8-1-2008

Low bandwidth eye tracker for scanning laser ophthalmoscopy

Zachary Harvey

Follow this and additional works at: <http://scholarworks.rit.edu/theses>

Recommended Citation

Harvey, Zachary, "Low bandwidth eye tracker for scanning laser ophthalmoscopy" (2008). Thesis. Rochester Institute of Technology. Accessed from

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.

Low Bandwidth Eye Tracker for Scanning Laser Ophthalmoscopy

by

Zachary G. Harvey

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science
in Computer Engineering

Supervised by

Assistant Professor Sonia Lopez Alarcon, PhD.
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
August 2011

Approved by:

Sonia Lopez Alarcon, PhD., Assistant Professor
Thesis Advisor, Department of Computer Engineering

Nathan Cahill, D.Phil., Associate Professor
Committee Member, Department of School of Mathematical Sciences

Roy Melton, PhD., Lecturer
Committee Member, Department of Department of Computer Engineering

Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Low Bandwidth Eye Tracker for Scanning Laser Ophthalmoscopy

I, Zachary G. Harvey, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Zachary G. Harvey

Date

Dedication

I would like to dedicate this work to Jennifer McLaughlin who's support has been a cornerstone for my accomplishments at RIT. I would like to thank my family for their emotional and financial help through my time at RIT. Additionally I would like to thank the many faculty and staff members of the Computer Engineering department at RIT for their amazing student support.

Acknowledgments

I am grateful for the all the help I received, financially and intellectually, from both my advisers, Sonia Lopez Alarcon and Nathan Cahill, and supervisor, Alfredo Dubra, who's guidance and vision made this work possible. I would like to thank my entire committee for taking the time to review my thesis and give feedback. Finally I would like to acknowledge the countless hours of hard work by the many people in the Dubra lab.

Abstract

Low Bandwidth Eye Tracker for Scanning Laser Ophthalmoscopy

Zachary G. Harvey

Supervising Professor: Sonia Lopez Alarcon, PhD.

Use of adaptive optics with scanning laser ophthalmoscopes (AOSLOs) has allowed for *in vivo*, noninvasive imaging of the human rod and cone photoreceptor mosaic. This modality could prove to be a valuable tool for clinicians in early diagnosis of retinal disease as well as provide invaluable insight for researchers. In order for these instruments to become practical in a clinical environment, many challenges must be overcome. Involuntary eye motion makes the use of AOSLOs particularly difficult as it increases imaging time, post-processing time, data storage requirements, and, most importantly, subject's chances of retinal damage due to light exposure.

The goal of this thesis is to develop a real time eye tracking and compensation system capable of overcoming slow eye drift. Data acquisition and synchronization software and electronics were developed for use with an AOSLO. A motion estimation technique based on normalized cross correlation NCC accelerated by CUDA enabled graphics cards was used as a basis for the tracking system. Motion prediction methods were developed and evaluated in order to increase the system bandwidth. Specifically, linear and quadratic extrapolation, discrete cosine transform extrapolation, and Kalman filtering techniques were used. These tracking methods were evaluated using simulated motion and real subjects.

Contents

Dedication	iii
Acknowledgments	iv
Abstract	v
Glossary	xv
1 Introduction	1
1.1 Adaptive Optics Overview	2
1.2 Scanning Laser Ophthalmoscopy	2
1.3 Adaptive optics in retinal imaging	5
1.4 AOSLO Image Formation	6
1.5 Thesis Objectives and Motivation	9
1.5.1 Eye Motion	9
1.5.2 Motivation	11
1.5.3 Objectives	11
1.6 Motion Estimation	13
1.7 Previous work	16
1.7.1 Duel Purkinje Imaging	16
1.7.2 Dither Scanners	17
1.7.3 Electrooculography eye tracking	17
1.7.4 Corneal Reflection	18
1.7.5 Limbus and Pupil Tracking	18

2	CUDA Overview	19
3	System Overview	28
3.1	Electronics and Hardware Overview	28
3.2	Software Overview	33
4	Tracking Methods	40
4.1	Linear and Quadratic Extrapolation	42
4.2	DCT Extrapolation	42
4.3	Kalman Filter	44
5	Experimental Results	46
5.1	Simulated Motion	46
5.2	Human Subjects	48
6	Conclusions	54
	Bibliography	57
A	The Kalman Filter	60

List of Tables

4.1	Summary of the Kalman filter algorithm. x_k is a column vector containing the current state, F is a matrix represents the system dynamics, u_{k-1} is a column vector containing the control input, G is a matrix that relates the control input to the system state, w_{k-1} represents noise in the system dynamics with covariance matrix Q_k , y_k is a column vector containing the measurements, H is a matrix relates the system state to the measurements, and v_k is a column vector representing the noise in the measurements with covariance matrix R_k	45
5.1	IQRs for the various tracking methods. The y IQR is the IQR of the displacements in the y direction, while the x IQR is the IQR of the displacements in the x direction. . . .	50
A.1	Summary of the Kalman filter algorithm.	64

List of Figures

1.1	Shows a high level diagram of an adaptive optics system. Light from the retina is sent to a wavefront sensor and a deformable mirror. image reflected from the deformable mirror will have higher contrast because the wavefront has been corrected.	3
1.2	Shows how use of a pinhole in a confocal scanning laser ophthalmoscope spatially filters reflections from different layers. The green lines represent the out of focus reflections while the red lines are the in focus reflection.	4
1.3	Example SLO image.	5
1.4	Shows a side by side comparison of a fundus image typically used in a clinical setting with an image of the same subject collected with an AOSLO. From top left to bottom right, Fundus photo, expanded area outlined in green box from fundus image, same area imaged with AOSLO, expanded area on AOSLO image. The large bright circles are cone photoreceptors while the smaller dots around them are rod photoreceptors.	7
1.5	Shows an example image collected with an AOSLO, scale bar is $20\mu\text{m}$. Larger bright dots are cones while smaller surrounding dots are rods.	8

- 1.6 Shows the path of the scanning raster on the retina (the path of the light created by the horizontal and vertical scanning mirrors). Arrows represent the direction of the scan. The blue arrow indicate that the image is being collected during this time while the red arrows indicate the inactive are of the image. Collecting during the backscan (following the red arrows) would result in more data being collected but the images would be flipped about the vertical axis with respect to one another 8
- 1.7 Shows calibration grids to determine the effect of the sinusoidal motion of the resonant scanner. The vertical grids on the left are spaced farther apart near the boundaries of the image, while the spacing in the middle is much more linear. The figure on the right is the same grid rotated 90 degrees for comparison. 9
- 1.8 Shows two images from the same image sequence. The image on the left is distorted due to a microsaccade. The orange box outlines the extent of this distortion. An undistorted image is shown on the left for reference. 10
- 1.9 Shows the result of image registration of AOSLO images. The two columns show a linear and logarithmic display of the same image. Each row has a different number of images combined. The first row is a single frame taken by an AOSLO. Notice the significant noise in the bottom left hand corner of the image. This is amplified by the logarithmic display. As more images are combined, this noise is significantly reduced and features become much more clear. 12

1.10	Shows zero padding of the images before performing the DFT in order to avoid aliasing artifacts due to periodic nature of DFT	15
1.11	Visual representation of DPI eye tracking. The orange line is the incoming light, the blue ray is the first Purkinje image, and the red line is the fourth Purkinje image. The dots represent the focal points of these images	16
2.1	Comparison of GPU and CPU hardware are allocation	20
2.2	Example kernel that adds two vectors, A and B, and store the result in vector C. threadIdx is a structure containing the x,y, and z components of the thread ID	21
2.3	Arrangement of threads across a thread grid. Each thread is represented by a kernel who has access to both the threadIdx and the blockIdx.	22
2.4	Example CUDA program	23
2.5	Performance comparison of DFTs using CuFFT, FFTW, Python Numpy, and Matlab	24
2.6	Performance comparison of matrix multiplication using CUBLAS, LAPACK++, Python Numpy, and Matlab	24
2.7	Flow diagram describing NCC calculations	26
2.8	NCC performance comparison, CUDA and Matlab	27

- 3.1 Timing diagram of the signals used to synchronize the optical scanners with the image acquisition. The vertical scanner is controlled using the signal shown at the top. $-V_{Amp}$ corresponds to the voltage required to rotate the scanner to $-\theta_{Amp}$ while V_{Amp} corresponds to the voltage required to rotate the scanner to θ_{Amp} . The image is collected between the beginning of ramp and time $TTop$. Once the scanner reaches the top of the raster it must return to the start of the next scan (time between $TTop$ and $TBottom$). During this time no images are collected. The next scan starts at the next rising edge of the h-sync. $TBottom$ determines the frame rate of the system (the frames per second is equal to $\frac{1}{TBottom}$). The rising edge of the v-sync tells the frame grabber to start acquiring a new frame. It is synchronized to the vertical scanner signal. The zero crossing circuit (or h-sync) creates a clock that is exactly in phase with the position of the horizontal scanner. Rising edges of the zero crossing circuit output tell the frame grabber to start acquiring a new line. The pixel clock is not shown but is a 40 MHz clock that is generated internally on the frame grabber. 30
- 3.2 Diagram describing how the actuators of the tip and tilt mirror are actually rotated. The figure on the left is the idealized location of the pistons controlling the tilt, while the figure on the right shows the actual configuration of the actuators. 31
- 3.3 High level system diagram 33
- 3.4 Program flow diagram for the tracking loop 35

- 3.5 Top Left: The acquisition settings panel allows the user to select which video channels are enabled and determine where the image sequences will be saved. Additionally, the user can put a threshold on the mean intensity value of the image so that only the brightest frames are saved Top Right: The optical scanners panel allows the user to manually adjust the video timing and field of view. Bottom Right: The tracking panel allows the user to adjust the setting for eye tracking. This is where the user can adjust the gain, prediction methods, NCC threshold, displacement threshold, etc.. 37
- 3.6 Video panel and FOV selector. The video panel displays a live feed of the acquisition. Above the video, a few statistics about the incoming images are displayed (min, max, and average pixel values). The FOV window allows the user to select a new FOV to use. 38
- 3.7 Image of the panel that allows the user to control the ROIs from the reference image as well as the shape and location of the incoming strips. The user selects the number of ROIs, x offsets, y offsets, and spacing between ROIs. 39
- 4.1 Shows the effects of the applying motion compensation during a raster scan. The orange box shows compensation being applied against the direction of the raster (stretching) and the green box shows correcting with the direction of the scanning (compression) 41

- 5.1 Percent error in the tracking signal for different simulated motions. The normalized peak velocity is the ratio of the peak velocity in the motion signal to the velocity of the raster scan. The color of the lines indicate the prediction method used. The line style corresponds to the number of updates made per frame, solid lines indicate once per frame (full frame), dotted lines indicate 4 updates per frame (coarse strips), and dashed lines indicate 8 updates per frame (fine strips). The frame rate of the system was 16.9 Hz. . . . 47
- 5.2 Residual displacements in image sequences collected using different tracking methods for two different subjects: No Tracking (NT), No Prediction (NP), Linear Extrapolation Full Frame (LFF), Quadratic Extrapolation Full Frame (QFF) Kalman Filter Full Frame (KFFF), Low Pass Filter Full Frame (LPFF), No Prediction Sub-Frame (NPSF), Linear Extrapolation Sub-Frame (LSF), Quadratic Extrapolation Sub-Frame (QSF), Kalman Filter Sub-Frame (KFSSF), Low Pass Filter Sub-Frame (LPSF). The box represents the inner quartile range (IQR) while bars represent the maximum and minimum of the displacements. 49
- 5.3 Average displacement per frame for image sequences using the various tracking methods. The green ellipses represent the standard deviation of the x displacements and y displacements. 52
- 5.4 Average displacement per frame for image sequences using the various tracking methods. The green ellipses represent the standard deviation of the x displacements and y displacements. 53

Glossary

A

- AO** Adaptive optics, method for improving image quality by correcting for distorted wavefronts, p. 1.
- AOSLO** Adaptive optics scanning laser ophthalmoscope, instrument for taking high resolution images of the retina, p. v.
- API** Application programming interface, collection of functions and data types in a third party library, p. 34.

