Semi-Supervised Pattern Recognition and Machine Learning for Eye-Tracking

Thomas B. Kinsman
thomask@rit.edu

Follow this and additional works at: https://scholarworks.rit.edu/theses

Recommended Citation

This Dissertation is brought to you for free and open access by 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.
Semi-Supervised Pattern Recognition and Machine Learning for Eye-Tracking

by

Thomas B. Kinsman

B.S.E.E. University of Delaware, 1983
M.S.E.C.E. Carnegie Mellon University, 1987

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Chester F. Carlson Center for Imaging Science College of Science Rochester Institute of Technology

November 20, 2015

Signature of the Author

Accepted by

Coordinator, Ph.D. Degree Program Date
The Ph.D. Degree Dissertation of Thomas B. Kinsman has been examined and approved by the dissertation committee as satisfactory for the dissertation required for the Ph.D. degree in Imaging Science.

Dr. Jeff Pelz, Dissertation Advisor

Dr. Mark Fairchild

Dr. Nathan Cahill

Dr. Carol Romanowski

Date
Title of Dissertation:
Semi-Supervised Pattern Recognition and Machine Learning for Eye-Tracking

I, Thomas B. Kinsman, hereby grant permission to Wallace Memorial Library of R.I.T. to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Signature

Date
Semi-Supervised Pattern Recognition and Machine Learning for Eye-Tracking

by

Thomas B. Kinsman

Submitted to the
Chester F. Carlson Center for Imaging Science
in partial fulfillment of the requirements
for the Doctor of Philosophy Degree
at the Rochester Institute of Technology

Abstract

The first step in monitoring an observer’s eye gaze is identifying and locating the image of their pupils in video recordings of their eyes. Current systems work under a range of conditions, but fail in bright sunlight and rapidly varying illumination. A computer vision system was developed to assist with the recognition of the pupil in every frame of a video, in spite of the presence of strong first-surface reflections off of the cornea.

A modified Hough Circle detector was developed that incorporates knowledge that the pupil is darker than the surrounding iris of the eye, and is able to detect imperfect circles, partial circles, and ellipses. As part of processing the image is modified to compensate for the distortion of the pupil caused by the out-of-plane rotation of the eye. A sophisticated noise cleaning technique was developed to mitigate first surface reflections, enhance edge contrast, and reduce image flare. Semi-supervised human input and validation is used to train the algorithm.

The final results are comparable to those achieved using a human analyst, but require only a tenth of the human interaction.
Acknowledgements

First and foremost I must acknowledge the love and support of my wife, Lynn E. Kinsman. Lynn adds much needed order to the chaos of my life. Both spiritually and financially, she has kept the family growing and the house running. Lynn proof-read several chapters of my proposal, kept the family together, supervised college visits for our oldest child, and supported me spiritually during the entire thesis processes. Lynn is my soul mate.

Jeff B. Pelz has been my steadfast adviser throughout the process, providing guidance and financial and academic support and encouragement over the years. For me it was not just a journey of finding the solutions, it was also a journey of understanding the methods well enough to teach them in the future. Jeff supported me in this, went to bat for me, fought more political arguments than I am aware of, and encouraged me to go out into the world and profess. Jeff is a friend and a confidant, and through his confidence in me I have gained wisdom.

Mark D. Fairchild was my first professor at RIT. Mark is a wonderful example of a dedicated professor and collaborator. I use some of his teaching methods to this day. We share an interest in furthering the state of education, the vision system, and anomalous-trichromacy.

I am sincerely grateful to Nathan D. Cahill. Nate gave guidance and inspiration throughout the process. Nate encouraged me to take advantage of the constrained nature and motions of the eye, and to develop the method for unwrapping the eye. Nate also provided additional guidance on the structure of this dissertation.

