores the corresponding phosphene index number for each pixel within the central 480 by 480 window of the full camera view. Pixels not belonging to a phosphene are assigned number zero. Once the phosphene index number is determined, the pixel is sampled by adding to a storage register that corresponds to that particular phosphene index number. This process repeats until all pixels have been sampled. Finally, an average is performed on all of the storage registers according to the number of pixels that are within each phosphene, and the results are stored in a separate set of storage registers.

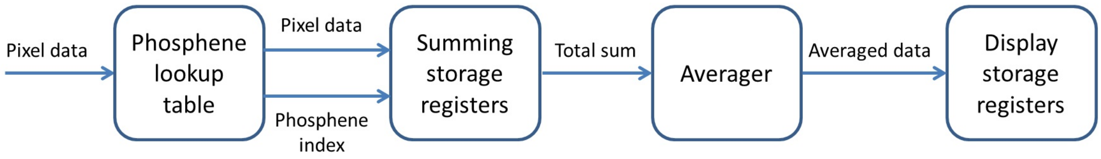

Figure 2.10: Averaging sampler implementation

2.3.3 Luminance Thresholding

Various studies [19, 37, 96] have shown that limited modulation of phosphene brightness is possible using a number of different techniques. However, there is some ambiguity in the possible number of distinguishable brightness levels achievable. The Hatpack system takes an optimistic approach at simulation of this property, having the option to display at 2, 4 or 8 levels of intensity or greyscale. This is achieved by using static thresholds to categorise the averaged samples of pixels (described in Section 2.3.2) into one of 2, 4 or 8 bands of intensity.

Since 10-bit storage registers are used for pixels, the full greyscale intensity range is 0 to 1023. Static threshold values have been defined by dividing this range evenly in order to create bands of intensity for 2, 4 and 8 level modes. Results of 2 and 4-level thresholding are shown in Figure 2.11. It is often difficult to perceive the results of the system in a static image form, therefore it is highly encouraged to view the videos listed in Appendix B and here is a link.

To avoid high frequency oscillation between intensity bands, a hysteresis feature was included. Two threshold values are used to define changes between intensity bands, instead of one value. When a phosphene's intensity is between the two thresholds, no change occurs. Figure 2.12 shows how hysteresis reduces the oscillation problem.

2.3.4 Dynamic Threshold Selection

It became apparent, through general use of the system, that variations of lighting in the environment resulted in poor scene conversion and representation. This was due to the use of fixed intensity values for greyscale thresholding. Although the Hatpack camera employs a built-in automatic gain controller, it is sometimes not enough to compensate for such lighting variations. A number of dynamic threshold selection functions have been developed in order to provide a possible solution.

Manual Threshold Selection

An initial solution to this problem was to allow the user to adjust thresholds via the remote control. The process of readjusting thresholds while wearing the Hatpack

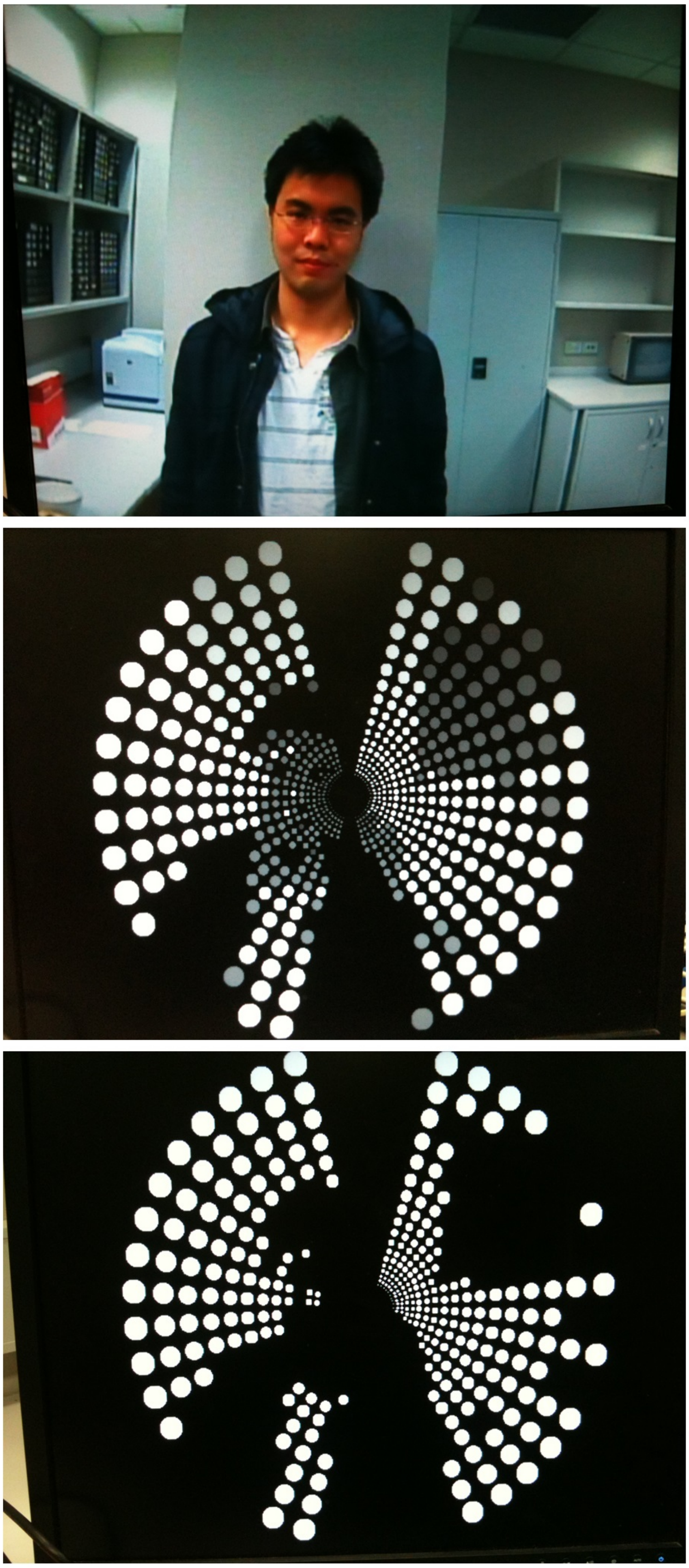

Figure 2.11: Thresholding: full resolution image (top), 4-level image (middle), binary image (bottom)

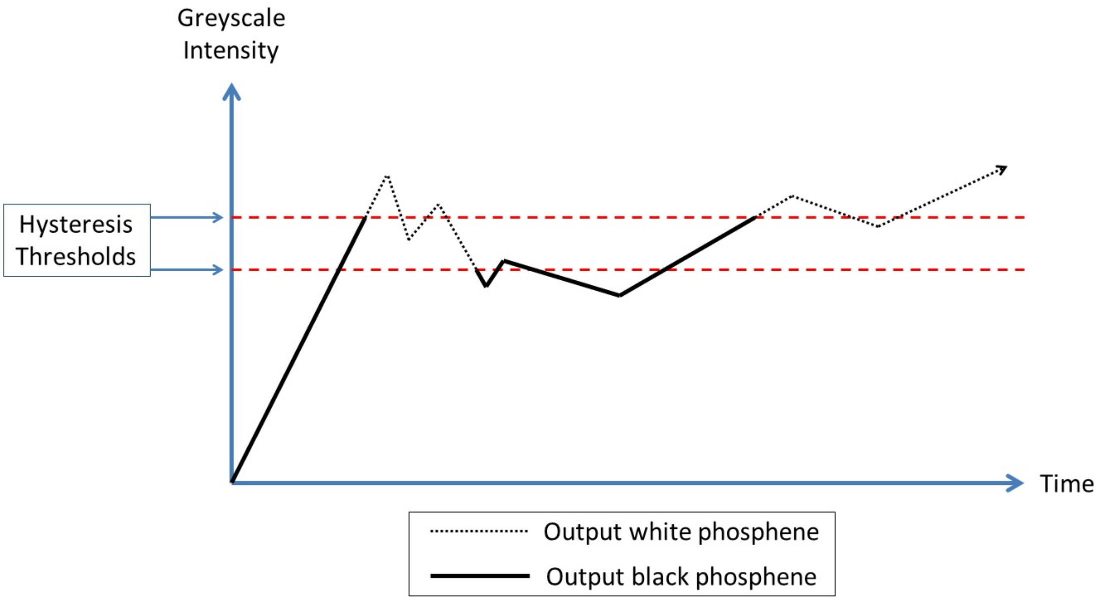

Figure 2.12: Binary thresholding with hysteresis

proved rather tedious however, with users often finding it difficult to correctly locate the IR receiver. It was also observed that users seemed to move back and forth between a range of thresholds in order to obtain a better understanding of a scene. Therefore a simpler interface is required that allowed for users to quickly change between thresholds. The solution chosen is to integrate a potentiometer control dial. The dial was mounted within a small plastic jiffy box and connected to the Hatpack sytem via heatshrinked cabling (see Figure 2.13). The potentiometer position was monitored using a digital I/O pin by measuring the time it took for a capacitor to discharge, which varies with the potentiometer resistance.

Threshold Ramping

A common approach that was found to be used with the manual control dial was to cycle back and forth through the whole range of thresholds. This seemed to allow momentary recognition of salient information in the scene at a variety different thresholds that could be investigate further by the user as needed. In an attempt to automate this process, a ramping function has been developed. The function makes use of a histogram in order set upper and lower threshold bounds to move between. This is done in real time and is dependent on current frame pixel data. After the bounds have been established, the threshold values are then incremented or decremented at set time intervals and following either a saw-tooth or triangular ramping pattern. The preferred pattern can be chosen by the user and the time interval can be adjusted so that a faster or slower ramp can be achieved. Although effective in

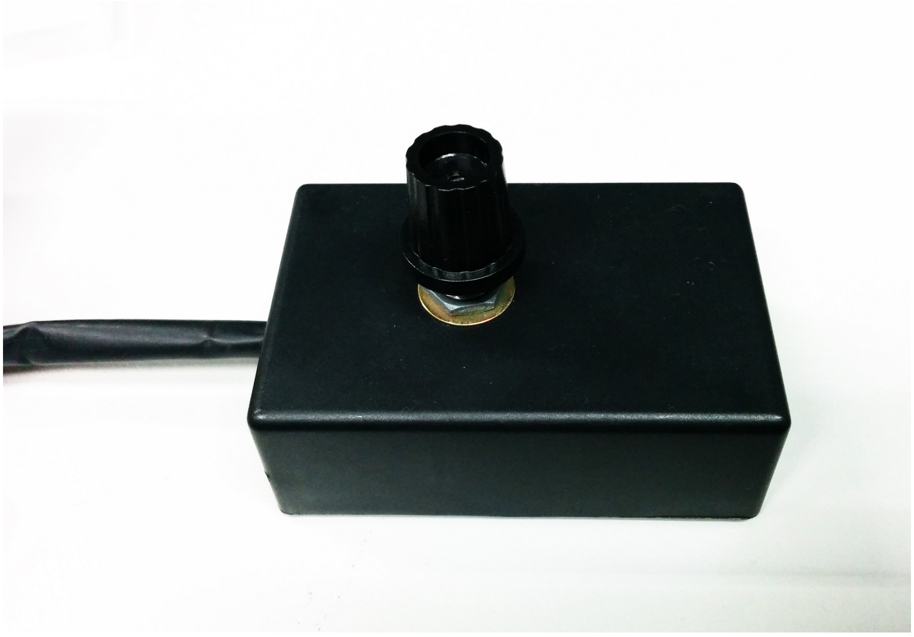

Figure 2.13: Manual threshold selection control dial

certain situations, this function was also found to induce nausea occasionally and so was excluded from further use and evaluation in testing.

Histogram Assisted Binary Threshold Selection

This function uses statistical information such as the range and spread of pixel data within a frame to automatically choose thresholds for the user. It is achieved through the generation of a histogram containing pixel intensity data. Since frame data in the system is only available one pixel at a time, implementation of a histogram is performed sequentially over the period of a single frame.

Storage registers are used as histogram ‘bins’. There are 20 bins, each of which store an 18-bit count of the pixels that fall within the bin’s allocated intensity range. The intensity range of pixels in the system is 0 to 1023, therefore each histogram bin has its own allocated intensity range of 51 levels, apart from the final bin which has a range of 55 levels. Only pixels within the center 480 by 480 square of a frame are used to make up the histogram. For each of these pixels, the value in the appropriate bin storage register is incremented. In hindsight, if the number of bins used was a power of 2, a simpler logic implementation for filling the histogram bins could have been achieved.

Once the histogram is constructed the bin with the maximum count is determined through the use of a conditional logic block. After this bin is found, an iterative approach is used to perform a discrete integral of the surrounding bins. Adjacent bin values are added one at a time and alternating from left to right of the bin with the maximum count. This method is performed over the remaining clock cycles in

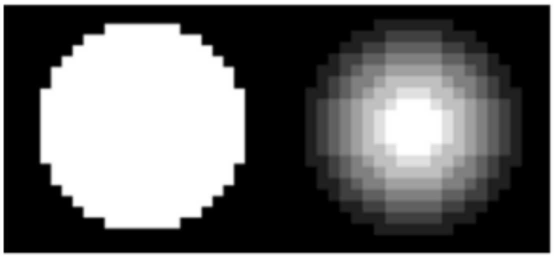

Figure 2.14: Phoshpene modelling: without Gaussian function (left), with Gaussian function (right)

the frame until a predefined threshold of 60% is reached. This threshold was found to work best in practical use of the system. At this point the bins with the highest (H) and lowest (L) valued intensity ranges in the integral are known. An average of the two ranges ($\frac{H+L}{2}$) is used to define the binary threshold.

2.3.5 Gaussian Phosphene Modelling

Stimulation of each electrode on the implant will produce a phenomenon in the patient's visual field known as a phosphene, whose appearance is somewhat similar to a bright spot of light [19]. Rather than simply using square pixels that perfectly line up with each other, it would be ideal to model the output visualisation based on what phosphenes would approximately look like. In the literature, one common approach is to model the phosphene using a 2D Gaussian mask [30]. The 2D Gaussian function is based on the standard distribution curve, except in two dimensions instead of one. This creates the appearance of a round 'spot' where the centre of the spot has the highest intensity value with the intensity values decreasing radially towards the outside edge of the spot, following the standard distribution curve. A comparison between a phosphene with and without the Gaussian function applied is shown in Figure 2.14.

In order to implement this method on FPGA, the intensity of each pixel within a phosphene needs to be weighted according to its Euclidian distance from the center of the phosphene. A set of discrete weightings that follow a Gaussian curve was generated for each different size of phosphene. A number of lookup tables have been generated using MATLAB scripts. Figure 2.15 outlines the use of these tables in the implementation. In order to get the Euclidian distance of a particular pixel, the coordinates of the phosphene's centre are required. The two LUTs (cent_pix_X and cent_pix_Y) achieve this, given that the phosphene number of the current pixel is available. The euc_dist LUT then provides a Euclidian distance value given the center pixel and current pixel coordinates. Taking as inputs the Euclidian distance and current ring number, the Gaussian weights LUT then provides the weightings.

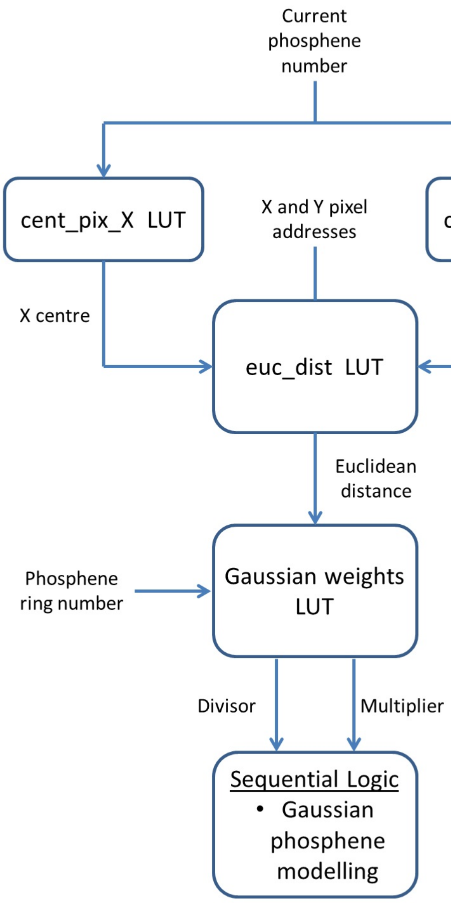

Figure 2.15: Gaussian phoshpene modelling implementation

To simplify the logic, a similar multiply and shift method was used for applying the weightings to pixels. This method is carried out in the sequential logic block on pixels that are to be output for display.

2.3.6 Dead Electrode Simulation

It is very likely that after implantation of a visual prosthesis there could be electrodes that do not function or are not able to elicit phosphenes in the patient's visual field. To simulate this the Hatpack has an implementation of a dead electrode or phosphene dropout feature. A simple lookup table is used to implement this feature. Using MATLAB, 3 pairs of 648 normally distributed random binary numbers have been generated to simulate 10%, 25%, and 50% of electrode failure. A MATLAB script was used to create a Verilog lookup table module (rand_masks) that would provide these random binary values as output. Taking as input the current phosphene number and selected dropout mode, the rand_masks LUT provides to the sequential logic block a binary value that would enable (binary 1) or disable (binary 0) the current phosphene from being displayed. Disabled phosphenes have a stored intensity of zero.

2.3.7 Edge Detection

Extracting edge information from an image can often be a useful way to show salient information in a scene. When used as the input to a visual prosthesis simulator, the edge result represents the main structural information, which in an uncluttered environment, can be a bit more intuitive to the user. A study done by Dowling et al. [39] compared the Sobel and Canny edge detection methods [23, 65] with a visual prosthesis simulator in order to determine which would give more useful mobility information. The results showed no significant difference between the two methods, and concluded that the Sobel operator was more suited for a visual prosthesis system due to its low computational cost. Guided by these results it has been chosen to implement Sobel edge detection. In order to implement Sobel edge detection simultaneous access of a 3 by 3 window of pixels is required. In the Hatpack system this meant that pixels needed to be buffered in a shift register. A 1920 by 10-bit shift register was used. This shift register buffers exactly three lines of pixels and has three taps or access points to the end pixels in each line. These end pixels are shifted into two consecutive sets of 10-bit registers, thus creating access to a 3 by 3 window of pixels from a frame. Similar implementations of this 'sliding window' function can be found in the works of Bailey [9] and Lim et al. [69] At each clock cycle a new pixel is inserted into the shift register, and the three pixels at the taps are shifted through the sets of 10-bit registers. Performing this method is equivalent to shifting the 3 by 3 window over every possible location in the frame in a sequential fashion. A discrete convolution of the Sobel edge operators is performed with the 9 available pixels (3 from taps and 6 from 10-bit registers) at each clock cycle and the result is stored in a temporary register. This value is then checked against a threshold in order to set the output edge image pixel values.

2.3.8 Infra-Red Remote Control

All functions within the Hatpack system implementation can be controlled by an infra-red remote control. Signals from the remote control are interpreted by the IR Receiver module and a unique code is generated as output. This code is used as input to the IR control combinational logic block. The logic is implemented as a large Verilog case statement that toggles or modifies a variety of control registers. These control registers are forwarded to all of the modules and sub-modules that they control. In each of these modules there are subsequent combinational and sequential logic conditional statements that alter the modules outputs according to these control signals.

2.3.9 Frame Rate Reduction

The ability of a person to detect motion is very important when it comes to mobility exercises in low resolution vision. A key factor that would limit one's ability to detect motion in the immediate environment is the lack of temporal resolution. Studies have shown that achievable stimulation rates of a visual prosthesis could be as low as four frames per second [38]. In order to simulate this temporal resolution and investigate the possible implications it may have on a patient's ability to move around, a frame rate reduction function has been implemented. The output frame rate of the Hatpack system can be changed in real time. The Hatpack has 8 different discrete frame rates available for selection (1, 2, 4, 8, 10, 15, 30 and 60 frames per second). Variable frame rate is achieved by holding the stored frame output data for the specific period of the chosen frame rate.

2.4 Extensions

2.4.1 Revised Mapping Implementation

Prior to starting the development of the Hatpack the specifications of the Gennaris device had not been fully defined. The number of electrodes to be used, and the orientation and location of implantation on the brain was approximate. Subsequent progression of the project provided more detailed specifications of the first-in-human device. In order to maintain a realistic simulation of the expected level of vision of the Gennaris, the Hatpack mapping needed to be updated following the new advice regarding the first-in-human device. The implementation of the visuotopic mapping function (see Section 2.3.1) made this a simple task as all that was required was a redefinition of the lookup tables used.

The mapping arrangement used to update the lookup tables was created by Dr. Wai Ho Li, who was a chief investigator on the project at the time. He developed MATLAB scripts, similar to the ones that were used in Section 2.3.1 of this work, to generate phosphene patterns that are more closely representative of the Gennaris given the new specifications. According to the Monash Vision Group, the first-in-human trial will involve the implantation of up to 11 tiles, each with 43 electodes, onto one side of the brain [78] similar to mock implantation image on the left of Figure 2.16. The outputs of Wai Ho's work have been used in conjunction with the scripts developed in the work of this thesis to update the Hatpack phosphene pattern and achieve the result shown in the bottom image of Figure 2.16. The revision of

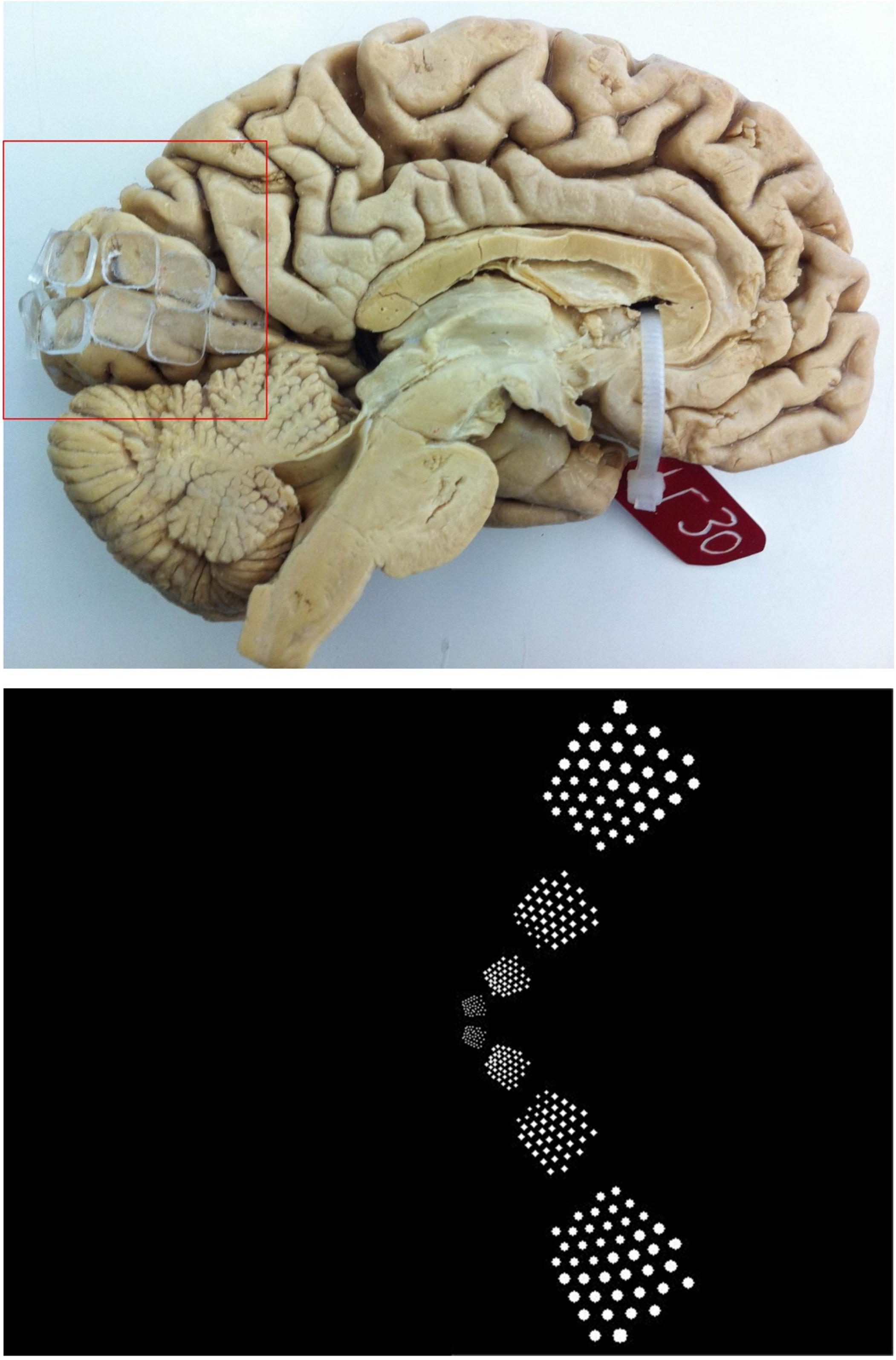

Figure 2.16: Revised mapping phosphene pattern. Placement of tiles on the left hemisphere of the brain (top image) will produce phosphenes in the right hemisphere of the visual field (bottom image).

the mapping was completed after the fourth psychophysics experiment and used as part of the fifth psychophysics experiment (see Chapter 3).

2.4.2 Otsu's Thresholding

Psychophysical testing with the Hatpack has highlighted the importance of selecting the most appropriate luminance threshold for an image (see Chapter 3). While the histogram-assisted automatic threshold selection mode developed for the Hatpack works well, there are some circumstances that will cause it to perform in a less than optimal way. An image with a multi-modal intensity distribution for example, will result in a threshold that is skewed towards whichever histogram maxima is of greater magnitude. In order to improve the implementation of the Hatpack automatic thresholding mode, alternative methods have been investigated for implementation.

A well known method used in computer vision and image processing is Otsu's thresholding method [83]. This method is based on the assumption that the images have bimodal histograms made up of two classes - 'foreground' and 'background' pixels. Given this assumption, an optimal threshold is calculated to separate the two classes by maximising the inter-class variance. A hardware implementation of the Otsu method was developed by Associate Professor Lindsay Kleeman. The code he developed was used (with permission) to replace the original automatic thresholding mode implemented in the Hatpack. As with the revised mapping implementation this replacement was completed after the fourth psychophysics experiment and used as part of the fifth psychophysics experiment (see Chapter 3).

2.5 Results and Discussion

Tables 2.1,2.2 and 2.3 show the logic and memory utilisation of the processing modules created for the Hatpack system. It was found that the total system logic element utilisation was rather minimal, using just 18% of the FPGA's logic resources. On the other hand the on-chip memory usage was rather high at 70%. This was as expected as the main LUT used for the visuotopic mapping implementation was large. It can be seen in the table that the histogram module uses almost half of the total logic utilisation of the system. This is because of the conditional logic that is used to determine the bin with the highest count. Using a smaller number of bins, or implementing an iterative approach would possibly decrease the logic utilisation for this module substantially, however both of these alternatives have drawbacks in terms of accuracy and latency.

Table 2.1: Logic utilisation of Hatpack modules

Module	Logic Cells (% of FPGA logic cells)
Whole system	20,061 / 114,480 (17.5 %)
Edge detection	314 / 114,480 (0.3 %)
Mapping and main functions	4738 / 114,480 (4 %)
Histogram	8258 / 114,480 (7 %)

Table 2.2: Logic register utilisation of Hatpack modules

Module	Logic Registers (% of FPGA logic registers)
Whole system	6236 (6 %)
Edge detection	146 (0.1 %)
Mapping and main functions	1224 (1 %)
Histogram	812 (0.7 %)

Figures 2.17,2.18,2.19 and 2.20 show some of the visual results of the each of the modes implemented in the system. It is worth noting that the results look much more intuitive when wearing the system. The system was also found to be more useful when viewing a non-static scene or when the camera sensor was being moved around (via head movements that are registered to the viewpoint by the user's ego motion sensors). This can be said to be due to the ability of the viewer to build up a mental model of the scene as more slightly differing temporal information is provided to them. Refer to the first video listed in Appendix B for a better indication of the system's operation.

The latency of the system (with no frame rate limitations enabled) is roughly one frame or about 17 ms. The majority of this latency is accrued by the mapping implementation which waits until the end of a frame to update the display storage registers. Updating these registers earlier would result in a 'tearing' effect in the output frame, as half the frame data would not be completely synchronised.

Prolonged testing of the system showed that battery life was roughly 4 hours from full charge. Preliminary testing hinted that certain tasks benefited from 4-level thresholding as opposed to binary (2-level), and that a frame rate of 4 frames per second was the lowest at which users could adequately navigate through a cluttered

Table 2.3: On-chip memory utilisation of Hatpack modules

Module	Memory bits (% of on-chip memory)
Whole system	2,774,535 (70 %)
Edge detection	19,140 (0.5 %)
Mapping and main functions	2,663,571 (66 %)
Histogram	0 (0 %)

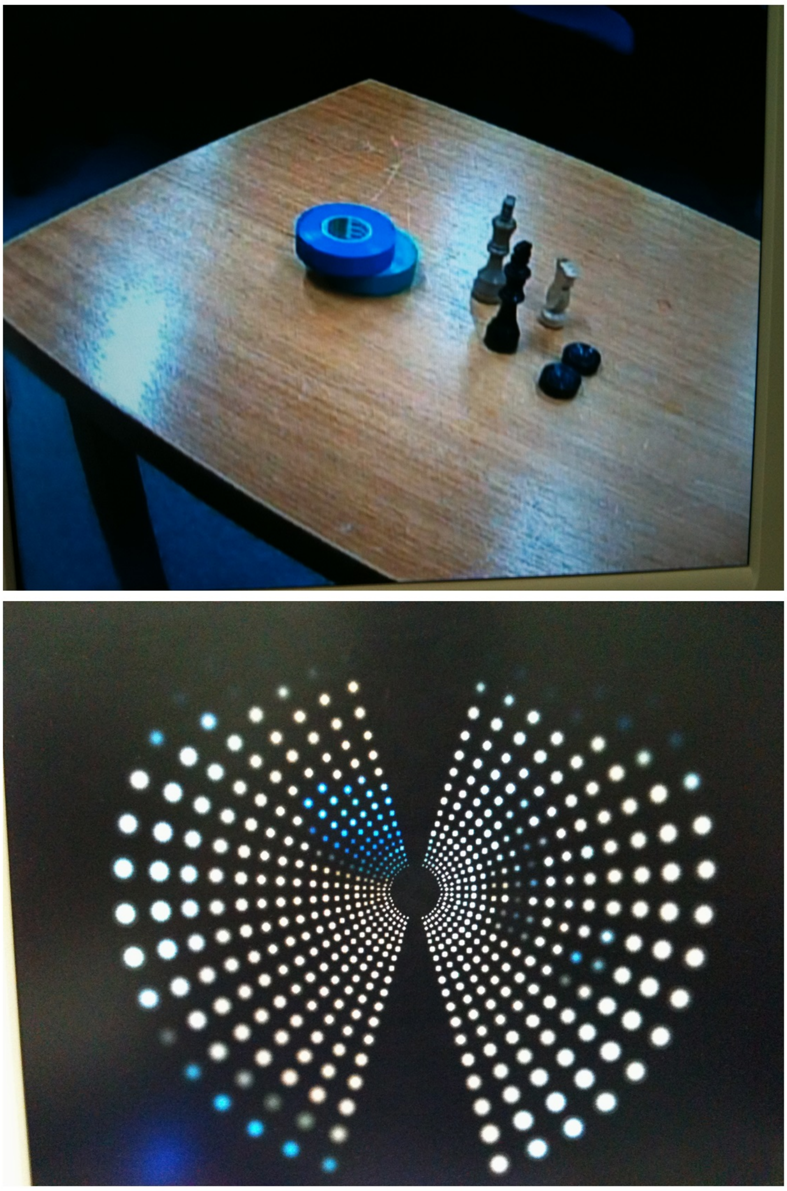

Figure 2.17: Visuotopic mapping (without greycale conversion and luminance thresholding)

environment. Edge detection proved useful in identifying objects in a non-cluttered, low-contrast environment.

2.5.1 System Limitations

In an actual cortical visual prosthesis, static electrode placement on the visual cortex means that the patient would not be able to scan over sections of the elicited phosphene pattern with movements of their eyes. In other words, phosphenes are locked to the gaze of the patient unlike the Hatpack system where the user is able to focus on different parts of the displayed pattern with eye movements. Simulating

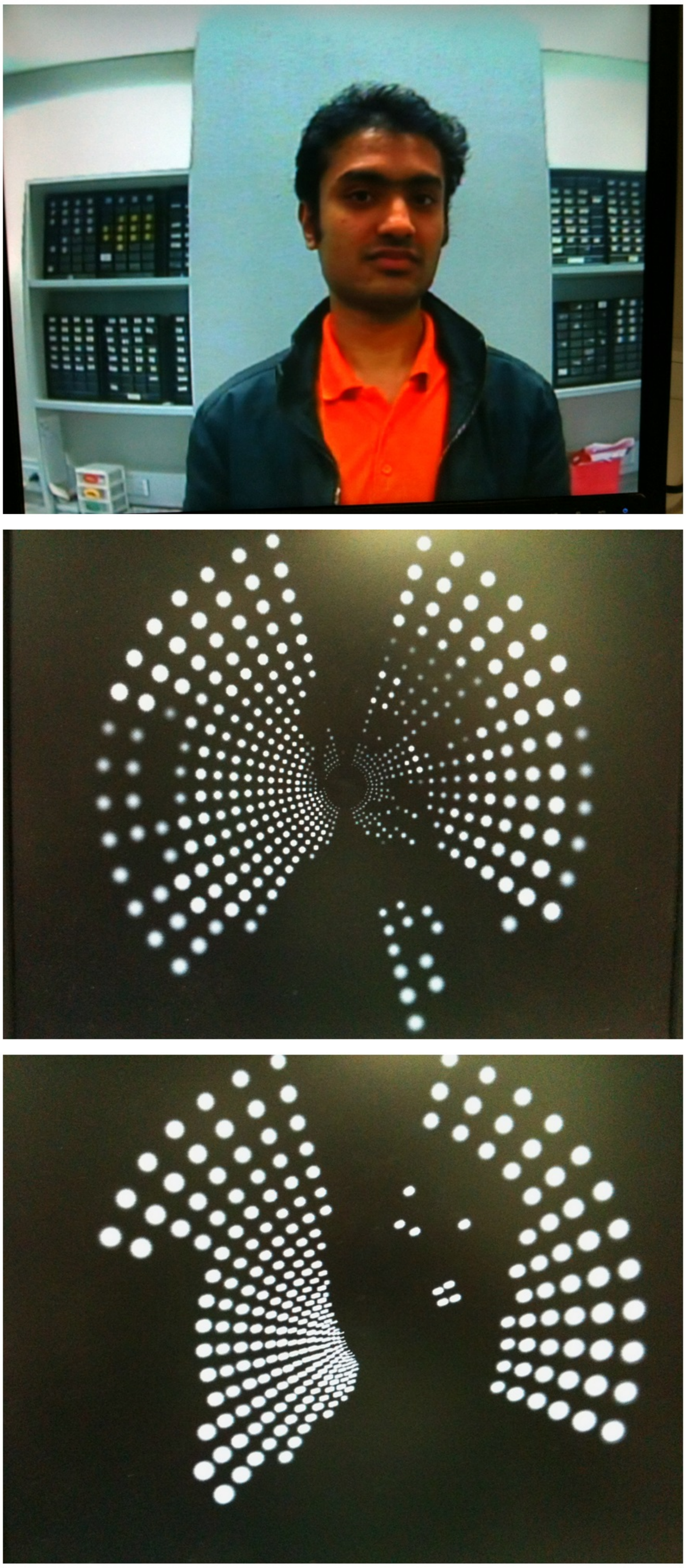

Figure 2.18: Luminance thresholding: full resolution image (top), 4-level (middle), binary (bottom)

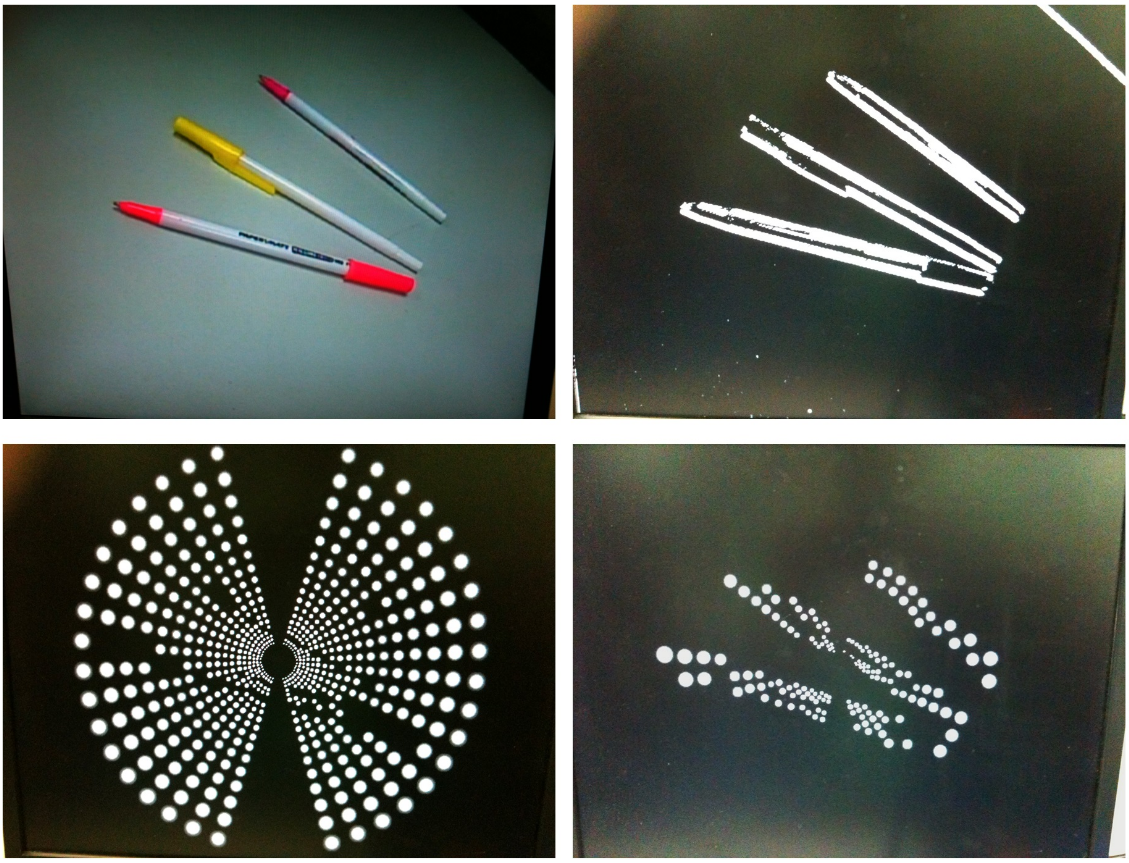

Figure 2.19: Edge detection: full resolution, extracted edges, output without edge detection, output with edge detection

this attribute would require an active eye-tracker and movement of the phosphene pattern along with the eyes.