C

- CCD** Charged coupled device, a light intensity sensor, p. 1.
- CSLM** Confocal scanning laser microscopy, method for imaging three dimensional volumes by spatially filtering reflections from out of focus layers with the use of a pinhole, p. 2.
- CUDA** Compute unified device architecture, architecture developed by NVidia corporation to allow for more flexible programming of the GPU, p. vii.

D

- DAQ** Data acquisition, p. 29.
- DCT** Discrete Cosine Transform, special version of the discrete Fourier transform that always results in real valued coefficients, p. vii.

F

FOV Field of view, area being imaged on the retina, p. xiii.

G

GUI Graphical user interface, p. 33.

I

IQR Inner quartile range, the difference between the third and first quartile. The range of the inner 50 of the data points, p. xiv.

L

LPF Low pass filter, p. 43.

M

MIL Matrox imaging library, API used to communicate to Matrox frame grabbers, p. 33.

N

NCC Normalized cross correlation, a measure of similarity between two matrices. In general it is a value between -1 and 1, p. v.

P

PMT Photomultiplier tube, sensitive light detector that converts light intensities to current, p. 4.

R

ROI Region of interest, an sub-area of a matrix or image, p. 34.

S

- SDK** Software development kit, tool chain distributed by vendors used to develop software for specific hardware or platform, p. 20.
- SLO** Scanning laser ophthalmoscope, method for imaging lateral areas of the retina based on the principles of confocal scanning laser microscopy, p. 2.
- SNR** Signal to noise ratio, measurement of how strong a signal is, p. 1.
- SWIG** Simplified wrapper interface generator, tool for creating software interfaces between low level programming languages and higher level programming languages, p. 33.

Chapter 1

Introduction

An adaptive optics scanning laser ophthalmoscope (AOSLO) is an instrument that takes high resolution images of the human retina. It forms an image by scanning a single beam of light over a field of view (FOV). This method provides better contrast and axial sectioning than images collected using a charged coupled device (CCD) array, however, these images are distorted by involuntary eye motion and suffer from low signal to noise ratios (SNR). In order to correct for this, many more images than required for post-processing are collected to ensure adequate feature coverage. Compensating for motion during image acquisition could reduce the number of images required for image registration, the process of aligning and combining images.

The objective of this thesis is to develop a real time eye tracking and compensation system for use with an AOSLO. The motion estimation method used was an adaptation of the registration process presented in [6]. Motion prediction methods were developed and evaluated in order to increase system bandwidth and decrease distortion induced by tracking. Linear and quadratic extrapolation, discrete cosine transform (DCT) extrapolation, and Kalman filtering techniques were implemented and tested using simulated eye motion with varying frequencies, as well as human subjects. The residual motion (i.e. displacement) is used as a metric to compare the performance of the different prediction methods, as well as their temporal response.

This chapter will explain the basic principles of adaptive optics (AO) and some background on a few different retinal imaging methods as well as the use of AO in conjunction with these methods. Eye motion will be discussed

and how it is perceived in AOSLO images. The registration process that is used as a foundation for the tracking system will be outlined. Finally, a brief perspective of the work done in the field of eye tracking and how this thesis fits into this history will be provided.

1.1 Adaptive Optics Overview

Adaptive optics is a technique for compensating for a distorted wavefront. It was originally developed for use in astronomy to correct for images blur caused by turbulence in the atmosphere. There are two main components in an adaptive optics system, a wavefront sensor and a deformable mirror.

When an wavefront passes through the lens in the eye, it becomes distorted. A wavefront sensor is a device with an array of small lenses, known as lenslets, in front of a CCD array. If the wavefront was perfectly uniform, the lenses would focus the incoming light into a uniformly spaced grid of spots. If the wavefront is distorted, these spots are displaced. Using these displacements, the wavefront can be reconstructed.

A deformable mirror is a device with a thin reflective film that is covering an array small actuators. Each actuator can push or pull to reshaped the thin film. Once the wavefront has been measured, that information can be fed to the deformable mirror to compensate for aberrations caused by the eye. When the distorted wavefront is reflected off of the deformable mirror, it returns corrected allowing for greater image contrast. A high level diagram of an adaptive optics system is shown in figure 1.1.

1.2 Scanning Laser Ophthalmoscopy

Scanning laser ophthalmoscopy (SLO) is based on the principles of confocal scanning laser microscopy (CSLM) [21]. CSLM allows for very fine axial sectioning and three dimensional reconstructions [3]. Light is reflected off a dichroic mirror and through an objective lens that focuses the spot on a single layer of a sample. Light is reflected from other layers as well, which if unfiltered, would decrease the contrast in the image. Light from the out

Figure 1.1: Shows a high level diagram of an adaptive optics system. Light from the retina is sent to a wavefront sensor and a deformable mirror. image reflected from the deformable mirror will have higher contrast because the wavefront has been corrected.

Figure 1.2: Shows how use of a pinhole in a confocal scanning laser ophthalmoscope spatially filters reflections from different layers. The green lines represent the out of focus reflections while the red lines are the in focus reflection.

of focus layers of the sample have a slightly shifted focal point than the in focus layer. A pinhole is placed directly in front of a light detector, at the focal point of the in focus reflected light. All out of focus light will be blocked by the pinhole and will not reach the detector. The detector is normally a photomultiplier tube (PMT), which converts the light intensity into a small current that can later be passed through a transimpedance amplifier and converted to voltage. This method is depicted in figure 1.2. With this setup, only a single spot can be imaged. Oscillating mirrors, such as scanning galvanometers or servo controlled mirrors, can be inserted before the light reaches the objective lens in order to move the spot across the sample. The motion is synchronized with acquisition electronics which are being used to digitize the voltage coming from the transimpedance amplifier. Each pixel in the image corresponds to the intensity of the reflected light from one scanning spot on the sample. Slower scanning speeds result in lower noise but longer imaging duration. CSLM is typically used for three dimensional reconstruction of tissue samples.

A fundus camera is a typical instrument found in clinical environments that provides a low resolution wide FOV. Fundus image resolution is limited by reflections from the many layers in the retina. An SLO corrects

Figure 1.3: Example SLO image.

this problem by using CSLM, where the objective lens is the lens in the eye. This allows for much better axial sectioning and contrast. Light safety standards limit the amount of light that can be shined into the eye. Additionally, the reflectivity of the retina is very low which results in very low SNR. This imaging method is limited by optical aberrations caused by the lens. An example image taken with an SLO is shown in figure 1.3.

1.3 Adaptive optics in retinal imaging

Adaptive optics was first used in retinal imaging in conjunction with a fundus camera by Liang *et al.*[12]. They were able to observe microscopic cellular features such as individual cone photoreceptors. This was taken one step further by Roorda *et al.*[17] who created the first AOSLO. AOSLOs have been improved by Dubra *et al.*[7] who are able to achieve higher resolution by making the optics non-planar. Using this instrument the entire rod and cone mosaic can be resolved up to 15 degrees retinal eccentricity. Adaptive optics has been incorporated in OCT systems as well [13, 10]. Zawadzki *et al.*[25] were able to achieve such high resolution that three dimensional volumes could be reconstructed.

Adaptive optics in retinal imaging shows tremendous potential both as a clinical instrument and a research tool. Being able to repeatedly image the

entire rod and cone photoreceptor, the cells responsible for vision, mosaic could be a valuable feature in early diagnosis of diseases, many of which cause irreversible damage to the retina. Additionally, being able to watch the progression of diseases could allow for better studies of how certain drugs affect the retina. Figure 1.4 shows the scale of improvement achievable through adaptive optics. Here, a subject has been imaged with both a traditional CCD array fundus camera as well as an AOSLO. The AOSLO image has been manually registered to the fundus image. A full AOSLO image is shown in figure 1.5.

1.4 AOSLO Image Formation

A scanning ophthalmoscope forms an image of the retina by scanning a single spot over the field of view in a raster scan like motion [17]. The raster is composed of two synchronized scanning mirrors. The horizontally scanning mirror is a free running open loop resonant scanner oscillating at a frequency of approximately 10 to 18 KHz. The vertical component of the raster is created by using a slower scanning galvanometer which is synchronized to the free running horizontal scanner. The motions are combined to form the raster as depicted in figure 1.6.

Figure 1.6 shows the idealized motion of the resonant scanner. In reality, its motion is sinusoidal. As the scanner moves toward the apex of its oscillation, the speed decreases. This causes the samples near the horizontal boundaries of the image to be much closer together, resulting in warping near the edges. This effect is minimized by only collecting pixels during the near linear region of the sinusoid but still requires post-processing to remove this distortion.

In order to correct for this, a one time calibration must be performed. Gratings, or dark vertical and horizontal lines, are imaged for each field of view in order to determine the parameters of the sinusoid. Although, only vertical gratings are required in order to extract the sinusoidal information, horizontal gratings are also taken so that the transformation can be chosen such that the pixels are square. An example calibration grid is shown in figure 1.7. The vertical gratings image is averaged down the columns to

Figure 1.4: Shows a side by side comparison of a fundus image typically used in a clinical setting with an image of the same subject collected with an AOSLO. From top left to bottom right, Fundus photo, expanded area outlined in green box from fundus image, same area imaged with AOSLO, expanded area on AOSLO image. The large bright circles are cone photoreceptors while the smaller dots around them are rod photoreceptors.