Carol Romanowski was my first graduate professor, and taught the first course I took in machine learning and data mining. In a sense she was behind me “before the beginning” of my Ph.D. journey. Now as I teach one of the classes that she taught, I constantly think of how she taught them. Carol is a source of guidance in all things, and I am extremely fortunate to have her serving on my committee. I am deeply indebted to Carol for her thorough review of this work and her many useful and enlightening comments. This thesis reads much better because of her, and I owe her a great debt.

Many people at RIT helped me navigate academic administration. These people include Sue Chan who assured that my coursework was on track, and Joyce French who kept my finances in order. Joe Pow is always at the center of the Center for Imaging Science. I would also like to thank Val Hemink, Mel Warren, Beth Lockwood and many others that helped on a day-to-day basis.

I would like to thank my parents, Betty and Bill, for their love and support, and for being good role models. I would like to thank my children, Duncan and Devon, for putting up with my long hours away as I tried to set a good role model. Again, I want to thank my wife, Lynn, or keeping the family happy and healthy.
I would like to thank John Kerekes and Anthony Vodacek for admitting me to the program. Dr. Kerekes also serves as the Ph.D. program coordinator, and Dr. Vodacek served as another adviser. They both provided the different point of view that was sometimes needed to see through noise.

I would also like to thank Maria Helguera for her help, support, and advice over the years. Dr. Helguera’s medical imaging classes emphasized the importance of contrast in any feature used for computer vision or machine learning.

I would like to thank Juan Cockburn, Harvey Rhody, Roger Gaborski, and Roxanne Canosa for numerous discussions of computer vision. Carl Salvaggio, Naval Rao, Jan van Aardt, and Peter Bajorski were also helpful for clarifying discussions on topics from LiDAR to statistics. Thanks are due to Roger Easton for periodic sessions on imaging mathematics, and to Emmett Ientilucci for additional tutoring on radiometry.

There are many students to thank. These include my study mates Brandon May and Kelly Canham. Office mates and the entire cadre of students that I went through the core courses with. Michael Presnar, Karl Walli, Jarad Herweg, and Brian Flusche were especially helpful. I also wish to thank the other graduate students in the Multidisciplinary Vision Research Laboratory, Susan Munn, Tommy Keane, Dong Wang, Preeti Vaidynathan, and Kate Walders.

Since research is a critical part of a Ph.D. quest, many of the staff of the Wallace library became colleagues. Most of the research librarians, especially Adwoa Boateng, Susan Mee, Linette Koren, Roman Koshykar, and Gus Kovalik knew me on a first name basis. Internet searches cannot reliably answer a question about the interaction of artwork and human vision, but Kari Horowicz can.

The Wallace Center also includes some helpers behind the scenes who should be known to those that follow. Tracey Melville and Diane Grabowski are in charge of thesis and dissertation binding. If you are not reading this in a bound form, but are instead reading it electronically, it is because of Jennifer Roeszies, the coordinator for electronic dissertations. Another secret super-hero for research is Morna Hilderbrand, Interlibrary Loan Librarian and Manager for the IDS Department. Morna Hilderbrand never failed to obtain any book or reference. Morna and her team connect the Wallace library to a network of libraries throughout US and beyond.

Finally, I wish to thank Lynn Kinsman for supporting me during this time, and allowing me the pleasure of this quest.
This research was supported in part by the National Science Foundation (grant number 0909588).
This dissertation is dedicated to those that come after me. To all students, everywhere, including my children, so that they know that with dedication and determination you can succeed.
Contents

Declaration ................................................. 1
Approval .................................................. 2
Copywrite ................................................ 3
Abstract .................................................. 4
Acknowledgements ....................................... 5
Dedication ............................................... 8

List of Figures ........................................... 16
List of Tables ............................................ 20