Electrode placement within the visual cortex will also be significantly varied from patient to patient and will need to be placed carefully to avoid blood vessels in the brain. This will mean that the pattern of phosphenes will not appear as neat and 'mathematically' located as is the case with the Hatpack.

2.6 Summary and Conclusions

This chapter has presented the development and implementation of the Hatpack simulator system - a wearable and portable, real time simulation system and psychophysical test platform for a cortically based bionic vision implant. The Hatpack has been made to represent, as closely as possible, the anticipated vision to be provided by the Monash Vision Group's Gennaris vision implant. It makes use of current neurophysiological models of visuotopy and overcomes some of the most common limitations of existing simulators, such as lack of portability and frame rate and latency issues. This has been achieved through an FPGA implementation.

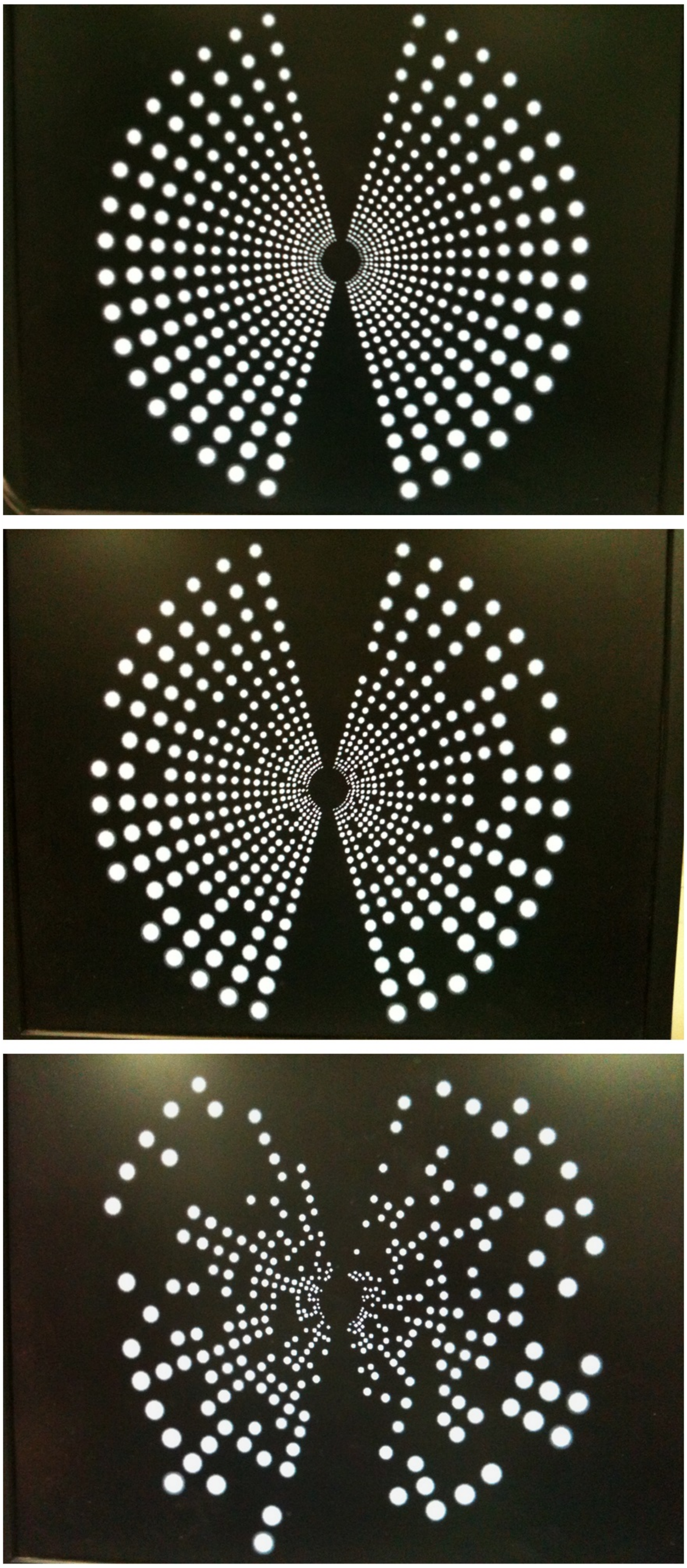

Figure 2.20: Dead electrode simulation: 0% (top), 10% (middle), 50% (bottom)

The Hatpack operates at 60 frames per second and with a constant latency of only 17 ms. It is lightweight, weighing only 3 kilograms, and low in power consumption, able to last up to 4 hours on a full charge. In addition to an implementation of the visuotopic mapping Monopole model through an innovative lookup table feature, the Hatpack also simulates variable electrode stimulation intensity through luminance thresholding, a 2D Gaussian model of the look of a phosphene, and the possibility of failed electrodes. Edge detection and a number of automatic thresholding functions have also been implemented.

A vision implant will produce phosphenes that are ‘gaze-contingent’. This means that they will remain in the same sections of the visual field regardless of eye movements. The Hatpack currently does not simulate this phenomena. While feedback from Hatpack use has suggested that scanning of the phosphene pattern does not seem to help, this would need further investigation and may prove useful as future work for the Hatpack. However, while the use of eye-tracking devices and real time movement of the phosphene pattern could potentially be used match the user’s focus point, gaze-contingency may still not be realistically achievable. This is due to the relatively high frequency of micro-saccades of the eye, and the latency of detecting the eye position and adjusting the image frame accordingly. If the latency from the movement of the eye to the movement of the image on the display is not less than the combined eye detection and image display times, then true gaze-contingency may not be practically realisable.

The Hatpack can be used to develop, test and evaluate new image processing functions and features as candidates for implementation in an end-user device. Evaluation would be done through the conduction of psychophysics experiments. A number of these experiments have been carried out with the Hatpack, the details of which are in the following chapter.

Chapter 3

Psychophysics Testing with Hatpack

The development of the Hatpack simulator descibed in the previous chapter has enabled the visualisation of a cortical bionic vision implant. More specifically, it simulates the level and characteristics of vision anticipated for the Monash Bionic Eye device - the Gennaris. It incorporates visuotopic mapping, which has been based on neurophysiological studies of the visual cortex. Also implemented are a number of vision processing functions that aid in the representation of scene information to the user at the highly limited resolution. The Hatpack provides a suitable platform for the testing of potential image processing functions and the evaluation of the abilities of users to complete certain tasks while restricted to the level of vision of a bionic eye implant. Psychophysical experiments can be performed on normally sighted individuals in order to test and evaluate implemented functions and user capabilities in real world situations, guiding the development of the Gennaris and other visual prosthesis devices.

This chapter details the design and conduction of five different psychophysics tests that have been completed through the course of this work. Each of the tests has involved the use of the Hatpack with a number of normally sighted volunteers attempting to complete tasks that resemble everyday activities. The development and evaluation of a simulator such as the Hatpack is not a linear process, and is often guided by user feedback. The psychophysics testing has provided much of this feedback and as a result the functionality and visual representation provided to participants has evolved throughout the different tests carried out. This is discussed in the relevant sections of this chapter. The contributions to this work by others is detailed in the relevant sections.

3.1 Introduction

Psychophysics has been defined by Bruce et al. as "the analysis of perceptual processes by studying the effect on a subject's experience or behaviour of systematically varying the properties of a stimulus along one or more physical dimensions" [20]. The term "psychophysics" was first coined by Gustav Fechner in 1860 [41] and more simply refers to quantitative investigation of the affects of physical stimuli on sensations and perceptions. Usually, the aim of a psychophysics experiment is to determine whether a subject can detect and identify a stimulus and differentiate between it and another stimulus, and possibly describe the sensations or perceptions experienced. Experimental stimuli are ideally objectively measurable and applied to one or more sensory domains (vision, hearing, touch, taste, smell) with varying thresholds.

In the field of bionic vision, psychophysics has been used mainly to assess the ability of patients to complete simple visual tasks that would be required in everyday use. As it has proved impractical to perform these tests on actual implanted patients, many research groups have developed simulators that can be used with normally sighted individuals. Performing psychophysics testing using simulators with normally sighted individuals allows for a quicker and easier development and test cycle. It can be done on a larger scale and in a safely controlled manner. Much work has been done with simulator systems and psychophysics, so much so that it has become a research field of its own called simulated prosthetic vision (SPV).

The SPV psychophysics work that has been completed prior to the beginning of the work presented in this thesis has predominantly come from four main research groups. The pioneering group was that of Cha, Horch and Normann from the University of Utah. Three separate studies were conducted by them in the early 1990s with the use of a simulator comprising a small video camera, miniature monochrome CRT monitor and a number perforated masks to give the effect of phosphenated vision. The first study aimed to investigate the relationship between visual acuity and the number of pixels and their spacing within a phosphene image. The acuity tests involved the presentation of the letter "E" on a computer screen at various sizes and orientations. The study concluded that 625 phosphenes presented within a visual field of 1.7° would be enough to provide a visual acuity of 20/30 [24]. The second study looked at the ability of users to read large text and the speeds they could achieve with the low vision of the simulator. The results showed that a 25 by 25 array of pixels projected on a foveal visual field of 1.7° is sufficient to provide reading rates near 170 words per minute for scrolling text and 100 words per minute for fixed text [26]. The final study completed by the group investigated mobility performance. Participants were required to walk through a maze of obstacles, while wearing the simulator, and walking speed and number of body contacts with obstacles was used as a performance metric. The findings indicated that a 25 by 25 array of pixels distributed within the foveal visual area could provide useful visually guided mobility in environments requiring a high degree of pattern recognition [25].

First to follow the seminal work of the Utah group were researchers from John Hopkins University School of Medicine, led by Gislin Dagnelie. A very extensive amount of work was completed by this group in this field from 2000 up until 2009. Work by the group mainly involved the investigation of the effects of varying phosphene grid parameters such as grid size, dot size, gap width, dropout rate and greyscale resolution on the users ability to complete certain tasks. Visual acuity, object recognition and reading tasks were looked at by Hayes et al. in [51]. This study concluded that the 16 by 16 array gave the highest performance for identifying shapes; that volunteers developed methods to overcome the visual limitations of the system; scanning was used on the tumbling E and reading tasks in order to enhance visual acuity; tactile information was used by most volunteers on the candy pour to improve accuracy; and that with systematic training patients may improve the utility of retinal prosthetic devices. Thompson et al. conducted tests requiring the recognition of faces [111], concluding that gradually swifter and more accurate facial recognition can be achieved when pixelised dot images are scanned over convolved facial images by using a mouse-pointing device. Dagnelie et al. investigated visual inspection and eye-hand coordination tasks [35], adequate reading tasks [33], and mobility tasks in both real and virtual environments [34]. The results of these studies suggested: that a 16 by 16 electrode array should allow paragraph reading; that simple visual inspection and eye-hand coordination tasks can be learned under pixelised conditions with very low resolution; that a retinal implant with as few as 60 electrodes may provide independent wayfinding abilities to the adventitiously blind however substantial practice and learning would be required. Later work from the group, carried out by Srivastava et al. focused on the simulation of a cortical prosthesis, and conducted similar visual inspection, eye-hand coordination and way finding tests as prior work [101]. This study concluded that the crude and limited information provided by the cortical visual prosthesis device under development could potentially help blind subjects to learn and get some assistance for conducting simple tasks in daily life.

Researchers from the Ophthalmology Clinic, Geneva University Hospitals conducted a number of detailed studies that investigated user performance of reading activities and the learning effects associated. Sommerhalder et al. showed that 300 electrodes would be needed for restoration of some reading ability [99], and about 600 for useful reading performance of full-page text [100]. Fornos et al. followed this work by investigating a real time pixelisation strategy [47] and the ability of users to complete simple pointing and manipulation tasks [46]. Fornos et al. concluded in these studies: that the spatial and temporal characteristics of image pixelisation play a role in artificial vision simulations; that four to five hundred contacts covering a 2 by 3 mm^2 retinal area is necessary to transmit sufficient visual information for full-page text reading.

The fourth research group was from the University of New South Wales in Australia. This group investigated the effects of filtering schemes and cutoff frequencies [29], visual tracking performance [50], and performance differences between a rectangular and hexagonal grid sampling and representation [27, 28]. These studies concluded: that clinically significant improvement can be achieved by adopting an optimised configuration of filter aperture; that a small improvement in simulated prosthetic visuo-motor function was achieved through implementation of an image analysis scheme involving overlapping Gaussian kernels; that implanted electrodes arranged in to form a hexagonal mosaic as opposed to existing rectangular arrangements will benefit the acuity a device can provide for an implantee; that acuity depends strongly on the way images were analysed prior to phosphene representation.

The work carried out by all of these groups has shown that psychophysics using simulated prosthetic vision is a viable way to investigate the potential capabilities of bionic vision users. It may also prove an ideal way to tune image processing functions and guide the development of vision devices although the absence of any correlation of this work to actual implants mean the simulated approach is yet to be fully validated in practice. The Hatpack system has been used in a similar way to explore potential of the level of vision and implemented functions of the Gennaris device.

The work in this chapter is not a formal set of clinical trials, and was a subsidiary goal that eventuated from the development of the Hatpack. The testing has been used to assess the functionality of the Hatpack system and improve the implementations that may someday be implemented in the Gennaris device. The experiments presented have been based loosely on the types of testing that has been done in other existing work that has been reviewed above. Since these tests involved human participants, ethics approval was required and acquired through Monash University Human Research Ethics Committee (MUHREC) with the following ethics approval project number: (CF12/0085 - 2012000030).

3.2 Experiment 1: Maze Test (Obstacle Avoidance)

Mobility and orientation is very important for navigation through everyday environments. Navigation or obstacle avoidance tests have been used by a number of research groups during their simulated prosthetic vision studies. These experiments usually involve a controlled space with a number of obstacles that the participant has to try and make their way through with minimal collisions. Examples of such work are in [25, 34, 101].

Since a key goal of this thesis is to potentially improve the navigational ability of bionic vision users, the first experiment that has been conducted is an obstacle avoidance task - the Maze Test. This test was designed and conducted in collaboration with Benedict Yong who was a research assistant for the Monash Vision Group at the time.

3.2.1 Experimental Setup

Seven test subjects were recruited for this experiment (6 male, 1 female). The maze test involved subjects walking through a indoor course scattered with obstacles. The obstacles were placed in one of 5 different arrangements that had been previously defined using a pseudo-random method of obstacle placement that would minimise as much as possible the ability to learn the environment. Care was taken to avoid creating arrangements that would allow for a significantly more difficult or simpler path to the end of the maze. Figure 3.1 shows the test area with one of the predefined obstacle arrangements. The obstacles were large cardboard boxes and office chairs with wheels. The 5 different configurations were divided up, one for each mode that was tested and kept consistent between subjects. Subjects were not allowed to see the obstacle layout before each test. The starting point was around the corner from the main rectangular maze area (see Figure 3.1), and the end point was at a table at the far end wall of the maze. Placed on the table was a small black box. When the subject successfully located and picked up the box, the test ended. Video footage of an example of this test can be found in Appendix B and here is a link.

For the test, both time to completion and number of collisions were recorded for all subjects. Subjects were allowed to touch the obstacles in the maze, so only unintentional collisions were counted. The 5 modes tested were a control mode similar to a normally sighted person (full resolution, full colour), and four non-control modes including 4-level luminance thresholding (full frame rate), binary thresholding (full frame rate) and reduced frame rate at 15 Hz and 4 Hz (both with 4-level thresholding). Subjects were given 2 minutes accommodation time just

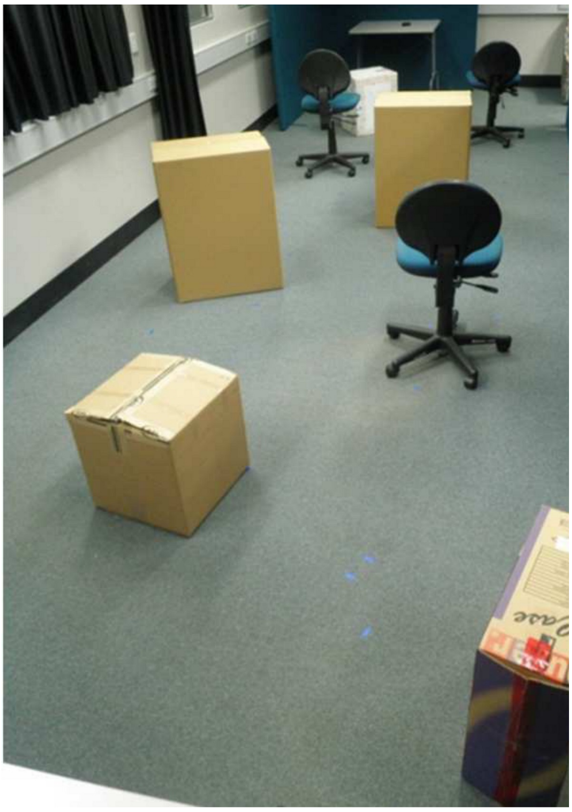

Figure 3.1: Layout of the maze test

before the test for each mode where they could adjust to using the system around a cardboard box and two chairs placed away from the actual maze area. Subjects were also given a minimum of 5 minutes break in between each test.

3.2.2 Results

The statistical method of two-sample T-test has been used to analyse the results. This method is a hypothesis test for determining whether the means of two populations differ on some outcome significantly. The data are collected from two random samples of independent observations [61]. The top graph in Figure 3.2 details the time to completion (in seconds) for each mode, averaged over the 7 subjects. The order of the modes from left to right in the graphs reflects the order that the subjects were tested in. The error bars show the standard error of 2 standard deviations away from the mean. 2-way, paired T-Tests were conducted between the control time and each of the non-control modes, as well as between the 4-level thresholding full frame rate mode and the other 3 modes (binary and both reduced frame rates). T-Tests will result in a ‘T-value’ that can be used for significance testing. In order to determine whether there is statistical significance, the T-value is compared against a standard table of t-values to determine a ‘p-value’. The p-value is the probability of obtaining a T-value. If the p-value falls below a specific threshold, then the result is

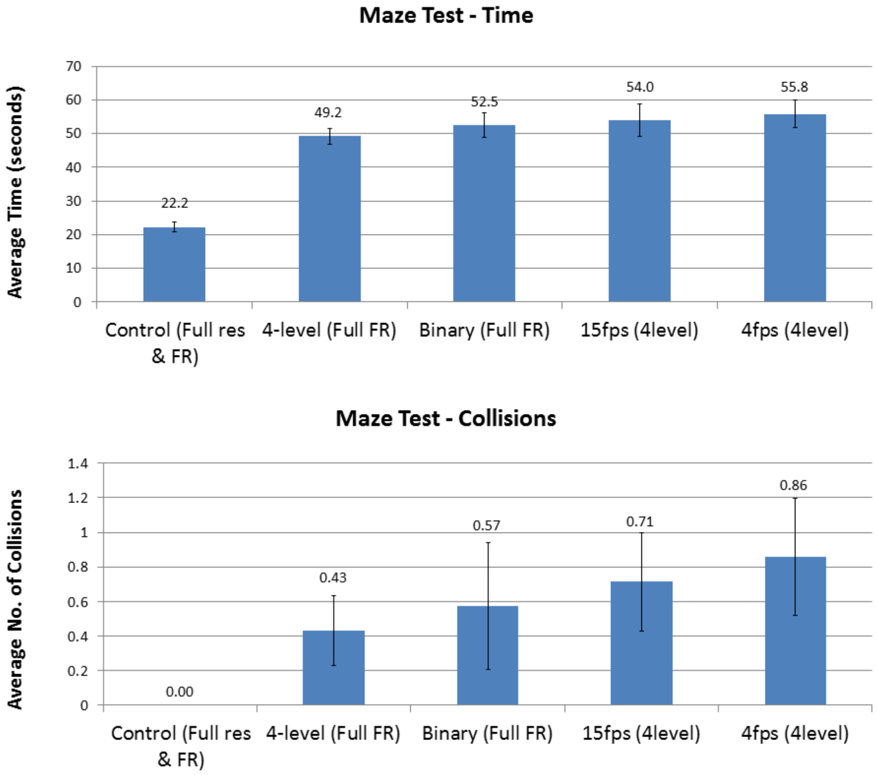

Figure 3.2: Graphs of time to completion and number of collisions for each mode for the maze test

said to be significant. A threshold of 0.05 relates to a 95% confidence interval, which is roughly two standard deviations from the mean, and is a common threshold used for significance testing [6]. The times taken for all the non-control modes were found to significantly higher ($p < 0.05$ for all) than the time for the control. The binary and reduced frame rate modes were slightly longer than the 4-level thresholding full frame rate mode, but all the non-control modes were within the statistical margin of error ($p > 0.05$ for all). The bottom graph in Figure 3.2 details the number of collisions for each mode, averaged over the 7 subjects. The error bars show the standard error for a 95% confidence interval. The average number of collisions was very low, due to a few of the subjects not colliding with anything in any of the modes, but the binary thresholding and reduced frame rate modes had more collisions on average than the 4-level thresholding full frame rate.

3.2.3 Discussion

The Maze Test results show that subjects take much longer to finish the test in any of the non-control modes compared to the control, and that although the binary and reduced frame rate modes took slightly longer to complete than the 4-level full frame rate mode, the difference was not statistically significant. This trend is also shown in the average number of collisions, but the standard error is too large to allow firm conclusions. From observations made while building and testing the system, reduction in greyscale intensity resolution and frame rate does increase the difficulty of most general tasks including navigational and obstacle avoidance tasks. Possible reasons for this not being made clear in this particular test's results are that the maze area was fairly small and straightforward so the task could be completed in a relatively short amount of time, and the number of test subjects was low, presenting a relatively large error. Also, the obstacles used in this test were large and obvious and so subjects may not have benefited a lot from an increased greyscale resolution and frame rate. Another problem could be the order of the modes in which the subjects were tested was made consistent and that the 'harder' modes were tested later. A learning effect just from repeated testing, even with the changing obstacle placement and accommodation time between tests, could cause a decrease in times for the later tested 'harder' modes and hence reduce differences between them and the earlier test 'easier' modes.

3.3 Experiment 2: Chessboard Test (Object Manipulation)

Interaction and manipulation of objects is a very common activity throughout everyday life. For normally sighted individuals such tasks usually involve the coordination of sight and touch sensory information with movement of an arm or leg. Without vision, blind individuals lose half of the available feedback, making object interaction much more difficult and inaccurate. The use of a visual prosthesis could restore some level of this visual servoing ability, and so a number of eye-hand coordination experiments have been carried out by research groups [34, 46, 101].

The second experiment that we have conducted is an object manipulation task involving the placement of chess pieces on a chessboard. This test was designed and conducted in collaboration with Benedict Yong.

3.3.1 Experimental Setup

The same 7 test subjects (6 male, 1 female) recruited for the Maze Test have been used as participants in this experiment. The task required subjects to sit down at a table with a chessboard in front of them and 16 chess pieces (8 black, 8 white) placed in a random pile to the left of the chessboard. The objective was for the subjects to sort and place any black coloured pieces on any white square in the bottom half of the chessboard, and the white pieces on black squares in the top

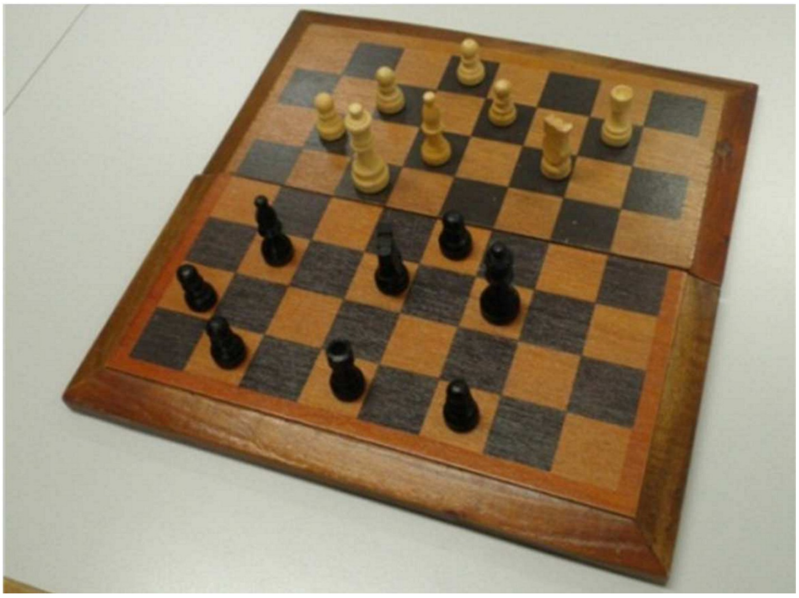

Figure 3.3: Example of completed chessboard test

half of the chessboard. Video footage of an example of this test can be found in Appendix B and here is a link.

For the test, both time to completion and number of mistakes were recorded for all subjects. For a piece to be considered as correctly placed, at least half of it had to be over the correct square. Another aspect to this experiment was to test for learning effects that come from repeated usage of the system. As such, the modes were tested in the order, from left to right, shown in the graphs of Figure 3.4 (all at full frame rate). A control mode was tested first (full resolution, full colour). This was followed by the non-control modes which were repeated in the order: binary luminance thresholding, 4-level thresholding, binary, 4-level, binary, and 4-level. Before the testing, subjects were asked to attempt the task without wearing the system in order to familiarise themselves with the task itself. The testing was conducted in a single session, with a minimum 1 minute break in between each test.

3.3.2 Results

The top graph in Figure 3.4 details time to completion (in seconds) for each mode, averaged over the 7 subjects. The order from left to right of the modes reflects the order the subjects were tested in, and how the non-control modes were tested repeatedly 3 times to examine learning effects. The error bars show the standard error. 2-way, paired T-Tests were conducted between the control time and each of the non-control modes, as well as between the binary and 4-level thresholding for each pair of repeated tests (eg. 1st binary with 1st 4-level). The times taken for all non-control modes were significantly longer than the control mode ($p < 0.05$ for

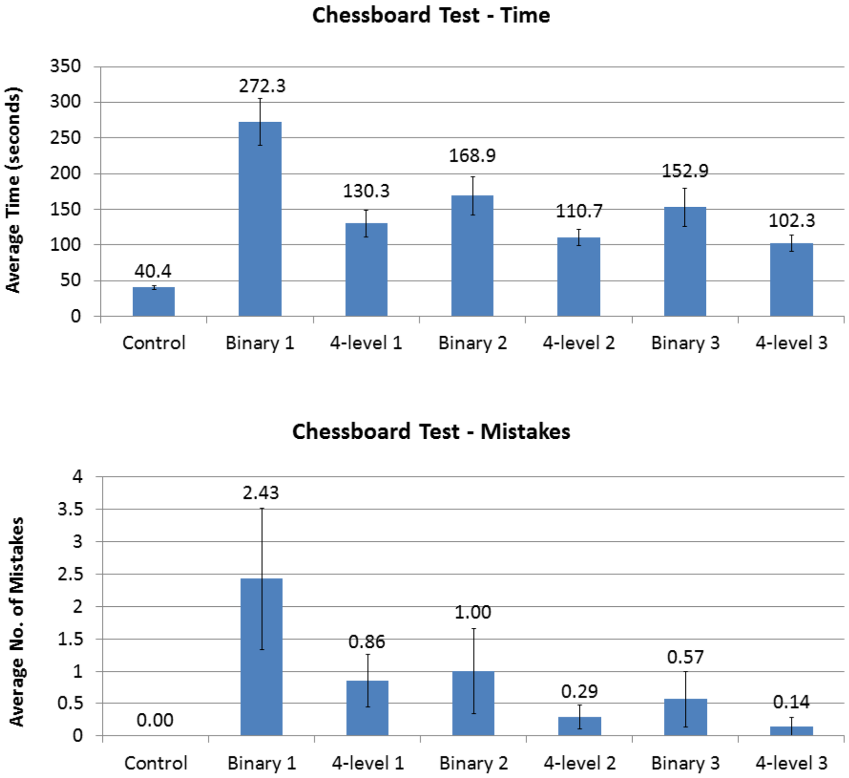

Figure 3.4: Graphs of time to completion and number of mistakes for the Chessboard Test

all). The times taken for the binary modes were significantly longer than the 4-level thresholding for the same repeated number of trial ($p < 0.05$ for the 1st and 2nd pairs of tests, p=0.063 for the 3rd pair). The times for all modes decreases with increasing number of repeated tests. The bottom graph in Figure 3.4 details the number of mistakes for each mode, averaged over the 7 subjects. The error bars show the standard error. The average number of mistakes was quite low due to some subjects not making any mistakes. The trend however clearly looks similar to the Chessboard Time graph with decreasing number of mistakes with repeated trials.

3.3.3 Discussion

For the Chessboard Test, the results demonstrate that the binary modes were significantly longer than the 4-level thresholding modes for each repeated test. The results also show that there is a clear downwards trend with increasing number of tests for both modes. The average number of mistakes also shows these trends, and that the binary has more mistakes than the 4-level and that both modes decrease over repeated testing, however the standard error is very large. The reason the tests were completed much faster on 4-level compared to binary is likely because this test is based primarily on contrast discrimination and the extra levels of grey available on the 4-level allow the subjects to be able to tell the difference between the dark and light chess pieces and chessboard as well as the grey table more rapidly. This shows that different tasks may benefit differently from various modes. A significant learning effect was evident as times and mistakes would decrease with repeated testing, probably leading to an eventual plateau point where times do not get much faster. It is apparent that as people keep repeating a task they are unfamiliar with, they will improve at it. This is quite likely to be the case with a visual prosthesis simulator, or even a patient with a visual prosthesis implant itself.

3.4 Experiment 3: Ball Interception Test (Motion and Interaction)

In the previous two experiments, the scene was void of other people and all objects were stationary when not moved by the participant. In a typical real life setting however, there are often other people around and they will be moving of their own free will. Given that stimulation studies have shown that distinguishable frame rates of early vision implants could be as low as four frames per second [38], it is useful to investigate how well a bionic vision user would be able to interact with a dynamic environment under these conditions.

The third experiment that has been conducted involves participants interacting with rolling balls while using the Hatpack with limited frame rates. This test was designed and conducted in collaboration with Benedict Yong.

3.4.1 Experimental Setup

Once again, the same 7 test subjects (6 male, 1 female) were used. The Ball Interception Test required subjects to sit down at a modified table tennis table. The table had a horizontally sliding ‘paddle’ mounted on the participant’s end and at the opposite end 4 fixed half-pipe ramps that balls could be rolled down (see Figure 3.5). The ramps were all the same length, had the same elevation angle, and had a ‘fast’ and ‘slow’ roll point for the balls. The ramps were partially covered by a cardboard screen so that the participant could not see which ramp the ball was rolling down until it reached the bottom. The objective was for the subjects to use the sliding paddle to block/intercept/stop the balls (similar to the video game ‘Pong’ or to ‘Air Hockey’ in an arcade).

The number of successfully intercepted balls was recorded for all subjects. To be considered as a successful interception, the ball was not allowed to hit the rail that the paddle was sliding along. In cases where there was ambiguity in the outcome (ie.

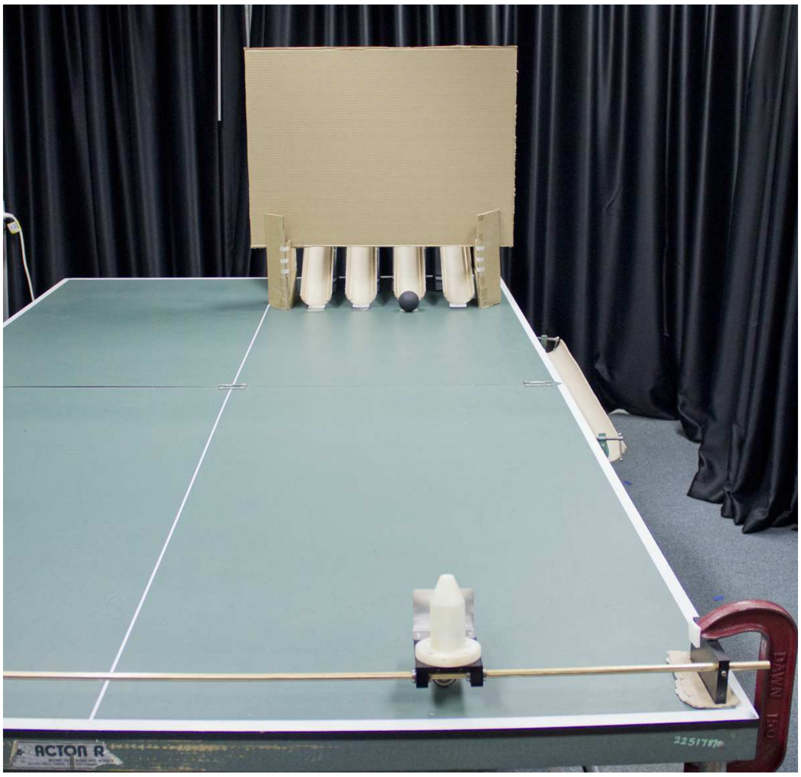

Figure 3.5: Ball interception test

balls getting stuck under paddle or between the paddle and side block), a note was made and that particular roll was repeated at the end of the trial. For each mode, 60 rolls were completed. A predefined randomised roll sequence was used that ensured each of the 4 ramps was used equally and also that there was an equal number of ‘fast’ and ‘slow’ rolls. The aim of this test was to observe the effects of lowered frame rates, and so the modes tested were: a control (full resolution, colour and framerate) and 4-level luminance thresholding at 4 Hz, 10 Hz, and 60 Hz. Subjects were given 5 minutes to familiarise themselves with the task before beginning the trials. The testing was conducted in a single session, with a minimum 1 minute break in between each mode tested.