Figure 1.5: Shows an example image collected with an AOSLO, scale bar is $20\mu\text{m}$. Larger bright dots are cones while smaller surrounding dots are rods.

Figure 1.6: Shows the path of the scanning raster on the retina (the path of the light created by the horizontal and vertical scanning mirrors). Arrows represent the direction of the scan. The blue arrow indicate that the image is being collected during this time while the red arrows indicate the inactive are of the image. Collecting during the backscan (following the red arrows) would result in more data being collected but the images would be flipped about the vertical axis with respect to one another

Figure 1.7: Shows calibration grids to determine the effect of the sinusoidal motion of the resonant scanner. The vertical grids on the left are spaced farther apart near the boundaries of the image, while the spacing in the middle is much more linear. The figure on the right is the same grid rotated 90 degrees for comparison.

produce a single vector. The locations of local minimum are used as the center points for each vertical lines and are used to fit the parameters of a sinusoid 1.1 in a least squared error sense.

$$X = A \sin\left[\frac{2\pi x}{N} + \phi\right] + \beta \quad (1.1)$$

Here, X is the position in the warped image while x is the true position of that pixel. The inverse of equation 1.1 is performed to determine the "dewarped" pixel locations. These locations do not fall on integer induces so interpolation is required and special care must be taken to account for the non-uniform spatial sampling.

1.5 Thesis Objectives and Motivation

1.5.1 Eye Motion

Eye motion varies greatly from subject to subject. Factors such as age and the presence of retinal disease play a large role in the a subject's ability to stabilize eye motion. For the purposes of this work, the main class of eye motions of interest is fixational eye motion. During an imaging session with AOSLOs, a subject is told fixate at a point on a grid that corresponds

Figure 1.8: Shows two images from the same image sequence. The image on the left is distorted due to a microsaccade. The orange box outlines the extent of this distortion. An undistorted image is shown on the right for reference.

to a specific location of interest on the retina. Despite the fact that subjects are fixating as best they possibly can, some types of motion still exist. This includes tremors, drift, and microsaccades. It has been argued that these involuntary movements of the eye play a critical role in the vision system. Tremors are fast jerky random movement with amplitudes of 10-20 arcseconds at frequencies of 200 Hz. When looking at the power spectra for tremors, it has been found that a majority of the energy is between 0-40 Hz while a secondary peak at 40-100 Hz also exists. Drift is continuous motion that can vary from 5-20 arcminutes/s in healthy retinas to 200 arcmins/s in diseased retinas. Microsaccades are very fast jerks that compensate for drift with amplitudes between 1 and 20 arcmin at a rate of 10 deg/s [22, 5, 16].

Microsaccades are very fast compared to the scanning speed of the raster which results in serious stretching and compression of the images (see figure 1.8). These motions are difficult to track using image processing techniques. Tremors, although have very high frequencies and result in video sequences that seem very jerky, do not cause problems in the post processing due to their small amplitudes. Eye drift moves the features of interest outside of the FOV but have very low frequencies. As a result, this type of motion is very easy to track using image processing techniques. The aim of this project is to overcome slow eye drift using passive, image processing techniques and prediction methods.

1.5.2 Motivation

The low SNR typical of AOSLO images dictates that many images at each location be collected and averaged together. Eye motion makes this process difficult for two reasons. Firstly, motion that is very fast compared to the scanning speed of the raster distorts the images. This makes offline image registration difficult and time consuming. Secondly, large motion amplitudes compared to the FOV size require that many images must be taken at each fixation location in order to ensure adequate feature coverage and good registration result. This is a problem because it increases post processing times and storage requirements. Additionally, it increases the duration of the imaging session which adds discomfort to the subject and increases the chances of retinal damage due to light exposure. Specifically, 400% more images are required at each location than needed to improve the SNR to an acceptable level (if 50 images are to be combined, then 200 images must be collected to ensure the feature coverage). This is illustrated in figure 1.9.

Motion compensation would be valuable in terms of making this imaging modality much more practical in a clinical environment which, in turn, would make this technology more approachable for biologists studying eye disease. By compensating for eye motion in real time, the features of interest can be stabilized and significantly less images would be required to ensure a successful registration result.

1.5.3 Objectives

The objective of this thesis is to develop the software and electronics for a real time motion tracking and compensation system for use with AOSLOs. Ideally this will be accomplished with minimal additional electronics and optics. Image processing techniques will be used to extract eye rotation from the incoming video stream. A tip and tilt mirror will be used to move the raster scanning to a different location based on this information. Prediction methods will be used in order to increase system bandwidth.

Figure 1.9: Shows the result of image registration of AOSLO images. The two columns show a linear and logarithmic display of the same image. Each row has a different number of images combined. The first row is a single frame taken by an AOSLO. Notice the significant noise in the bottom left hand corner of the image. This is amplified by the logarithmic display. As more images are combined, this noise is significantly reduced and features become much more clear.

1.6 Motion Estimation

Involuntary eye motion distorts images collected by AOSLOs. Typically, offline registration of the image sequences is used to overcome the distortion caused by involuntary eye motion and low SNR. For the purposes of this work, the registration process outlined [6] is used as the basis for the tracking system and will be discussed below.

Given the high level of magnification provided by adaptive optics, small angles of rotation of the eye are approximated as lateral translations. Therefore the pixel mapping between two AOSLO images can be described as,

$$\begin{aligned} X &= x + \epsilon_x(x, y) \\ Y &= y + \epsilon_y(x, y) \end{aligned} \quad (1.2)$$

where (X, Y) is the location of the pixel on the reference frame, (x, y) is the location of the same pixel on some subsequent, or floating, frame, and (ϵ_x, ϵ_y) represent the approximated translation that occurred between the reference frame and the floating frame. Horizontal scanning frequencies are sufficiently fast to assume that the eye is stationary during a single line scan. Therefore the pixel mapping can be approximated as a function of the vertical location of the pixel.

$$\begin{aligned} X &= x + \epsilon_x(y) \\ Y &= y + \epsilon_y(y) \end{aligned} \quad (1.3)$$

Images are divided into strips and compared to a manually selected reference frame containing minimal distortion and significant overlap with the other images in the sequence. The normalized cross correlation (NCC) is used as a similarity metric as defined below,

$$C_{R,S}(m, n) = \frac{\sum_{i,j} R(i, j) S(m + i, n + j)}{\sqrt{\sum_{i,j} R(i, j)^2 \sum_{p,q} S(p, q)^2}} \quad (1.4)$$

where R represents the reference image, S represents the current strip, and m and n are the translations in the x and y direction. The summations

are performed only over the resulting overlapping region after translating the strip (m, n) pixels from the center of the reference image. The value $C_{R,S}(m, n)$ is measure of match between the reference frame and the current strip that has been translated by a factor (m, n) . In general this value is in the interval $[-1, 1]$, however, because images have only positive intensity values, the result of the correlation is in the interval $[0, 1]$ where a value of 1 means a perfect match and 0 represents no correlation. Calculating the correlation with all possible values for m and n would be an $O(n^2)$ operation. Fourier domain techniques are used to reduce the computational complexity to an $O(N \log N)$ operation. Using this, the solution space for the transformation can be computed using the formula below.

$$C_{R,S} = \frac{\mathcal{F}^{-1}(\mathcal{F}(R)^* \mathcal{F}(S))}{\sqrt{\mathcal{F}^{-1}(\mathcal{F}(P_R)^* \mathcal{F}(S^2)) \mathcal{F}^{-1}(\mathcal{F}(R^2)^* \mathcal{F}(P_S))}} \quad (1.5)$$

Here the $\mathcal{F}$ is the discrete Fourier transform and $\mathcal{F}^{-1}$ is the inverse discrete Fourier transform. P_R represents the unit step function over the support of the reference image and P_S is the unit step function over the support of the current strip. The $*$ symbol represent complex conjugate. The images and step functions are zero padded in order to avoid periodic aliasing caused by the discrete Fourier transform as depicted in figure 1.10. The NCC was chosen for is computational complexity as well as it's robustness to fluctuations in brightness and contrast. This robustness can be shown by thinking of the images R and S as $N * M$ dimensional vectors and rewriting 1.4 as the dot product of the current strip and reference frame divided by the norm of the two vectors as shown in the equation below.

$$C_{R,S}(m, n) = \frac{\langle R, S_{m,n} \rangle}{\|R\| \|S_{m,n}\|} = \cos(\theta) \quad (1.6)$$

Where θ is the angle between the two vectors R and $S_{m,n}$. The angle between the two vectors does not depend on their norm which corresponds to the brightness in the image. This allows for fluctuations in mean intensity between images within a sequence due to tear film dynamics.

Figure 1.10: Shows zero padding of the images before performing the DFT in order to avoid aliasing artifacts due to periodic nature of DFT

C_R, S is a matrix whose element at location (i, j) represents the NCC between the reference image and current strip translated by a factor of $i - \frac{M_R}{2} - \frac{M_S}{2}$ in the y direction and $j - \frac{N_R}{2} - \frac{N_S}{2}$ in the x direction. The translation is determined by finding the maximum across a ROI of the NCC matrix such that a minimum amount of overlap between the current strip and the reference frame is guaranteed. If the maximum is found at a location (i_{max}, j_{max}) then the displacements, $\epsilon_x(y)$ and $\epsilon_y(y)$ become,

$$\begin{aligned}\epsilon_x(y) &= j_{max} - \frac{N_R}{2} - \frac{N_S}{2} \\ \epsilon_y(y) &= i_{max} - \frac{M_R}{2} - \frac{M_S}{2}\end{aligned}\tag{1.7}$$

Image registration software uses this method to align images collected from AOSLOs and then average them together in order to increase SNR. The tracking system developed in this work will use the same process during acquisition and feed the results to the optical scanners.

Figure 1.11: Visual representation of DPI eye tracking. The orange line is the incoming light, the blue ray is the first Purkinje image, and the red line is the fourth Purkinje image. The dots represent the focal points of these images

1.7 Previous work

1.7.1 Duel Purkinje Imaging

In 1973 T.N. Cornsweet and H.D. Crane developed the first non-invasive method for tracking eye rotations within an acceptable accuracy [4]. This was accomplished through the use of Purkinje reflections. Shining light into the eye results in four reflections, the first off of the surface of the cornea, the second off the rear surface of the cornea, the third off the front surface of the lens, and the fourth off the rear surface of the lens. These reflected images are known as the first through the fourth Purkinje images respectively. Eye rotations result in separations of the first and fourth reflections as the focal points for the rear of the lens and the front of the cornea shift vertically with respect to one another.

This method is robust lateral head translations as this causes the reflections to move together. Only rotations will change the separations between the images. Although this method has been improved significantly, it still requires a sophisticated imaging system. An additional light source and tip and tilt mirror is required to ensure the Purkinje images stay cantered on a photo diode. Another issue associated with duel Purkinje imaging is

that eye rotations measured at the cornea do not necessarily correspond to motion observed on the retina by an AOSLO.

1.7.2 Dither Scanners

Other, more accurate eye tracking systems have been proposed for scanning laser ophthalmoscopy. Specifically, Daniel Hammer *et. al.* developed an eye tracking system for use with AOSLOs [9, 8]. Their approach used dither scanners to trace a circle around a user controlled feature at a frequency of 8 KHz. The reflection of the scan is recorded. The phase shift between two consecutive scans indicated the amount of eye motion. This signal is fed into a lock-in amplifier that outputs a voltage that is proportional to the phase shift. This voltage is sent to additional scanners that adjust the raster to compensate for motion. Although this method for tracking achieves excellent bandwidth (1Khz), it also increases overall system complexity. In addition to the two scanners need to create the circular beam, two scanners are needed to adjust the raster. Another concern with this method is that it increase the subject's exposure to light, as an additional light source is needed to produce the tracking signal. Additional electronics are also needed such as the lock in amplifier and the control system used to steer the raster scan.

Similar registration techniques, such as the one proposed in [20], have been performed on-line during acquisition [23]. The goal of this work, however, was to extract the motion in order to apply a stimulus at a very precise location, not to adjust the raster scan.

1.7.3 Electroocularography eye tracking

Recently, there has been an increase in interest in measuring electroocularography (EOG) signals as method for tracking eye motion [2] in computer human interaction systems. This technique use electrodes that are placed around the eye to measure the voltage potential between the top of the eye and the forehead for vertical rotations and the bottom of the eye and the forehead for horizontal rotations. The voltage difference is attributed to the

higher metabolic rate at the retina. In practice this method is very difficult and inaccurate. Although there is very high temporal resolution (only limited by the speed of the digitization) the average voltage changes $4 \mu\text{V}$ per degree making it very susceptible to noise. Additionally, the use of electrodes can be very uncomfortable for the subject. Due to the high magnification of AOSLO images, this differential is not sensitive enough [24].

1.7.4 Corneal Reflection

This technique uses a head mounted infrared illumination source that is shined on the cornea. A camera is attached to the same mount facing the eye. Because the curvature of the cornea differs from that of the eye, as the eye rotates, the distance between the reflection of the IR source and the center of the pupil changes. Digital image processing techniques are used to extract the position of the corneal reflection and pupil. This method is noninvasive and is relatively inexpensive, however, the spatial and temporal accuracy is limited by the camera and would not provide enough resolution to work for an AOSLO [24].

1.7.5 Limbus and Pupil Tracking

This method is very similar to the corneal reflection in that it uses a head mounted camera. Instead of finding the distance between the pupil and a reflected spot, it relies on the shape of the limbus as viewed by the camera to determine the rotation of the eye. This method uses the RANSAC algorithm to fit ellipses of different sizes to the limbus. The accuracy with this method is limited for the same reasons as the corneal reflection methods. [18]

Chapter 2

CUDA Overview

In order to achieve real time performance, general purpose graphics processing is used. Specifically NVidia's compute unified device architecture (CUDA) was targeted. A brief overview of the architecture and a performance evaluation that relates to the processing used in this project is described here. GPUs have the potential to increase performance one to two orders of magnitude depending on the algorithm. Additionally, flexible programming environments and extensive library support make this emerging technology very approachable.

As the demands for graphics processing have increased, GPU vendors have been forced to make a more flexible graphics pipeline. NVidia, took this flexibility to the next level by allowing the pipeline to be used for general purpose computation. GPUs have evolved into massively parallel architectures capable of many mega-flops (millions of floating point operations per second). CUDA is NVidia's attempt to reach a broader market of people who have high demands for numerical computing [15]. This includes image processing, video encoding/decoding, scientific computing, particle simulations and more. What give GPUs an advantage over CPUs is the amount of area dedicated to floating point hardware on chip. Consider the diagram shown in figure 2.1. We see that the amount of area for cache, DRAM, and control logic is much greater for a CPU. This allows for a more flexible programming environment that can handle a large variety of tasks. GPUs, on the other hand, tend to share resources such as cache and control logic across many arithmetic logic units (ALUs). This allows for more space for number crunching at the cost of flexibility. In fact, CUDA works most efficiently when all ALUs are performing the same operation on different elements

Figure 2.1: Comparison of GPU and CPU hardware are allocation

in memory. Although this may seem impractical, many algorithms can be phrased in this way. For instance, fast Fourier transforms, convolution, and edge detection algorithms all perform very well on this hardware.

The CUDA SDK provides two methods for programming with GPUs: the driver SDK, and runtime API. The driver SDK is a lower level library that requires CUDA code to be separated in a different file from the host code, which is subsequently loaded and compiled dynamically at run time. Although this method is more in-line with the OpenCL standard developed by the Kronos group, it is incompatible with NVidia's numerical libraries which were developed using the runtime API.

The runtime API is a set of functions and extended C-like syntax used to communicate with the GPU. Work is divided into small tasks, known as kernels. Each kernel is it's own thread given a unique thread ID that directly corresponds to a location within a vector, matrix, or volume. This gives the kernel a logical way to access data within arrays. Kernels are denoted using a special global keyword that is placed before the function. An example kernel is shown in figure 2.2. Threads are arranged in blocks, each block is has a user defined x and y dimension. Blocks are arranged into a grid, which can has an x, y, and z dimension. A diagram showing the arrangement of threads across a grid is shown in figure 2.3. Each thread is allowed a small amount of local memory. Threads within a single block have a pool of shared local

```

// Kernel definition
__global__ void VecAdd(float* A, float* B, float* C)
{
    int i = threadIdx.x;
    C[i] = A[i] + B[i];
}

int main()
{
    ...
    // Kernel invocation with N threads
    VecAdd<<<1, N>>>(A, B, C);
}

```

Figure 2.2: Example kernel that adds two vectors, A and B, and store the result in vector C. threadIdx is a structure containing the x,y, and z components of the thread ID

memory whose access time is comparable to cache but is user controlled. In addition, there are two globally accessible memories, global memory and texture memory. Texture memory is specialized memory that is optimized for interpolation which is commonly used in graphics processing. Global memory is the largest pool, however is not cached and has considerable latencies to access. Special care should be taken by the programmer to ensure that memory accesses to global memory are coalesced. Data copied from the host RAM to the global GPU memory are done explicitly. An example program using the CUDA runtime API is shown in figure 2.4.

NVidia provides libraries to perform many common numerical computing tasks. Specifically the CUBLAS and the CuFFT libraries are the most useful for this work. CUDA was compared to other numerical computing libraries in terms of performance. Matrix multiplication is used during the dewarping of the images. CUBLAS matrix multiplication is compared to Matlab, LAPACK, and Python numpy. The results of this comparison are shown in figure 2.6. The CuFFT library is used to calculate the NCC. This library is compared to FFTW, Matlab and Python. These results are summarized in figure 2.5.

For the purposes of this work, the most important calculation performed is

Figure 2.3: Arrangement of threads across a thread grid. Each thread is represented by a kernel who has access to both the threadIdx and the blockIdx.

```

// Host code
int main()
{
    int N = ...;
    size_t size = N * sizeof(float);

    // Allocate input vectors h_A and h_B in host memory
    float* h_A = (float*)malloc(size);
    float* h_B = (float*)malloc(size);

    // Initialize input vectors
    ...

    // Allocate vectors in device memory
    float* d_A;
    cudaMalloc((void**)&d_A, size);
    float* d_B;
    cudaMalloc((void**)&d_B, size);
    float* d_C;
    cudaMalloc((void**)&d_C, size);

    // Copy vectors from host memory to device memory
    cudaMemcpy(d_A, h_A, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_B, h_B, size, cudaMemcpyHostToDevice);

    // Invoke kernel
    int threadsPerBlock = 256;
    int blocksPerGrid =
        (N + threadsPerBlock - 1) / threadsPerBlock;
    VecAdd<<<blocksPerGrid, threadsPerBlock>>>(d_A, d_B, d_C, N);

    // Copy result from device memory to host memory
    // h_C contains the result in host memory
    cudaMemcpy(h_C, d_C, size, cudaMemcpyDeviceToHost);

    // Free device memory
    cudaFree(d_A);
    cudaFree(d_B);
    cudaFree(d_C);

    // Free host memory
    ...
}

```

Figure 2.4: Example CUDA program

Figure 2.5: Performance comparison of DFTs using CuFFT, FFTW, Python Numpy, and Matlab

Figure 2.6: Performance comparison of matrix multiplication using CUBLAS, LAPACK++, Python Numpy, and Matlab

the NCC in Fourier domain as described in equation 1.5. It is important to note that a majority of the equation can be precomputed because the reference frame does not change very often. Specifically the DFT of the reference frame and the normalization factor relating to the reference frame only needs to be calculated once. Additionally the DFT of the unit step functions, P_S and P_R , only needs to be computed once provided the dimensions of the images do not change. After looking at the performance of the CuFFT library, it was found that FFTs on matrices with dimensions that were powers of two were significantly more efficient than non power of two dimensions. There are two reasons for this effect. From an algorithm standpoint, the FFT performs better with powers of two because the data set can be broken up into $\log_2 N$ stages while with non powers of two, extra stages are required to ensure the correct result. The second reason is that the GPU resources are most effectively utilized with powers of two dimensions. Specifically the GPU can take advantage of shared memory which has much better access times than global memory [14]. Because of this, all images are zero padded to the nearest power of two. A flow diagram of the memory buffers used and the operations performed on them is shown in figure 2.7.

The performance of the NCC implementation was compared to a MATLAB implementation. The raw computation times for various image sizes is shown in figure 2.8. As can be seen from the graph, the NCC calculation has a staircase like pattern. This is caused by the zero padding to the next power of two. As a result, many image dimensions are padded to the same size. Analysis was performed using a GTX 285 with 512 MB of DRAM. This particular architecture has no double precision support, instead double precision calculations are done using multiple single precision calculations. This explains the discrepancy in performance between the single and double precision NCC computation times. Two orders of magnitude performance increase was achieved using a CUDA enabled graphics card.

Figure 2.7: Flow diagram describing NCC calculations

Figure 2.8: NCC performance comparison, CUDA and Matlab

Chapter 3

System Overview

3.1 Electronics and Hardware Overview

An AOSLO described in [7] was used for as the basis for the eye tracking system that is the object of this study. The only modification made was the replacement of the scanning galvanometer (which can only be adjusted in one dimension) used for the vertical component of the raster scan with a tip-tilt piezo-ceramic driven mirror provided by Physik Instrumente (Karlsruhe, Germany) so that tracking could be performed in both the horizontal and vertical directions.