3.4.2 Results

The graph shown in Figure 3.6 details the number of successfully intercepted balls for each mode, averaged over the 7 subjects. Once again, the order of the modes reflects the order the subjects were tested in, and the error bars show standard error. 2-way, paired T-Tests were conducted between the control mode and each of the non-control modes, as well as between the relative non-control modes (4 Hz vs 10 Hz, 10 Hz vs 60 Hz). The number of intercepted balls for each of the non-control modes was significantly shorter than the control mode ($p < 0.05$ for all). There was also a significant difference between the non-control modes themselves ($p < 0.05$ for

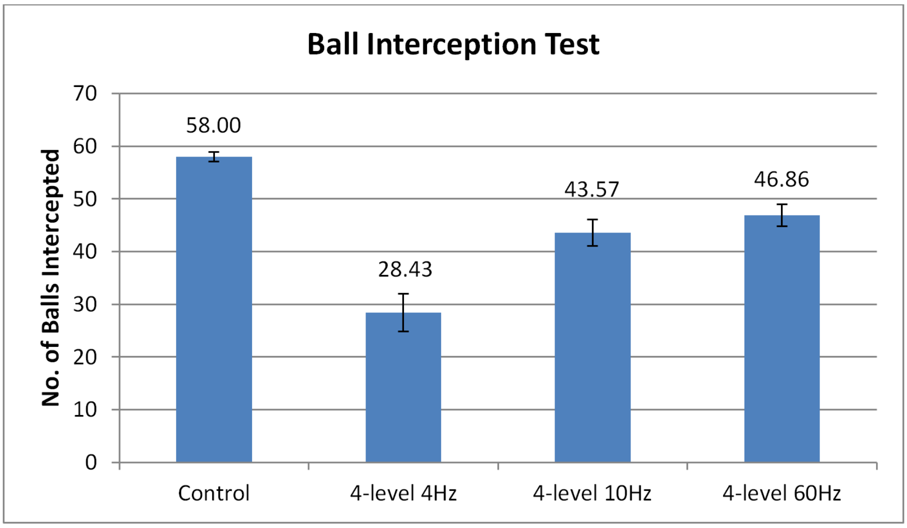

Figure 3.6: Graph of number of intercepted balls for Ball Interception Test

both 4 Hz vs 10 Hz, and 10 Hz vs 60 Hz). The graph shows an upward trend in intercepted balls as the frame rate is increased, however it is possible that this is partly attributed to learning.

3.4.3 Discussion

The Ball Interception Test results show an upward trend in successfully intercepted balls as the frame rate is increased. This was as expected, as a lower frame rate would give a user less time to react and would not allow for smooth tracking of the ball's movement. The differences between the control mode and non-control modes, as well as between the consecutive control modes themselves all showed significant differences in results. It was noticed however, that a considerable number of 'misses' that occurred in the tests were due to misjudgement of where the user thought they had actually moved the paddle to (the paddle would be offset from the balls final location by only a small margin). This is clearly to do with the ability of the participant to coordinate themselves with the paddle placement without actually looking at it. It was not possible to view both the paddle and the rolling ball at the same time, except when the ball reached very close to the paddle, and so the participants had to learn the relative position of the paddle through trial and error. Although, some time was allowed to 'calibrate' with a simple hand-eye coordination task before each test, the subject's still seemed to have some trouble.

3.5 Experiment 4: Shape Identification

From the observations made during the first three experiments, as well as general use of the Hatpack, it became evident that the selection of luminance thresholds

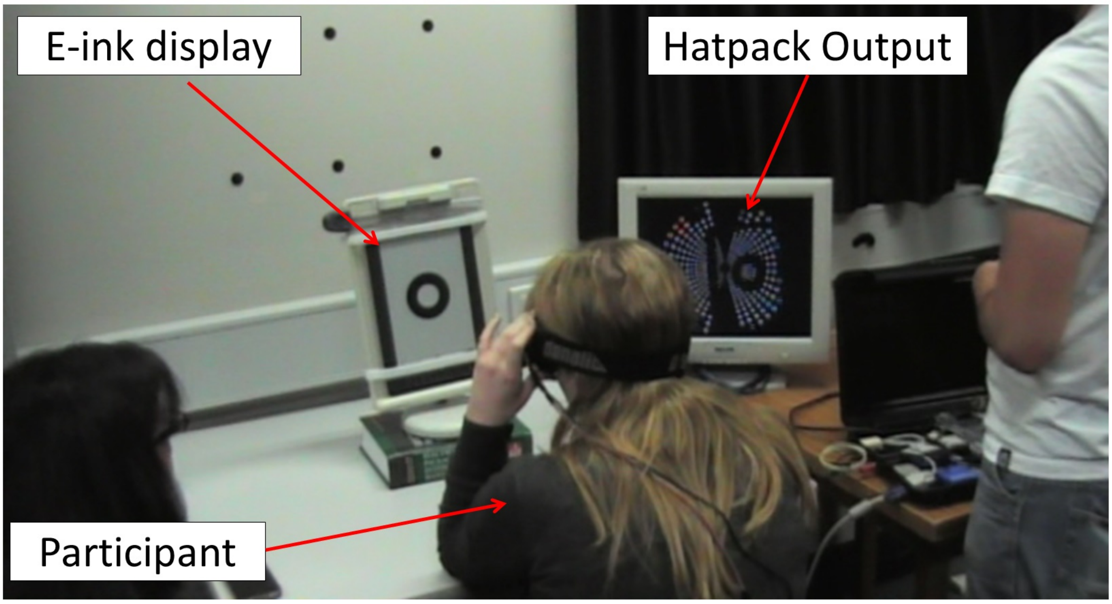

Figure 3.7: Shape test setup

can have a significant impact on the presented phosphene image. In order to mitigate the adverse effects of poor lighting and minimal scene contrast, manual and automatic luminance thresholding modes were added to the Hatpack system (See Section 2.3.4). Initial testing of these modes hinted at differing user preference and perceived intuitiveness.

The fourth experiment carried out involved a set of tests to establish which luminance threshold selection method would be most pragmatic and useful for implementation in the end-user implantation device. The design of these tests and the statistical analysis of the results have been completed by Dr. Collette Mann. Dr. Dennis Lui assisted during conduction of the testing.

3.5.1 Experimental Setup

Recruited for this experiment, were 18 participants, all of whom were either Monash University staff or students (8 male, 10 female). The set of tests required participants to sit in front of an e-ink display and attempt to identify displayed shapes which varied in shade, while wearing the Hatpack set to one of the three luminance thresholding modes: Static, Automatic or Manual. The test setup is shown in Figure 3.7. The reflective illumination properties of the e-ink display meant that lighting could be carefully controlled and kept consistent throughout all trials.

Participants were initially given the Hatpack and allowed 5-10 minutes accommodation time for use of the three modes. Participants indicated the order they felt was the easiest mode to the hardest mode, by identifying shapes on an e-ink display screen. The shapes used for the accommodation time were different to the

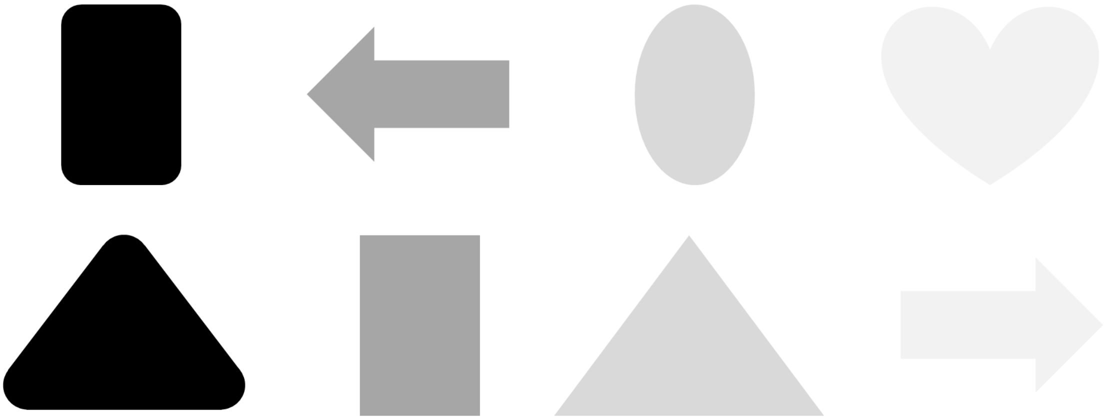

Figure 3.8: Shape test shapes and shades

shapes used in the actual testing. Once the order of the threshold selection modes was decided, participants were instructed that there were eight possible shapes to identify, different to the test shapes they had just viewed, in four monochromatic shades (black; 35%, 15% and 5% grey). They were also told the identity of the eight shapes, but they were not shown the shapes prior to testing. The eight shapes (Figure 3.8) included: a triangle with pointed corners or rounded corners, a rectangle with pointed or rounded corners, a love heart, a left-facing or right-facing arrow and an oval. Shapes were presented one at a time in a pre-randomised order with the possibility of each shape being presented multiple times in any given trial to remove the possibility of guesswork. Tests were completed in blocks of 32 shapes for each mode, with a total of 96 trials and 1-2 minutes rest between each block. Each test of 96 trials contained an equal number of the four possible shades, i.e. each shade 24 times. Participants were instructed to say “pass” if they could not identify the shape. This answer was classified as a ‘no response’. The participant’s timed response was then recorded following each trial.

3.5.2 Results

The statistical analysis of the results of this experiment has been completed by Dr. Collette Mann, using IBM SPSS Statistics version 20. Two sets of data were analysed; Response Accuracy as a function of Mode, and Response Accuracy as a function of Shade Dimension. Response Time was not analysed as the stop time was arbitrary, dependent on the participant. General Linear Model was used for both analyses controlling for age, gender and total number of no responses (TotalNR).

Response Accuracy (threshold mode)

The statistical method of one-way analysis of variance (ANOVA) was used to analyse the results. This method involves the division of variation in a set of observations

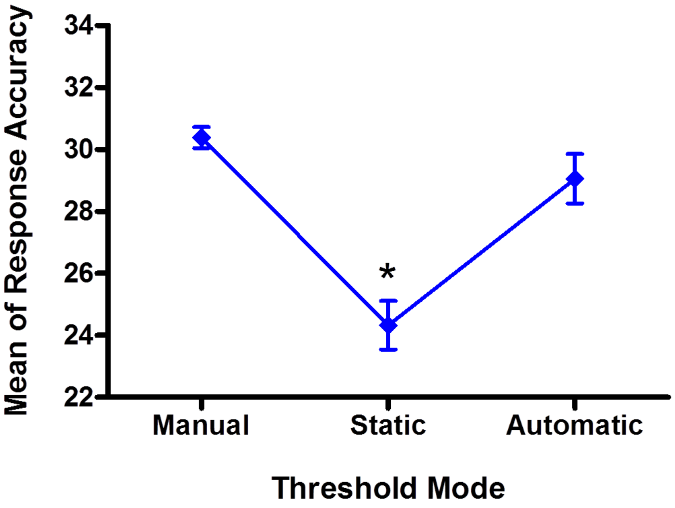

Figure 3.9: Graph of response accuracy as a function of threshold mode for shape test

into distinct components, in order to determine whether there are any significant differences between the means of three or more independent groups [36]. A one-way repeated measures ANOVA was conducted to determine if there were significant differences between the three luminance threshold selection modes (Manual, Static and Automatic) for participants accurately identifying eight distinctive shapes. Age, gender and TotalNR were included as covariates. A test of within-subjects showed no main effect for accuracy (p=0.932) or any interaction effects between accuracy and amount of no responses, i.e. TotalNR (p=0.062, Greenhouse-Geisser corrected) although this was trending; accuracy and age (p=0.869); or accuracy and gender (p=0.832). There was a main between-subject effect for TotalNR (p<0.001) demonstrating that some participants found identifying the shapes significantly more difficult than others resulting in more “pass” responses for these subjects. There were also main between-subject effects for age (p=0.024) and gender (p=0.015) with younger subjects performing better than the older participants; and males more correctly identifying the shapes than females. Pairwise comparisons between the three modes showed that the Static mode was significantly harder for participants to correctly identify shapes between both the Manual (p<0.001) and Automatic (p=0.003) modes as shown in Figure 3.9. However, there were no significant differences between the Manual and Automatic modes (p=0.502).

Response Accuracy (shade dimension)

A one-way repeated measures ANOVA was conducted to determine if there were significant differences in accuracy detecting the four shade dimensions (black; 35%,

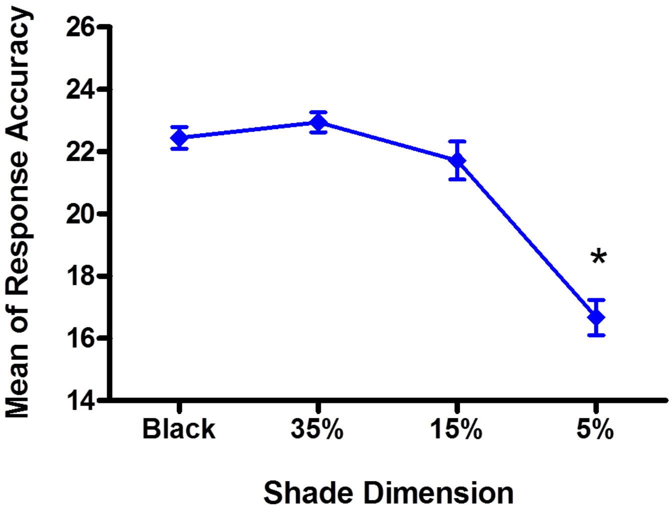

Figure 3.10: Graph of response accuracy as a function of shade dimension for shape test

15% and 5% grey). Age, gender and TotalNR were included as covariates. A within-subjects test showed there was a significant main effect of accuracy (p=0.028) indicating that participants could correctly identify shapes for some shades, but it was more difficult for others. Pairwise comparisons showed that participants found the 5% shade of grey significantly harder to identify shapes than the other shade dimensions (p<0.001 for all shades) as illustrated in Figure 3.10. There were no significant differences between the other three shades. Main effects for between-subjects tests remained the same. As with the threshold mode, there were no interactions between accuracy and age (p=0.647) or accuracy and gender (p=0.419). However, there was an interaction between accuracy and TotalNR (p=0.001), demonstrating that the number of "pass" responses was very dependent on the shade dimension, in particular the 5% grey shade.

3.5.3 Discussion

As is evident in the results, both the Manual and Automatic modes showed significantly better performance than the Static mode in terms of response accuracy. However, when 'no responses' were removed in the statistical analysis, the Static mode was not significantly different to the other two modes. This indicates that were no incorrect responses in this mode, and participants either answered correctly or could not identify the shapes at all, and therefore chose to 'pass'. There was also no statistically significant difference between the Manual and Automatic modes, suggesting that more and varied tests would be required, such as involving mobility and/or manipulation, in order to conclude that one mode performs significantly better than others. The test comparing shade dimensions was more conclusive with shape identification significantly worse when the shade was set to 5% grey indicating that in all modes, participants either incorrectly, or could not, recognise the shapes.

3.6 Experiment 5: Sock Sorting

Following progression of the Gennaris device development, the expectations of the first in human trial had changed. A more likely indication of the number of electrodes to be implanted and their ideal arrangement on the brain was made known. As a result the Hatpack visuotopic mapping was modified soon after conducting the Shape Identification experiment (See Section 2.4.1). Furthermore, the implemented automatic thresholding mode was updated with an implementation of the Otsu's thresholding method (See Section 2.4.2). Results from the fourth experiment indicated that a more interactive test maybe required to conclude the type of luminance thresholding mode that performs significantly better than others.

To evaluate user performance with the updated mapping and automatic thresholding mode a fifth eye-hand and manipulation experiment was conducted, involving the sorting of socks of varying shades. The design of this experiment and the statistical analysis of the results have been completed by Dr. Collette Mann.

3.6.1 Experimental Setup

There were 14 participants recruited for this experiment, with an equal number of male and females. The test required participants to sit in front of a table while wearing the Hatpack, and attempt to sort a jumbled pile of 36 socks according to shade (black, white or grey). During the testing the Hatpack was set to one of four different mode combinations, with either manual or automatic thresholding and at a dropout rate of either 0% or 50%. The test setup is shown in Figure 3.11.

Participants were given 5-10 minutes of accommodation time prior to beginning the test. In this time they were provided with only one of each shade of sock. They were encouraged to try and establish their own method of shade discrimination, and were allowed to switch between manual and automatic thresholding modes freely upon request. Following the accomodation time, the Hatpack display was turned off and all 36 socks were placed in a jumbled pile on the table towards the left of the participant. The participant had their hand guided by one of the testers to the location of the pile. They were instructed that only two socks could be picked up from the pile at a time. At least one sock needed to be sorted before the participant was allowed to pick up another, alternatively they could replace a sock to the pile and pick up another. When the subject was ready, the Hatpack display was turned on and the test would begin. Placement of the socks into the three separate piles

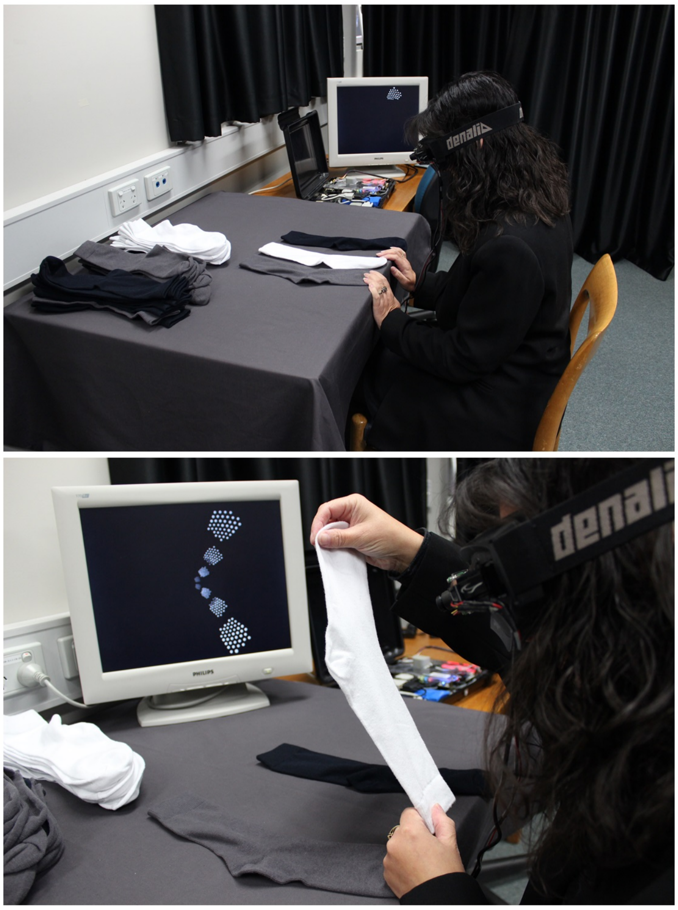

Figure 3.11: Sock test setup

was completed by the tester, after being handed the sock by the participant with a verbal indication of what they thought the shade was. Participants were given a verbal warning from the tester when only two socks were left on the table. There were four different combinations of two test conditions (threshold mode and dropout percentage). The order that the participant would be tested was chosen from a list of 16 possibilities, and so none of the participants were tested in the same order. A total of 144 trials were completed by each subject, i.e. 36 socks sorted 4 times. Participants were allowed to take a rest at any point during the experiment, but not during a test that had already begun.

3.6.2 Results

The statistical analysis of the results of this experiment has been completed by Dr. Collette Mann, using IBM SPSS Statistics version 20. Three sets of data were analysed; Response Accuracy as a function of Sock Shade, Response Accuracy as a function of Luminance Threshold Mode for both Dropout conditions, and Response Time as a function of Dropout for both Thresholding conditions. General Linear Model was used for all three analyses.

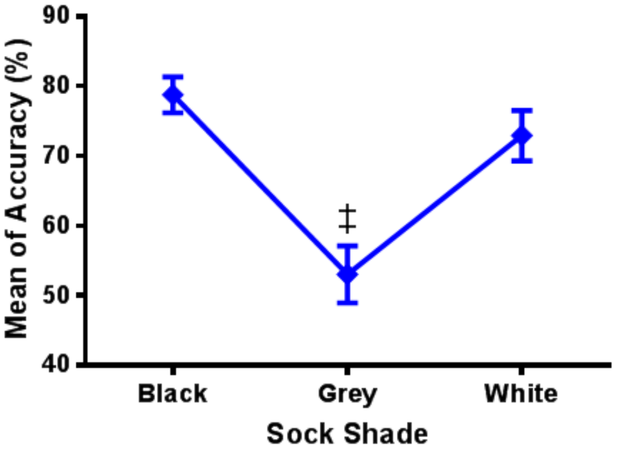

Figure 3.12: Graph of response accuracy as a function of sock shade for sock test

Response Accuracy (Sock Shade)

A one-way repeated measures analysis of variance (ANOVA) was conducted to determine if there were significant differences between the three sock shades (black, white, grey) for participants accurately sorting socks. A test of within-subjects showed a main effect for accuracy ($p<0.001$) but no interactions between threshold mode and dropout. Pairwise comparisons showed that grey socks were significantly harder to identify than black or white, this is shown in Figure 3.12 ($p<0.001$). There were no significant differences in identifying black from white socks however ($p=0.432$).

Response Accuracy (Luminance Threshold Mode)

A one-way ANOVA was conducted to determine if there were significant differences between the two thresholding modes (manual and automatic) for participants accurately sorting socks. Figure 3.13 shows the results for accuracy as a function of luminance threshold mode for each of the dropout rates. Pairwise comparisons showed no significant difference between Automatic and Manual thresholding modes ($p>0.05$). Dropout rate did not affect performance in Automatic mode ($p=0.208$). Dropout rate was significantly different in Manual mode with 50% dropout rate more difficult than no dropout in Manual mode ($p=0.04$).

Response Time (Dropout Mode)

A one-way ANOVA was conducted to determine if there were significant differences between the two dropout rates (0% and 50%) for participants accurately sorting socks. Figure 3.14 shows the results for response time as a function of dropout rate for each of the threshold modes. Pairwise comparisons showed that with both modes

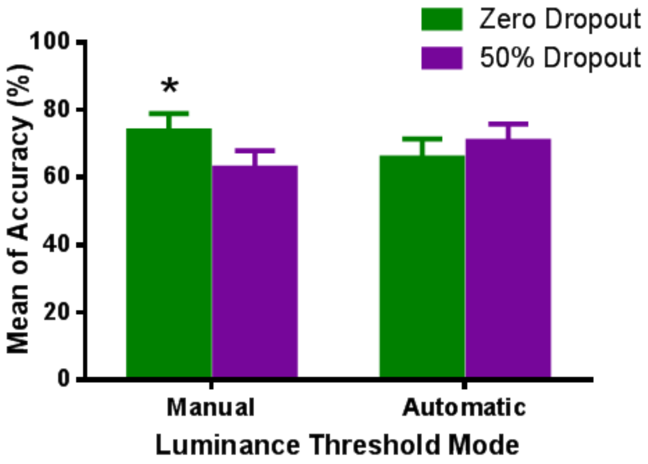

Figure 3.13: Graph of response accuracy as a function of threshold mode for sock test

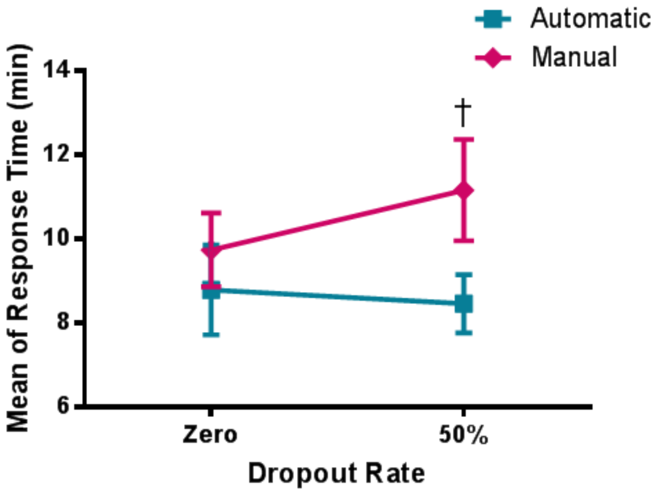

Figure 3.14: Graph of response time as a function of dropout rate for sock test

at 50% dropout rate, correct identification was significantly quicker in Automatic than Manual (p=0.006).

3.6.3 Discussion

Grey socks were significantly harder to identify than the other two shades. This was expected as the phosphenes were presented in binary intensity form and depending on the threshold, the grey socks would show up as either intensity. From observations during the testing, some participants were able to recognise this peculiarity and use it to distinguish the difference. The strategy that appeared to work most well involved putting the sock down on the table on its own and scanning over it slowly. Since the colour of the cloth on the top of the table was of a similar intensity to the grey sock, a grey sock would often be indistinguishable from the tablecloth regardless of the threshold used. White and black socks however would contrast with the table cloth and adjusting the threshold or allowing the automatic thresholding to readjust would usually allow the determination of the sock being either black or white. Another method employed by participants was to hold the socks in the air and try to determine its shade with contrast to the white wall or black curtain that was in view. This seemed less effective as grey socks would appear changed in appearance depending on threshold. However, one user was able to notice a 'shimmering' effect when viewing a grey sock and was able to consistently distinguish the shades with this method rather quickly. It is likely that if subjects were informed of the most suitable methods, they may have performed better. There was no difference between automatic and manual modes for accuracy. This was unexpected as it was thought that the manual mode would provide more accuracy. The 50% dropout rate did worsen accuracy significantly, but only for the manual mode. For the automatic mode it had the opposite effect, improving accuracy, however this was not significant. Methods used for distinguishing sock shades could be the cause of this. For response time, the automatic mode showed a significant improvement over manual for 50% dropout. This was expected due to the time it takes to adjust the threshold by hand and loss of the use of that hand during the process. While there was not a significant difference for the 0% dropout rate, it seemed to be trending in the same way. Overall, manual mode seems to be less beneficial, however this cannot be confirmed from the results of this experiment. User preference seemed to be towards the automatic in this experiment, which is the opposite of the fourth experiment. Effectiveness of either mode could be dependent on circumstances.

3.7 Summary and Conclusions

This chapter has presented five different psychophysics experiments that have been carried out using the Hatpack system. The testing investigated the abilities of normally sighted individuals to complete a range of different tasks that resembled everyday activities, while limited by the low vision expected of a visual prosthesis.

The first of the experiments involved a mobility and orientation task requiring navigation through a maze of obstacles. The results of this test suggested that reduction in greyscale intensity resolution and frame rate increases the difficulty of navigational and obstacle avoidance tasks. However this was not a significant result, possibly due to the small size of the maze and obstacles being 'too obvious'. There was a significant difference between the control mode and all other modes though, with the time taken for participants to complete the maze being more than double. This shows that there is much room for improvement. A learning effect could have also attributed to similar performance during the 'harder' modes.

The second experiment was a contrast discrimination and object manipulation task requiring the placement of chess pieces on a chessboard. The results of this test showed that the binary luminance thresholding modes caused a significant increase in the time required to complete the task. This shows the importance of having the ability to present multiple phosphene intensity levels. A considerable learning effect was observed over repeated testing. This indicates that bionic vision users may need significant practice and training once implanted as everyday tasks may need familiarisation with the newly provided visual input.

The third experiment investigated the limitations of low frame rate on interaction with a dynamic environment. The test involved participants attempting to intercept rolling balls. Significant increases in performance were observed as the frame rate was increased. This highlights the need for achieving higher stimulation framerates in vision prostheses. Also observed was a mismatch between the subjects' arm location and the perceived location of the ball. This may mean that new prosthesis patients will require training and practice to 'calibrate' themselves with the visual input.

The fourth experiment was a contrast and object discrimination task requiring the user to identify shapes of different shades. The results showed that the static luminance thresholding mode was significantly worse than manual or automatic. This highlights the need for the ability to change thresholds during use of a bionic vision device. There was no significant difference shown between the other two modes though, and so neither could be concluded as better. More and varied testing involving physical interaction with the environment may allow for that conclusion.

The fifth experiment was another contrast discrimination task, but involved interaction with the environment. Participants were required to sort socks of different shade. The results showed that grey socks were significantly harder to identify than the other two shades. This is a further indication of the need for multiple phosphene intensity levels in a prosthesis. However the results of this test show that if a device was limited to just two levels, there are ways to allow such abilities through the exploitation of the thresholding functionality. Standardised methods and training may be required though to allow for this. Manual mode was indicated as less beneficial in this test, but this is likely to be dependent on the circumstances of the situation and environment.

The completion of these experiments have highlighted the need for more greyscale intensity resolution, higher framerate, more advanced thresholding implementations, and post-operative training so that patients can get the most out of their implant. While the margin of errors observed in comparisons of the modes in these tests were mostly too large to allow firm conclusions to be made, a statistically significant difference was consistently observed between control modes and non-control modes.

This shows that completing tasks under limited vision anticipated of a visual prosthesis is significantly harder and that there is a need for further research into ways of reducing the complexity of such tasks in order to provide better quality of life for patients. Different approaches are likely to be needed in order to achieve this and close the gap between the abilities of normally sighted individuals and those with the low vision of a bionic eye. Observations made over the course of this work and modification of the Hatpack system has highlighted the limitations of purely visual sensing. Poor lighting conditions resulted in poor representations of the scene due to low contrast. On the other hand, suitably lit environments that were very cluttered or consisted of high contrast patterns often resulted in a confusing representation at low resolution. If for example the area used for the first experiment were to consist of carpet that had highly contrasting squares on it, it is very likely that the participant would be unable to differentiate between the carpet and obstacles, especially since there is no ability to convey depth information. This has motivated an investigation into the use of alternate sensing technology, which could possibly provide solutions to such issues. An investigation into advanced sensing integration and implementation of more advanced processing techniques has been completed and is presented in the chapters that follow.

Chapter 4

Least Squares Plane Fitting and Depth Sensor Error Characterisation

The results of the psychophysics tests outlined in Chapter 3 indicate that the navigational abilities with 2D visual sensing alone may not be sufficient in some environments. Poor lighting conditions, cluttered environments, and undesired peripheral content can quite easily affect a 2D camera image, resulting in a confusing bionic vision representation. The 3D domain allows exploitation of structural information in a scene, avoiding many of these issues and allowing a clearer environmental representation. Depth sensors, such as the Microsoft Kinect v2, provide a low cost entry point into the 3D sensing domain. These sensors provide information about the distance to objects in the sensed environment in the form of dense point cloud data. This data has the necessary accuracy, resolution and range to allow for the implementation of more sophisticated vision processing. For example highlighting pixels in a bionic eye corresponding to a floor plane is an effective aid for blind navigation in cluttered environments.

This chapter presents the development of a plane fitting algorithm that is suitable for implementation on an FPGA platform with depth data input provided by a Kinect v2 sensor. Also presented is an analysis and characterisation of sensor error and the propagation of that error to the fitting of planes. The content of this chapter provides the necessary foundations for the hardware implementation and applications that are discussed in the chapter that follows.

4.1 Introduction

Quick and effective navigation through an environment is often reliant on the knowledge of surfaces and objects within it and the available free space to move among them. For a normally sighted person, this knowledge is typically gained through vision. Upon view of a new scene, a normally sighted person can almost immediately identify the locations of various paths and obstacles and build a map within their minds. They can then traverse this map as they see fit. For blind individuals this is not the case and they must rely on other senses to build up this map. This is usually a more lengthy and possibly dangerous task and a dynamic environment can prove even more difficult.

The advent of visual prosthesis devices could allow blind individuals to regain this visual mapping ability. Unfortunately the naivety of early prostheses may mean that identification of objects and surfaces is difficult. Furthermore, first generation prostheses provide no perception of the depth of objects in view. Therefore detection of surfaces and objects in the scene through other means could assist the user in navigation. The detection of planes and planar objects is one way to achieve this. Planes are prolific in everyday environments. Floors, walls, doors, windows, stairs, tables, chairs, bookshelves, electronic screens are but a few examples of the types of planar surfaces and objects that one would come across in a typical day. Detecting planes using 2D sensing however is not feasible, and so an extension into the 3D domain is required.

Other work has explored the use of Microsoft Kinect depth sensors [67, 68, 71, 106, 108]. This sensor has a compact design, low cost, considerably less power consumption than industrial 3D sensing systems, and has a vast open source software support network. The second version Kinect offers higher depth image resolution (512 by 424 pixels), more accurate time-of-flight depth sensing and increased standard operating range (0.5 m to 4.5 m), making it a suitable choice for 3D plane fitting. In order to integrate 3D sensing in a bionic eye system, two main problems need to be solved. Firstly, there is currently no support for direct connection of Kinect sensors to FPGA development boards. Secondly, a solely software based implementation would preclude the aims of low latency, high frame rate operation. A unified computer-FPGA architecture can achieve these goals. This will be discussed further in Chapter 5.

The application of plane detection to object recognition and navigational tasks is likely to prove very useful for a bionic vision user. Some good examples of such applications are the plane based staircase detection work completed by [108] and the low contrast hazard avoidance work by [75] which involves the detection of ground planes. The plane detection implementations presented in these works incorporate variations of RANSAC, an iterative method that estimates parameters of a mathematical model from a set of observed data containing outliers [44]. This algorithm is unsuitable for the purposes of this work due to the processing time. Instead, a least squares plane fitting method [92] has been investigated for implementation. The implementation involves fitting planes to localised patches around points in depth data. This plane fitting method is highly parallelisable and able to be reduced to a form involving integer additions, multiplications and a few divide operations.

In this chapter the groundwork for the implementation of real time hardware-based plane fitting is presented. An algorithm is developed based on the least squares method of plane fitting. The sensor error of a Kinect v2 is analysed, and error models for the various plane fitting parameters are derived in order to specify logic precision requirements, statistical testing of the validity of a plane fit, and achievable plane fitting angle resolution.