A frame grabber is a PCI card that is used to collect video data. The video data can be either digital or analog. In the case of the system described here, the photo detector converts the reflected light into an analog voltage. The digitilization of the voltage must be synchronized with the optical scanners. In order to accomplish this, three digital clocks are used: the horizontal-sync (h-sync), the vertical-sync (v-sync), and the pixel clock. The rising edge of the v-sync indicates that the start of a new raster scan, the rising edge of the h-sync indicates the start of a new horizontal scan from the resonant scanner, and the rising edge of the pixel clock indicates the a new pixel should be acquired. These clocks can be fed into or generated by the frame grabber.

The frequency of the resonant scanner (oscillating in the horizontal direction) can not be altered, only the amplitude of the scan can be controlled. The scanner outputs a voltage that is proportional to its current position. A zero crossing circuit is used to generate a clock that is in phase with the motion of the scanner. This clock is used as the h-sync and is fed into the

frame grabber.

The angle of the vertical scanner is proportional to an input voltage that is centered at zero volts. If no tracking is performed, then the angle should vary linearly between $\pm\theta Amp$. Spatial sampling density and frame rate are determined by the time between the bottom of the raster scan ($-\theta Amp$) and the top of the raster scan ($+\theta Amp$). Faster scanning means lower spatial sampling but higher temporal sampling. Different parameters are chosen based on the field of view (FOV) that is desired and the purpose of the imaging. For a majority of the time the vertical scanning is independent of the horizontal scanning. The only synchronization between the horizontal and vertical scanning is that as soon as a raster scan finishes, the start of the next begins on the rising edge of the h-sync clock.

A National Instruments (NI) data acquisition (DAQ) card (PCIe 7841R) is used to generate the control input for the vertical scanning. Another purpose of the DAQ card is to output the v-sync clock. This clock toggles logic state when the vertical scanner reaches $\pm\theta Amp$. These signals are generated using on board Xilinx Virtex V FPGA. A Matrox Helios frame grabber is used for image acquisition and generation of the 40 MHz pixel clock which is phase locked to the h-sync. A timing diagram showing how the v-sync and h-sync relate to the vertical scanning input voltage is shown in figure 3.1.

Eye motion compensation in the vertical direction is added to the ramp. If the slope of the ramp is α with offset β and the tracking compensation is Δy , then the output of the DAQ card is $v = \alpha t + \beta + \Delta y$. This combination is done on the Xilinx FPGA. Compensation in the horizontal direction, Δx is directly output to the tip-tilt mirror.

The timing diagram in figure 3.1 shows the idealized voltage that should be sent to the vertical scanner if the actuator controlling the vertical tilt of the mirror were directly above the point of rotation. In reality, the axis of the actuators for the tip and tilt mirror for this project are rotated by 45 degrees. A diagram describing this issue is shown in figure 3.2

In order to account for this, the control signals must be rotated by 45 degrees.

Figure 3.1: Timing diagram of the signals used to synchronize the optical scanners with the image acquisition. The vertical scanner is controlled using the signal shown at the top. $-V_{Amp}$ corresponds to the voltage required to rotate the scanner to $-\theta_{Amp}$ while V_{Amp} corresponds to the voltage required to rotate the scanner to θ_{Amp} . The image is collected between the beginning of ramp and time T_{Top} . Once the scanner reaches the top of the raster it must return to the start of the next scan (time between T_{Top} and T_{Bottom}). During this time no images are collected. The next scan starts at the next rising edge of the h-sync. T_{Bottom} determines the frame rate of the system (the frames per second is equal to $\frac{1}{T_{Bottom}}$). The rising edge of the v-sync tells the frame grabber to start acquiring a new frame. It is synchronized to the vertical scanner signal. The zero crossing circuit (or h-sync) creates a clock that is exactly in phase with the position of the horizontal scanner. Rising edges of the zero crossing circuit output tell the frame grabber to start acquiring a new line. The pixel clock is not shown but is a 40 MHz clock that is generated internally on the frame grabber.

Figure 3.2: Diagram describing how the actuators of the tip and tilt mirror are actually rotated. The figure on the left is the idealized location of the pistons controlling the tilt, while the figure on the right shows the actual configuration of the actuators.

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} \Delta x \\ \alpha t - \beta + \Delta y \end{pmatrix} \quad (3.1)$$

Here, Δx is the horizontal tracking signal, Δy is the vertical tracking signal, α is the slope of the ramp that would be sent to the scanner had there been no rotation, β is the offset, t is time, and θ is rotation of the actuators. Using $\theta = 45$ the x and y signals become,

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{\Delta x}{\sqrt{2}} - \frac{\alpha t - \beta + \Delta y}{\sqrt{2}} \\ \frac{\Delta x}{\sqrt{2}} + \frac{\alpha t - \beta + \Delta y}{\sqrt{2}} \end{pmatrix}. \quad (3.2)$$

Another consideration that must be taken into account is the rate at which the tracking signal is changed. Abrupt changes could damage the scanner and should be avoided. In light of this, the tracking signals Δx and Δy are actually changed at a rate that is proportional to the speed of the vertical scanning. In other words, when a new offset is calculated, the signals Δx and Δy change at a rate $\pm p\alpha t$, where p is a value determined at runtime by a configuration file and the sign is determined by the direction of change, until the new offset has been reached.

Ensuring the tracking signal changed slowly and the math regarding the rotation of the actuators was done using a Xilinx Virtex V FPGA which is on the NI DAQ card. This allowed for a simpler software design and also minimized the communication over the PCI bus which was being used for image data and GPU calculations. The FPGA was programed using LabView FPGA module and communication between the software and the FPGA was done via the NI FPGA C API.

As discussed in the motion estimation section of this thesis the main equation used for determining the translation of the motion is,

$$C_{R,S} = \frac{\mathcal{F}^{-1}(\mathcal{F}(R)^* \mathcal{F}(S))}{\sqrt{\mathcal{F}^{-1}(\mathcal{F}(P_R)^* \mathcal{F}(S^2)) \mathcal{F}^{-1}(\mathcal{F}(R^2)^* \mathcal{F}(P_S))}} \quad (3.3)$$

This calculation is accelerated using a CUDA enabled graphics card (GTX-285) using the CuFFT library. For a description and performance evaluation of CUDA and the CUDA SDK see appendix 2. The software interfaces

Figure 3.3: High level system diagram

to the CUDA card via the CUDA driver and the frame grabber using the Matrox Imaging Library, MIL. A photo multiplier tube (PMT) is used to convert the reflected light into a voltage, which is digitized by the frame grabber. A high level diagram of the system is shown in figure 3.3.

3.2 Software Overview

The software consists of three main components: user interface, hardware control and acquisition, and the tracking control loop. The user interface was created using python and the cross platform graphical user interface (GUI) library wxPython. This was chosen for its ease of use and flexibility.

Because the rest of the software was written in C/C++ for performance reasons, custom python modules needed to be created to handle the communication between C/C++ and python. These modules were created with the aid of a tool called simplified interface wrapper generated (SWIG). SWIG takes C++ classes and generates an interface to higher level languages such as python, perl, java, and C#.

Hardware control and acquisition was developed using the hardware vender's

application programming interfaces (API) using C/C++. The most sophisticated part of this software is the Matrox imaging library (MIL) control module. MIL allows for callback functions to be registered when certain events occur, such as when a certain scan line is reached or the start of a new frame. As strips of the image arrive, the MIL control module informs the control loop via a semaphore of new data. Careful attention must be paid in order to ensure that the callback functions that are registered do not take too long to execute. Processing, such as image rotation and hard drive I/O are handled in a separate thread in order to minimize call back execution time. Synchronization tools such as semaphores, events, and barriers are used to ensure that data is safely handled in both threads.

The third component of the software is the tacking loop. The purpose of this part is to wait for incoming strips of the image, calculate the normalized cross correlation (NCC), perform the prediction methods that will be discussed later, and applying the offsets to the optical scanners. The maximum normalized correlation value, which is a measure of similarity between the two images, is compared to a user controlled threshold to ensure that there is a reasonably high confidence in the calculation. Additionally, the location of the maximum correlation value in the NCC matrix is compared to a user controlled threshold to ensure that there was sufficient overlap between the current strip and the reference frame. If the overlap is too low, the NCC typically is unreliable and should be discarded. A high level flow diagram of the tracking loop is shown in figure 3.4.

Another aspect of the software was the flexibility for choosing how the NCC was calculated. Using the entire reference frame for equation 3.3 would be a very time consuming operation and in it would increase the update latency. Instead, the reference frame is divided into small sections, or regions of interest (ROIs). As strips of the current raster scan arrive, the NCC is calculated using only the ROI on the reference frame that corresponds to where the strip would be if there were no displacement. For instance, if a strip arrived who's center line was the 100th in the image, the ROI from the reference frame that has the most coverage of line 100 would be used as R in equation 3.3. The ROIs on the reference frame as well as the dimensions and locations of the strips from the incoming frames can be

Figure 3.4: Program flow diagram for the tracking loop

changed dynamically at run time from the user interface. Screen shots of the software are shown below figures 3.6, 3.5, and 3.7.

Reducing the size of the ROIs increases performance and allows for more updates per raster scan. Different dimensions for the ROIs were experimented with in order to find a balance between NCC accuracy (which would increase with larger ROIs) and performance (which would increase with smaller ROIs). Specifically, three different settings were chosen: one update per raster scan (full frame tracking), 4 updates per frame (coarse sub frame tracking), and 8 updates per frame (fine sub frame tracking).

Figure 3.5: Top Left: The acquisition settings panel allows the user to select which video channels are enabled and determine where the image sequences will be saved. Additionally, the user can put a threshold on the mean intensity value of the image so that only the brightest frames are saved Top Right: The optical scanners panel allows the user to manually adjust the video timing and field of view. Bottom Right: The tracking panel allows the user to adjust the setting for eye tracking. This is where the user can adjust the gain, prediction methods, NCC threshold, displacement threshold, etc..

Figure 3.6: Video panel and FOV selector. The video panel displays a live feed of the acquisition. Above the video, a few statistics about the incoming images are displayed (min, max, and average pixel values). The FOV window allows the user to select a new FOV to use.

Figure 3.7: Image of the panel that allows the user to control the ROIs from the reference image as well as the shape and location of the incoming strips. The user selects the number of ROIs, x offsets, y offsets, and spacing between ROIs.

Chapter 4

Tracking Methods

After the translation between the ROIs in the reference and current images has been estimated, the voltage required for compensating for this shift is multiplied by a user-selected gain and is applied to the optical scanners. This direct method of applying compensation has a couple of drawbacks. Firstly, applying the full compensation at once and as soon as the calculation is completed results in vertical stretching and horizontal shearing of the image. Secondly, if there is little overlap between the reference and current image the NCC can result in wildly inaccurate translations with large magnitudes. This distortion, that adds to that introduced by the eye motion, leads to errors in subsequent motion estimation (see figure 4).

In order to explore this issue, two different motion estimation methods were used, full frame and sub-frame. When using the whole image (i.e. full-frame) the motion compensation is best applied while the vertical (slow) scanner is returning to the top of the raster and no portion of the image is being collected. This avoids adding distortion to the recorded images, but it does however, limit the correction to a single update per frame. Estimating the motion and updating multiple times per frame (sub-frame method) can increase the effective bandwidth of the eye tracker, but at the potential cost of distorting the images while the motion correction is applied. Two extreme cases using coarse and fine horizontal images strips as ROIs were evaluated.

In order to increase the bandwidth and thus reduce the amplitude of the motion compensation required at each update, four different predictive algorithms were evaluated, namely: linear extrapolation, quadratic extrapolation, discrete cosine transform (DCT) extrapolation (low-pass filter), and Kalman filtering. These prediction methods output a displacement that is

Figure 4.1: Shows the effects of the applying motion compensation during a raster scan. The orange box shows compensation being applied against the direction of the raster (stretching) and the green box shows correcting with the direction of the scanning (compression)

used to steer the optical scanners.

4.1 Linear and Quadratic Extrapolation

The linear and quadratic prediction methods use a user-defined number of image displacements from previous ROIs, to extrapolate the motion using first and second order polynomials respectively as defined in equation 4.1.

$$\begin{aligned}\hat{x} &= c_1 + c_2T + c_3T^2 \\ \hat{x} &= c_1 + c_2T\end{aligned}\tag{4.1}$$

Using this assumption we can find the parameters $c_{1,2,3}$ that minimize the squared error between the model and the previous displacements by solving the following system of equations using N data points,

$$\begin{pmatrix} N & \sum_{i=1}^N t_i & \sum_{i=1}^N t_i^2 \\ \sum_{i=1}^N t_i & \sum_{i=1}^N t_i^2 & \sum_{i=1}^N t_i^3 \\ \sum_{i=1}^N t_i^2 & \sum_{i=1}^N t_i^3 & \sum_{i=1}^N t_i^4 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^N x_i \\ \sum_{i=1}^N x_i t_i \\ \sum_{i=1}^N x_i t_i^2 \end{pmatrix}\tag{4.2}$$

where t_i is the time stamp for the i th update and x_i is the displacement measured at time stamp t_i . For the case of linear extrapolation, only the upper 2×2 area of left most matrix is used. This system of linear equations is solved using the LAPACK library using which decomposes the matrix into an upper and lower diagonal matrix and uses backward substitution to solve for the parameters. Once the constants are calculated, the time stamp for the next strip is computed and fed into equation 4.1.

4.2 DCT Extrapolation

A signal can be decomposed into an finite sum of cosines of different frequencies. The amplitude of each cosine corresponds to the power of that frequency in the signal. In order to extract those coefficients the discrete

cosine transform (DCT) II is used which is defined in equation 4.3.

$$p_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right] \quad (4.3)$$

Each DCT coefficient, p_k , is calculated with a sum of N evenly spaced samples of the signal. These coefficients relate to the original signal through the following equation,

$$x_k = \frac{1}{2}x_0 + \sum_{n=1}^{N-1} p_n \cos\left[\frac{\pi n}{N}\left(k + \frac{1}{2}\right)\right]. \quad (4.4)$$

DCTs always result in real valued coefficients provided that the original signal is real valued. This transform is very similar to the Fourier transform except that the signal is assumed to have even symmetry. The value k in equation 4.4 does not have to be an integer number. In this way the DCT coefficients can be used to interpolate a value that wasn't originally sampled. Because each DCT coefficient corresponds to a cosine with a greater frequency than the previous coefficient, by only retaining some of the coefficients in the summation in equation 4.4 the signal can be effectively filtered. The DCT coefficients can be used to extrapolate the signal by using a value of k that is greater than N .

For the eye tracking system described in this work, DCT extrapolation or low pass filter (LPF) extrapolation was used as one method of prediction. This would be done by taking previously calculated displacements and perform the DCT II as described in equation 4.3 and then using equation 4.4 with k equal to $N + 1$ and only retaining some user controlled number of DCT coefficients. This prediction would then be fed back to the scanners.

This method was chosen because it would be more robust to noisy measurements because the tracking signal would be filterer. Unfortunately the assumption that the signal has even symmetry is not correct, additionally the strips are not necessarily evenly distributed in time so the DCT of these samples is only an approximation.

4.3 Kalman Filter

Essentially the Kalman filter algorithm has three main purposes:

1. Combine noisy measurements from different sensors of the same quantity into a single more accurate one
2. Use a physical model of how a quantity changes over time to improve the reliability and accuracy of a measurement
3. Estimate the parameters of the physical model or states that may not be directly observable

For the eye tracking system, the NCC is the only measurement of the displacement, so only the later two purposes are used here. The main motivation for using the Kalman filter is that if the current image has very low overlap with the reference, then the NCC can give wildly inaccurate translations with large magnitudes. LPF can be used to smooth out the tracking signal, but ultimately outliers will still have an impact on the tracking signal. By making a smoothness or physical constraint on what is expected of the signal, the Kalman filter can make a more accurate version of the signal.

One way to explain how the Kalman filter works, is to think of the rotation of the eye as a random variable, x . The NCC at time step k gives us an estimate of that random variable with some probability of being correct. This probability distribution is assumed to be Gaussian with some known variance R_k . The larger the variance the less certain the measurement. R_k is a function of overlap between the two images and the correlation value. If the correlation is very low or a very large displacement is observed, then R_k would be much larger. In order to improve the NCC an assumption about the motion of the eye is made. In this work a very simple model is assumed where the rotational velocity of the eye is assumed to be constant between any two measurement. This model also has some probability of being correct with an assumed Gaussian probability distribution with some variance Q_k . Using the simple constant velocity assumption, the noise is caused by the acceleration. The Kalman filter then takes both the measurement and the assumption and combine them to get an approximation of the true rotation

Model	Predict	Update
$x_k = Fx_{k-1} + Gu_{k-1} + w_{k-1}$ $y_k = Hx_k + v_k$	$\hat{x}_k^- = Fx_{k-1}^+ + Gu_{k-1}$ $P_k^- = FP_{k-1}^+F^T + Q_{k-1}$	$K_k = P_k^- H^T (H_k P_k^- H^T + R_k)^{-1}$ $\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-)$ $P_k^+ = (I - K_k H)P_k^-$

Table 4.1: Summary of the Kalman filter algorithm. x_k is a column vector containing the current state, F is a matrix represents the system dynamics, u_{k-1} is a column vector containing the control input, G is a matrix that relates the control input to the system state, w_{k-1} represents noise in the system dynamics with covariance matrix Q_k , y_k is a column vector containing the measurements, H is a matrix relates the system state to the measurements, and v_k is a column vector representing the noise in the measurements with covariance matrix R_k .

of the eye with some probability distribution of being correct and associated variance, P_k , that is lower than both R_k and Q_k .

An analytical derivation of the Kalman filter is described in appendix A. A brief summary of the result is shown in table 4.1. The Kalman filter is divided into two steps, predicting and updating. Prediction is performed by using the assumption of constant velocity between any two measurements. Using kinematics equations, this results in the following parameters for the Kalman filter [1],

$$x_k = \begin{pmatrix} x \\ \dot{x} \end{pmatrix}, \quad F = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}, \quad G = \begin{pmatrix} \Delta t^2/2 \\ \Delta t \end{pmatrix}, \quad Q = GG^T \sigma_a^2, \quad (4.5)$$

where σ_a^2 is the variance in the acceleration expected and can be adjusted at runtime by the user. The update step is performed by incorporating the translation calculated by the NCC and determining the variance associated with that measurement. This variance is determined by the correlation value from the NCC, lower correlation results in higher, or less certain, variance.

There are two issues associated with this implementation of the Kalman filter, accurately determining the covariance matrices R_k and Q_k and the poor accuracy associated with the simple constant velocity assumption. These two issues are the main source of error when using this tracking method.

Chapter 5

Experimental Results

5.1 Simulated Motion

Simulated motion was created by attaching a non-resonant galvanometric optical scanner to a model eye formed by a lens and a piece of paper. The scanner was driven with a sinusoidal voltage with a fixed amplitude of 1.54 degrees (approximately 2X the field of view), and varying frequencies to test the performance of the eye tracker. For each frequency, the tracking and motion signals were recorded with a digital oscilloscope. The peak velocity was calculated for each frequency and normalized by the velocity of the vertical component of the imaging raster. The results are summarized in figure 5.1.

The relative error is the norm of the error divided by the norm of the motion signal. Error is plotted against normalized peak velocity. This is the peak velocity in the injected motion divided by the vertical scanning speed. Higher normalized velocities indicate higher motion speeds. This metric was chosen because it is a measure of the speed of the motion compared to the sampling frequency. If the motion is equal to the speed of the vertical scanner (normalized peak velocity = 1.0), then each line of the image would be exactly the same (if the motion was opposing the raster scan) or each line in the image would be spaced by the FOV size (if the motion is with the direction of the raster scan) resulting in a very sparse sampling. In this way the normalized peak velocity is a measure of the distortion in the video sequence.

The lines in figure 5.1 follow the expected pattern. Generally speaking, full frame tracking performs the worst, as the features being tracked fall out

Figure 5.1: Percent error in the tracking signal for different simulated motions. The normalized peak velocity is the ratio of the peak velocity in the motion signal to the velocity of the raster scan. The color of the lines indicate the prediction method used. The line style corresponds to the number of updates made per frame, solid lines indicate once per frame (full frame), dotted lines indicate 4 updates per frame (coarse strips), and dashed lines indicate 8 updates per frame (fine strips). The frame rate of the system was 16.9 Hz.

of the field of view before the compensation has been applied. The more updates per frame, the better the performance. Fine strips were able to track motion that was up to 0.65 times the speed of the vertical scanning, while full frame tracking methods failed after 0.2. This is because the number of updates per raster scan directly relates to the bandwidth of the system. If there is only one update per frame, by the time the correction has been applied, it could have been too late. The sinusoidal motion that was injected into the system worked well for the prediction methods discussed earlier. Kalman filtering methods performed best for sub frame tracking while LPF method performed best for full frame tracking. Linear and quadratic extrapolation failed to show significant improvement.