4.2 Least Squares Plane Fitting of Depth Image Data

Given a set of depth image points $\{u_i, v_i, z_i\}(where\ i = 1, \ldots, m)$, it is possible to find a plane of the form $z = Au + Bv + C$ where u and v are image coordinates and z is the depth coordinate measured at that image location (see Figure 4.1). With a depth sensor such as the Kinect v2 there is a unique z for each (u, v), therefore the plane z can be expressed as a function of u and v.

$$z = Au + Bv + C \tag{4.1}$$

As mentioned in the chapter introduction, the least squares algorithm is based on the minimisation of the residuals of the points from the proposed curve or surface, a plane in this case. The sum of the squared errors between z_i and the plane values $(Au_i + Bv_i + C)$ is minimised using the following error function:

$$E(A, B, C) = \sum_{i=1}^{m} (z_i - (Au_i + Bv_i + C))^2 \tag{4.2}$$

The error function in Equation (4.2) is quadratic in $A,\ B,\ C$ therefore there will be only one minimum. This enables the global minimum to be determined analytically. The minimum of this error function will occur when the gradient, with respect to $[A\ B\ C]^T$, is equal to zero (ie. $\nabla E = [0, 0, 0]$):

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \nabla E = -2\sum_{i=1}^{m} \left((z_i - (Au_i + Bv_i + C)) \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \right) \tag{4.3}$$

To allow for the solution of A, B, C, Equation (4.3) can be rewritten as the following system of linear equations:

$$\begin{bmatrix} \sum_{i=1}^{m} u_i^2 & \sum_{i=1}^{m} u_i v_i & \sum_{i=1}^{m} u_i \\ \sum_{i=1}^{m} u_i v_i & \sum_{i=1}^{m} v_i^2 & \sum_{i=1}^{m} v_i \\ \sum_{i=1}^{m} u_i & \sum_{i=1}^{m} v_i & \sum_{i=1}^{m} 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} u_i z_i \\ \sum_{i=1}^{m} v_i z_i \\ \sum_{i=1}^{m} z_i \end{bmatrix} \tag{4.4}$$

By choosing $(u_i, v_i), \ldots, (u_m, v_m)$ to be integer coordinates inside a square mask centred at (0,0) then $\sum u_i = \sum v_i = \sum u_i v_i = 0$ and Equation (4.4) simplifies to:

$$\begin{bmatrix} \sum_{i=1}^{m} u_i^2 & 0 & 0 \\ 0 & \sum_{i=1}^{m} v_i^2 & 0 \\ 0 & 0 & \sum_{i=1}^{m} 1 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} u_i z_i \\ \sum_{i=1}^{m} v_i z_i \\ \sum_{i=1}^{m} z_i \end{bmatrix} \tag{4.5}$$

The first matrix in Equation (4.5) is a diagonal matrix and so its inverse is also a diagonal. The solution of the equation becomes:

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \frac{1}{\sum_{i=1}^{m} u_i^2} & 0 & 0 \\ 0 & \frac{1}{\sum_{i=1}^{m} v_i^2} & 0 \\ 0 & 0 & \frac{1}{\sum_{i=1}^{m} 1} \end{bmatrix} \begin{bmatrix} \sum_{i=1}^{m} u_i z_i \\ \sum_{i=1}^{m} v_i z_i \\ \sum_{i=1}^{m} z_i \end{bmatrix} \tag{4.6}$$

Finally, A, B and C can be expressed as follows:

$$A = \frac{\sum_{i=1}^{m} u_i z_i}{\sum_{i=1}^{m} u_i^2} \qquad B = \frac{\sum_{i=1}^{m} v_i z_i}{\sum_{i=1}^{m} v_i^2} \qquad C = \frac{\sum_{i=1}^{m} z_i}{n} \tag{4.7}$$

4.2.1 Sparsely Sampled 11 by 11 Patch Implementation

The implementation developed fits planes to a patch of pixels surrounding a particular data point. This process is carried out for all points in the depth image frame. A patch size of $(2k + 1)$ by $(2k + 1)$ depth pixels is used. Only 9 sparse points are utilised from this patch for the plane fit. The reason for this is that when using a small patch size, such as 3 by 3 ($k = 1$), fitted planes can very easily be influenced by noise. This is due to the short base line used to estimate a plane normal. Conversely, while using a larger patch size may mitigate these effects, the implementation of the least squares operations in hardware becomes much more expensive due to the larger number of points. A suitable compromise is reached by choosing an 11 by 11 patch size ($k = 5$) and sampling a subset of those points with sufficient redundancy to provide a robust plane fit. In Section 4.3.4 it is shown that this patch size results in small angle errors in fitted plane normals. The patch for $k = 5$ is shown below.

$$\begin{bmatrix} z_{(1,1)} & \cdots & z_{(1,6)} & \cdots & z_{(1,11)} \\ \vdots & \ddots & \vdots & & \vdots \\ z_{(6,1)} & \cdots & z_{(6,6)} & \cdots & z_{(6,11)} \\ \vdots & & \vdots & \ddots & \vdots \\ z_{(11,1)} & \cdots & z_{(11,6)} & \cdots & z_{(11,11)} \end{bmatrix} \tag{4.8}$$

The sparsely sampled 3 by 3 matrix is constructed using the 9 patch points as follows:

$$\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix} = \begin{bmatrix} z_{(1,1)} & z_{(1,6)} & z_{(1,11)} \\ z_{(6,1)} & z_{(6,6)} & z_{(6,11)} \\ z_{(11,1)} & z_{(11,6)} & z_{(11,11)} \end{bmatrix} \tag{4.9}$$

Assuming a patch centred at (0,0), the corresponding u and v indices are:

$$u\ indices : \begin{bmatrix} -k & 0 & k \\ -k & 0 & k \\ -k & 0 & k \end{bmatrix}$$

$$v\ indices : \begin{bmatrix} -k & -k & -k \\ 0 & 0 & 0 \\ k & k & k \end{bmatrix} \tag{4.10}$$

$$where\ k = \frac{patch\ length - 1}{2} = 5$$

The 9 points are evenly spaced and share the same centre as the patch, therefore the equations of (4.7) are still applicable. It is also possible to further simplify the A and B equations. The following are the revised equations for A, B and C (where rows and columns are with respect to the sparsely sampled 3 by 3 matrix from (4.9), and $n = 9$).

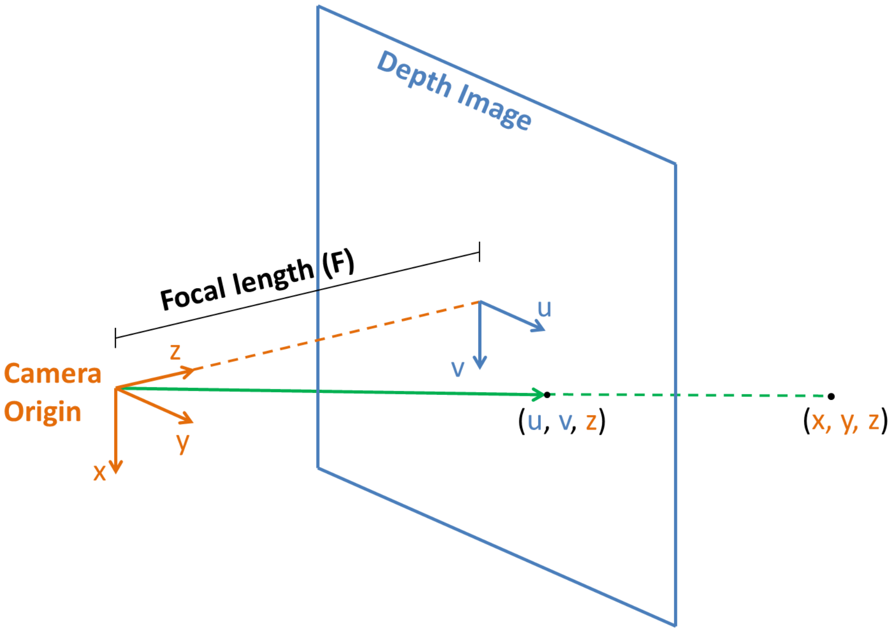

Figure 4.1: Relationship between depth image coordinates (u, v, z) and world coordinates (x, y, z)

$$A = \frac{\sum_{i=1}^{m} u_i z_i}{\sum_{i=1}^{m} u_i^2} = \frac{(\sum(column\ 3) - \sum(column\ 1))}{6k}$$

$$B = \frac{\sum_{i=1}^{m} v_i z_i}{\sum_{i=1}^{m} v_i^2} = \frac{(\sum(row\ 3) - \sum(row\ 1))}{6k} \tag{4.11}$$

$$C = \frac{\sum_{i=1}^{m} z_i}{n} = \frac{\sum_{i=1}^{9} z_i}{9}$$

A further advantage of this approach is that since the denominator is constant, division can be replaced by a multiplication by the pre-calculated inverse, which is preferable as multiplication is more efficient than division.

4.2.2 Transformation to World Coordinates

Input depth data points will be provided by a Microsoft Kinect v2 sensor. The coordinate frame units of this data has the u and v axes expressed in pixel integers and the z axis range values in millimetres. For consistency these are transformed to a world coordinate frame. Figure 4.1 shows x, y, z in the world coordinates in millimetres and u and v are the pixel integer index values of the depth image plane at focal point of F from the depth sensor. A pinhole camera model can be used

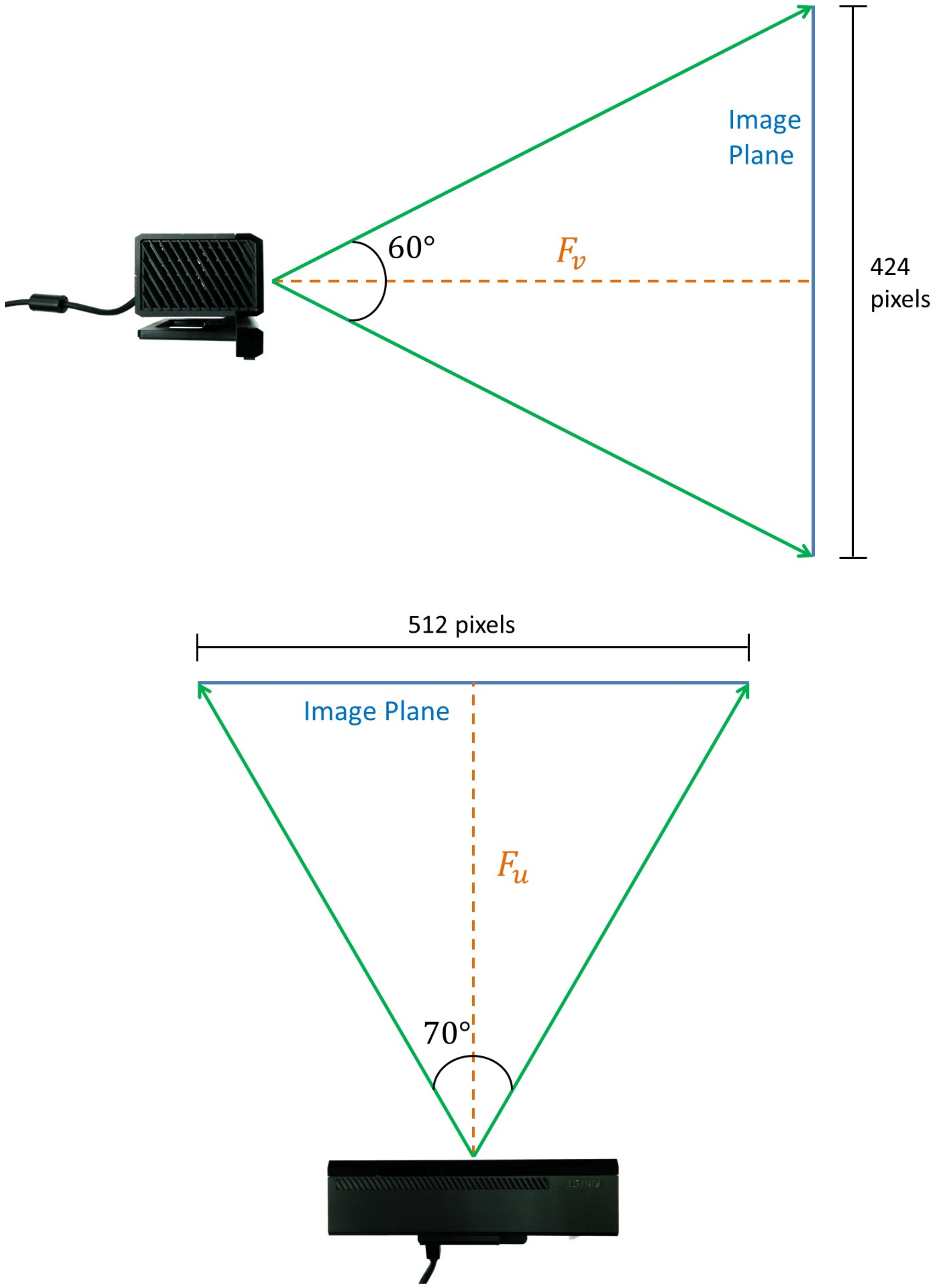

Figure 4.2: Relationship between field of view, image height and width, and focal values of the Kinect v2 sensor

to calculate the focal point. Given that the field of view of the Kinect v2 sensor is specified to be 70° and 60° (horizontal and vertical axes respectively) with pixel width and height $w = 512$ and $h = 424$, the value of F can be found as follows (refer to Figure 4.2):

$$\begin{aligned} For\ the\ horizontal\ axis: \qquad F_u &= \frac{w}{2tan\left(\frac{70}{2}\right)} = 365.6\ pixels \\ For\ the\ vertical\ axis: \qquad F_v &= \frac{h}{2tan\left(\frac{60}{2}\right)} = 367.2\ pixels \end{aligned} \tag{4.12}$$

The u and v axes focal values F_u and F_v in Equation 4.12 are different. Ideally these values should be equal for square pixels. To allow for rounding errors in the nominal 70° and 60° field of view values, an average of the u and v axes focal values is used.

$$F = \frac{F_u + F_v}{2} \approx 366\ pixels \tag{4.13}$$

The principle of similar triangles can be used to find the following scaling factors that relate u and v to x and y.

$$u = \frac{F}{z}x \qquad\qquad v = \frac{F}{z}y \tag{4.14}$$

A plane can be fitted to an image patch centred around some point (u_c, v_c) using the equations from (4.11). The relative patch pixel coordinates are:

$$u_p = u - u_c \qquad\qquad v_p = v - v_c \tag{4.15}$$

Where u and v are the pixel image coordinates. The fitted plane equation is then:

$$z = A(u - u_c) + B(v - v_c) + C \tag{4.16}$$

Substituting the equations from (4.14) gives the following equation, which is now expressed in terms of the world coordinates x, y and z.

$$z = \left(\frac{AF}{z}\right)x + \left(\frac{BF}{z}\right)y + C - u_cA - v_cB \tag{4.17}$$

4.2.3 Plane Normal Vector

A normal vector can be used to characterise the orientation and distance of a fitted plane from the origin. Rearranging Equation (4.17) to the form $f(x, y, z) = 0$, gives:

$$z.(z + Au_c + Bv_c - C) - AFx - BFy = 0 \tag{4.18}$$

A normal vector to the surface defined by $f(x, y, z) = 0$ can be represented by the gradient $\nabla f(x, y, z)$.

$$N = \nabla f(x, y, z) = \begin{bmatrix} \frac{\partial F}{\partial x} \\ \frac{\partial F}{\partial y} \\ \frac{\partial F}{\partial z} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} \tag{4.19}$$

$$\nabla f(x, y, z) = -AF\hat{\imath} - BF\hat{\jmath} + (2z + Au_c + Bv_c - C)\hat{k} \tag{4.20}$$

Where $\hat{\imath}$, $\hat{\jmath}$, $\hat{k}$ are unit vectors along the x, y and z axes. Evaluating $\nabla f(x, y, z)$ at $z = C$ (the best fit value of the patch centred at (u_c, v_c)) gives:

$$\nabla f(x, y, z) = -AF\hat{\imath} - BF\hat{\jmath} + (C + Au_c + Bv_c)\hat{k} \tag{4.21}$$

Therefore at $z = C$, the normal vector $[\ \alpha\ \beta\ \gamma\ ]^T$ of the fitted plane will be:

$$N = \nabla f(x, y, z) = \begin{bmatrix} \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} -AF \\ -BF \\ (C + Au_c + Bv_c) \end{bmatrix} \tag{4.22}$$

Finally, the equation of a plane at $z = C$ with normal vector N can be expressed as follows, where κ is some constant:

$$\alpha x + \beta y + \gamma z = \kappa \tag{4.23}$$

Through substitution of (4.14) and (4.22) into (4.23), the value of κ can be written as follows:

$$\begin{aligned} \kappa &= (-AF)\left(\frac{Cu_c}{F}\right) + (-BF)\left(\frac{Cv_c}{F}\right) + (C^2 + ACu_c + BCv_c) \\ &= C^2 \end{aligned} \tag{4.24}$$

There are now equations defined for all of the plane parameters α, β, γ and κ. In order to use these plane parameters to detect planes of specific orientation and offset from the origin, the parameters need to be normalised. The geometric method of normalisation would require a square root function, a sum of squared plane parameter values and a division, which is expensive to implement in hardware. Instead it has been chosen to normalise the plane parameters by dividing by the sum of absolute values of α, β and γ referred to as the L_1 norm (4.25). While this approach may give significant errors in normalisation, this is not important to the result of this work because a planar region would all have the same normalisation error, and would have the same calculated equation.

$$\lambda = |\alpha| + |\beta| + |\gamma| \tag{4.25}$$

The normalised plane parameters are:

$$\begin{bmatrix} \alpha_n \\ \beta_n \\ \gamma_n \\ \kappa_n \end{bmatrix} = \frac{1}{\lambda} \begin{bmatrix} \alpha \\ \beta \\ \gamma \\ \kappa \end{bmatrix} = \left[\frac{1}{|\alpha| + |\beta| + |\gamma|}\right] \begin{bmatrix} -AF \\ -BF \\ (C + Au_c + Bv_c) \\ C^2 \end{bmatrix} \tag{4.26}$$

4.3 Error Characterisation of Depth Sensor

The hardware implementation of the plane fitting algorithm (see Chapter 5) makes use of fixed point arithmetic for simpler, faster logic compared to floating point. Fixed point and floating point operations (multiplies and divides) can produce results with more bits than the operands, therefore information loss is a possibility. An appropriate level of logic precision has to be defined in order for the solution to provide satisfactory results. Since the input to the algorithm is in integer form, this means defining an appropriate number of fractional bits for any further operations. An analysis and characterisation of the sensor errors is useful for optimising the number of fractional bits required. Futhermore, by modelling the error in the plane fit parameters, suitable thresholds can be defined for the rejection of non-plane patches as well as the acceptable difference thresholds when matching parameters for the detection applications that are discussed in Chapter 5. The following section outlines the analysis and characterisation of the sensor data and the plane fitting errors.

4.3.1 Depth Value (Range) Error

The depth value error of a Kinect v1 sensor was analysed by Khoshelham and Elberink via the use of the RANSAC algorithm to fit planes to depth data of a door at varying distances [60]. Pinto et al. have used the same method to compare the depth value error of three sensors, one of which being a Kinect v2 [87]. In a similar manner, the depth value error of the Kinect v2 sensor has been analysed in this work, albeit using a least squares method for plane fitting. Note that the method used is a best-case analysis of the sensor error as the plane used is facing directly towards the sensor and data points are sampled centrally within the image. Furthermore the plane used was a white surface and so reflectance of IR light was optimal.

Depth images have been captured of a scene with a large, flat planar surface (a whiteboard) in view. The deviations of the whiteboard from a perfect plane have been characterised using a metre-long steel straight edge (see Figure 4.3). By moving and rotating the straight edge at different angles on the whiteboard the deviations from a plane are estimated to be less than 1 millimetre. Kinect v2 depth images were captured at intervals of 0.25 m from distances ranging from 0.75 m to 7.75 m from the planar surface. In MATLAB dense plane fitting was performed on regions of each image that made up the whiteboard to produce a ground truth for the measured data. Difference error values were calculated using this ground truth. The sample autocorrelation function was computed for the residual errors of a row and column of the whiteboard plane measured at about 3 metres and plotted with 95% confidence intervals (see Figures 4.4 and 4.5). It can be seen in both figures that there is strong correlation in neighbouring residuals up to two pixels away. However, at a lag of 5 pixels there is little correlation. So for a selection of $k = 5$, patch depth values are statistically independent. The sample autocorrelation function was also computed for various distances from 0.75 m to 7.75 m to confirm for a lag of 5 pixels there is no dependency on range. In the first graph of Figure 4.6 a parabolic fit to the standard deviation of errors with distance from the sensor is shown. The quadratic error fit also applies to data from a 30° inclined plane as shown in the second graph of Figure 4.6. The results gained are quite similar and support the work of Pinto et al. [87].

4.3.2 Resultant Error Models for Plane Fitting Implementation

Having found the mean and standard deviation of depth value error, models for the errors in the parameters of the plane fitting algorithm can be developed. The following outlines briefly the derivation of error models for the algorithm previously

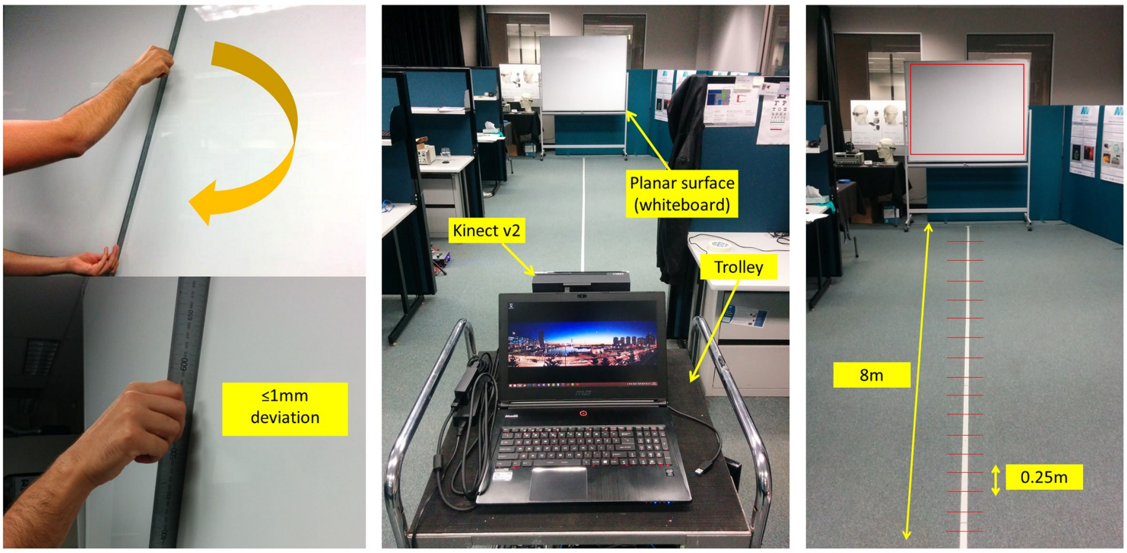

Figure 4.3: Collection of statistical data for range error analysis

presented. A more detailed derivation of equations (4.38) to (4.41) in this chapter are provided in Appendix A.

As shown previously in Section 4.2, for a plane with the form $z = Au + Bv + C$, the parameters A, B, and C can be found using the following simplified equations from (4.7).

$$A = \frac{\sum_{i=1}^{m} u_i z_i}{\sum_{i=1}^{m} u_i^2} \qquad B = \frac{\sum_{i=1}^{m} v_i z_i}{\sum_{i=1}^{m} v_i^2} \qquad C = \frac{\sum_{i=1}^{m} z_i}{n} \tag{4.27}$$

The z values are assumed to be mutually independent normal random variables and so the variance and standard deviation of the above parameters can be found as follows. A, B and C can be represented as a linear combinations of z_i:

$$\begin{aligned} A &= \frac{1}{\sum_{i=1}^{m} u_i^2} \left(\sum_{i=1}^{m} u_i z_i \right) \\ B &= \frac{1}{\sum_{i=1}^{m} v_i^2} \left(\sum_{i=1}^{m} v_i z_i \right) \\ C &= \frac{1}{n} \sum_{i=1}^{m} z_i \end{aligned} \tag{4.28}$$

Therefore, assuming $z_i = (z_{true} + e_i)$, where $e_i \sim N(0, \sigma_z)$, all estimates of A, B, and C will have zero mean errors and variances as follows:

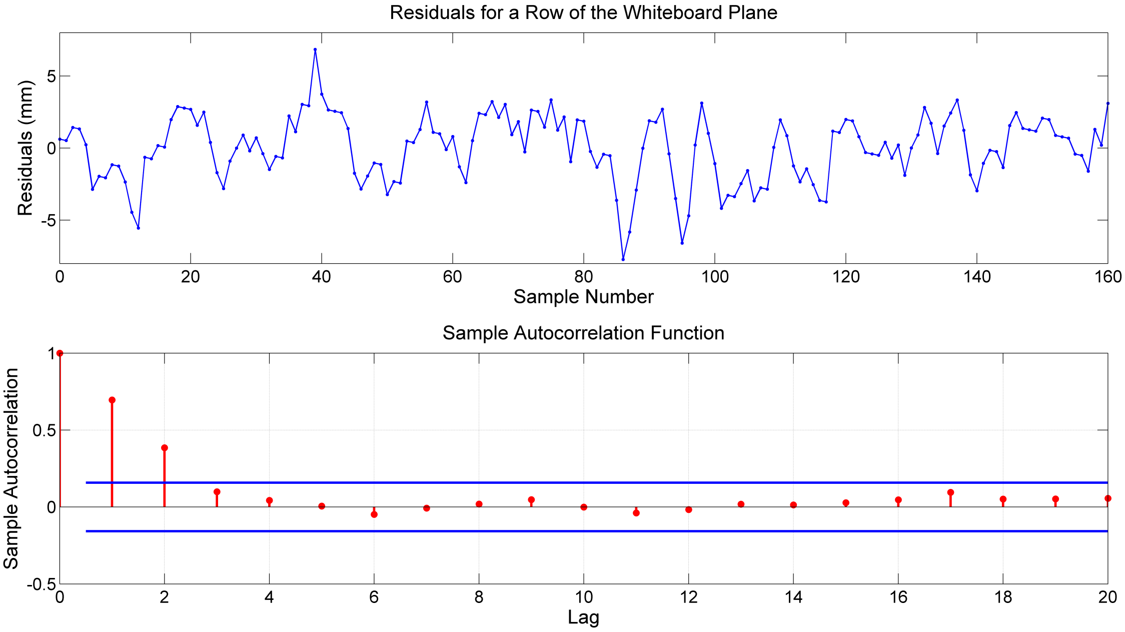

Figure 4.4: Autocorrelation function plot for a row of the whiteboard at a distance of about 3 metres

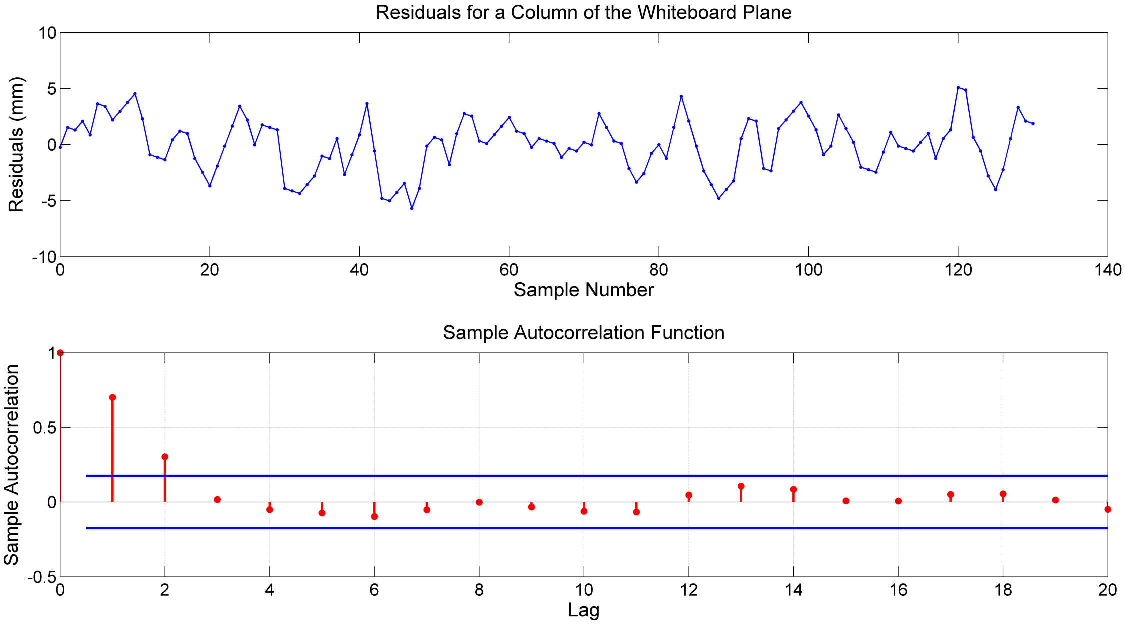

Figure 4.5: Autocorrelation function plot for a column of the whiteboard at a distance of about 3 metres

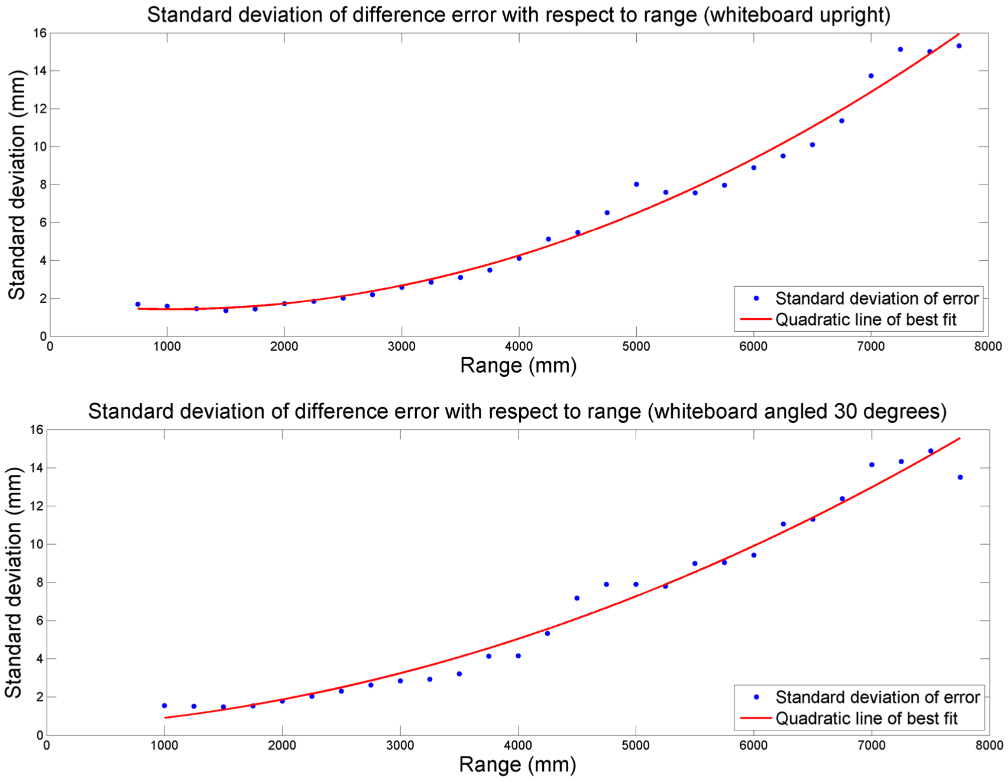

Figure 4.6: The graphs show the measured standard deviations of difference errors with respect to range for the whiteboard facing towards the sensor plane (top graph) and rotated 30° upwards from the sensor plane.

$$
\begin{aligned}
\sigma_A^2 &= \left[\frac{1}{\sum_{i=1}^m u_i^2}\right]^2 \left(\sum_{i=1}^m u_i^2 \sigma_i^2\right) = \frac{\sigma_z^2}{\sum_{i=1}^m u_i^2} \\
\sigma_B^2 &= \left[\frac{1}{\sum_{i=1}^m v_i^2}\right]^2 \left(\sum_{i=1}^m v_i^2 \sigma_i^2\right) = \frac{\sigma_z^2}{\sum_{i=1}^m v_i^2} \\
\sigma_C^2 &= \frac{1}{n^2} \sum_{i=1}^m \sigma_i^2 = \frac{\sigma_z^2}{n}
\end{aligned}
\tag{4.29}
$$

and standard deviations σ_A, σ_B and σ_C

From Equation (4.10) the standard deviations of A and B can be expressed in terms of k as follows:

$$\begin{aligned}\sigma_A &= \frac{\sigma_z}{k\sqrt{6}} \\ \sigma_B &= \frac{\sigma_z}{k\sqrt{6}} \\ \sigma_C &= \frac{\sigma_z}{3}\end{aligned} \tag{4.30}$$

From Equation (4.30) it is apparent that the standard deviation of errors scales inversely with patch size. Therefore larger patch sizes reduce errors. From Equation (4.22) it can be seen that α and β are scaled versions of A and B and so the variances follow as scaled versions of Equation (4.29):

$$\begin{aligned}\sigma_\alpha^2 &= Var[-AF] = F^2\frac{\sigma_z^2}{\sum_{i=1}^m u_i^2} = F^2\frac{\sigma_z^2}{6k^2} \\ \sigma_\beta^2 &= Var[-BF] = F^2\frac{\sigma_z^2}{\sum_{i=1}^m v_i^2} = F^2\frac{\sigma_z^2}{6k^2}\end{aligned} \tag{4.31}$$

$$\textit{and standard deviations } \sigma_\alpha \textit{ and } \sigma_\beta$$

To find the variance of γ the expression in Equation (4.22) needs to be expanded as follows, so that all correlated terms are grouped.

$$\begin{aligned}\gamma &= C + Au_c + Bv_c \\ &= \frac{\sum_{i=1}^m z_i}{n} + u_c\frac{\sum(column3) - \sum(column1)}{30} \\ &\quad + v_c\frac{\sum(row3) - \sum(row1)}{30} \\ &= z_1(\frac{1}{9} + \frac{u_c}{30} - \frac{v_c}{30}) + z_2(\frac{1}{9} - \frac{v_c}{30}) + z_3(\frac{1}{9} + \frac{u_c}{30} - \frac{v_c}{30}) \\ &\quad + z_4(\frac{1}{9} - \frac{u_c}{30}) + z_5(\frac{1}{9}) + z_6(\frac{1}{9} + \frac{u_c}{30}) \\ &\quad + z_7(\frac{1}{9} - \frac{u_c}{30} + \frac{v_c}{30}) + z_8(\frac{1}{9} + \frac{v_c}{30}) \\ &\quad + z_9(\frac{1}{9} - \frac{u_c}{30} + \frac{v_c}{30})\end{aligned} \tag{4.32}$$

By choosing the worst case values for u_c and v_c of 256 and 212 respectively, then the variance can be expressed as follows:

$$\begin{aligned}
\sigma_\gamma^2 &= \sigma_z^2(\frac{-697}{45})^2 + \sigma_z^2(\frac{-313}{45})^2 + \sigma_z^2(\frac{71}{45})^2 \\
&+ \sigma_z^2(\frac{-379}{45})^2 + \sigma_z^2(\frac{1}{9})^2 + \sigma_z^2(\frac{389}{45})^2 \\
&+ \sigma_z^2(\frac{-61}{45})^2 + \sigma_z^2(\frac{323}{45})^2 + \sigma_z^2(\frac{707}{45})^2 \\
&= \sigma_z^2(\frac{33149}{45}) \\
&\approx (27.14\sigma_z)^2
\end{aligned} \tag{4.33}$$

For the variance of κ, the following approximation can be used:

$$\begin{aligned}
&For\ some\ f = aX^b \\
&\sigma_f^2 \approx (abX^{b-1}\sigma_X)^2 = (\frac{fb\sigma_X}{X})^2
\end{aligned} \tag{4.34}$$

Giving the following variance for κ:

$$\sigma_\kappa^2 \approx (2C\sigma_C)^2 = (\frac{2C\sigma_z}{3})^2 \tag{4.35}$$

From equations (4.31,4.33,4.35) standard deviations for the four plane parameters α, β, γ and κ can be defined:

$$\begin{aligned}
\sigma_\alpha &= F\frac{\sigma_z}{k\sqrt{6}} \\
\sigma_\beta &= F\frac{\sigma_z}{k\sqrt{6}} \\
\sigma_\gamma &= \sigma_z\sqrt{\frac{33149}{45}} \approx 27.14\sigma_z \\
\sigma_\kappa &\approx \frac{2C\sigma_z}{3}
\end{aligned} \tag{4.36}$$

To find the variances of the normalised plane parameters α_n, β_n, γ_n and κ_n, the following approximation can be used:

$$\begin{aligned}
&For\ some\ f = \frac{X}{Y} \\
&\sigma_f^2 \approx f^2\left[\left(\frac{\sigma_X}{X}\right)^2 + \left(\frac{\sigma_Y}{Y}\right)^2 - 2\frac{\sigma_{XY}}{XY}\right]
\end{aligned} \tag{4.37}$$