The motion estimation method discussed earlier assumes rigid transformations for strips of the image. When motion causes stretching, compression, or sheering in the image, this assumption fails. This results in very low correlation value from equation 1.4. Because normalized peak velocity is a measure of how distorted the images are, Figure 5.1 shows the robustness of the assumptions made about rigid transformations of strips of the images.

5.2 Human Subjects

The eye tracker was evaluated in two different human subjects. Sequences of retinal imaged were recorded using each of the tracking methods discussed in section 4. The sequences were processed offline using the registration software discussed in [6]. Using the NCC, the average displacement per frame from a manually selected reference frame was calculated. The displacements were shifted by the median and plotted in a box and whiskers plot shown in figure 5.2. A box and whiskers plot is way of visualizing data sets. A horizontal line is drawn at the minimum, lower 25% of the data, the median, the lower 75% of the data, and the maximum. The inner quartile range (IQR), the distance between the lower 25% and lower 75% of the data, is boxed. If the subject had excellent fixation, most of the data points would be very close to the median. The IQR is a measure of how compacted the displacements are. The smaller the IQR the more stabilized the image sequence. For instance, subject two has significantly worse fixation which

Figure 5.2: Residual displacements in image sequences collected using different tracking methods for two different subjects: No Tracking (NT), No Prediction (NP), Linear Extrapolation Full Frame (LFF), Quadratic Extrapolation Full Frame (QFF) Kalman Filter Full Frame (KFFF), Low Pass Filter Full Frame (LPFF), No Prediction Sub-Frame (NPSF), Linear Extrapolation Sub-Frame (LSF), Quadratic Extrapolation Sub-Frame (QSF), Kalman Filter Sub-Frame (KFSF), Low Pass Filter Sub-Frame (LPSF). The box represents the inner quartile range (IQR) while bars represent the maximum and minimum of the displacements.

is indicated by the height of the box (*i.e.* the IQR) in the box plots.

Microsaccads are not capable of being tracked as they cause too much distortion in the image for equation 1.4 to work properly. This explains why the maximum and minimum are not improved with tracking turned on. Additionally when tracking fails, the FOV does not cover the reference ROI and it can not recover. This can occur after a blink, when the subjects fixation is drastically shifted or if the NCC failed and steered the imaging raster away from the tracking features.

The IQR was reduced significantly with tracking using the various methods. This implies that 50% of the residual displacements were brought closer to zero when tracking was turned on. Table 5.1 summarizes these results. The IQR for subject one was reduced by a maximum of 58.1% while subject two saw an maximum improvement of 88%. The difference in improvement is due to the fact that subject two has significantly worse

	Subject 1		Subject 2	
	<i>y</i> IQR	<i>x</i> IQR	<i>y</i> IQR	<i>x</i> IQR
NT	80	133	108.5	147
NPFF	37.5	23	48	50
LFF	68	52	109.5	86.5
QFF	45	33	233	142
KFFF	77	56	73	87
LPFF	56	58	148.5	147
NPSF	33.5	20.5	13	10
LSF	23	25	13	14.5
QSF	31.5	31	24	28
KFSF	38	43.5	42.5	53
LPSF	36	64	83.5	92.5

Table 5.1: IQRs for the various tracking methods. The *y* IQR is the IQR of the displacements in the *y* direction, while the *x* IQR is the IQR of the displacements in the *x* direction.

fixation than subject one.

There is no significant difference between the IQR for displacements in the *x* direction versus displacements in the *y* direction. This suggests that the tracking methods discussed in this work are suitable for both directions. Additionally, it suggests that the NCC is equally robust to image shear as it is to image stretch or compression.

The *x* and *y* displacements were plotted for each frame in the video sequence. The standard deviation is shown on these figures as an ellipse with a width equal to the standard deviation of the *x* displacements and the height equal to the standard deviation of the *y* displacements. The displacements for subject one are shown in Figure 5.3, and the displacements for subject two are shown in Figure 5.4. These plots show that the standard deviation is smaller when tracking is turned on and more displacements are within one standard deviation of zero. Tracking failures are shown as outliers that fall outside of the green ellipse. These are largely caused by the subject blinking.

In human subjects, prediction methods performed less favorably compared to tracking with no prediction. This is due to the simplified assumptions made by the prediction methods. More accurate models of eye motion

could improve the performance. Tremor makes fitting a simple linear or quadratic model very difficult. It also causes variable velocity with lots of acceleration. The Kalman filter model assumes constant velocity between measurement updates, and this is clearly not the case with actual eye motion. The low pass filtering method described earlier requires the DCT of the previous N equally spaced data points, however, these points are generally not equally spaced which can cause the extrapolation to give erratic results.

Figure 5.3: Average displacement per frame for image sequences using the various tracking methods. The green ellipses represent the standard deviation of the x displacements and y displacements.

Figure 5.4: Average displacement per frame for image sequences using the various tracking methods. The green ellipses represent the standard deviation of the x displacements and y displacements.

Chapter 6

Conclusions

AOSLOs allow for noninvasive high resolution imaging of the human retina. This imaging modality has the potential to become an import tool in early disease diagnosis and research. Serious engineering challenges make these systems difficult and unpractical to uses. Specifically, involuntary eye motion increases the number of images required for post processing. This problem is compounded by factors such as age and the presence of retinal disease.

By compensating for eye motion during image acquisition the number of images required can be significantly reduced. This not only allows for shorter imaging sessions, but also reduces post processing times, storage requirements, and light exposure for the subject.

In this thesis, a low bandwidth eye tracking system was developed for an AOSLO. Eye motion was extracted using the NCC accelerated by a CUDA enabled graphics card. Prediction methods were used to increase system bandwidth and reduce the amount of compensation required at each update. Four prediction methods were developed and evaluated: linear extrapolation, quadratic extrapolation, DCT extrapolation, and Kalman filtering.

Performance evaluation of the CUDA implementation of the NCC algorithm was performed. Computation times for matrix multiplication and FFTs for various image sizes was collected using CUDA and the competing CPU based libraries. It was found that CUDA can provide a factor of 5X improvement in performance for FFTs and a factor of 10X improvement in matrix multiplication over competing libraries. A MATLAB NCC implementation was made and compared to the performance of the CUDA

version. Two orders of magnitude improvement can be seen using the single precision implementation with CUDA over MATALAB. This drastic increase in performance allows for many more motion updates per raster scan and thus increases performance of the tracking system.

Characterization was performed for both simulated and actual eye motion. Using the methods discussed here, eye motion with speeds of up to 0.65 times the vertical raster scanning speed can be tracked. Using data collected from two subjects, a 50% to 80% improvement in the IQR of the residual image displacement was achieved. This results in significant reduction in the number of frames required to perform post processing. It reduces imaging time and thus reduces discomfort for the subject and decreases chances of retinal damage due to light exposure.

The main factor limiting the bandwidth of the system was compression and shearing of the images. As processing power increases, more sophisticated transformation models can be assumed while retaining real-time performance. Specifically equation 1.2 can be adjusted to include a term for these distortions.

Although the prediction methods showed a performance increase for simulated motion, when tested with human subjects the tracking system performed better with no prediction. This is largely due to the simplified assumptions made about the motion.

Additional work is required to find better prediction methods. Specifically the Kalman filter model could be improved by using a more sophisticated model of the motion. Other, more advanced, models exist, however, were not implemented due to time constraints. The white noise jerk model, exponentially autocorrelated acceleration model [1], and the oculomotor plant mechanical model [11] would have been good choices. Another fault of the Kalman filter was the assumption of acceleration being white noise when in fact it is colored.

The tracking system developed here has potential to be used for more than just reducing imaging requirements. Automated imaging sessions, on-line image registration, and spatially selective image acquisition are other possible uses for such a system. A more automated and practical imaging system can help bring this imaging modality from primarily a research tool

to widespread adoption as a clinical instrument.

Bibliography

- [1] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation and Tracking: Principles, Techniques, and Software*. Artech House, Inc, 1993.
- [2] Andreas Bulling, Jamie A. Ward, Hans Gellersen, and Gerhard Troster. Eye movement analysis for activity recognition using electrooculography. *IEEE Trans. on pattern analysis and machine intelligence*, 33:741–753, 2011.
- [3] Nathan S. Claxton, Thomas J. Fellers, and Michael W. Davidson. Laser scanning confocal microscopy, July 2011.
- [4] T. N. Cornsweet and H. D. Crane. Accurate two-dimensional eye tracker using first and fourth purkinje images. *J. Opt. Soc. Am.*, 63:921–928, 1973.
- [5] R.W. Ditchburn and B. L. Ginsborg. Involuntary eye movements during fixation. *The Journal of physiology*, 119:1–17, 1953.
- [6] Alfredo Dubra and Zachary Harvey. Registration of 2d images from fast scanning ophthalmic instruments. In *Lecture Notes on Computer Science*, 6204, pages 60–71, 2010.
- [7] Alfredo Dubra and Yusufu Sulai. Reflective afocal broadband adaptive optics scanning ophthalmoscope. *Biomedical Optics Express*, 2:1757–1768, 2011.

- [8] R. D. Ferguson, D. X. Hammer, C. E. Bigelow, N. V. Iftimia, T. E. Ustun, S. A. Burns, A. E. Elsner, and D. R. Williams. Tracking adaptive optics scanning laser ophthalmoscope. In *Proc. SPIE 6138*, pages 232–240, 2006.
- [9] D. X. Hammer, R. D. Gerguson, J. C. Magill, M. A. White, A. E. Elsner, and R. H. Webb. Image stabilization for scanning laser ophthalmoscopy. *Opt. Express*, 10:1542–1549, 2010.
- [10] B. Hermann, E. J. Fernandez, A. Unterhuber, H. Scattmann, A. F. Fercher, W. Drexler, P. M. Prieto, and P. Artal. Adaptive-optics ultrahigh-resolution optical coherence tomography. *Optics Letters*, 29:2142–2144, 2004.
- [11] Oleg V. Komogortsev. Eye movement prediction by kalman filter with integrated linear horizontal oculomotor plant mechanical model. In *ETRA '08*, pages 229–284, 2008.
- [12] Junzhong Liang, David R. Williams, and Donald T. Miller. Supernormal vision and high-resolution retinal imaging through adaptive optics. *J. Opt. Soc. Am. A*, 14:2884–2892, 1997.
- [13] Donald T. Miller, Junle Qu, Ravi S. Jonnal, and Karan Thorn. Coherence gating and adaptive optics in the eye. In *Proc. SPIE 4956*, pages 65–72, 2003.
- [14] NVidia. *CUDA CUFFT Library*, 3.1 edition, 2010.
- [15] NVidia. *NVIDIA CUDA Programming Guide*, 3.0 edition, 2010.
- [16] Lorrin A. Riggs, John C. Armington, and Floyd Ratliff. Motions of the retinal image during fixation. *Journal of the OSA*, 44:315–321, 1954.