Firstly, standard deviation equations for λ need to be defined. From equation (4.25), it is known that $\lambda = |\alpha| + |\beta| + |\gamma|$. Therefore there are 8 different cases for evaluating variance, as each of α, β, γ could have a positive or negative result. It can be shown however, that there are only 4 unique cases, and furthermore only the best and worst case variances are of interest (See Appendix A). This leaves the two following cases for λ:

$$\begin{aligned} \lambda_1 &= (\alpha) + (\beta) + (\gamma) = -FA - FB + (C + Au_c + Bv_c) \\ \lambda_2 &= (\alpha) + (\beta) + (-\gamma) = -FA - FB - (C + Au_c + Bv_c) \end{aligned} \tag{4.38}$$

Using a similar method as was done for σ_γ^2 in equations (4.32 and 4.33), the following variance equations for the two cases of λ can be found:

$$\begin{aligned} \sigma_{\lambda_1}^2 &= \frac{\sigma_z^2 53749}{225} \approx (15.456\sigma_z)^2 \\ \sigma_{\lambda_2}^2 &= \frac{\sigma_z^2 1081477}{225} \approx (69.329\sigma_z)^2 \end{aligned} \tag{4.39}$$

There is also need to find the covariances of each of the plane parameters α, β, γ and κ with the scaling factor λ. This can be done through the products of each, removing the non-correlated cross-terms (See Appendix A). The following covariances can be defined:

$$\begin{aligned} &For\ best\ case\ of\ \sigma_{\lambda_1} = 15.456\sigma_z, \\ &\sigma_{\alpha\lambda_1} = \frac{1342\sigma_z^2}{5} \approx 16.38^2\sigma_z^2 \\ &\sigma_{\beta\lambda_1} = \frac{9394\sigma_z^2}{25} \approx 19.38^2\sigma_z^2 \\ &\sigma_{\gamma\lambda_1} = -\frac{91187\sigma_z^2}{225} \approx -20.13^2\sigma_z^2 \\ &\sigma_{\kappa\lambda_1} = \frac{\sigma_z^3}{81} \end{aligned} \tag{4.40a}$$

$$\begin{aligned}
&For\ worst\ case\ of\ \sigma_{\lambda_2} = 69.329\sigma_z,\\
&\sigma_{\alpha\lambda_2} = \frac{37942\sigma_z^2}{25} \approx 38.96^2\sigma_z^2\\
&\sigma_{\beta\lambda_2} = \frac{35258\sigma_z^2}{25} \approx 37.55^2\sigma_z^2\\
&\sigma_{\gamma\lambda_2} = -\frac{422677\sigma_z^2}{225} \approx -43.34^2\sigma_z^2\\
&\sigma_{\kappa\lambda_2} = -\frac{\sigma_z^3}{81}
\end{aligned} \tag{4.40b}$$

Substituting equations (4.36, 4.39 and 4.40) into equation (4.37) gives the following equations for the variances of the normalised plane parameters α_n, β_n, γ_n and κ_n:

$$\begin{aligned}
&For\ best\ case\ of\ \sigma_{\lambda_1} = 15.456\sigma_z,\\
&\sigma_{\alpha_n}^2 \approx \sigma_z^2\left(\frac{\alpha}{\lambda_1}\right)^2\left[\frac{29.88^2}{\alpha^2} + \frac{15.46^2}{\lambda_1^2} - \frac{23.17^2}{\alpha\lambda_1}\right]\\
&\sigma_{\beta_n}^2 \approx \sigma_z^2\left(\frac{\beta}{\lambda_1}\right)^2\left[\frac{29.88^2}{\beta^2} + \frac{15.46^2}{\lambda_1^2} - \frac{27.41^2}{\beta\lambda_1}\right]\\
&\sigma_{\gamma_n}^2 \approx \sigma_z^2\left(\frac{\gamma}{\lambda_1}\right)^2\left[\frac{27.14^2}{\gamma^2} + \frac{15.46^2}{\lambda_1^2} + \frac{28.47^2}{\gamma\lambda_1}\right]\\
&\sigma_{\kappa_n}^2 \approx \sigma_z^2\left(\frac{\kappa}{\lambda_1}\right)^2\left[\frac{(0.66C)^2}{\kappa^2} + \frac{15.46^2}{\lambda_1^2} - \frac{0.024\sigma_z}{\kappa\lambda_1}\right]
\end{aligned} \tag{4.41a}$$

$$\begin{aligned}
&For\ worst\ case\ of\ \sigma_{\lambda_2} = 69.329\sigma_z,\\
&\sigma_{\alpha_n}^2 \approx \sigma_z^2\left(\frac{\alpha}{\lambda_2}\right)^2\left[\frac{29.88^2}{\alpha^2} + \frac{69.33^2}{\lambda_2^2} - \frac{55.09^2}{\alpha\lambda_2}\right]\\
&\sigma_{\beta_n}^2 \approx \sigma_z^2\left(\frac{\beta}{\lambda_2}\right)^2\left[\frac{29.88^2}{\beta^2} + \frac{69.33^2}{\lambda_2^2} - \frac{53.11^2}{\beta\lambda_2}\right]\\
&\sigma_{\gamma_n}^2 \approx \sigma_z^2\left(\frac{\gamma}{\lambda_2}\right)^2\left[\frac{27.14^2}{\gamma^2} + \frac{69.33^2}{\lambda_2^2} + \frac{61.3^2}{\gamma\lambda_2}\right]\\
&\sigma_{\kappa_n}^2 \approx \sigma_z^2\left(\frac{\kappa}{\lambda_2}\right)^2\left[\frac{(0.66C)^2}{\kappa^2} + \frac{69.33^2}{\lambda_2^2} + \frac{0.024\sigma_z}{\kappa\lambda_2}\right]
\end{aligned} \tag{4.41b}$$

4.3.3 Fractional Bit Precision

From the error models for A, B, C, the required fractional bits for the hardware implementation can be determined. As can be seen in the graphs from Figure 4.6, the standard deviation of errors varies with range and incident angle. However, it is the best case range error that will define the logic precision requirements as the noise margins will always be above this value. From the upper graph, the best case value is $\sigma_z = 1.5\ mm$. Given that $k = 5$ and $n = 9$, a substitution can be made into Equations (4.29 and 4.30) to get:

$$\begin{aligned}
\sigma_A &= \frac{\sigma_z}{k\sqrt{6}} = 0.122\ mm/pixel \\
\sigma_B &= \frac{\sigma_z}{k\sqrt{6}} = 0.122\ mm/pixel \\
\sigma_C &= \frac{\sigma_z}{\sqrt{n}} = 0.5\ mm
\end{aligned} \tag{4.42}$$

Therefore the noise standard deviation will always be above 0.122 and a fractional bit precision of 8 bits for parameters A and B has been chosen. This is more than is required to give an unbiased representation and allows a safety margin for improvement in precision.

4.3.4 Selecting Patch Size

The patch size is $(2k + 1)$ by $(2k + 1)$ and is selected to give sufficient angular resolution for plane normals. A small value of $k = 1$ results in poor discrimination of planes and their normal vectors. A simple case of a plane with normal in the z direction is sufficient to inform the choice of k. Suppose we have a patch at $u_c = v_c = 0$ measuring the plane. For the case of a plane with normal vector $(0, 0, 1)$, $\lambda = C$ and equation (4.26) becomes:

$$\begin{bmatrix} \alpha_n \\ \beta_n \\ \gamma_n \\ \kappa_n \end{bmatrix} = \begin{bmatrix} \frac{\alpha}{C} \\ \frac{\beta}{C} \\ \frac{\gamma}{C} \\ \frac{\kappa}{C} \end{bmatrix} = \begin{bmatrix} \frac{-AF}{C} \\ \frac{-BF}{C} \\ 1 \\ C \end{bmatrix} \tag{4.43}$$

$$\alpha_n x + \beta_n y + \gamma_n z = C$$

For the small values of α_n due solely to measurement errors, the standard deviation σ_{α_n} can be approximated by applying equations (4.36 and 4.37) with $X \approx 0$:

$$\sigma_{\alpha_n} \approx \frac{\sigma_\alpha}{C} = \frac{F}{k\sqrt{6}} \times \frac{\sigma_z}{C} \tag{4.44}$$

For a range of 4 metres, $C = 4000\ mm$, $\sigma_z \approx 4\ mm$ (see Figure 4.6), $F = 366$ pixels, and σ_{α_n} represents the angle error, in radians, of the plane normal to the z axis projected on the x-z plane:

$$\sigma_{\alpha_n} = \frac{366}{k\sqrt{6}} \times \frac{4}{4000} \approx \frac{0.15}{k} \tag{4.45}$$

Using $k = 5$ results in σ_{α_n} of 0.03 radians or 1.7°. σ_{β_n} is similar. This represents a reasonable compromise between good angular resolution of planes and the benefits of a small patch size for applications in bionic vision.

4.3.5 Threshold for Rejecting Non-Planes

Given the plane equation $z = Au_p + Bv_p + C$, the difference between a measured patch depth value z_i and the value calculated using the fitted plane parameters A, B, and C, $diff_i$ has an expected value of 0.

$$diff_i = z_i - (Au_p + Bv_p + C) \tag{4.46}$$

Since there are 9 points in the patch, there are 9 different cases for the variance σ^2_{diff} $(i = 1, 2, \dots, 9)$ $(u_p = -5, 0, 5)$ $(v_p = -5, 0, 5)$. However, it can be shown that there are only 3 unique solutions.

$$\begin{aligned} For\ z_1, z_3, z_7, z_9: &\quad \sigma^2_{diff} = \frac{5\sigma_z^2}{9} \\ For\ z_2, z_4, z_6, z_8: &\quad \sigma^2_{diff} = \frac{13\sigma_z^2}{18} \\ For\ z_5: &\quad \sigma^2_{diff} = \frac{8\sigma_z^2}{9} \end{aligned} \tag{4.47}$$

The worst case variance can be used to set a threshold of two standard deviations to cover 95% of the distribution. So for a particular point to accepted as part of a plane:

$$|z_i - (Au_p + Bv_p + C)| \quad \leq \quad T = 2\sigma_{diff} = \frac{2\sigma_z\sqrt{8}}{3} \tag{4.48}$$

4.3.6 Depth Dependency of Errors in Normalised Plane Parameters

The thresholds for matching of normalised plane parameters could use the standard deviation equations from (4.41) and set error limits of two standard deviations to cover 95% of the distribution. These would need to be recalculated for every pixel in each patch since there is a dependency on the plane parameters and depth. In this section it is shown that α_n, β_n and γ_n linearly increase with depth for a plane with the same normal vector, measured from the same patch centre in the depth camera image. This provides insight for empirically selecting a simpler fixed threshold in the hardware implementation based on a depth and suitable maximum plane orientation.

Suppose there is a plane with a unit vector normal of $[a\ b\ c]^T$ (ie. $a^2+b^2+c^2 = 1$), with equation as follows:

$$ax + by + cz = d \tag{4.49}$$

Where d is the minimum distance of the plane to the origin. This plane can be expressed in image depth coordinates u, v, z by substituting equation (4.14) into equation (4.49):

$$a\frac{uz}{F} + b\frac{vz}{F} + cz = d \tag{4.50}$$

From equation (4.15) it is known that $u = u_c + u_p$ and $v = v_c + v_p$, where (u_c, v_c) is the patch centre and (u_p, v_p) are local patch coordinates. Therefore the following can be defined:

$$G(u_p, v_p, z) = au_p + bv_p + (cF + au_c + bv_c) - \frac{dF}{z} = 0 \tag{4.51}$$

The tangential plane centred on (u_c, v_c) is given by ∇G:

$$\begin{aligned} \frac{\partial G}{\partial u_p} &= a \\ \frac{\partial G}{\partial v_p} &= b \\ \frac{\partial G}{\partial z} &= \frac{dF}{z^2} = \frac{dF}{C_z^2} \\ where\ C_z &= z\ at\ (u_c, v_c) \end{aligned} \tag{4.52}$$

This gives the patch plane equation:

$$au_p + bv_p + \frac{dF}{C_z^2}z = D \tag{4.53}$$

The constant D can be found from $u_p = v_p = 0$ and $z = C_z$ to be $\frac{dF}{C_z}$. Therefore the plane equation is $au_p + bv_p + \frac{dF}{C_z^2} z = \frac{dF}{C_z}$. Re-arranging gives:

$$z = \left(-\frac{a}{dF} C_z^2\right) u_p + \left(-\frac{b}{dF} C_z^2\right) v_p + C_z \tag{4.54}$$

So from the previous notation of A, B, C, α, β and comparing with (4.54):

$$\begin{aligned} A &= -\frac{aC_z^2}{dF} \qquad \textit{and} \qquad \alpha = -AF = \frac{aC_z^2}{d} \\ B &= -\frac{bC_z^2}{dF} \qquad \textit{and} \qquad \beta = -BF = \frac{bC_z^2}{d} \\ C &= C_z \end{aligned} \tag{4.55}$$

Suppose now there is a plane observed at varying depths but with the same normal vector and the same (u_c, v_c). The value of d can be found as a function of C_z from equation (4.50) with $z = C_z$, $u = u_c$ and $v = v_c$:

$$\begin{aligned} d &= a\frac{u_c C_z}{F} + b\frac{v_c C_z}{F} + cC_z \\ &= \frac{C_z}{F}\left[au_c + bv_c + cF\right] \end{aligned} \tag{4.56}$$

Substituting this into equation (4.55) gives the following:

$$\begin{aligned} \alpha &= \frac{aC_z^2}{d} = \frac{aFC_z}{au_c + bv_c + cF} = k_\alpha C_z \\ \beta &= \frac{bC_z^2}{d} = \frac{bFC_z}{au_c + bv_c + cF} = k_\beta C_z \\ \gamma &= C_z + Au_c + Bv_c = C_z(1 - \frac{k_\alpha u_c}{F} - \frac{k_\beta v_c}{F}) = k_\gamma C_z \end{aligned} \tag{4.57}$$

Where k_α, k_β and k_γ are constants for the given plane normal and patch centre. So for the same planes with the same normal vector and observed from the same patch centre $\alpha = k_\alpha C_z$, $\beta = k_\beta C_z$, $\gamma = k_\gamma C_z$, and λ can be defined as follows:

$$\begin{aligned}\lambda &= |\alpha| + |\beta| + |\gamma| \\ &= [|k_\alpha| + |k_\beta| + |k_\gamma|]\, C_z \\ &= k_\lambda C_z\end{aligned} \tag{4.58}$$

Hence,

$$\begin{aligned}\alpha_n &= \frac{\alpha}{\lambda} = \frac{k_\alpha}{k_\lambda} \\ \beta_n &= \frac{\beta}{\lambda} = \frac{k_\beta}{k_\lambda} \\ \gamma_n &= \frac{\gamma}{\lambda} = \frac{k_\gamma}{k_\lambda}\end{aligned} \tag{4.59}$$

Where all $k's$ are independent of the depth C_z. Note that α, β, γ, λ all scale linearly with depth C_z. Applying this to σ_{α_n}, σ_{β_n} and σ_{γ_n} from equation (4.41) gives:

$$\begin{aligned}\sigma_{\alpha_n} &= \sigma_z \left|\frac{\alpha}{\lambda}\right| \sqrt{\frac{29.88^2}{\alpha^2} + \frac{69.33^2}{\lambda^2} - \frac{55.09^2}{\alpha\lambda}} \\ &= \frac{\sigma_z}{C_z}\sqrt{\frac{1}{k_\lambda^2}29.88^2 + \frac{k_\alpha^2}{k_\lambda^4}69.33^2 - \frac{k_\alpha}{k_\lambda^3}55.09^2} \\ \\ \sigma_{\beta_n} &= \sigma_z \left|\frac{\beta}{\lambda}\right| \sqrt{\frac{29.88^2}{\beta^2} + \frac{69.33^2}{\lambda^2} - \frac{53.11^2}{\beta\lambda}} \\ &= \frac{\sigma_z}{C_z}\sqrt{\frac{1}{k_\lambda^2}29.88^2 + \frac{k_\beta^2}{k_\lambda^4}69.33^2 - \frac{k_\beta}{k_\lambda^3}53.11^2} \\ \\ \sigma_{\gamma_n} &= \sigma_z \left|\frac{\gamma}{\lambda}\right| \sqrt{\frac{27.14^2}{\gamma^2} + \frac{69.33^2}{\lambda^2} + \frac{61.3^2}{\gamma\lambda}} \\ &= \frac{\sigma_z}{C_z}\sqrt{\frac{1}{k_\lambda^2}27.14^2 + \frac{k_\gamma^2}{k_\lambda^4}69.33^2 + \frac{k_\gamma}{k_\lambda^3}61.3^2}\end{aligned} \tag{4.60}$$

Note that in Figure 4.6, the standard deviation of depth measurements, σ_z, is a quadratic function of z (or C_z in equation (4.60)) and so σ_z scales with C_z^2 and σ_{α_n}, σ_{β_n} and σ_{γ_n} then scale with C_z. For σ_{κ_n} it can be shown from equation (4.41) to scale with $C_z^{2.5}$ as follows:

$$\begin{aligned}\sigma_{\kappa_n} &= \sigma_z \left|\frac{\kappa}{\lambda}\right| \sqrt{\frac{(0.66C)^2}{\kappa^2} + \frac{69.33^2}{\lambda^2} + \frac{0.024\sigma_z}{\kappa\lambda}} \\ &= \sigma_z \sqrt{\frac{1}{k_\lambda^2}0.6^2 + \frac{1}{k_\lambda^4}69.33^2 + \frac{\sigma_z}{k_\lambda^3 C_z}0.024}\end{aligned} \tag{4.61}$$

Since σ_z scales with C_z^2, the following is obtained:

$$\sigma_{\kappa_n} = C_z^2 \sqrt{\frac{1}{k_\lambda^2}0.6^2 + \frac{1}{k_\lambda^4}69.33^2 + \frac{0.024}{k_\lambda^3}C_z} \tag{4.62}$$

Assuming that the first two terms in the square root are negligible in comparison to the third, it can then be shown:

$$\begin{aligned}\sigma_{\kappa_n} &\approx C_z^2 \sqrt{\frac{0.024}{k_\lambda^3}C_z} \\ &= C_z^{2.5} \sqrt{\frac{0.024}{k_\lambda^3}}\end{aligned} \tag{4.63}$$

4.4 Summary and Conclusions

This chapter has presented the derivation of a new hardware plane fitting algorithm as well as an analysis of the depth sensor error and propagation of those errors to the normalised plane parameters.

The plane fitting algorithm is based on least squares and lends itself well to an FPGA implementation. By using a patch-based approach and making assumptions about the centring of the patch coordinates, the least squares plane fitting method has been simplified to a number of simple addition and multiplication operations and a few fixed value division operations. This is beneficial for the hardware implementation as it results in a simpler logic implementation that can be run at a high clock frequency.

A transformation of the fitted plane equation to a world coordinate system has been outlined. This transformation achieves consistency in the coordinate system and facilitated the derivation of equations for the plane normal vector parameters α, β and γ from Section 4.2.3. Subsequently normalised plane parameter equations have been derived. These equations allow for the detection of planes of specific orientation and offset from the origin.

The depth sensor error standard deviation has been found to have a parabolic trend increasing with distance. The results agree with existing literature and show a minimum error of 1.5 mm (at 1.5 m depth) and a maximum error of 15 mm (at 7.75 m depth). Based on this range error model, the propagation of error throughout the plane fitting algorithms has been analysed and resultant error models for plane parameters have been defined. Using the error models derived for the A, B and C parameters, the fractional bit precision of the hardware implementation has been defined as 8 fractional bits. In a similar manner, the distance dependency of errors for rejecting non-planes and matching the normalised plane parameters have been established in Sections 4.3.5 and 4.3.6 respectively.

The error analysis in this chapter can be said to have followed a best-case approach, as some contributing factors to error have been ignored. As will be shown in the following chapter, the attenuation of IR illumination results in high noise levels towards the edges of the depth image. The range error analysis in this chapter has not involved peripheral parts of the depth image. Lens distortion and error due to attenuation reflected IR light have also been ignored. To provide more accurate models of sensor errors, these factors would need to be taken into account, and this is a possible avenue for future work.

An autocorrelation analysis has been performed and shown that there is significant correlation of residuals for up to two pixels away. This means that neighbouring pixels cannot be assumed independent. In the case of the error analysis and modelling performed in this work, sampled patch pixels were evenly spaced at a distance of 5 pixels and so can be assumed independent. The patch size and sample spacing is determined by the selection of k. In Section 4.3.4 the selection of $k = 5$ is justified through the analysis of the angular error of the plane normal, which was found to have a standard deviation of 0.03 radians or 1.7° (at 4 m depth).

The work presented in this chapter provides the basis for the hardware implementation of plane fitting that will be discussed in the next chapter.

Chapter 5

Hardware Implementation of Plane Fitting and Applications to Bionic Vision

In the previous chapter, the development of a least squares plane fitting algorithm was presented. A patch based approach has been used in order to allow a fast FPGA implementation to be tractable. A real time implementation of this algorithm applied to detection of objects in a scene is anticipated to be a useful navigational aid for blind individuals.

This chapter investigates the feasibility of integrating 3D depth sensing into bionic vision systems as well as the application of plane fitting of this data to object detection as a form of navigational aid. A hardware design based on the algorithm developed in Chapter 4 is implemented and used for floor and tabletop plane detection in a real time end-to-end system.

5.1 Introduction

In the fields of robotics and computer vision, sensing of three dimensional environments and objects is necessary for many different applications. This can be achieved through the use of a variety of different sensing technologies, including stereo-vision, laser scanners, and triangulation-based systems. Recent advances in real time depth sensing however, such as the Kinect v1 and Kinect v2 [76], have facilitated such applications at a low cost while maintaining sufficient sensing accuracy and resolution. Many applications in 3D perception and recognition have been investigated using these devices in recent years [21, 52, 64, 104].

With increasing interest in visual prosthesis development, some researchers have investigated the potential use of 3D sensing for bionic vision purposes [67, 68, 71, 75, 106, 108]. A fundamental building block for many 3D applications is recognising and characterising planes in depth frames [53, 79, 89, 98, 107], and applications of this have been used in some work to complement the low spatial resolution of a vision implant and aid the user in navigational tasks [75, 108]. These implementations usually involve complex computations and real time operation in a portable form has been facilitated by high performance laptop computers. In order to incorporate such functionality into an end-user bionic vision device, comparable processing power would be required, and this is not likely to be feasible with current computer technology given the limitations of size and weight of planned prosthesis systems.

A possible alternative solution is the use of Field Programmable Gate Array (FPGA) technology. FPGAs offer a high level of parallelism, which often allows for significant speed-up of performance in comparison to CPU implementations, with a fraction of the operating clock speed and required hardware. This is especially useful when implementing image processing algorithms [9, 10, 62] that involve identical and independent operations. FPGAs are flexibile in terms of architecture and can be exported to an Application Specific Integrated Circuit (ASIC). This would allow for even more speed and reduction in power consumption. An ASIC could be integrated into an end-user device without significantly affecting physical size or power consumption.

Implementations of least squares algorithms on FPGA hardware exist in literature, such as the work of Yang et al. [119]. This work however implements an iterative approach to solving linear equations via the Cholesky decomposition method, and cannot directly be applied to plane fitting as is achieved in the work of this chapter. To the best of knowledge of the author, there have been no FPGA implementations of the specific case of fitting planes to 3D depth data in real time via a least squares method. A hardware implementation of the HK segmentation method developed by Besl and Jain [16], was developed by Tellioglu [109] in order to classify curvatures within 3D point cloud data. The method involved the calculation of the mean (H) and Gaussian (K) curvature of a 128 by 128 set of point cloud data in order to classify it as part of one of 8 different types of curved surfaces. In this work, the detection of planar surfaces is possible (when H and K are zero), however fitting of the data to an actual plane equation is not performed and so the segmentation of specific planes within the scene is not possible. There have been a number of FPGA-based image processing and computer vision implementations [10, 15, 31, 58, 62, 69] however the use of depth or 3D data is quite uncommon in the field, limited mainly to biomedical imaging applications [66]. This is possibly due to a lack of support for direct integration of 3D sensors such as the Kinect v2 with FPGA hardware. Recent advances in FPGA technology however, have enabled very high speed communication with external devices such as computers. PCI express for example, has become a very popular inclusion in current high-end FPGA development boards [17, 110, 118]. This could provide a possible way to bridge the gap between a 3D sensor and an FPGA, and would be quite interesting to investigate.

The real time processing of depth data using an FPGA system is investigated in this chapter through simulation and implementation of the plane fitting algorithm covered in Chapter 4. It is shown to be achievable and that plane fitting in particular is a good fit for a hardware solution.

5.2 Hardware Solution

The plane fitting algorithm developed in Chapter 4 has been implemented using the Verilog Hardware Description Language. The hardware solution exploits the parallelism of operations in the algorithm. In order to do this, all of the depth pixels within the patch that is currently being processed are made available simultaneously. Since depth data is supplied in a consecutive stream, this is achieved through the use of a sliding window implementation as outlined in Figure 5.1. Incoming depth pixel data is buffered in First In First Out (FIFO) row buffers using Altera's RAM-based Shift Register Megacore [2]. The FIFO is able to be tapped at various points, allowing the patch data to be made available. The tapped data is stored into a series of individual logic-based shift registers. This creates a sliding window effect, making available on each clock cycle a new 11 by 11 patch of data, the centre of which is offset one pixel horizontally from the last patch. Since only 9 evenly spaced points are required from the patch, the number of logic-based shift registers can be reduced to just 3 rows (indicated by the bold blocks in Figure 5.1).

As the patch pixel data is made available, it is sent to a pipelined processing block that performs the plane fitting algorithm in a number of smaller steps or pipeline stages. This pipelined architecture allows for a higher achievable clock frequency, throughput and framerate, while maintaining a latency, with respect to the top left of the patch, of 10 rows of the depth image. An overview of the various pipeline stages is given in Figure 5.2. At the end of all the pipeline stages, the normalised plane parameters α_n, β_n, γ_n and κ_n are made available for later processing (see Section 5.4). A binary 'Pval' signal is also made available at the end of the pipeline. This signal is a valid bit resulting from the recalculation of plane values for the current patch according to the A, B and C parameters achieved. If the difference between actual depth values and the calculated plane values falls outside of some threshold (that can be set according to Chapter 4 Section 4.3.5 Equation (4.48)), then the current patch would be not be considered a valid plane (Pval bit would be '0'). The current implementation uses a constant threshold based on a maximum range as discussed in Section 5.4 below.

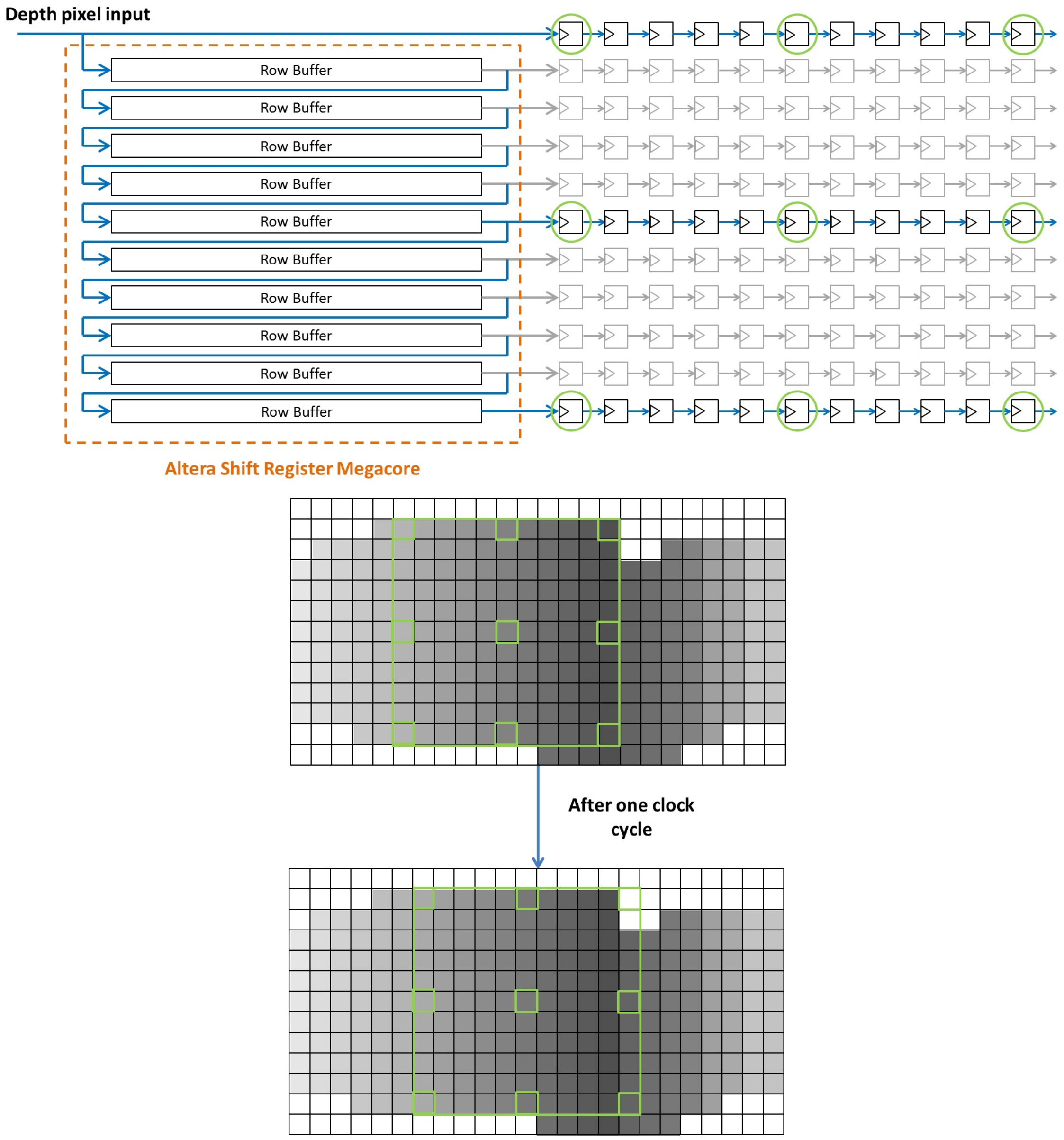

Figure 5.1: Block diagram of sliding window implementation. Depth pixels are input to the FIFO on every clock cycle. The tapped outputs of the Megacore are simultaneously shifted into logic-based shift registers on each clock cycle creating a sliding window effect.

Two depth pixels are delivered to the system per clock cycle from tightly coupled on-chip memory. A second copy of the plane fitting block from Figure 5.2 is then instantiated to double the processing rate of the implementation. The blocks work in parallel, as indicated in the diagram in Figure 5.3, and output the normalised plane parameters for two consecutive patches per clock cycle. Note that to achieve both processing and output of two patches per clock cycle, the sliding window implementation shown in Figure 5.1 is modified to receive 32 bits (2 pixels) per clock, and six 32-bit logic-based shift registers would be used in place of the eleven 16-bit shift registers in the diagram. The location of the window in the bottom of Figure 5.1 would also shift by two positions after each clock cycle.

5.3 System Implementation

In order to demonstrate a real time implementation of the plane fitting module from Section 5.2, an end-to-end system has been developed. This system needed to be able to interface with the Kinect v2 sensor, which provides the depth data input, as well as some form of display device for providing visual output of the results. Since there is currently no support for connection of Kinect v2 depth sensors to FPGA development boards, a hybrid hardware-software architecture has been developed. A suitable platform for this architecture incorporates an FPGA chip with a substantial amount of high speed logic and on-chip memory, and a high speed communication channel for interfacing with a computer, which in turn would interface with the Kinect v2 sensor and a display monitor. The Terasic TR4 FPGA development board [110] (Figure 5.4) has been chosen for the system. The TR4 is equipped with a Stratix IV FPGA chip [4] that has 228,000 logic elements, 17,133 kilobits of on-chip memory, support for DDR3 off-chip memory, and two PCI express communication ports.

An overview of the end-to-end system is given in Figure 5.5. Depth frames are captured by the Kinect v2 sensor and sent down to the host computer via USB 3.0 interface at a resolution of 512 by 424 by 16 bits and a framerate of 30 frames per second. The frames are buffered by the host computer and sent down individually to the memory on the FPGA board via PCI express link for processing. Upon completion of processing a frame, the FPGA signals the computer and the data is transferred from the FPGA memory back up to the host computer, where it can be displayed on the monitor.

The implementation of the hardware portion of the system is shown in a detailed block diagram in Figure 5.6. Frame data transferred from the host computer is stored directly into FPGA on-chip dual-port memory. Also stored in that memory are various control and threshold values. The second interface port of the memory

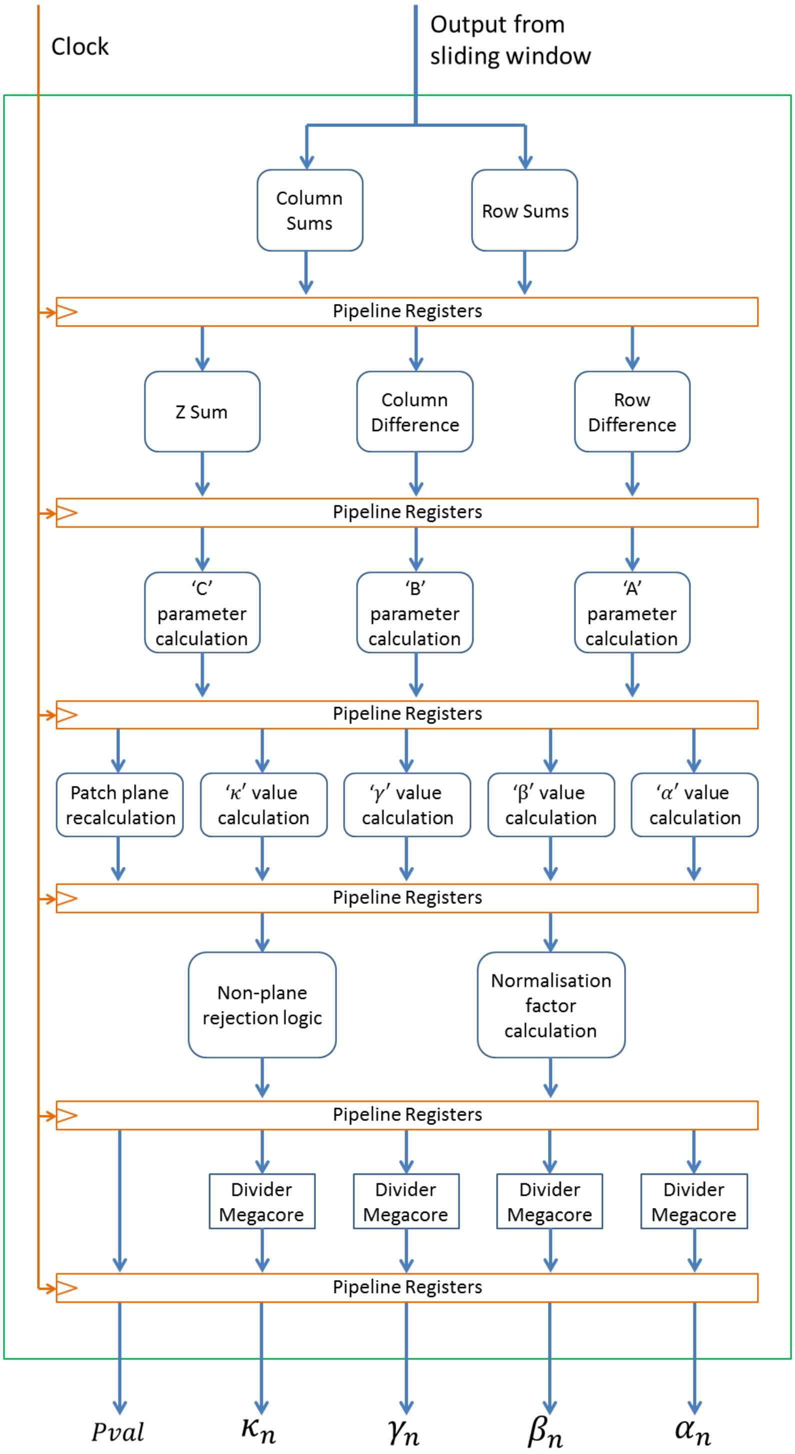

Figure 5.2: Block diagram of plane fitting pipeline implementation. Each stage of the pipeline is made up of combinational logic (blue blocks), the results of which are buffered by the pipeline registers (orange) on every clock cycle. This allows for multiple patches of data to be operated on simultaneously at various stages of the plane fitting algorithm.

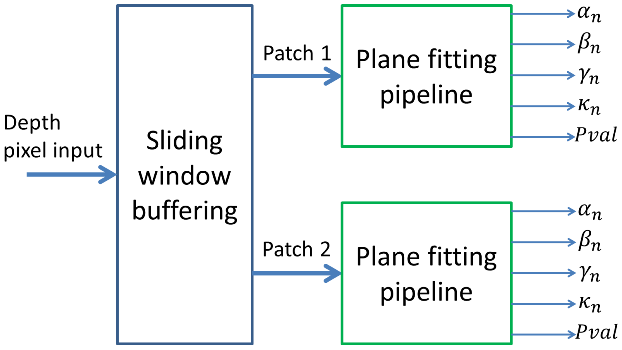

Figure 5.3: Dual patch processing implementation. The use of two copies of the plane fitting pipeline allows the processing of two patches per clock cycle.

Figure 5.4: The TR4 FPGA development board

is connected to a custom Altera Qsys component that has been developed as part of this work. This component instantiates the plane fitting module explained in Section 5.2, and implements a state machine that provides a controlled interface to the frame data stored in memory. The Qsys system integration tool facilitates the connection of the various parts of the system via standardised communication protocols. It also allows the instantiation of IP cores. IP cores provided by Altera have been used in this system for the PCI express interface, the dual-port on-chip memory controller, and the variable division blocks. The state machine in the plane fitting module polls a control byte stored in memory. When the host computer completes a transfer of a frame to the FPGA memory, it signals to the custom component by setting this control flag. When the flag is set, the component begins reading frame data into the plane fitting module and writing the processed data back into a separate section of the on-chip memory. When the frame has been completely processed, the custom

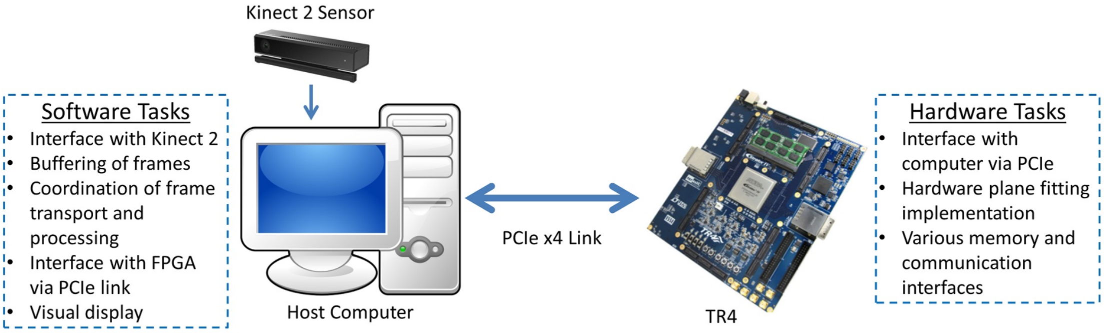
Kinect 2 Sensor
Software Tasks
Interface with Kinect 2
Buffering of frames
Coordination of frame transport and processing
Interface with FPGA via PCIe link
Visual display
Host Computer
PCIe x4 Link
TR4
Hardware Tasks
Interface with computer via PCIe
Hardware plane fitting implementation
Various memory and communication interfaces

Figure 5.5: High level end-to-end system overview

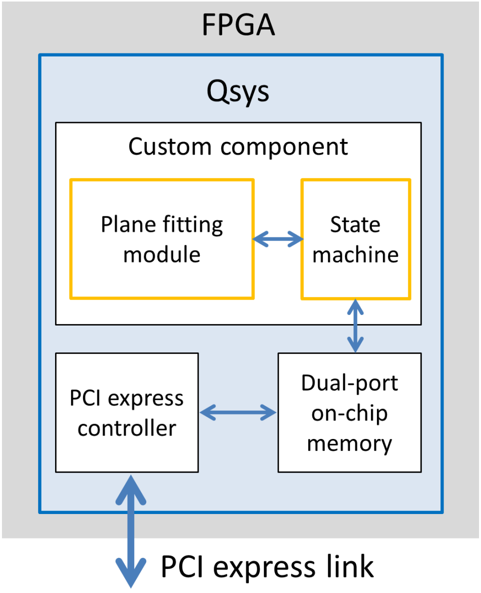

Figure 5.6: Block diagram of hardware section of end-to-end system

component resets the control flag, to indicate to the host computer that it can read the data in the processed section of memory.

5.4 Applications

Although more general, this plane fitting system has been developed for bionic vision patients, and applications are predominantly navigational aids. Through the detection and segmentation of objects, paths, doorways, obstacles and/or free space, the effort required by a bionic vision user to navigate through everyday environments may be reduced. As a proof of concept, two different examples of detection applications have been implemented in this system: ground plane detection and table top detection.

By making the assumption that the Kinect v2 sensor is at a fixed height and fixed y axis orientation (camera roll) when moving through the environment, the implementation of both detection applications can be reduced to relatively simple comparison logic in hardware. More sophisticated applications could compensate for orientation changes with inertial sensors. The normalised plane parameters calculated by the plane fitting module are compared to expected values that relate to either a ground plane or table top. If the differences fall within a specified threshold, then the current pixel is flagged as part of the type of plane that is attempting to be detected. Ideally this threshold would be calculated separately for each pixel and patch using the equations from Chapter 4 Section 4.3.2 Equation (4.41). However in these results it was found suitable to determine the thresholds empirically, avoiding unnecessary logic usage. Furthermore, it is not critical that a fixed 95% of the planes be matched as long as enough matches create a discernible pattern for a blind user given the low resolution of the vision implant. The expected plane parameter values can either be predefined and hard coded into the logic, or sampled from a particular region in the frame of data in real time upon a button press. Visual results for these two applications are shown in Section 5.5.

5.5 Results and Performance

The results of this work can be broken down into 3 main parts: simulation, end-to-end system integration and real time performance with applications. The following subsections outline the testing methods used and results gained for each.

5.5.1 Simulation

Initial development and testing of the plane fitting algorithm explained in Chapter 4 was completed using the MATLAB 2013a software package [74]. Actual Kinect v2 depth image data was used as input, and the implementation made use of simple fixed point arithmetic and bitwise logic operations that mimics the hardware solution. The hardware implementation was then completed using Verilog HDL in a testbench environment using Altera's Quartus 13.1 (service pack 2) software [3]. Finally, the implementation was verified using Mentor Graphics's ModelSim (Altera Edition) simulation software package [5].

A block diagram of the testbench setup is shown in Figure 5.7. The testbench simulation provided the Design Under Test (DUT) with the same input data as was used in MATLAB and verified the outputs with the MATLAB generated output data. The ModelSim simulation output matched the MATLAB output correctly. Tables 5.1 and 5.2 show logic resource usage and maximum clock rate characteristics

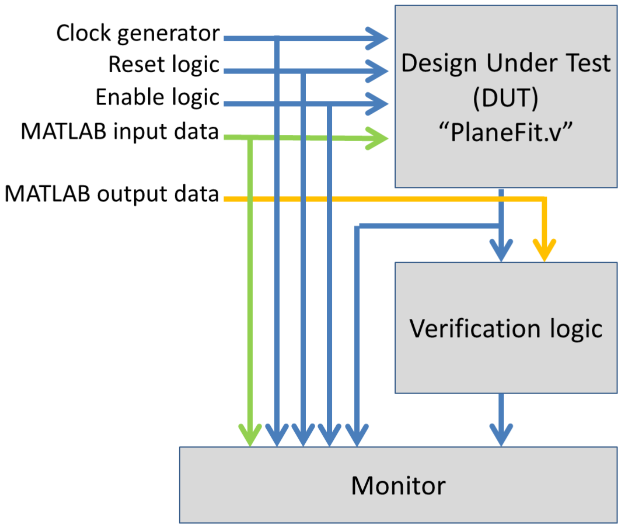

Figure 5.7: Block diagram of testbench

Table 5.1: Resource usage for plane fitting module

Resource	Usage
Logic elements	33,027 / 182,400 (18%)
Memory bits	92,638 / 14,625,792 (<1 %)
18-bit multiplier blocks	64 / 1,288 (5%)

estimated by the Quartus compilation for the plane fitting module (using a Stratix IV EP4SGX230KF40C2 FPGA chip).

As can be seen in Table 5.1, a low proportion (<18%) of logic resources was needed. The majority of this logic is attributed to the divider blocks however, with only about 2000 logic elements used for the other parts of the implementation ($\approx$ 1%). On-chip memory and multiplier block usage for the plane fitting module is also quite small (<1% and 5% respectively). The compilation timing analysis (Table 5.2) estimated a maximum operating frequency of 54.11 MHz, which can be related to a maximum theoretical frame rate of 480 FPS (assuming data input/output rate of two 16-bit pixels per clock cycle). Given that the Kinect v2 sensor captures depth frames at a maximum frame rate of 30 FPS, the frequency required to run the module at frame rate is only 3.4 MHz. Power consumption of integrated circuits are proportional to the clock frequencies they operate at. Running at this lower clock rate of 3.4 MHz would therefore mean that there would be significantly less power usage in comparison to an implementation on a GPU-based system, which may be run at clock frequencies in the order of 1 GHz.

Table 5.2: Timing analysis

Property	Value
Maximum operating frequency	54.11 MHz
Theoretical max. frame rate	480 FPS

5.5.2 End-to-End System Integration

As described in Section 5.3, practical real time implementation of the plane fitting implementation requires an end-to-end system. Along with plane fitting, this system has number of peripheral tasks such as various communication and memory interfaces, as well as management and display of data, each with processing time and latencies. So while the simulation results of Section 5.5.1 report a maximum theoretical frame rate of 480 frames per second (FPS), the actual achievable frame rate is lower.

There are two main bottlenecks that limit the maximum achievable framerate. The first occurs at the memory interface between the custom component and the on-chip memory. Read and write transfers take 2 clock cycles each to complete, and cannot be performed simultaneously. Therefore the plane fitting module would process a frame of data in 8.7 ms, dropping the theoretical maximum frame rate by a factor of about 4 to 115 FPS. The second major bottleneck involves preparing, transferring, and displaying of frame of data carried out by the host computer. The latency of transferring a frame of data over the PCI express link was measured to be 2.4 ms for both downstream and upstream transfers. Buffering and display of the frame data had a measured processing time of 3.5 ms. Therefore the overall software processing time is 8.3 ms. Taking into account the processing time of the hardware modules, the maximum achievable framerate is 60 FPS. This however ignores operating system scheduling delays that cause the performance to be lower. The following section (5.5.3) covers the real time operating performance for the complete system, and display results.

5.5.3 Real Time Performance and Visual Results

Since the Kinect v2 sensor provides depth frames at 30 FPS, the system's real time performance has been tested with pre-captured streams of data. For the tests, 450 frames were first loaded into the computer memory from captured files on the hard drive. The frames were then sent one by one down to the FPGA for plane fitting and ground or table top detection. The results were read back and displayed. This process was repeated for all 450 frames and the total processing time was used to find the average framerate of operation. The average framerate achieved was 60 FPS, as estimated in the previous section. This surpasses the Kinect v2 sensing rate by a factor of two.

Visual results of the ground plane and table top detection are given in Figure 5.8. A sample of 6 frames out of the 450 processed frames are shown. The left column contains unprocessed depth artificially-coloured frames, and the middle and right columns show the results of the ground plane and table top detection applications respectively. A video of the same can be found in Appendix B and here is a link. Note that the video runs at a lower framerate than what is actually achieved during real time operation. In the results, white pixels correspond to correctly detected plane pixels that are part of either the ground or a table top of a set height. The thresholds on α_n, β_n, γ_n were set to 0.05 and the threshold on κ_n was set to 120 mm. Due to the falloff of IR illumination, the peripheral sections of the depth image can be quite noisy and cause errors in detection. This is evident in the ground plane detection results of Figure 5.8.

5.6 Discussion

From the Verilog simulation results it can be seen that the maximum processing framerate that can be achieved by the hardware plane fitting module is 480 FPS. While this is well above the required rate to keep up with the delivery of frames from the Kinect v2 (30 FPS), there are some assumptions made in order to achieve this rate in a real time system. These assumptions are that the input data can be delivered at a rate of two 16-bit pixels per clock cycle, and that the output data can be simultaneously consumed at the same rate. Given adequate FPGA resources, these assumptions can be improved in order to increase the throughput and effectively increase the framerate. If the assumption was to be made that the input and output data rates are four pixels per clock for example, and the buffering and number of plane fitting pipelines increased accordingly, the maximum framerate (at the same clock frequency) would be doubled to 960 FPS. Similarly, for higher input/output data rates such as 8 or 16 pixels per clock cycle, the maximum framerates would increase to 1920 FPS and 3840 FPS respectively, limited only by logic resources. Since only 18% of logic elements have been utilised this is feasible. It is important to note though, that increasing the number of patches that need to be processed in parallel will make the sliding window implementation more complex, therefore logic and routing resources required may increase quickly.

The maximum achievable framerate is limited by the highest processing and/or latency times and the memory bandwidth of the various stages within the implementation. There are two main bottlenecks within this system's particular implementation. The first is the interface between data stored in memory and the plane

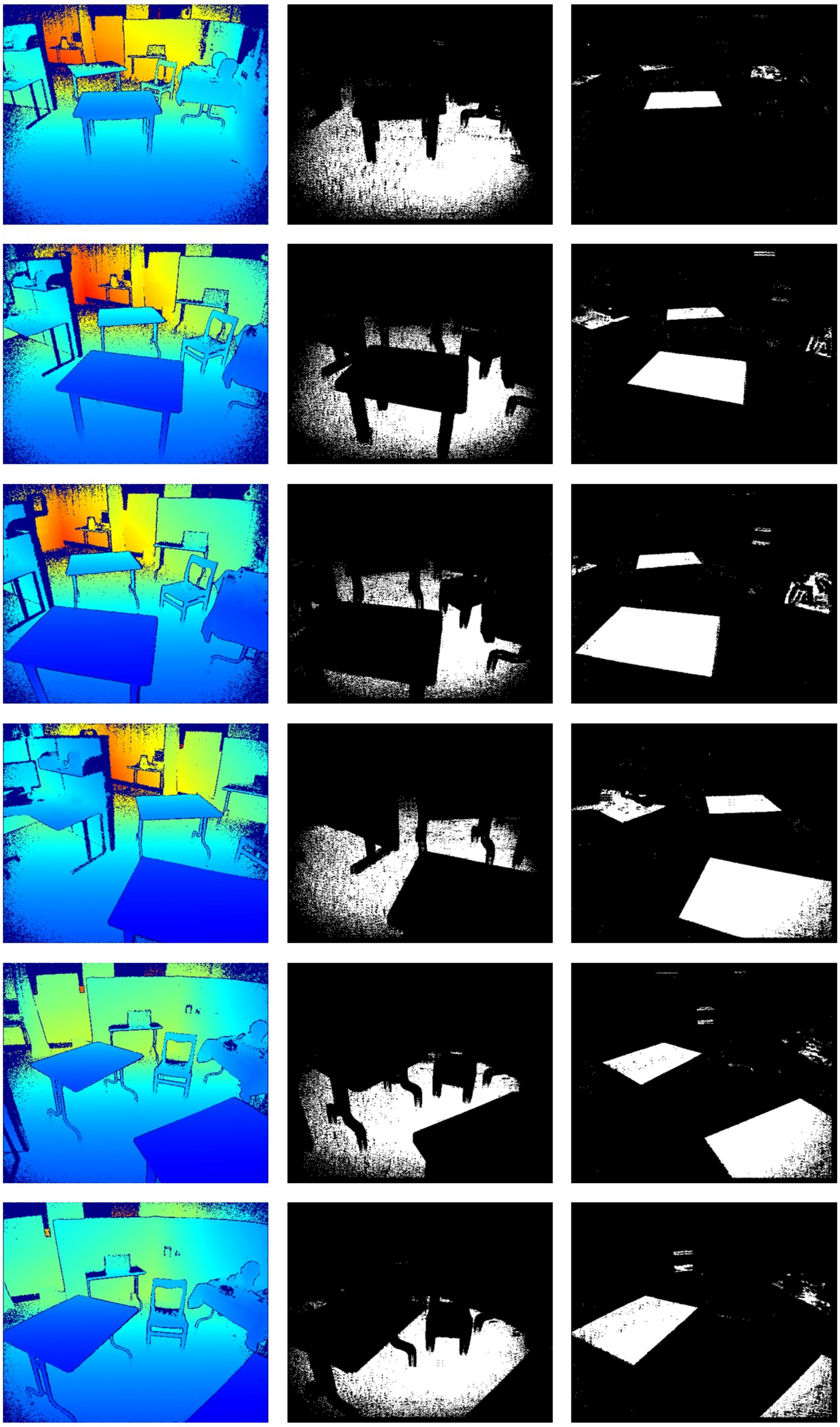

Figure 5.8: Visual results of the floor and table top detection applications are shown above. The artificially-coloured depth frames (left column) were used as input. The middle and right columns are the output results for floor and table top detection respectively. The white pixels highlight the correctly detected planes. Detection errors are noticeable around the edges of the frames. This is due to the falloff of Kinect v2 IR illumination.

fitting module. Data is able to be read and written from the memory in blocks of 32 bits. However, read and write transactions each have 1 clock cycle latency. Furthermore, read and write operations cannot occur simultaneously unlike the simulation results. This is because the second interface port of the on-chip memory is connected to the PCI express communication port for data polling purposes. This causes a reduction in performance of 4 times to a framerate of 115 FPS, hence any throughput improvements that could have been made to the plane fitting module would have no overall benefit. A possible improvement to this bottleneck could be to investigate the use of separated memory blocks for the unprocessed and processed data. This would mean separate interface buses and simultaneous reads and writes. This could improve throughput by a factor of 2.

The second bottleneck in the system is computer software processing time and data transfer overheads. As was stated in Section 5.5.2 of the results, the total software time was measured to be 8.3 ms. This means that even if the hardware solution process and latency time is negligible, the maximum framerate achievable is 120 FPS.

As can be seen in Figure 5.8 of the results, the ground plane and table top detection applications are successful. Real time testing of the whole system has been able to achieve a maximum framerate of 60 FPS. While this is half of the theoretical maximum framerates that have been estimated in the previous discussion, it still surpasses the framerate of the Kinect v2 sensor by a factor of 2.

The visual results have alluded to the fact that the level of noise in the depth data increases significantly towards the peripheral edges of the image. This is likely to be due to the attenuation of IR illumination and the reflected IR light from surfaces in the scene. Lens distortion could also be a contributing factor. As mentioned in the previous chapter, a best-case approach has been taken for the analysis of depth sensor error and so the error models developed would likely be unsuitable for data far from the centre of the depth image. A more detailed analysis of the error would be necessary to account for the effects of IR attenuation and lens distortion and this could be done for future work.

The error models for the normalised plane parameters developed in Chapter 4 have not been used in the implementation of applications in this chapter. For the specific case of highlighting floor or tabletop planes for a bionic vision system, it is has been found unnecessary due to the low resolution output of an end-user device. Instead the thresholds used have been determined empirically. This has been based on a maximum range and has been kept constant for all depth data points. This is sufficient to provide a suitably representative image of the detected planes. It also avoids unnecessary logic implementation that would be required to calculate a threshold for every pixel of the image. The variability of plane parameters becomes much more critical for tasks such as plane tracking and in these cases the error model equations will need to be used. A hardware implementation of the developed error models could be investigated for future work.

Other future work could include more advanced applications and functionality. Inertial sensing and/or tracking of planes in the environment could be incorporated. This would allow for the compensation of changes to camera position and orientation, as would be quite likely in practical use of a bionic vision system. Navigational cues could be displayed on screen to help users in everyday tasks. More advanced object and surface detection could be implemented to identify to the user things such as chairs, doorways, bins, stairs, paths, etc.

5.7 Summary and Conclusions

Presented in this chapter has been the design and implementation of the plane fitting algorithm from Chapter 4 on FPGA hardware. The hardware implementation has been developed using Verilog and simulated to achieve a maximum framerate of 480 FPS. To facilitate a real time implementation on actual hardware, a hybrid hardware-software end-to-end system has been implemented. This system integrates the hardware plane fitting module with depth data from a Microsoft Kinect v2 sensor through a PC and high speed PCI express communication bus.

The plane fitting implementation has been applied to two different bionic vision detection applications - ground plane and tabletop detection, and the whole system is able to run in real time at 60 FPS. The two detection applications provide a proof of concept of the feasibility of integrating a depth sensor with a bionic vision device in order to aid a user in navigational tasks. Extensions to these applications could be in the form of detection of more complicated objects and inclusion of inertial sensing and tracking of planes for egomotion measurement.

Although the system is capable of operating at twice the rate that frames can be provided from the Kinect v2 sensor, a number of bottlenecks were observed that prevent the system reaching framerates nearer to that of the simulation results. These are mainly due to memory interfaces and data transfer latencies. While possible solutions have been discussed, their realisation may not provide any further benefit in bionic vision applications as it is unlikely prostheses will require framerates exceeding 30 frames per second. The work presented in this chapter also has uses in the fields of robotics and computer vision however, and investigation into maximising real time performance would be attractive in these cases, although outside the scope of this thesis. Another limitation of the system is that the use of a host computer to interface with the Kinect sensor potentially hinders portability if it were to be incorporated into a bionic vision system. This could be solved via development of a USB driver to connect directly to the Kinect from the FPGA, and could be an avenue for future work.

The work of this chapter has shown the potential benefit of real time plane fitting of depth sensing for improving bionic vision systems.

Chapter 6

Summary and Future Work

Research into the development of bionic vision devices has been progressing quickly in recent years, with research groups around the world working towards commercial products. However, very limited spatial and temporal resolution is expected of early implants. Due to the electrical interaction properties that exist between neighbouring stimulating electrodes in brain tissue, it is currently only feasible for implants to use a maximum of several hundred electrodes within the foveal area. This results in the same number of elicited visual points. Furthermore, the achievable framerate is likely to be as low as four frames per second and the control of phosphene intensity could be as coarse as two levels. This thesis has investigated the capabilities of users under the anticipated level of vision of the Monash Vision Group's Gennaris device and has explored options for potentially improving future patients' performance in everyday tasks such as navigation, pattern recognition and object manipulation. The key achievements and findings of the work presented in this thesis can be summarised as follows.

6.1 Summary of Key Achievements and Findings

Hatpack Simulator

In the work presented in Chapter 2, a portable real time simulator called the Hatpack has been developed based on a Field Programmable Gate Array (FPGA) architecture. The system takes into account current neurophysiological models of visuotopy, with an implementation of the Monopole model to mimic the non-linear mapping of points on the visual cortex to points in the visual field. The Hatpack is lightweight, immersive and completely mobile allowing it to be used for untethered testing away from a lab bench or office desktop. The exploitation of parallel operations on the

FPGA allows the Hatpack to operate at 60 frames per second with a constant latency of only 17 ms. Its power consumption is low enough for it to last 4 hours on a full charge.

In addition to the visuotopic mapping implementation, the Hatpack also simulates other characteristics expected of a vision implant including: limited control of phosphene intensity via luminance thresholding; the appearance of a phosphene by using a 2D Gaussian mask implementation; the possibility of failed electrodes; low temporal resolution via framerate reduction; and the ability to control the system and change modes and thresholds in real time via an IR remote control interface and manual threshold control dial. A number of different automatic threshold selection methods as well as an edge detection mode have also been implemented.

The Hatpack has been tested extensively, having been used for five different sets of psychophysical tests over the period of a few years. In this time there have been no major issues with its operation, thus proving its robustness and suitability for use as a psychophysical testing tool. Furthermore, there were no significant complaints regarding comfort of the head gear following the integration of the head mounted display into the ski goggles.

Psychophysics Testing

Five different psychophysics tests have been conducted as part of the work presented in Chapter 3. These involved the use of the Hatpack with normally sighted individuals, and the completion of tasks that resemble everyday activities. The first test was a navigational task requiring users to try and make their way through a maze of obstacles. The second test was an eye-hand coordination and contrast discrimination problem requiring interaction and placement of chess pieces on a chessboard. The third experiment required participants to attempt to stop rolling balls at varying framerates. In the fourth experiment participants were presented with different shapes of varying shades on an e-ink display and were required to identify the shape. The final experiment required participants to try and sort socks of different shades.

It was found that users were able to complete all of the tasks successfully albeit with varying proficiencies. However, the results have shown that it is significantly more difficult to complete the tasks under the simulated phosphene vision than when in a control mode that was similar to normal sight. This has shown that there is still much room for improvement to allow bionic vision users to complete such tasks with similar proficiency as a normally sighted person. Other observations from the results included the need for multiple levels of greyscale intensity, higher framerate, more advanced thresholding implementations, and post-operative training so that patients can get the most out of their implant.

General use of the Hatpack and the conduction of the psychophysics testing has revealed the adverse effects of poor lighting and cluttered environments on the ability of the system to provide a clear representation of the scene. This has prompted the need for an alternate sensing approach and has motivated the investigation into the feasibility of integrating a 3D depth sensor and advanced processing techniques into a bionic vision system.

Plane Fitting and Sensor Error Analysis

Presented in Chapter 4 is the development of a new plane fitting algorithm based on the least squares method. This algorithm employs a patch based approach and has been tailored for a suitable implementation on FPGA hardware. The operations required to calculate the fitted plane parameters have been reduced to a form involving a number of simple addition and multiplication operations and a few fixed value divisions. This allows for a simpler logic implementation that could be run at higher clock frequencies.

The potential variability of depth sensor data has been investigated via the analysis of errors in a Microsoft Kinect v2 sensor which was also presented in this chapter. This analysis allows for the specification of logic precision requirements, statistical testing of the validity of a plane fit, and achievable plane fitting angle resolution. A best-case range error model was obtained through statistical analysis of depth frames and found to have a parabolic trend increasing with distance. Using this model the propagation of errors to the plane fitting parameters has been investigated and general models for the errors in the plane fit parameters have been derived. From these models acceptable difference thresholds when matching parameters for the floor and tabletop detection applications can be obtained.

Real Time Hardware Implementation of Plane Fitting and Bionic Vision Applications

Chapter 5 details a real time implementation of the hardware plane fitting algorithm developed as part of Chapter 4 in a hybrid hardware-software end-to-end system. The plane fitting solution has been developed using Verilog and first simulated in a testbench environment. The simulation reported a maximum theoretical framerate of 480 frames per second. The logic usage of the FPGA was found to be only 18% however, meaning that it may be feasible to scale up the implementation to achieve theoretical framerates of up to 3840 frames per second. This is assuming a data rate of 16 pixels per clock.

Following simulation, an end-to-end system was put together in order to implement this module in real time on an FPGA. This system integrated a Kinect v2 sensor with a TR4 FPGA board via a computer and high speed PCI express communication bus. Plane fitting was applied to the detection and highlighting of tables and free floor space in the system and a framerate of 60 frames per second has been achieved.

6.2 Future Work

The work of this thesis is by no means complete and in fact the results of this work have quite possibly opened up new avenues for investigation. A number of limitations of this work exist and these provide the most obvious paths for future work. However, the systems and functionality developed as part of this thesis could be applied for other purposes and are likely to have use in other fields of research as well. Possible future work that could be done to improve upon this project and potential areas of application outside the field of medical bionics are outlined as follows.

Improvements to the Hatpack

The phosphenes that will be evoked in a cortical prosthesis will be gaze-contingent. This means that they will remain in the same location in the visual field regardless of movements of the eyeball. This property has not been implemented in the Hatpack and could be part of future work for the system. Wearers of the Hatpack have alluded to possibility of the lack of gaze-contingency not making a difference, however this can only be confirmed through testing and comparison of the two cases. Furthermore, the revised mapping of the Hatpack involves phosphenes that are displayed in only one of the two visual hemispheres and this seems to force users to scan over the phosphenes more as there is a natural tendency for looking towards the centre of a pattern displayed on the screen.

More image processing and computer vision algorithms could be investigated for implementation as part of the Hatpack features. The implementation of face detection for example would be useful to indicate to a user if there is someone within the immediate environment that is looking towards them. Recognition of the faces could also allow the user distinguish who they are looking at. This information could be conveyed to the user through an auditory form through the integration of audio output. Other functions could include various object and shape detection implementations which could possibly be presented to the user in the form of simplified icons or letters. These could be quite useful for navigational purposes, via the highlighting of known landmarks, pathways, doorways, street signs, etc.

More Hatpack Psychophysics Testing

The psychophysics experiments that have been completed as part of this work are not a formalised set of clinical trials. They are somewhat simplistic and possible future work could be to conduct larger scale tests that are more difficult, have a wider range of tasks and more exhaustively test the functions that are implemented on the Hatpack system.

The tests conducted as part of this thesis have all been indoors. It would be quite interesting to see how a participant would perform in an outdoor setting. A possible task could be for a participant to try and navigate from the inside of an office building to a nearby bus stop or car park. This would also test the ability of the Hatpack system and its functions to deal with the high dynamic range of lighting between the two.

It would also be quite interesting to further investigate learning effects. Long-term testing involving many hours of continuous Hatpack use could be a possible option. This could be repeated over a number of weeks or even months in order to determine the point of diminishing return in terms of learning.

Extensions to 3D Sensing and Applications

The implementation of the 3D end-to-end system of Chapter 5 is a proof of concept system and quite general. The inclusion of inertial sensing could be part of future work to allow for the compensation of changes to camera position and orientation. This would be quite likely in practical use of a bionic vision system. Navigational cues could be displayed on screen to help users in everyday tasks. More advanced object and surface detection could be implemented to identify to the user things such as chairs, doorways, bins, stairs, paths, and other objects and landmarks that would be useful to users.

Advancements in FPGA technology have meant that many FPGAs are now equipped with hard wired implementations of powerful ARM processor cores integrated into the microchip. These system-on-chip devices could provide a simpler way to interface a depth sensor with the FPGA as the need for a computer may no longer be necessary. This could also mean that a portable version of the system could be developed similar to the Hatpack.

The floor space and tabletop detection applications implemented have used thresholds that have been determined empirically based on a maximum range. Ideally these would need to be calculated for each pixel in a patch using the error models developed in Chapter 4. However these models are quite general and for the specific case of highlighting planes for bionic visions applications it is not necessary to be as accurate given the low resolution, so long as a suitable representative image can be achieved. For more critical applications such as the tracking of planes, the variability of the normalised plane parameters becomes important for accuracy and so the use of the models developed would be necessary. An investigation into a hardware implementation of these error model equations could be completed for future work.

The error analysis performed in Chapter 4 was completed using a best-case approach. The effects of attenuation of IR illumination and reflection, and lens distortion have been ignored. From the results of the applications in Chapter 5 it is evident that the noise level of depth data increases significantly towards the peripheral edges of the depth frames. This means that the developed error model equations would not be suitable for the outer parts of the image. A more detailed analysis of error would be useful for properly defining the variability of depth sensor data.

Psychophysics trials would be another area for future work. This could be done in a similar way the experiments carried out with the Hatpack system. Low resolution mapping would need to incorporated and the mapping implementation within the Hatpack could be used for this. With first in-human trials approaching however, the possibility of conducting testing with actual implanted patients is a likely option. Due to the modular implementation of the Gennaris device, alternative hardware for sensing and processing can be investigated without the need for further surgery or modification to the stimulating tiles implanted within the patient's brain.

Use in Other Fields of Research

The fields of Robotics and Computer Vision could find use in the work completed in this thesis. The low resolution mapping implementation used in the Hatpack along with some of the processing functions could be used for robotic systems that are limited to low resolution sensing. Likewise, plane fitting is a fundamental building block of many 3D robotics and computer vision applications and the ability to achieve the level of real time performance observed in this work could prove very useful in these fields.

The hybrid computer-FPGA architecture used for the end-to-end system allows for the potential of integrating a number of different sensors simultaneously. The fusion of inertial and GPS sensors into the system could allow for the implementation of Simultaneous Localisation and Mapping [7] functionality. This has been widely researched in the field of robotics and could prove a useful addition for a bionic vision system.

6.3 Concluding Remarks

The development of the Hatpack has provided a robust and flexible real time system that is suitable for use as a psychophysical test platform. By closely representing the anticipated level of the Gennaris, it also provides a useful visual tool that can be used handle the expectations of the general public and family and friends of future patients. The experiments carried out with the Hatpack have shown not surprisingly that it is significantly harder to complete tasks that resemble everyday activities under the limited resolution of a bionic vision device. This has motivated an investigation into the possibility of integrating alternate sensing approaches to potentially close the gap between the abilities of normally sighted and blind individuals. The development of the real time end-to-end system has shown that it is in fact feasible to incorporate 3D depth sensing into a bionic vision device. Furthermore, the application of planefitting to free floor space and tabletop detection was achieved at a framerate of 60 frames per second.

The Hatpack system has influenced the development of the Gennaris and given confidence that useful outcomes will be obtained with the resolution and framerate that is anticipated of initial implementations. It has also provided good publicity and convinced and inspired sponsors to invest in the future of the project. Hopefully the results of this work, in particular the integration of 3D sensing, will be incorporated into future revisions of the Gennaris and other vision implants in order to improve the quality of life of future patients and allow them to regain abilities that were lost with their sight.