- [17] Austin Roorda, Fernando Romero-Borja, III William Donnelly, Hope Queener, Thomas Hebert, and Melanie Campbell. Adaptive optics scanning laser ophthalmoscopy. *Optics Express*, 10:405–412, 2002.
- [18] Wayne J. Ryan, Andrew T. Duchowske, and Stan T. Birchfield. Limbus / pupil switching for wearable eye tracking under variable lighting conditions. In *ETRA*, pages 61–64, 2008.
- [19] Dan Simon. *Optimal State Estimation*. John Wiley and Sons, 2006.
- [20] Scott Stevenson and Austin Roorda. Correcting for miniature eye movements in high resolution scanning laser ophthalmoscopy. In *Proc. SPIE 5688*, 145, 2005.
- [21] Robert H. Webb and George W. Hughes. Scanning laser ophthalmoscope. *IEEE Trans. on Biomedical Engineering*, 28:488–492, 1981.
- [22] Stephen G. Whittaker, James Budd, and Roger W. Cummings. Eccentric fixation with macular scotoma. *Investigative Ophthalmology and Visual Science*, 29:268–278, 1988.
- [23] Qiang Yang, David W. Arathorn, Pavan Tiruveedula, Curits R. Vogel, and Austin Roorda. Design of an integrated hardware interface for aoslo image capture and cone-targeted stimulus delivery. *Virtual Journal for Biomedical Optics.*, 5:17841–17858, 2010.
- [24] Laurence R. Young and David Sheena. Eye-movement measurement techniques. *American Psychologist*, 30:315–330, 1975.
- [25] Robert J. Zawadzki, Stacey S. Choi, Alfred R. Fuller, Julia W. Evans, Bernd Hamann, and John S. Werner. Cellular resolution volumetric in vivo retina imaging with adaptive optics-optical coherence tomography. *Optics Express*, 17:4084–4094, 2009.

Appendix A

The Kalman Filter

The discussion of the Kalman filter provided here is a summary of the derivation in [19]. From a high level, the goal of the Kalman filter is the following:

1. Combine noisy measurements from different sensors of the same quantity into a single more accurate one
2. Use a physical model of how a quantity changes over time to improve the reliability and accuracy of a measurement
3. Estimate the parameters of the physical model or states that may not be directly observable

Suppose that the process that is being measured can be modeled using a discrete time linear system of the following form.

$$x_k = Fx_{k-1} + Gu_{k-1} + w_{k-1} \quad (\text{A.1})$$

Where x_k is a column vector of m elements containing the parameters, or state, of the model at time step k , F is an $m \times m$ matrix relating the state at time $k - 1$ to the state at time k , u_{k-1} is an n element column vector containing the control input to the system at time step $k - 1$, G is an $m \times n$ matrix relating the control input to the system state, and w_{k-1} represents the error in the model at time step $k - 1$. The measurements, y_k , of the process are related to the state using the following equation.

$$y_k = Hx_k + v_k \quad (\text{A.2})$$

Where y_k is j element column vector containing the measurements at time step k , H is $j \times m$ matrix that relates the current system state to the current measurements, and v_k is a j element column vector representing the noise in the measurement at time step k . It is assumed that the noise processes w_k and v_k are white, zero-mean, uncorrelated, and have a known covariance matrices Q_k and R_k respectively. The best estimate of x_k given all of the measurements up to and including the measurement at time step k is defined as the following.

$$\hat{x}_k^+ = E[x_k | y_1, y_2, \dots, y_k] \quad (\text{A.3})$$

Here $E[X]$ is the expected value of X and is defined as

$$E[X] = \sum_i^N x_i p_i \quad (\text{A.4})$$

Where p_i is the probability of the value x_i occurring. The estimate at time k given all of the measurements, $\hat{x}_k^+$, is also known as the *a posteriori* estimate. The prediction of x_k given all of the measurements up to but not including the measurement at time step k is known as the *a priori* estimate and is given by the following equation.

$$\hat{x}_k^- = E[x_k | y_1, y_2, \dots, y_{k-1}] \quad (\text{A.5})$$

The certainty of the estimate at time step k is given by the trace of the covariance matrix which is defined as

$$P_k = E[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T] \quad (\text{A.6})$$

Where $\bar{x}_k$ is the expected value of x at time k . It is assumed that the expected value and covariance of the state at time $k = 0$ is known. We would like to come up with a prediction of x_1 using our knowledge of the current state

$\hat{x}_0^+$. To do this we take the expected value of equation A.1.

$$\begin{aligned}\hat{x}_1^- &= E[Fx_0 + Gu_0 + w_0] \\ &= FE[x_0] + Gu_0 + E[w_0] \\ &= F\hat{x}_0^+ + Gu_0\end{aligned}\tag{A.7}$$

This says that given no additional information, our best guess for the next state is to use the system dynamics to predict. In general we can say that

$$\hat{x}_k^- = F\hat{x}_{k-1}^+ + Gu_{k-1}\tag{A.8}$$

We also need to determine the effect of the prediction on the covariance. The goal would be to determine P_k^- given P_{k-1}^+ , $\hat{x}_k^-$, and $\hat{x}_{k-1}^+$. To do this we take the covariance of equation A.1.

$$\begin{aligned}E[(x_k - \hat{x}_k^-)(\dots)^T] &= E[(Fx_{k-1} + Gu_{k-1} + w_{k-1} - \hat{x}_k^-)(\dots)^T] \\ &= E[(Fx_{k-1} + Gu_{k-1} + w_{k-1} - (F\hat{x}_{k-1}^+ + Gu_{k-1}))(\dots)^T] \\ &= E[(F(x_{k-1} - \hat{x}_{k-1}^+) + w_{k-1})(\dots)^T] \\ &= E[F(x_{k-1} - \hat{x}_{k-1}^+)(x_{k-1} - \hat{x}_{k-1}^+)^T F^T + w_{k-1}w_{k-1}^T + \\ &\quad F(x_{k-1} - \hat{x}_{k-1}^+)w_{k-1}^T + w_{k-1}(x_{k-1} - \hat{x}_{k-1}^+)^T F^T] \\ P_k^- &= FP_{k-1}^+ F^T + Q_{k-1}\end{aligned}\tag{A.9}$$

Here we have a way of determining the *a priori* estimate of the state as well as the covariance associated with it. The next step is to incorporate the measurements into the estimate and determine this effect on the covariance (*i.e.* using the measurement y_k determine our best guess of the state, $\hat{x}_k^+$, and calculate the covariance P_k^+). The Kalman filter is a recursive estimator, meaning that the new estimates require only the previous estimate and covariance. This is desirable because as the number of measurements grows, the amount computation required for all measurements would also grow. The general form of the Kalman filter update is shown in the equation below.

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-)\tag{A.10}$$

The error in the update equation is shown below.

$$\begin{aligned}
\epsilon_{\hat{x}_k^+} &= x_k - \hat{x}_k^+ \\
&= x_k - \hat{x}_k^- - K_k(y_k - H\hat{x}_k^-) \\
&= \epsilon_{\hat{x}_k^-} - K_k(Hx_k + v_k - H\hat{x}_k^-) \\
&= \epsilon_{\hat{x}_k^-} - K_k H(x_k - \hat{x}_k^-) - K_k v_k \\
&= \epsilon_{\hat{x}_k^-} - K_k H \epsilon_{\hat{x}_k^-} - K_k v_k \\
&= (I - K_k H) \epsilon_{\hat{x}_k^-} - K_k v_k
\end{aligned} \tag{A.11}$$

The covariance of the estimation error is defined as,

$$\begin{aligned}
P_{\epsilon_{\hat{x}_k^+}} &= E[\epsilon_{\hat{x}_k^+} \epsilon_{\hat{x}_k^+}^T] \\
&= E[((I - K_k H) \epsilon_{\hat{x}_k^-} - K_k v_k)(\dots)^T] \\
&= (I - K_k H) E[\epsilon_{\hat{x}_k^-} \epsilon_{\hat{x}_k^-}^T] (I - K_k H)^T - \\
&\quad K_k E[v_k \epsilon_{\hat{x}_k^-}^T] (I - K_k H)^T - (I - K_k H) E[\epsilon_{\hat{x}_k^-} v_k^T] K_k^T + \\
&\quad K_k E[v_k v_k^T] K_k^T
\end{aligned} \tag{A.12}$$

The measurement noise (v_k) is uncorrelated with the prediction error ($\epsilon_{\hat{x}_k^-}$) therefore,

$$\begin{aligned}
E[v_k \epsilon_{\hat{x}_k^-}^T] &= E[v_k] E[\epsilon_{\hat{x}_k^-}] \\
&= 0.
\end{aligned} \tag{A.13}$$

Using equation A.13 and equation A.12 we obtain

$$P_{\epsilon_{\hat{x}_k^+}} = (I - K_k H) P_{\epsilon_{\hat{x}_k^-}} (I - K_k H)^T + K_k R_k K_k^T. \tag{A.14}$$

We would like to choose the Kalman gain, K_k , such that it minimizes the sum of the variances of the state variables(*i.e.* choose K_k such that it minimizes the trace of the covariance matrix $P_{\epsilon_{\hat{x}_k^+}}$). If B is a symmetric matrix

Model	Predict	Update
$x_k = Fx_{k-1} + Gu_{k-1} + w_{k-1}$ $y_k = Hx_k + v_k$	$\hat{x}_k^- = Fx_{k-1}^+ + Gu_{k-1}$ $P_k^- = FP_{k-1}^+F^T + Q_{k-1}$	$K_k = P_k^- H^T (H_k P_k^- H^T + R_k)^{-1}$ $\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-)$ $P_k^+ = (I - K_k H)P_k^-$

Table A.1: Summary of the Kalman filter algorithm.

then we can use the following identity,

$$\frac{\delta \text{Tr}(ABA^T)}{\delta A} = 2AB. \quad (\text{A.15})$$

Minimizing equation A.14 becomes,

$$\frac{\delta \text{Tr}(P_k)}{\delta K_k} = 2(I - K_k H)P_{\epsilon_{\hat{x}_k^-}}(-H^T) + 2K_k R_k. \quad (\text{A.16})$$

Setting the derivative to zero and rearranging terms leads to the following expression,

$$K_k = P_{\epsilon_{\hat{x}_k^-}} H_k^T (H_k P_{\epsilon_{\hat{x}_k^-}} H_k^T + R_k)^{-1} \quad (\text{A.17})$$

We can now summarize the Kalman filter as a two step process (predict and update) in the following table.