Appendix A

Derivations of Equations (4.38) to (4.41)

A.1 Variance of the Scaling Factor

From Equation (4.25), the scaling factor is shown to be $\lambda = |\alpha| + |\beta| + |\gamma|$. There are 8 different cases, since α, β, γ can all have either positive or negative results. So λ can be one of the following:

$$\begin{aligned}
\lambda_1 &= (\alpha) + (\beta) + (\gamma) = -FA - FB + (C + Au_c + Bv_c) \\
\lambda_2 &= (\alpha) + (\beta) + (-\gamma) = -FA - FB - (C + Au_c + Bv_c) \\
\lambda_3 &= (\alpha) + (-\beta) + (\gamma) = -FA + FB + (C + Au_c + Bv_c) \\
\lambda_4 &= (\alpha) + (-\beta) + (-\gamma) = -FA + FB - (C + Au_c + Bv_c) \\
\lambda_5 &= (-\alpha) + (\beta) + (\gamma) = FA - FB + (C + Au_c + Bv_c) \\
\lambda_6 &= (-\alpha) + (\beta) + (-\gamma) = FA - FB - (C + Au_c + Bv_c) \\
\lambda_7 &= (-\alpha) + (-\beta) + (\gamma) = FA + FB + (C + Au_c + Bv_c) \\
\lambda_8 &= (-\alpha) + (-\beta) + (-\gamma) = FA + FB - (C + Au_c + Bv_c)
\end{aligned} \tag{A.1}$$

In order to find variance equations for the 8 cases listed above, they need to be first represented in terms of the individual patch elements z_1 to z_9, then expanded and grouped according to correlated terms. This working is shown as follows for λ_1:

$$\begin{aligned}
\lambda_1 &= (\alpha) + (\beta) + (\gamma) \\
&= -FA - FB + (C + Au_c + Bv_c) \\
&= -F\frac{\sum(column\ 3) - \sum(column\ 1)}{30} - F\frac{\sum(row\ 3) - \sum(row\ 1)}{30} \\
&\quad + \frac{\sum_{i=1}^{m} z_i}{n} + u_c\frac{\sum(column\ 3) - \sum(column\ 1)}{30} \\
&\quad + v_c\frac{\sum(row\ 3) - \sum(row\ 1)}{30} \\
&= -\frac{F}{30}(z_3 + z_6 + z_9 - z_1 - z_4 - z_7) - \frac{F}{30}(z_7 + z_8 + z_9 - z_1 - z_2 - z_3) \\
&\quad + \frac{1}{9}(z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9) + \frac{u_c}{30}(z_3 + z_6 + z_9 - z_1 - z_4 - z_7) \\
&\quad + \frac{v_c}{30}(z_7 + z_8 + z_9 - z_1 - z_2 - z_3) \\
&= z_1(\frac{1}{9} + \frac{F}{15} - \frac{u_c}{30} - \frac{v_c}{30}) + z_2(\frac{1}{9} + \frac{F}{30} - \frac{v_c}{30}) + z_3(\frac{1}{9} + \frac{u_c}{30} - \frac{v_c}{30}) + z_4(\frac{1}{9} + \frac{F}{30} - \frac{u_c}{30}) \\
&\quad + z_5(\frac{1}{9}) + z_6(\frac{1}{9} - \frac{F}{30} + \frac{v_c}{30}) + z_7(\frac{1}{9} - \frac{u_c}{30} + \frac{v_c}{30}) + z_8(\frac{1}{9} - \frac{F}{30} + \frac{v_c}{30}) \\
&\quad + z_9(\frac{1}{9} - \frac{F}{15} + \frac{u_c}{30} + \frac{v_c}{30})
\end{aligned} \tag{A.2}$$

Substituting in the values of $F = 366$, $u_c = 256$ and $v_c = 212$ gives:

$$\begin{aligned}
\lambda_1 &= z_1(\frac{401}{45}) + z_2(\frac{236}{45}) + z_3(\frac{71}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) + z_6(\frac{-32}{9}) + z_7(\frac{-61}{45}) \\
&\quad + z_8(\frac{-226}{45}) + z_9(\frac{-391}{45})
\end{aligned} \tag{A.3}$$

Similarly, the remaining cases of λ are:

$$\begin{aligned}\lambda_2 = {} & z_1(\frac{359}{9}) + z_2(\frac{862}{45}) + z_3(\frac{-71}{45}) + z_4(\frac{928}{45}) + z_5(\frac{-1}{9}) + z_6(\frac{-938}{45}) + z_7(\frac{61}{45}) \\ & + z_8(\frac{-872}{45}) + z_9(\frac{-361}{9})\end{aligned} \tag{A.4}$$

$$\begin{aligned}\lambda_3 = {} & z_1(\frac{-697}{45}) + z_2(\frac{-862}{45}) + z_3(\frac{-1027}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) + z_6(\frac{-32}{9}) + z_7(\frac{1037}{45}) \\ & + z_8(\frac{872}{45}) + z_9(\frac{707}{45})\end{aligned} \tag{A.5}$$

$$\begin{aligned}\lambda_4 = {} & z_1(\frac{697}{45}) + z_2(\frac{-236}{45}) + z_3(\frac{-1169}{45}) + z_4(\frac{928}{45}) + z_5(\frac{-1}{9}) + z_6(\frac{-938}{45}) + z_7(\frac{1159}{45}) \\ & + z_8(\frac{226}{45}) + z_9(\frac{-707}{45})\end{aligned} \tag{A.6}$$

$$\begin{aligned}\lambda_5 = {} & z_1(\frac{-697}{45}) + z_2(\frac{236}{45}) + z_3(\frac{1169}{45}) + z_4(\frac{-928}{45}) + z_5(\frac{1}{9}) + z_6(\frac{938}{45}) + z_7(\frac{-1159}{45}) \\ & + z_8(\frac{-226}{45}) + z_9(\frac{707}{45})\end{aligned} \tag{A.7}$$

$$\begin{aligned}\lambda_6 = {} & z_1(\frac{697}{45}) + z_2(\frac{862}{45}) + z_3(\frac{1027}{45}) + z_4(\frac{-34}{9}) + z_5(\frac{-1}{9}) + z_6(\frac{32}{9}) + z_7(\frac{-1037}{45}) \\ & + z_8(\frac{-872}{45}) + z_9(\frac{-707}{45})\end{aligned} \tag{A.8}$$

$$\begin{aligned}\lambda_7 = {} & z_1(\frac{-359}{9}) + z_2(\frac{-862}{45}) + z_3(\frac{71}{45}) + z_4(\frac{-928}{45}) + z_5(\frac{1}{9}) + z_6(\frac{938}{45}) + z_7(\frac{-61}{45}) \\ & + z_8(\frac{872}{45}) + z_9(\frac{361}{9})\end{aligned} \tag{A.9}$$

$$\begin{aligned}\lambda_8 = {} & z_1(\frac{-401}{45}) + z_2(\frac{-236}{45}) + z_3(\frac{-71}{45}) + z_4(\frac{-34}{9}) + z_5(\frac{-1}{9}) + z_6(\frac{32}{9}) + z_7(\frac{61}{45}) \\ & + z_8(\frac{226}{45}) + z_9(\frac{391}{45})\end{aligned} \tag{A.10}$$

Given these equations, the following variances have been found:

$$\begin{aligned}\sigma_{\lambda_1} = {} & \sigma_z^2(\frac{401}{45})^2 + \sigma_z^2(\frac{236}{45})^2 + \sigma_z^2(\frac{71}{45})^2 + \sigma_z^2(\frac{34}{9})^2 + \sigma_z^2(\frac{1}{9})^2 + \sigma_z^2(\frac{-32}{9})^2 + \sigma_z^2(\frac{-61}{45})^2 \\ & + \sigma_z^2(\frac{-226}{45})^2 + \sigma_z^2(\frac{-391}{45})^2 \\ = {} & \sigma_z^2\frac{53749}{225} \approx (15.456\sigma_z)^2\end{aligned} \tag{A.11}$$

$$\begin{aligned}\sigma_{\lambda_2} = {} & \sigma_z^2(\frac{359}{9})^2 + \sigma_z^2(\frac{862}{45})^2 + \sigma_z^2(\frac{-71}{45})^2 + \sigma_z^2(\frac{928}{45})^2 + \sigma_z^2(\frac{-1}{9})^2 + \sigma_z^2(\frac{-938}{45})^2 + \sigma_z^2(\frac{61}{45})^2 \\ & + \sigma_z^2(\frac{-872}{45})^2 + \sigma_z^2(\frac{-361}{9})^2 \\ = {} & \sigma_z^2\frac{1081477}{225} \approx (69.329\sigma_z)^2\end{aligned} \tag{A.12}$$

$$\begin{aligned}\sigma_{\lambda_3} = {} & \sigma_z^2(\frac{-697}{45})^2 + \sigma_z^2(\frac{-862}{45})^2 + \sigma_z^2(\frac{-1027}{45})^2 + \sigma_z^2(\frac{34}{9})^2 + \sigma_z^2(\frac{1}{9})^2 + \sigma_z^2(\frac{-32}{9})^2 \\ & + \sigma_z^2(\frac{1037}{45})^2 + \sigma_z^2(\frac{872}{45})^2 + \sigma_z^2(\frac{707}{45})^2 \\ = {} & \sigma_z^2\frac{519301}{225} \approx (48.042\sigma_z)^2\end{aligned} \tag{A.13}$$

$$\begin{aligned}\sigma_{\lambda_4} &= \sigma_z^2(\frac{697}{45})^2 + \sigma_z^2(\frac{-236}{45})^2 + \sigma_z^2(\frac{-1169}{45})^2 + \sigma_z^2(\frac{928}{45})^2 + \sigma_z^2(\frac{-1}{9})^2 + \sigma_z^2(\frac{-938}{45})^2 \\ &\quad + \sigma_z^2(\frac{1159}{45})^2 + \sigma_z^2(\frac{226}{45})^2 + \sigma_z^2(\frac{-707}{45})^2 \\ &= \sigma_z^2\frac{24637}{9} \approx (52.321\sigma_z)^2\end{aligned} \tag{A.14}$$

$$\begin{aligned}\sigma_{\lambda_5} &= \sigma_z^2(\frac{-697}{45})^2 + \sigma_z^2(\frac{236}{45})^2 + \sigma_z^2(\frac{1169}{45})^2 + \sigma_z^2(\frac{-928}{45})^2 + \sigma_z^2(\frac{1}{9})^2 + \sigma_z^2(\frac{938}{45})^2 \\ &\quad + \sigma_z^2(\frac{-1159}{45})^2 + \sigma_z^2(\frac{-226}{45})^2 + \sigma_z^2(\frac{707}{45})^2 \\ &= \sigma_z^2\frac{24637}{9} \approx (52.321\sigma_z)^2\end{aligned} \tag{A.15}$$

$$\begin{aligned}\sigma_{\lambda_6} &= \sigma_z^2(\frac{697}{45})^2 + \sigma_z^2(\frac{862}{45})^2 + \sigma_z^2(\frac{1027}{45})^2 + \sigma_z^2(\frac{-34}{9})^2 + \sigma_z^2(\frac{-1}{9})^2 + \sigma_z^2(\frac{32}{9})^2 \\ &\quad + \sigma_z^2(\frac{-1037}{45})^2 + \sigma_z^2(\frac{-872}{45})^2 + \sigma_z^2(\frac{-707}{45})^2 \\ &= \sigma_z^2\frac{519301}{225} \approx (48.042\sigma_z)^2\end{aligned} \tag{A.16}$$

$$\begin{aligned}\sigma_{\lambda_7} &= \sigma_z^2(\frac{-359}{9})^2 + \sigma_z^2(\frac{-862}{45})^2 + \sigma_z^2(\frac{71}{45})^2 + \sigma_z^2(\frac{-928}{45})^2 + \sigma_z^2(\frac{1}{9})^2 + \sigma_z^2(\frac{938}{45})^2 \\ &\quad + \sigma_z^2(\frac{-61}{45})^2 + \sigma_z^2(\frac{872}{45})^2 + \sigma_z^2(\frac{361}{9})^2 \\ &= \sigma_z^2\frac{1081477}{225} \approx (69.329\sigma_z)^2\end{aligned} \tag{A.17}$$

$$\begin{aligned}\sigma_{\lambda_8} &= \sigma_z^2(\frac{-401}{45})^2 + \sigma_z^2(\frac{-236}{45})^2 + \sigma_z^2(\frac{-71}{45})^2 + \sigma_z^2(\frac{-34}{9})^2 + \sigma_z^2(\frac{-1}{9})^2 + \sigma_z^2(\frac{32}{9})^2 \\ &\quad + \sigma_z^2(\frac{61}{45})^2 + \sigma_z^2(\frac{226}{45})^2 + \sigma_z^2(\frac{391}{45})^2 \\ &= \sigma_z^2\frac{53749}{225} \approx (15.456\sigma_z)^2\end{aligned} \tag{A.18}$$

It can be seen that there are only 4 unique cases. Furthermore, only the best and worst case variances are of interest. This leaves the following two variances that have been shown in Equation (4.39) of Chapter 4:

$$\begin{aligned} \sigma_{\lambda_1}^2 &= \frac{\sigma_z^2 53749}{225} \approx (15.456\sigma_z)^2 \\ \sigma_{\lambda_2}^2 &= \frac{\sigma_z^2 1081477}{225} \approx (69.329\sigma_z)^2 \end{aligned} \tag{A.19}$$

A.2 Covariances of Plane Parameters with the Scaling Factor

The covariances of each of the plane parameters α, β, γ, and κ with the scaling factor λ are required for the approximation that will be used to find the variance of the normalised plane parameters. These covariances can be found by working out the product of each of the pairs of variables, and removing the non-correlated cross-terms. The derivation of the covariance of α with the best and worst case λ values is outlined in detail as an example. The products of the two pairs are:

$$\begin{aligned} \alpha\lambda_1 = {} & \big[\frac{61}{5}(z_1 - z_3 + z_4 - z_6 + z_7 - z_9)\big] \times \big[z_1(\frac{401}{45}) + z_2(\frac{236}{45}) + z_3(\frac{71}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) \\ & - z_6(\frac{32}{9}) - z_7(\frac{61}{45}) - z_8(\frac{226}{45}) - z_9(\frac{391}{45})\big] \\ = {} & (\frac{14396 z_1 z_2}{225}) - (\frac{1342 z_1 z_3}{15}) + (\frac{34831 z_1 z_4}{225}) - (\frac{14396 z_2 z_3}{225}) + (\frac{61 z_1 z_5}{45}) + (\frac{14396 z_2 z_4}{225}) \\ & - (\frac{11407 z_1 z_6}{75}) - (\frac{671 z_3 z_4}{25}) + (\frac{4148 z_1 z_7}{45}) - (\frac{14396 z_2 z_6}{225}) - (\frac{61 z_3 z_5}{45}) - (\frac{13786 z_1 z_8}{225}) \\ & + (\frac{14396 z_2 z_7}{225}) + (\frac{5429 z_3 z_6}{225}) + (\frac{61 z_4 z_5}{45}) - (\frac{5368 z_1 z_9}{25}) + (\frac{2684 z_3 z_7}{75}) - (\frac{1342 z_4 z_6}{15}) \\ & - (\frac{14396 z_2 z_9}{225}) + (\frac{13786 z_3 z_8}{225}) + (\frac{6649 z_4 z_7}{225}) - (\frac{61 z_5 z_6}{45}) + (\frac{3904 z_3 z_9}{45}) - (\frac{13786 z_4 z_8}{225}) \\ & + (\frac{61 z_5 z_7}{45}) - (\frac{11407 z_4 z_9}{75}) - (\frac{671 z_6 z_7}{25}) - (\frac{61 z_5 z_9}{45}) + (\frac{13786 z_6 z_8}{225}) + (\frac{33611 z_6 z_9}{225}) \\ & - (\frac{13786 z_7 z_8}{225}) - (\frac{1342 z_7 z_9}{15}) + (\frac{13786 z_8 z_9}{225}) + (\frac{24461 z_1^2}{225}) - (\frac{4331 z_3^2}{225}) + (\frac{2074 z_4^2}{45}) \\ & + (\frac{1952 z_6^2}{45}) - (\frac{3721 z_7^2}{225}) + (\frac{23851 z_9^2}{225}) \end{aligned} \tag{A.20}$$

$$
\begin{aligned}
\alpha\lambda_2 = {} & \big[\frac{61}{5}(z_1 - z_3 + z_4 - z_6 + z_7 - z_9)\big] \times \big[z_1(\frac{359}{9}) + z_2(\frac{862}{45}) - z_3(\frac{71}{45}) + z_4(\frac{928}{45}) - z_5(\frac{1}{9}) \\
& - z_6(\frac{938}{45}) + z_7(\frac{61}{45}) - z_8(\frac{872}{45}) - z_9(\frac{361}{9})\big] \\
= {} & (\frac{52582z_1z_2}{225}) - (\frac{37942z_1z_3}{75}) + (\frac{166103z_1z_4}{225}) - (\frac{52582z_2z_3}{225}) - (\frac{61z_1z_5}{45}) + (\frac{52582z_2z_4}{225}) \\
& - (\frac{55571z_1z_6}{75}) - (\frac{6771z_3z_4}{25}) + (\frac{113216z_1z_7}{225}) - (\frac{52582z_2z_6}{225}) + (\frac{61z_3z_5}{45}) - (\frac{53192z_1z_8}{225}) \\
& + (\frac{52582z_2z_7}{225}) + (\frac{61549z_3z_6}{225}) - (\frac{61z_4z_5}{45}) - (\frac{22021z_1z_9}{45}) - (\frac{2684z_3z_7}{75}) - (\frac{37942z_4z_6}{75}) \\
& - (\frac{52582z_2z_9}{225}) + (\frac{53192z_3z_8}{225}) + (\frac{60329z_4z_7}{225}) + (\frac{61z_5z_6}{45}) + (\frac{114436z_3z_9}{225}) - (\frac{53192z_4z_8}{225}) \\
& - (\frac{61z_5z_7}{45}) - (\frac{55571z_4z_9}{75}) - (\frac{6771z_6z_7}{25}) + (\frac{61z_5z_9}{45}) + (\frac{53192z_6z_8}{225}) + (\frac{167323z_6z_9}{225}) \\
& - (\frac{53192z_7z_8}{225}) - (\frac{37942z_7z_9}{75}) + (\frac{53192z_8z_9}{225}) + (\frac{21899z_1^2}{45}) + (\frac{4331z_3^2}{225}) + (\frac{56608z_4^2}{225}) \\
& + (\frac{57218z_6^2}{225}) + (\frac{3721z_7^2}{225}) + (\frac{22021z_9^2}{45})
\end{aligned}
\tag{A.21}
$$

The covariances of these two equations will only involve the components that have correlated cross-terms and can be expressed as:

$$
\begin{aligned}
\sigma_{\alpha\lambda_1} &= (\frac{24461\sigma_z^2}{225}) - (\frac{4331\sigma_z^2}{225}) + (\frac{2074\sigma_z^2}{45}) + (\frac{1952\sigma_z^2}{45}) - (\frac{3721\sigma_z^2}{225}) + (\frac{23851\sigma_z^2}{225}) \\
&= \sigma_z^2(\frac{1342}{5}) \\
&\approx (16.38\sigma_z)^2
\end{aligned}
\tag{A.22}
$$

$$
\begin{aligned}
\sigma_{\alpha\lambda_2} &= (\frac{21899\sigma_z^2}{45}) + (\frac{4331\sigma_z^2}{225}) + (\frac{56608\sigma_z^2}{225}) + (\frac{57218\sigma_z^2}{225}) + (\frac{3721\sigma_z^2}{225}) + (\frac{22021\sigma_z^2}{45}) \\
&= \sigma_z^2(\frac{37942}{25}) \\
&\approx (38.96\sigma_z)^2
\end{aligned}
\tag{A.23}
$$

In a similar manner the products and covariances of the other plane parameters (β,γ and κ) with the scaling factor λ have been calculated and are as follows (note that the uncorrelated cross-terms have been excluded from the expressions for the products for convenience):

$$\begin{aligned}
\beta\lambda_1 &= \big[\frac{61}{5}(z_1 + z_2 + z_3 - z_7 - z_8 - z_9)\big] \\
&\times \big[z_1(\frac{401}{45}) + z_2(\frac{236}{45}) + z_3(\frac{71}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) - z_6(\frac{32}{9}) - z_7(\frac{61}{45}) \\
&- z_8(\frac{226}{45}) - z_9(\frac{391}{45})\big] \\
&= \ldots + (\frac{24461z_1^2}{225}) + (\frac{14396z_2^2}{225}) + (\frac{4331z_3^2}{225}) + (\frac{3721z_7^2}{225}) + (\frac{13786z_8^2}{225}) + (\frac{23851z_9^2}{225})
\end{aligned} \tag{A.24}$$

$$\begin{aligned}
\beta\lambda_2 &= \big[\frac{61}{5}(z_1 + z_2 + z_3 - z_7 - z_8 - z_9)\big] \\
&\times \big[z_1(\frac{359}{9}) + z_2(\frac{862}{45}) - z_3(\frac{71}{45}) + z_4(\frac{928}{45}) - z_5(\frac{1}{9}) - z_6(\frac{938}{45}) + z_7(\frac{61}{45}) \\
&- z_8(\frac{872}{45}) - z_9(\frac{361}{9})\big] \\
&= \ldots + (\frac{21899z_1^2}{45}) + (\frac{52582z_2^2}{225}) - (\frac{4331z_3^2}{225}) - (\frac{3721z_7^2}{225}) + (\frac{53192z_8^2}{225}) + (\frac{22021z_9^2}{45})
\end{aligned} \tag{A.25}$$

$$\begin{aligned}
\sigma_{\beta\lambda_1} &= (\frac{24461\sigma_z^2}{225}) + (\frac{14396\sigma_z^2}{225}) + (\frac{4331\sigma_z^2}{225}) + (\frac{3721\sigma_z^2}{225}) + (\frac{13786\sigma_z^2}{225}) + (\frac{23851\sigma_z^2}{225}) \\
&= \sigma_z^2(\frac{9394}{25}) \\
&\approx (19.38\sigma_z)^2
\end{aligned} \tag{A.26}$$

$$\begin{aligned}
\sigma_{\beta\lambda_2} &= (\frac{21899\sigma_z^2}{45}) + (\frac{52582\sigma_z^2}{225}) - (\frac{4331\sigma_z^2}{225}) - (\frac{3721\sigma_z^2}{225}) + (\frac{53192\sigma_z^2}{225}) + (\frac{22021\sigma_z^2}{45}) \\
&= \sigma_z^2(\frac{35258}{25}) \\
&\approx (37.55\sigma_z)^2
\end{aligned} \tag{A.27}$$

$$\begin{aligned}
\gamma\lambda_1 = \big[& z_1(\frac{-697}{45}) + z_2(\frac{-313}{45}) + z_3(\frac{71}{45}) + z_4(\frac{-379}{45}) + z_5(\frac{1}{9}) + z_6(\frac{389}{45}) + z_7(\frac{-61}{45}) \\
&+ z_8(\frac{323}{45}) + z_9(\frac{707}{45})\big] \times \big[z_1(\frac{401}{45}) + z_2(\frac{236}{45}) + z_3(\frac{71}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) \\
&- z_6(\frac{32}{9}) - z_7(\frac{61}{45}) - z_8(\frac{226}{45}) - z_9(\frac{391}{45})\big] \\
= \ldots &- (\frac{279497z_1^2}{2025}) - (\frac{73868z_2^2}{2025}) + (\frac{5041z_3^2}{2025}) - (\frac{12448z_4^2}{405}) + (\frac{z_5^2}{81}) - (\frac{12448z_6^2}{405}) \\
&+ (\frac{3721z_7^2}{2025}) - (\frac{72998z_8^2}{2025}) - (\frac{276437z_9^2}{2025})
\end{aligned} \tag{A.28}$$

$$\begin{aligned}
\gamma\lambda_2 = \big[& z_1(\frac{-697}{45}) + z_2(\frac{-313}{45}) + z_3(\frac{71}{45}) + z_4(\frac{-379}{45}) + z_5(\frac{1}{9}) + z_6(\frac{389}{45}) + z_7(\frac{-61}{45}) \\
&+ z_8(\frac{323}{45}) + z_9(\frac{707}{45})\big] \times \big[z_1(\frac{359}{9}) + z_2(\frac{862}{45}) - z_3(\frac{71}{45}) + z_4(\frac{928}{45}) - z_5(\frac{1}{9}) \\
&- z_6(\frac{938}{45}) + z_7(\frac{61}{45}) - z_8(\frac{872}{45}) - z_9(\frac{361}{9})\big] \\
= \ldots &- (\frac{250223z_1^2}{405}) - (\frac{269806z_2^2}{2025}) - (\frac{5041z_3^2}{2025}) - (\frac{351712z_4^2}{2025}) - (\frac{z_5^2}{81}) - (\frac{364882z_6^2}{2025}) \\
&- (\frac{3721z_7^2}{2025}) - (\frac{281656z_8^2}{2025}) - (\frac{255227z_9^2}{405})
\end{aligned} \tag{A.29}$$

$$\begin{aligned}
\sigma_{\gamma\lambda_1} &= (\frac{279497\sigma_z^2}{2025}) - (\frac{73868\sigma_z^2}{2025}) + (\frac{5041\sigma_z^2}{2025}) - (\frac{12448\sigma_z^2}{405}) + (\frac{\sigma_z^2}{81}) - (\frac{12448\sigma_z^2}{405}) \\
&\quad + (\frac{3721\sigma_z^2}{2025}) - (\frac{72998\sigma_z^2}{2025}) - (\frac{276437\sigma_z^2}{2025}) \\
&= \sigma_z^2(\frac{91187}{225}) \\
&\approx -(20.13\sigma_z)^2
\end{aligned} \tag{A.30}$$

$$\begin{aligned}
\sigma_{\gamma\lambda_2} &= (\frac{250223\sigma_z^2}{405}) - (\frac{269806\sigma_z^2}{2025}) - (\frac{5041\sigma_z^2}{2025}) - (\frac{351712\sigma_z^2}{2025}) - (\frac{\sigma_z^2}{81}) - (\frac{364882\sigma_z^2}{2025}) \\
&\quad - (\frac{3721\sigma_z^2}{2025}) - (\frac{281656\sigma_z^2}{2025}) - (\frac{255227\sigma_z^2}{405}) \\
&= \sigma_z^2(\frac{-422677}{225}) \\
&\approx -(43.34\sigma_z)^2
\end{aligned} \tag{A.31}$$

$$\begin{aligned}
\kappa\lambda_1 &= \big[\frac{1}{9}(z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8 + z_9)\big]^2 \\
&\quad \times \big[z_1(\frac{401}{45}) + z_2(\frac{236}{45}) + z_3(\frac{71}{45}) + z_4(\frac{34}{9}) + z_5(\frac{1}{9}) - z_6(\frac{32}{9}) - z_7(\frac{61}{45}) \\
&\quad - z_8(\frac{226}{45}) - z_9(\frac{391}{45})\big] \\
&= \ldots + (\frac{401z_1^3}{3645}) + (\frac{236z_2^3}{3645}) + (\frac{71z_3^3}{3645}) + (\frac{34z_4^3}{729}) + (\frac{z_5^3}{729}) - (\frac{32z_6^3}{729}) \\
&\quad - (\frac{61z_7^3}{3645}) - (\frac{226z_8^3}{3645}) - (\frac{391z_9^3}{3645})
\end{aligned} \tag{A.32}$$

$$\begin{aligned}
\kappa\lambda_2 &= \big[\frac{1}{9}(z_1+z_2+z_3+z_4+z_5+z_6+z_7+z_8+z_9)\big]^2 \\
&\times \big[z_1(\frac{359}{9}) + z_2(\frac{862}{45}) - z_3(\frac{71}{45}) + z_4(\frac{928}{45}) - z_5(\frac{1}{9}) - z_6(\frac{938}{45}) + z_7(\frac{61}{45}) \\
&- z_8(\frac{872}{45}) - z_9(\frac{361}{9})\big] \\
&= \ldots + (\frac{359z_1^2}{729}) + (\frac{862z_2^2}{3645}) - (\frac{71z_3^2}{3645}) + (\frac{928z_4^2}{3645}) - (\frac{z_5^2}{729}) - (\frac{938z_6^2}{3645}) \\
&+ (\frac{61z_7^2}{3645}) - (\frac{872z_8^2}{3645}) - (\frac{361z_9^2}{729})
\end{aligned} \tag{A.33}$$

$$\begin{aligned}
\sigma_{\kappa\lambda_1} &= (\frac{401\sigma_z^3}{3645}) + (\frac{236\sigma_z^3}{3645}) + (\frac{71\sigma_z^3}{3645}) + (\frac{34\sigma_z^3}{729}) + (\frac{\sigma_z^3}{729}) - (\frac{32\sigma_z^3}{729}) \\
&- (\frac{61\sigma_z^3}{3645}) - (\frac{226\sigma_z^3}{3645}) - (\frac{391\sigma_z^3}{3645}) \\
&= \frac{\sigma_z^3}{81} \\
&\approx 0.012\sigma_z^3
\end{aligned} \tag{A.34}$$

$$\begin{aligned}
\sigma_{\kappa\lambda_2} &= (\frac{359\sigma_z^3}{729}) + (\frac{862\sigma_z^3}{3645}) - (\frac{71\sigma_z^3}{3645}) + (\frac{928\sigma_z^3}{3645}) - (\frac{\sigma_z^3}{729}) - (\frac{938\sigma_z^3}{3645}) \\
&+ (\frac{61\sigma_z^3}{3645}) - (\frac{872\sigma_z^3}{3645}) - (\frac{361\sigma_z^3}{729}) \\
&= -\frac{\sigma_z^3}{81} \\
&\approx -0.012\sigma_z^3
\end{aligned} \tag{A.35}$$

The equations for all the covariances of Equation (4.40) from Chapter 4 have been derived and are summarised as follows:

$$
\begin{aligned}
&For\ best\ case\ of\ \sigma_{\lambda_1} = 15.456\sigma_z,\\
&\sigma_{\alpha\lambda_1} = \frac{1342\sigma_z^2}{5} \approx 16.38^2\sigma_z^2\\
&\sigma_{\beta\lambda_1} = \frac{9394\sigma_z^2}{25} \approx 19.38^2\sigma_z^2\\
&\sigma_{\gamma\lambda_1} = -\frac{91187\sigma_z^2}{225} \approx -20.13^2\sigma_z^2\\
&\sigma_{\kappa\lambda_1} = \frac{\sigma_z^3}{81}
\end{aligned}
\tag{A.36}
$$

$$
\begin{aligned}
&For\ worst\ case\ of\ \sigma_{\lambda_2} = 69.329\sigma_z,\\
&\sigma_{\alpha\lambda_2} = \frac{37942\sigma_z^2}{25} \approx 38.96^2\sigma_z^2\\
&\sigma_{\beta\lambda_2} = \frac{35258\sigma_z^2}{25} \approx 37.55^2\sigma_z^2\\
&\sigma_{\gamma\lambda_2} = -\frac{422677\sigma_z^2}{225} \approx -43.34^2\sigma_z^2\\
&\sigma_{\kappa\lambda_2} = -\frac{\sigma_z^3}{81}
\end{aligned}
$$

A.3 Standard Deviations of Normalised Plane Parameters

It is now possible to use the approximation in Equation (4.37) of Chapter 4 (restated below) to find the variances of the normalised plane parameters α_n, β_n, γ_n and κ_n.

$$
\begin{aligned}
&For\ some\ f = \frac{X}{Y}\\
&\sigma_f^2 \approx f^2\left[\left(\frac{\sigma_X}{X}\right)^2 + \left(\frac{\sigma_Y}{Y}\right)^2 - 2\frac{\sigma_{XY}}{XY}\right]
\end{aligned}
\tag{A.37}
$$

The standard deviations of α, β, γ and κ from Equation (4.36) of Chapter 4 are as follows:

$$
\begin{aligned}
\sigma_\alpha &\approx 29.88\sigma_z \\
\sigma_\beta &\approx 29.88\sigma_z \\
\sigma_\gamma &\approx 27.14\sigma_z \\
\sigma_\kappa &\approx 0.66C\sigma_z
\end{aligned}
\tag{A.38}
$$

Variance and Standard Deviation of α_n

For α_n, the variance using the best-case λ_1 is given by:

$$
\begin{aligned}
\sigma_{\alpha_n}^2 &\approx \left(\frac{\alpha}{\lambda_1}\right)^2 \left[\left(\frac{\sigma_\alpha}{\alpha}\right)^2 + \left(\frac{\sigma_{\lambda_1}}{\lambda_1}\right)^2 - 2\frac{\sigma_{\alpha\lambda_1}}{\alpha\lambda_1}\right] \\
&= \left(\frac{\alpha}{\lambda_1}\right)^2 \left[\left(\frac{29.88\sigma_z}{\alpha}\right)^2 + \left(\frac{15.46\sigma_z}{\lambda_1}\right)^2 - 2\frac{(16.38\sigma_z)^2}{\alpha\lambda_1}\right] \\
&= \sigma_z^2 \left(\frac{\alpha}{\lambda_1}\right)^2 \left[\frac{29.88^2}{\alpha^2} + \frac{15.46^2}{\lambda_1^2} - \frac{23.17^2}{\alpha\lambda_1}\right]
\end{aligned}
\tag{A.39}
$$

The standard deviation would then be:

$$
\sigma_{\alpha_n} = \sigma_z \left|\frac{\alpha}{\lambda_1}\right| \sqrt{\frac{29.88^2}{\alpha^2} + \frac{15.46^2}{\lambda_1^2} - \frac{23.17^2}{\alpha\lambda_1}}
\tag{A.40}
$$

Similarly for the worst-case λ_2, the variance and standard deviation of α_n is given by:

$$
\begin{aligned}
\sigma_{\alpha_n}^2 &\approx \left(\frac{\alpha}{\lambda_2}\right)^2 \left[\left(\frac{\sigma_\alpha}{\alpha}\right)^2 + \left(\frac{\sigma_{\lambda_2}}{\lambda_2}\right)^2 - 2\frac{\sigma_{\alpha\lambda_2}}{\alpha\lambda_2}\right] \\
&= \left(\frac{\alpha}{\lambda_2}\right)^2 \left[\left(\frac{29.88\sigma_z}{\alpha}\right)^2 + \left(\frac{69.33\sigma_z}{\lambda_2}\right)^2 - 2\frac{(38.96\sigma_z)^2}{\alpha\lambda_2}\right] \\
&= \sigma_z^2 \left(\frac{\alpha}{\lambda_2}\right)^2 \left[\frac{29.88^2}{\alpha^2} + \frac{69.33^2}{\lambda_2^2} - \frac{55.09^2}{\alpha\lambda_2}\right]
\end{aligned}
\tag{A.41}
$$

$$
\sigma_{\alpha_n} = \sigma_z \left|\frac{\alpha}{\lambda_2}\right| \sqrt{\frac{29.88^2}{\alpha^2} + \frac{69.33^2}{\lambda_2^2} - \frac{55.09^2}{\alpha\lambda_2}}
\tag{A.42}
$$

Variance and Standard Deviation of β_n

For β_n, the variance using the best-case λ_1 is given by:

$$\begin{aligned}\sigma^2_{\beta_n} &\approx \left(\frac{\beta}{\lambda_1}\right)^2 \left[\left(\frac{\sigma_\beta}{\beta}\right)^2 + \left(\frac{\sigma_{\lambda_1}}{\lambda_1}\right)^2 - 2\frac{\sigma_{\beta\lambda_1}}{\beta\lambda_1}\right] \\ &= \left(\frac{\beta}{\lambda_1}\right)^2 \left[\left(\frac{29.88\sigma_z}{\beta}\right)^2 + \left(\frac{15.46\sigma_z}{\lambda_1}\right)^2 - 2\frac{(19.38\sigma_z)^2}{\beta\lambda_1}\right] \\ &= \sigma_z^2 \left(\frac{\beta}{\lambda_1}\right)^2 \left[\frac{29.88^2}{\beta^2} + \frac{15.46^2}{\lambda_1^2} - \frac{27.41^2}{\beta\lambda_1}\right]\end{aligned} \tag{A.43}$$

The standard deviation would then be:

$$\sigma_{\beta_n} = \sigma_z \left|\frac{\beta}{\lambda_1}\right| \sqrt{\frac{29.88^2}{\beta^2} + \frac{15.46^2}{\lambda_1^2} - \frac{27.41^2}{\beta\lambda_1}} \tag{A.44}$$

Similarly for the worst-case λ_2, the variance and standard deviation of β_n is given by:

$$\begin{aligned}\sigma^2_{\beta_n} &\approx \left(\frac{\beta}{\lambda_2}\right)^2 \left[\left(\frac{\sigma_\beta}{\beta}\right)^2 + \left(\frac{\sigma_{\lambda_2}}{\lambda_2}\right)^2 - 2\frac{\sigma_{\beta\lambda_2}}{\beta\lambda_2}\right] \\ &= \left(\frac{\beta}{\lambda_2}\right)^2 \left[\left(\frac{29.88\sigma_z}{\beta}\right)^2 + \left(\frac{69.33\sigma_z}{\lambda_2}\right)^2 - 2\frac{(37.55\sigma_z)^2}{\beta\lambda_2}\right] \\ &= \sigma_z^2 \left(\frac{\beta}{\lambda_2}\right)^2 \left[\frac{29.88^2}{\beta^2} + \frac{69.33^2}{\lambda_2^2} - \frac{53.11^2}{\beta\lambda_2}\right]\end{aligned} \tag{A.45}$$

$$\sigma_{\beta_n} = \sigma_z \left|\frac{\beta}{\lambda_2}\right| \sqrt{\frac{29.88^2}{\beta^2} + \frac{69.33^2}{\lambda_2^2} - \frac{53.11^2}{\beta\lambda_2}} \tag{A.46}$$

Variance and Standard Deviation of γ_n

For γ_n, the variance using the best-case λ_1 is given by:

$$
\begin{aligned}
\sigma^2_{\gamma_n} &\approx \left(\frac{\gamma}{\lambda_1}\right)^2 \left[\left(\frac{\sigma_\gamma}{\gamma}\right)^2 + \left(\frac{\sigma_{\lambda_1}}{\lambda_1}\right)^2 - 2\frac{\sigma_{\gamma\lambda_1}}{\gamma\lambda_1}\right] \\
&= \left(\frac{\gamma}{\lambda_1}\right)^2 \left[\left(\frac{27.14\sigma_z}{\gamma}\right)^2 + \left(\frac{15.46\sigma_z}{\lambda_1}\right)^2 - 2\frac{-(20.13\sigma_z)^2}{\gamma\lambda_1}\right] \qquad \text{(A.47)} \\
&= \sigma_z^2 \left(\frac{\gamma}{\lambda_1}\right)^2 \left[\frac{27.14^2}{\gamma^2} + \frac{15.46^2}{\lambda_1^2} + \frac{28.47^2}{\gamma\lambda_1}\right]
\end{aligned}
$$

The standard deviation would then be:

$$
\sigma_{\gamma_n} = \sigma_z \left|\frac{\gamma}{\lambda_1}\right| \sqrt{\frac{27.14^2}{\gamma^2} + \frac{15.46^2}{\lambda_1^2} + \frac{28.47^2}{\gamma\lambda_1}} \qquad \text{(A.48)}
$$

Similarly for the worst-case λ_2, the variance and standard deviation of γ_n is given by:

$$
\begin{aligned}
\sigma^2_{\gamma_n} &\approx \left(\frac{\gamma}{\lambda_2}\right)^2 \left[\left(\frac{\sigma_\gamma}{\gamma}\right)^2 + \left(\frac{\sigma_{\lambda_2}}{\lambda_2}\right)^2 - 2\frac{\sigma_{\gamma\lambda_2}}{\gamma\lambda_2}\right] \\
&= \left(\frac{\gamma}{\lambda_2}\right)^2 \left[\left(\frac{27.14\sigma_z}{\gamma}\right)^2 + \left(\frac{69.33\sigma_z}{\lambda_2}\right)^2 - 2\frac{-(43.34\sigma_z)^2}{\gamma\lambda_2}\right] \qquad \text{(A.49)} \\
&= \sigma_z^2 \left(\frac{\gamma}{\lambda_2}\right)^2 \left[\frac{27.14^2}{\gamma^2} + \frac{69.33^2}{\lambda_2^2} + \frac{61.3^2}{\gamma\lambda_2}\right]
\end{aligned}
$$

$$
\sigma_{\gamma_n} = \sigma_z \left|\frac{\gamma}{\lambda_2}\right| \sqrt{\frac{27.14^2}{\gamma^2} + \frac{69.33^2}{\lambda_2^2} + \frac{61.3^2}{\gamma\lambda_2}} \qquad \text{(A.50)}
$$

Variance and Standard Deviation of κ_n

For κ_n, the variance using the best-case λ_1 is given by:

$$\begin{aligned}
\sigma^2_{\kappa_n} &\approx \left(\frac{\kappa}{\lambda_1}\right)^2 \left[\left(\frac{\sigma_\kappa}{\kappa}\right)^2 + \left(\frac{\sigma_{\lambda_1}}{\lambda_1}\right)^2 - 2\frac{\sigma_{\kappa\lambda_1}}{\kappa\lambda_1}\right] \\
&= \left(\frac{\kappa}{\lambda_1}\right)^2 \left[\left(\frac{(0.66C)\sigma_z}{\kappa}\right)^2 + \left(\frac{15.46\sigma_z}{\lambda_1}\right)^2 - 2\frac{(0.0012\sigma_z)^3}{\kappa\lambda_1}\right] \\
&= \sigma_z^2 \left(\frac{\kappa}{\lambda_1}\right)^2 \left[\frac{(0.66C)^2}{\kappa^2} + \frac{15.46^2}{\lambda_1^2} - \frac{0.024\sigma_z}{\kappa\lambda_1}\right]
\end{aligned} \tag{A.51}$$

The standard deviation would then be:

$$\sigma_{\kappa_n} = \sigma_z \left|\frac{\kappa}{\lambda_1}\right| \sqrt{\frac{(0.66C)^2}{\kappa^2} + \frac{15.46^2}{\lambda_1^2} - \frac{0.024\sigma_z}{\kappa\lambda_1}} \tag{A.52}$$

Similarly for the worst-case λ_2, the variance and standard deviation of κ_n is given by:

$$\begin{aligned}
\sigma^2_{\kappa_n} &\approx \left(\frac{\kappa}{\lambda_2}\right)^2 \left[\left(\frac{\sigma_\kappa}{\kappa}\right)^2 + \left(\frac{\sigma_{\lambda_2}}{\lambda_2}\right)^2 - 2\frac{\sigma_{\kappa\lambda_2}}{\kappa\lambda_2}\right] \\
&= \left(\frac{\kappa}{\lambda_2}\right)^2 \left[\left(\frac{(0.66C)\sigma_z}{\kappa}\right)^2 + \left(\frac{69.33\sigma_z}{\lambda_2}\right)^2 - 2\frac{-(0.0012\sigma_z)^3}{\kappa\lambda_2}\right] \\
&= \sigma_z^2 \left(\frac{\kappa}{\lambda_2}\right)^2 \left[\frac{(0.66C)^2}{\kappa^2} + \frac{69.33^2}{\lambda_2^2} + \frac{0.024\sigma_z}{\kappa\lambda_2}\right]
\end{aligned} \tag{A.53}$$

$$\sigma_{\kappa_n} = \sigma_z \left|\frac{\kappa}{\lambda_2}\right| \sqrt{\frac{(0.66C)^2}{\kappa^2} + \frac{69.33^2}{\lambda_2^2} + \frac{0.024\sigma_z}{\kappa\lambda_2}} \tag{A.54}$$

The variances and standard deviations of all the normalised plane parameters have been derived. A summary of the variances is given below.

$For\ best\ case\ of\ \sigma_{\lambda_1} = 15.456\sigma_z,$

$$\begin{aligned}
\sigma^2_{\alpha_n} &\approx \sigma^2_z \left(\frac{\alpha}{\lambda_1}\right)^2 \left[\frac{29.88^2}{\alpha^2} + \frac{15.46^2}{\lambda_1^2} - \frac{23.17^2}{\alpha\lambda_1}\right] \\
\sigma^2_{\beta_n} &\approx \sigma^2_z \left(\frac{\beta}{\lambda_1}\right)^2 \left[\frac{29.88^2}{\beta^2} + \frac{15.46^2}{\lambda_1^2} - \frac{27.41^2}{\beta\lambda_1}\right] \\
\sigma^2_{\gamma_n} &\approx \sigma^2_z \left(\frac{\gamma}{\lambda_1}\right)^2 \left[\frac{27.14^2}{\gamma^2} + \frac{15.46^2}{\lambda_1^2} + \frac{28.47^2}{\gamma\lambda_1}\right] \\
\sigma^2_{\kappa_n} &\approx \sigma^2_z \left(\frac{\kappa}{\lambda_1}\right)^2 \left[\frac{(0.66C)^2}{\kappa^2} + \frac{15.46^2}{\lambda_1^2} - \frac{0.024\sigma_z}{\kappa\lambda_1}\right]
\end{aligned} \tag{A.55a}$$

$For\ worst\ case\ of\ \sigma_{\lambda_2} = 69.329\sigma_z,$

$$\begin{aligned}
\sigma^2_{\alpha_n} &\approx \sigma^2_z \left(\frac{\alpha}{\lambda_2}\right)^2 \left[\frac{29.88^2}{\alpha^2} + \frac{69.33^2}{\lambda_2^2} - \frac{55.09^2}{\alpha\lambda_2}\right] \\
\sigma^2_{\beta_n} &\approx \sigma^2_z \left(\frac{\beta}{\lambda_2}\right)^2 \left[\frac{29.88^2}{\beta^2} + \frac{69.33^2}{\lambda_2^2} - \frac{53.11^2}{\beta\lambda_2}\right] \\
\sigma^2_{\gamma_n} &\approx \sigma^2_z \left(\frac{\gamma}{\lambda_2}\right)^2 \left[\frac{27.14^2}{\gamma^2} + \frac{69.33^2}{\lambda_2^2} + \frac{61.3^2}{\gamma\lambda_2}\right] \\
\sigma^2_{\kappa_n} &\approx \sigma^2_z \left(\frac{\kappa}{\lambda_2}\right)^2 \left[\frac{(0.66C)^2}{\kappa^2} + \frac{69.33^2}{\lambda_2^2} + \frac{0.024\sigma_z}{\kappa\lambda_2}\right]
\end{aligned} \tag{A.55b}$$

Appendix B

Video Footage

The following is a list of web links to video footage of the Hatpack Simulator and visual results of the end-to-end system of Chapter 5. These videos are also supplied on the CD-ROM disk attached to this thesis.

Hatpack Simulator Example Footage

https://www.youtube.com/watch?v=oAxaNloHVHg

Psychophysics Maze Test Example Footage

https://www.youtube.com/watch?v=2byh1qQfWGQ

Psychophysics Chessboard Test Example Footage

https://www.youtube.com/watch?v=gIVrnsk04LA

Ground Plane and Tabletop Detection Results of Hardware Plane Fitting End-to-end System

https://www.youtube.com/watch?v=CtDp8-Cqi4I

Vita

Publications arising from this thesis include:

H. Josh, B. Yong, L. Kleeman. A Real-time FPGA-based Vision System for a Bionic Eye. In *Proceedings of Australasian Conference on Robotics and Automation (ACRA)*, Melbourne, Australia, 2011.

H. Josh, B. Yong, L. Kleeman. Mobile, Real-Time Simulator for a Cortical Visual Prosthesis. In *Proceedings of the International Conference on Biomedical Electronics and Devices*, Villamoura, Portugal, 2012.

H. Josh, B. Yong, L. Kleeman. A Real-Time and Portable Bionic Eye Simulator. In *Biomedical Engineering Systems and Technologies* (pp. 51-67), 2013.

H. Josh, C. Mann, L. Kleeman, D. Lui. Psychophysics Testing of Bionic Vision Image Processing Algorithms Using an FPGA Hatpack. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Melbourne, Australia, 2013.

Submitted work:

H. Josh, L. Kleeman. A Novel Hardware Plane Fitting Implementation and Applications for Bionic Vision. **Submitted to** *Machine Vision and Applications*, 2015.

Permanent Address: Department of Electrical and Computer Systems Engineering
Monash University
Australia

This thesis was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this thesis were written by Glenn Maughan and modified by Dean Thompson and David Squire of Monash University.

References

[1] Allen, P., Yeoh, J., McCombe, M., Heriot, W., Luu, C., Ayton, L., Nayagam, D., Shepherd, R., Blamey, P. and Guymer, R. [2013]. Bionic vision australia-implantation of a suprachoroidal retinal prosthesis-results for the first participants, *Investigative Ophthalmology & Visual Science* **54**(15): 1031–1031.

[2] Altera Corporation [2014]. Altera: RAM-based shift register megafunction user guide, Webpage. (last accessed June 25, 2015).
URL: *https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug_shift_register_ram_based.pdf*

[3] Altera Corporation [2015a]. Altera Quartus II Design Software, Webpage. (last accessed June 25, 2015).
URL: *https://www.altera.com/products/design-software/fpga-design/quartus-ii/overview.html*

[4] Altera Corporation [2015b]. Altera Stratix IV FPGAs, Webpage. (last accessed June 25, 2015).
URL: *https://www.altera.com/products/fpga/stratix-series/stratix-iv/overview.html*

[5] Altera Corporation [2015c]. ModelSim-Altera Software, Webpage. (last accessed June 25, 2015).
URL: *https://www.altera.com/products/design-software/model---simulation/modelsim-altera-software.html*

[6] Argyrous, G. [2011]. *Statistics for research: with a guide to SPSS*, Sage Publications.

[7] Aulinas, J., Petillot, Y., Salvi, J. and Lladó, X. [2008]. The SLAM problem: A survey, *Frontiers in Artificial Intelligence and Applications* **184**(1): 363–371.

[8] Backus, B. T., Fleet, D. J., Parker, a. J. and Heeger, D. J. [2001]. Human cortical activity correlates with stereoscopic depth perception., *Journal of neurophysiology* **86**(4): 2054–2068.

[9] Bailey, D. G. [2011]. *Design for Embedded Image Processing on FPGAs*, John Wiley & Sons.

[10] Bailey, D. G. and Bouganis, C. S. [2008]. Reconfigurable foveated active vision system, *Proceedings of the 3rd International Conference on Sensing Technology, ICST 2008*, pp. 162–167.

[11] Bak, M., Girvin, J. P., Hambrecht, F. T., Kufta, C. V., Loeb, G. E. and Schmidt, E. M. [1990]. Visual Sensations Produced by Intracortical Microstimulation of the Human Occipital Cortex, *Medical and Biological Engineering and Computing* **28**(3): 257–259.

[12] Balasubramanian, M., Polimeni, J. and Schwartz, E. L. [2002]. The V1-V2-V3 complex: Quasiconformal dipole maps in primate striate and extra-striate cortex, *Neural Networks* **15**(10): 1157–1163.

[13] Bear, M. F., Connors, B. W. and Paradiso, M. A. [2007]. *Neuroscience*, Vol. 2, Lippincott Williams & Wilkins.

[14] Behrend, M. R., Ahuja, A. K., Humayun, M. S., Chow, R. H. and Weiland, J. D. [2011]. Resolution of the epiretinal prosthesis is not limited by electrode size, *IEEE transactions on neural systems and rehabilitation engineering* **19**(4): 436–442.

[15] Benedetti, A. and Perona, P. [1998]. Real-time 2-D feature detection on a reconfigurable computer, *1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–593.

[16] Besl, P. J. and Jain, R. C. [1986]. Invariant surface characteristics for 3d object recognition in range images, *Computer vision, graphics, and image processing* **33**(1): 33–80.

[17] Bittner, R. [2009]. Bus Mastering PCI Express In An FPGA, *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, pp. 273–276.

[18] Braille, L. [1829]. Procedure for writing words, music and plain song using dots for the use of the blind and made available to them, *Royal Institution of Blind Youth, Paris* .

[19] Brindley, G. S. and Lewin, W. S. [1968]. The Visual Sensations Produced by Electrical Stimulation of the Medial Occipital Cortex., *The Journal of Physiology* **196**(2): 479–493.

[20] Bruce, V., Green, P. R. and Georgeson, M. A. [2003]. *Visual perception: Physiology, psychology, & ecology*, Psychology Press.

[21] Butkiewicz, T. [2014]. Low-cost Coastal Mapping using Kinect v2 Time-of-Flight Cameras, *Oceans - St John's*, pp. 1–9.

[22] Button, J. and Putnam, T. [1962]. Visual responses to cortical stimulation in the blind, *J Iowa Med Soc* **52**(1): 17–21.

[23] Canny, J. [1986]. A computational approach to edge detection., *IEEE transactions on pattern analysis and machine intelligence* **8**(6): 679–698.

[24] Cha, K., Horch, K. and Normann, R. A. [1992a]. Simulation of a phosphene-based visual field: visual acuity in a pixelized vision system., *Annals of biomedical engineering* **20**(4): 439–49.

[25] Cha, K., Horch, K. W. and Normann, R. a. [1992b]. Mobility performance with a pixelized vision system., *Vision research* **32**(7): 1367–1372.

[26] Cha, K., Horch, K. W., Normann, R. a. and Boman, D. K. [1992]. Reading speed with a pixelized vision system., *Journal of the Optical Society of America. A, Optics and image science* **9**(5): 673–677.

[27] Chen, S. C., Hallum, L. E., Lovell, N. H. and Suaning, G. J. [2005a]. Learning prosthetic vision: a virtual-reality study., *IEEE transactions on neural systems and rehabilitation engineering*, Vol. 13, pp. 249–55.

[28] Chen, S. C., Hallum, L. E., Lovell, N. H. and Suaning, G. J. [2005b]. Visual acuity measurement of prosthetic vision: a virtual-reality simulation study., *Journal of neural engineering* **2**(1): S135–45.

[29] Chen, S. C., Lovell, N. H. and Suaning, G. J. [2004]. Effect on prosthetic vision visual acuity by filtering schemes, filter cut-off frequency and phosphene matrix: a virtual reality simulation., *Annual International Conference of the IEEE Engineering in Medicine and Biology Society.*, Vol. 6, pp. 4201–4204.

[30] Chen, S. C., Suaning, G. J., Morley, J. W. and Lovell, N. H. [2009]. Simulating prosthetic vision: I. Visual models of phosphenes, *Vision Research* **49**(12): 1493–1506.

[31] Cho, J., Mirzaei, S., Oberg, J. and Kastner, R. [2009]. Fpga-based face detection system using Haar classifiers, *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, pp. 103–111.

[32] Coulombe, J., Sawan, M. and Gervais, J.-F. [2007]. A highly flexible system for microstimulation of the visual cortex: Design and implementation, *Biomedical Circuits and Systems, IEEE Transactions on* **1**(4): 258–269.

[33] Dagnelie, G., Barnett, D., Humayun, M. S. and Thompson, R. W. [2006]. Paragraph text reading using a pixelized prosthetic vision simulator: Parameter dependence and task learning in free-viewing conditions, *Investigative Ophthalmology and Visual Science* **47**(3): 1241–1250.

[34] Dagnelie, G., Keane, P., Narla, V., Yang, L., Weiland, J. and Humayun, M. [2007]. Real and virtual mobility performance in simulated prosthetic vision., *Journal of Neural Engineering* **4**(1): S92–S101.

[35] Dagnelie, G., Walter, M. and Yang, L. [2006]. Playing checkers: detection and eyehand coordination in simulated prosthetic vision, *Journal of Modern Optics* **53**(9): 1325–1342.

[36] Dixon, W. J., Massey, F. J. et al. [1969]. *Introduction to statistical analysis*, Vol. 344, McGraw-Hill New York.

[37] Dobelle, W. H. and Mladejovsky, M. G. [1974]. Phosphenes produced by electrical stimulation of human occipital cortex, and their application to the development of a prosthesis for the blind., *The Journal of Physiology* **243**(2): 553–576.

[38] Dobelle, W. H., Mladejovsky, M. G., Evans, J. R., Roberts, T. S. and Girvin, J. P. [1976]. "Braille" reading by a blind volunteer by visual cortex stimulation., *Nature* **259**(5539): 111–112.

[39] Dowling, J. a., Maeder, A. and Boles, W. [2004]. Mobility enhancement and assessment for a visual prosthesis., *Medical Imaging* **5369**: 780–791.

[40] Duncan, R. O. and Boynton, G. M. [2003]. Cortical magnification within human primary visual cortex correlates with acuity thresholds, *Neuron* **38**(4): 659–671.

[41] Fechner, G. [1966]. *Elements of Psychophysics. Vol. I.*, New York.

[42] Fehervari, T., Matsuoka, M., Okuno, H. and Yagi, T. [2010]. Real-Time Simulation of Phosphene Images Evoked by Electrical Stimulation of the Visual Cortex, *Neural Information Processing. Theory and Algorithms*, pp. 171–178.

[43] Ferrndez, J., Liao, E., Bonomini, M. and Fernndez, E. [2007]. An adaptable multichannel architecture for cortical stimulation, *in* J. Mira and J. lvarez (eds), *Bio-inspired Modeling of Cognitive Tasks*, Vol. 4527 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 143–152.

[44] Fischler, M. a. and Bolles, R. C. [1981]. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM* **24**(6): 381–395.

[45] Foerster, O. [1929]. Beitrage zur pathophysiologie der sehbahn und der spehsphare, *J Psychol Neurol* **39**: 435–463.

[46] Fornos, A. P., Sommerhalder, J., Pittard, A., Safran, A. B. and Pelizzone, M. [2008]. Simulation of artificial vision: IV. Visual information required to achieve simple pointing and manipulation tasks, *Vision Research* **48**(16): 1705–1718.

[47] Fornos, A. P., Sommerhalder, J., Rappaz, B., Safran, A. B. and Pelizzone, M. [2005]. Simulation of artificial vision, III: Do the spatial or temporal characteristics of stimulus pixelization really matter?, *Investigative Ophthalmology and Visual Science* **46**(10): 3906–3912.

[48] Ghovanloo, M. and Najafi, K. [2007]. A wireless implantable multichannel microstimulating system-on-a-chip with modular architecture, *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **15**(3): 449–457.

[49] Hadjinicolaou, A. E., Leung, R. T., Garrett, D. J., Ganesan, K., Fox, K., Nayagam, D. a. X., Shivdasani, M. N., Meffin, H., Ibbotson, M. R., Prawer, S. and O'Brien, B. J. [2012]. Electrical stimulation of retinal ganglion cells with diamond and the development of an all diamond retinal prosthesis, *Biomaterials* **33**(24): 5812–5820.

[50] Hallum, L. E., Suaning, G. J., Taubman, D. S. and Lovell, N. H. [2005]. Simulated prosthetic visual fixation, saccade, and smooth pursuit., *Vision research* **45**(6): 775–88.

[51] Hayes, J. S., Yin, V. T., Piyathaisere, D., Weiland, J. D., Humayun, M. S. and Dagnelie, G. [2003]. Visually Guided Performance of Simple Tasks Using Simulated Prosthetic Vision, *Artificial Organs* **27**(11): 1016–1028.

[52] Henry, P., Krainin, M., Herbst, E., Ren, X. and Fox, D. [2012]. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments, *The International Journal of Robotics Research* **31**(5): 647–663.

[53] Holz, D., Holzer, S., Rusu, R. B. and Behnke, S. [2011]. Real-Time Plane Segmentation Using RGB-D Cameras, *RoboCup* (D): 306–317.

[54] Hornig, R., Zehnder, T. and Velikay-Parel, M. [2008]. The IMI retinal implant system, *Artificial Sight* pp. 111–128.

[55] Horton, J. C. and Hoyt, W. F. [1991]. The representation of the visual field in human striate cortex. A revision of the classic Holmes map., *Archives of ophthalmology* **109**(6): 816–24.

[56] Humayun, M. S., de Juan, E., Dagnelie, G., Greenberg, R. J., Propst, R. H. and Phillips, D. H. [1996]. Visual perception elicited by electrical stimulation of retina in blind humans, *Archives of Ophthalmology* **114**(1): 40–46.

[57] Humayun, M. S., Dorn, J. D., Da Cruz, L., Dagnelie, G., Sahel, J. A., Stanga, P. E., Cideciyan, A. V., Duncan, J. L., Eliott, D., Filley, E., Ho, A. C., Santos, A., Safran, A. B., Arditi, A., Del Priore, L. V. and Greenberg, R. J. [2012]. Interim results from the international trial of second sight's visual prosthesis, *Ophthalmology* **119**(4): 779–788.

[58] Johnston, C., Gribbon, K. and Bailey, D. [2005]. FPGA based remote object tracking for real-time control, *1st International Conference on Sensing Technology*, pp. 66–71.

[59] Kelly, S. K., Shire, D. B., Chen, J., Doyle, P., Gingerich, M. D., Drohan, W. A., Theogarajan, L. S., Cogan, S. F., Wyatt, J. L. and Rizzo, J. F. [2009]. Realization of a 15-Channel, Hermetically-Encased Wireless Subretinal Prosthesis for the Blind, *31st Annual International Conference of the IEEE EMBS*, pp. 200–203.

[60] Khoshelham, K. and Elberink, S. O. [2012]. Accuracy and resolution of kinect depth data for indoor mapping applications, *Sensors* **12**(2): 1437–1454.

[61] Kish, L. [2004]. *Statistical design for research*, Vol. 83, John Wiley & Sons.

[62] Kraft, M., Schmidt, A. and Kasinski, A. [2008]. High-speed image feature detection using fpga implementation of fast algorithm, *Proceedings of the VISAPP*, pp. 174–179.

[63] Krause, F. and Schum, H. [1931]. Neue deutsche chirurgie, *Enke, Stuttgart* .

[64] Lachat, E., Macher, H., Mittet, M.-a., Landes, T. and Grussenmeyer, P. [2015]. First Experiences With Kinect V2 Sensor for Close Range 3D Modelling, *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XL-5/W4, pp. 93–100.

[65] Lee, J., Haralick, R. and Shapiro, L. [1987]. Morphologic edge detection, *IEEE Journal on Robotics and Automation* **3**(2): 142–156.

[66] Li, J., Papachristou, C. and Shekar, R. [2005]. An FPGA-Based Computing Platform for Real-Time 3D Medical Imaging and its Application to Cone-Beam CT Reconstruction, *Journal of Imaging Science and Technology* **49**(3): 237–245.

[67] Li, W. H. [2015]. A Fast and Flexible Computer Vision System for Implanted Visual Prostheses, *Computer Vision - ECCV 2014 Workshops*, Vol. 8927, pp. 686–701.

[68] Li, W. H., Tang, T. J. J. and Lui, W. L. D. [2013]. Going Beyond Vision to Improve Bionic Vision, *2013 20th IEEE International Conference on Image Processing (ICIP)*, pp. 1555–1558.

[69] Lim, Y. K., Kleeman, L. and Drummond, T. [2013]. Algorithmic methodologies for FPGA-based vision, *Machine Vision and Applications* **24**(6): 1197–1211.

[70] Lowery, A. J. [2013]. Introducing the Monash vision group's cortical prosthesis, *2013 20th IEEE International Conference on Image Processing (ICIP)*, pp. 1536–1539.

[71] Lui, W. L. D., Browne, D., Kleeman, L., Drummond, T. and Li, W. H. [2012]. Transformative Reality: Improving bionic vision with robotic sensing, *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBS)*, pp. 304–307.

[72] Maberley, D. a. L., Hollands, H., Chuo, J., Tam, G., Konkal, J., Roesch, M., Veselinovic, A., Witzigmann, M. and Bassett, K. [2006]. The Prevalence of Low Vision and Blindness in Canada, *Eye* **20**(3): 341–346.

[73] Mathieson, K., Loudin, J., Goetz, G., Huie, P., Wang, L., Kamins, T. I., Galambos, L., Smith, R., Harris, J. S., Sher, A. et al. [2012]. Photovoltaic retinal prosthesis with high pixel density, *Nature photonics* **6**(6): 391–397.

[74] MathWorks Inc. [2015]. MATLAB Software Package, Webpage. (last accessed June 25, 2015).
URL: *http://au.mathworks.com/products/matlab/*

[75] McCarthy, C., Walker, J. G., Lieby, P., Scott, A. and Barnes, N. [2015]. Mobility and low contrast trip hazard avoidance using augmented depth, *Journal of Neural Engineering* **12**(1).

[76] Microsoft [2015]. Microsoft Kinect for Windows v2 Sensor, Webpage. (last accessed June 26, 2015).
URL: *https://www.microsoft.com/en-us/kinectforwindows/*

[77] Monash Vision Group [2015a]. Monash vision direct to brain bionic eye, Webpage. (last accessed July 12, 2015).
URL: *http://www.monash.edu.au/bioniceye/index.html*

[78] Monash Vision Group [2015b]. MVG Annual Report 2014, Webpage. (last accessed July 12, 2015).
URL: *http://www.monash.edu.au/bioniceye/assets/documents/mvg-annual-report-2014.pdf*

[79] Nguyen, V., Harati, A. and Siegwart, R. [2007]. A lightweight SLAM algorithm using Orthogonal planes for indoor mobile robotics, *IEEE International Conference on Intelligent Robots and Systems*, pp. 658–663.

[80] Niu, J., Liu, Y., Ren, Q., Zhou, Y., Zhou, Y. and Niu, S. [2008]. Vision implants: An electrical device will bring light to the blind, *Science in China, Series F: Information Sciences* **51**(1): 101–110.

[81] Normann, R. A. [2007]. Technology insight: future neuroprosthetic therapies for disorders of the nervous system., *Nature clinical practice. Neurology* **3**(8): 444–452.

[82] Ohta, J., Tokuda, T., Kagawa, K., Sugitani, S., Taniyama, M., Uehara, A., Terasawa, Y., Nakauchi, K., Fujikado, T. and Tano, Y. [2007]. Laboratory investigation of microelectronics-based stimulators for large-scale suprachoroidal transretinal stimulation (STS)., *Journal of neural engineering* **4**(1): S85–S91.

[83] Otsu, N. [1979]. A Threshold Selection Method from Gray-Level Histograms, *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-9**(1): 62–66.

[84] Penfield, W. and Jasper, H. [1954]. *Epilepsy and the functional anatomy of the human brain.*, Little, Brown & Co.

[85] Penfield, W. and Rasmussen, T. [1952]. *The cerebral cortex of man*, MacMillan Company.

[86] Pfaffenberger, C. J., Scott, J. P., Fuller, J. L., Ginsburg, B. E., Biefelt, S. W. et al. [1976]. *Guide dogs for the blind: their selection, development, and training.*, Elsevier Scientific Publishing Company.

[87] Pinto, A. M., Costa, P., Moreira, A. P., Rocha, L. F., Veiga, G. and Moreira, E. [2015]. Evaluation of Depth Sensors for Robotic Applications, *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 139–143.

[88] Polimeni, J. R., Balasubramanian, M. and Schwartz, E. L. [2006]. Multi-area visuotopic map complexes in macaque striate and extra-striate cortex., *Vision research* **46**(20): 3336–59.

[89] Poppinga, J., Vaskevicius, N., Birk, A. and Pathak, K. [2008]. Fast plane detection and polygonalization in noisy 3D range images, *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pp. 3378–3383.

[90] Reel, R. [1964]. Blind man's walking cane. US Patent 3,158,162.
URL: *https://www.google.com/patents/US3158162*

[91] Resnikoff, S., Pascolini, D., Etya'ale, D., Kocur, I., Pararajasegaram, R., Pokharel, G. P. and Mariotti, S. P. [2004]. Global data on visual impairment in the year 2002, *Bulletin of the World Health Organization* **82**(11): 844–851.

[92] Rorres, C. and Anton, H. [1979]. *Applications of Linear Algebra, 2nd Edition*, Wiley.

[93] Sakaguchi, H., Kamei, M., Fujikado, T., Yonezawa, E., Ozawa, M., Cecilia-Gonzalez, C., Ustariz-Gonzalez, O., Quiroz-Mercado, H. and Tano, Y. [2009]. Artificial vision by direct optic nerve electrode (AV-DONE) implantation in a blind patient with retinitis pigmentosa, *Journal of Artificial Organs* **12**(3): 206–209.

[94] Schira, M. M., Tyler, C. W., Spehar, B. and Breakspear, M. [2010]. Modeling magnification and anisotropy in the primate foveal confluence, *PLoS Computational Biology* **6**(1): 1–10.

[95] Schira, M. M., Wade, A. R. and Tyler, C. W. [2007]. Two-dimensional mapping of the central and parafoveal visual field to human visual cortex., *Journal of neurophysiology* **97**(6): 4284–4295.

[96] Schmidt, E. M., Bak, M. J., Hambrecht, F. T., Kufta, C. V., O'Rourke, D. K. and Vallabhanath, P. [1996]. Feasibility of a visual prosthesis for the blind based on intracortical microstimulation of the visual cortex., *Brain : a journal of neurology* **119 (Pt 2**: 507–522.

[97] Schwartz, E. L. [1977]. Spatial mapping in the primate sensory projection: Analytic structure and relevance to perception, *Biological Cybernetics* **25**(4): 181–194.

[98] Simon, G. and Berger, M.-O. [2002]. Pose estimation for planar structures, *IEEE Computer Graphics and Applications* **22**(6).

[99] Sommerhalder, J., Oueghlani, E., Bagnoud, M., Leonards, U., Safran, A. B. and Pelizzone, M. [2003]. Simulation of artificial vision: I. Eccentric reading of isolated words, and perceptual learning, *Vision Research* **43**(3): 269–283.

[100] Sommerhalder, J., Rappaz, B., De Haller, R., Fornos, A. P., Safran, A. B. and Pelizzone, M. [2004]. Simulation of artificial vision: II. Eccentric reading of full-page text and the learning of this task, *Vision Research* **44**(14): 1693–1706.

[101] Srivastava, N. R., Troyk, P. R. and Dagnelie, G. [2009]. Detection, eye-hand coordination and virtual mobility performance in simulated vision for a cortical visual prosthesis device., *Journal of neural engineering* **6**(3): 035008.

[102] Srivastava, N., Troyk, P. and Bradley, D. [2007]. Fpga based visual prosthesis device for testing visual perception on non human primates, *Electro/Information Technology, 2007 IEEE International Conference on*, IEEE, pp. 21–25.

[103] Stingl, K., Bartz-Schmidt, K. U., Besch, D., Braun, A., Bruckmann, A., Gekeler, F., Greppmaier, U., Hipp, S., Hörtdörfer, G., Kernstock, C., Koitschev, A., Kusnyerik, A., Sachs, H., Schatz, A., Stingl, K. T., Peters, T., Wilhelm, B. and Zrenner, E. [2013]. Artificial vision with wirelessly powered subretinal electronic implant alpha-IMS., *Proceedings. Biological sciences / The Royal Society* **280**(1757): 1–8.

[104] Stückler, J. and Steffens, R. [2011]. Real-Time 3D Perception and Efficient Grasp Planning for Everyday Manipulation Tasks, *European Conference on Mobile Robots (ECMR)*, number September, pp. 1–6.

[105] Suaning, G., Lavoie, P., Forrester, J., Armitage, T. and Lovell, N. [2006]. Microelectronic retinal prosthesis: Iii. a new method for fabrication of high-density hermetic feedthroughs, *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, IEEE, pp. 1638–1641.

[106] Tang, T. J. J. and Li, W. H. [2014]. An Assistive EyeWear Prototype that interactively converts 3D Object Locations into Spatial Audio, *Proceedings of the 2014 ACM International Symposium on Wearable Computers*, pp. 119–126.

[107] Tang, T. J. J., Lui, W. L. D. and Li, W. H. [2011]. A lightweight approach to 6-DOF plane-based egomotion estimation using inverse depth, *2011 Australasian Conference on Robotics and Automation (ACRA)*, pp. 1–10.

[108] Tang, T. J. J., Lui, W. L. D. and Li, W. H. [2012]. Plane-based detection of staircases using inverse depth, *Australasian Conference on Robotics and Automation (ACRA)*.

[109] Tellioglu, Z. H. [2010]. *Real Time 3D Surface Feature Extraction on FPGA*, PhD thesis, Middle East Technical University.

[110] Terasic Inc. [2013]. Terasic TR4 Development Board, Webpage. (last accessed June 25, 2015).
URL: *http://tr4.terasic.com/*

[111] Thompson, R. W., Barnett, G. D., Humayun, M. S. and Dagnelie, G. [2003]. Facial Recognition Using Simulated Prosthetic Pixelized Vision, *Investigative Ophthalmology & Visual Science* **44**(11): 5035–5042.

[112] van Rheede, J. J., Kennard, C. and Hicks, S. L. [2010]. Simulating prosthetic vision: Optimizing the information content of a limited visual display., *Journal of Vision* **10**(14): 1–14.

[113] Velázquez, R. [2010]. Wearable assistive devices for the blind, *Lecture Notes in Electrical Engineering* **75**: 331–349.

[114] Veraart, C., Raftopoulos, C., Mortimer, J. T., Delbeke, J., Pins, D., Michaux, G., Vanlierde, A., Parrini, S. and Wanet-Defalque, M. C. [1998]. Visual Sensations Produced by Optic Nerve Stimulation Using an Implanted Self-Sizing Spiral Cuff Electrode, *Brain Research* **813**(1): 181–186.

[115] Veraart, C., Wanet-Defalque, M.-C., Gérard, B., Vanlierde, A. and Delbeke, J. [2003]. Pattern recognition with the optic nerve visual prosthesis, *Artificial Organs* **27**(11): 996–1004.

[116] Wandell, B. a., Dumoulin, S. O. and Brewer, A. a. [2007]. Visual field maps in human cortex., *Neuron* **56**(2): 366–383.

[117] World Health Organisation [2014]. Visual impairment and blindness, Webpage. (last accessed June 19, 2015).
URL: *http://www.who.int/mediacentre/factsheet/fs282/en/*

[118] Xilinx Inc. [2015]. PCI Express (PCIe) - Xilinx PCIe Solution Portfolio, Webpage. (last accessed July 14, 2015).
URL: *http://www.xilinx.com/products/technology/pci-express.html*

[119] Yang, D., Peterson, G. D., Li, H. and Sun, J. [2009]. An fpga implementation for solving least square problem, *Field Programmable Custom Computing Machines, 2009. FCCM'09. 17th IEEE Symposium on*, IEEE, pp. 303–306.

[120] Zhao, Y., Lu, Y., Tian, Y., Li, L., Ren, Q. and Chai, X. [2010]. Image processing based recognition of images with a limited number of pixels using simulated prosthetic vision, *Information Sciences* **180**(16): 2915–2924.