1 Introduction ............................................. 22
  1.1 Problem Statement .................................... 23
    1.1.1 System Requirements .............................. 24
    1.1.2 Calibration ....................................... 24
    1.1.3 Mobile Eye Tracking Requirements ............... 25
  1.2 Data Inundation Problems and Issues .................. 25
    1.2.1 Data Inundation in Frames ....................... 25
    1.2.2 Data Inundation in Pixels ....................... 26
    1.2.3 Yarbus Software .................................. 26
    1.2.4 Data Reduction Techniques for Analysis ......... 27
    1.2.5 Pupil Location .................................. 27
  1.3 Processing Stages .................................... 28
    1.3.1 Automated Approach .............................. 28
    1.3.2 Use of Semi-Supervised Learning ................. 29
  1.4 Innovations ......................................... 30

2 Workflow and Primary Passes ............................ 33
  2.1 Overview of Primary Processing Passes ............... 34
  2.2 First Pass and Statistical Collection ................ 36
    2.2.1 Background Removal Statistics ................. 36
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.7.5</td>
<td>Other Methods</td>
<td>71</td>
</tr>
<tr>
<td>4.8</td>
<td>An Adaptive Noise Removal For Edge Detection</td>
<td>72</td>
</tr>
<tr>
<td>4.8.1</td>
<td>Leveraging Duality of Segmentation and Edge Detection</td>
<td>72</td>
</tr>
<tr>
<td>4.9</td>
<td>An Adaptive Edge Preserving Filter</td>
<td>72</td>
</tr>
<tr>
<td>4.9.1</td>
<td>Example Algorithm Using Eight Regions</td>
<td>73</td>
</tr>
<tr>
<td>4.9.2</td>
<td>Comparison in One Dimension</td>
<td>74</td>
</tr>
<tr>
<td>4.9.3</td>
<td>The Secret of the Shapes</td>
<td>76</td>
</tr>
<tr>
<td>4.10</td>
<td>Performance Optimization</td>
<td>81</td>
</tr>
<tr>
<td>4.10.1</td>
<td>Fast Variance Computation</td>
<td>81</td>
</tr>
<tr>
<td>4.10.2</td>
<td>Algorithm in Pseudocode</td>
<td>82</td>
</tr>
<tr>
<td>4.10.3</td>
<td>Re-weighting the Variance</td>
<td>83</td>
</tr>
<tr>
<td>4.10.4</td>
<td>Failure Modes</td>
<td>85</td>
</tr>
<tr>
<td>4.11</td>
<td>Additional Benefits</td>
<td>85</td>
</tr>
<tr>
<td>4.11.1</td>
<td>Edge Enhancement in Transition Regions</td>
<td>85</td>
</tr>
<tr>
<td>4.11.2</td>
<td>Edge Direction</td>
<td>86</td>
</tr>
<tr>
<td>4.12</td>
<td>Results – Before and After Images</td>
<td>86</td>
</tr>
<tr>
<td>4.13</td>
<td>Comparison to Other Methods</td>
<td>96</td>
</tr>
<tr>
<td>4.13.1</td>
<td>Unsharp Masking</td>
<td>96</td>
</tr>
<tr>
<td>4.13.2</td>
<td>Bilateral Edge Detection</td>
<td>96</td>
</tr>
<tr>
<td>4.14</td>
<td>Conclusions</td>
<td>98</td>
</tr>
<tr>
<td>5</td>
<td>The Hough Transform</td>
<td>100</td>
</tr>
<tr>
<td>5.1</td>
<td>Overview</td>
<td>100</td>
</tr>
<tr>
<td>5.2</td>
<td>The Hough Transform Basics</td>
<td>101</td>
</tr>
<tr>
<td>5.2.1</td>
<td>The Hough Transform for Circle Detection</td>
<td>101</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Ideal versus Actual Votes</td>
<td>106</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Hough Parameter Space for Circles</td>
<td>106</td>
</tr>
<tr>
<td>5.3</td>
<td>The Focused Hough Circle Detector</td>
<td>107</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Example Incorporating Dark Edges</td>
<td>107</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Traditional Approach is Susceptible to Noise</td>
<td>110</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Example on an Eye Image</td>
<td>111</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Novel Contributions</td>
<td>113</td>
</tr>
<tr>
<td>5.4</td>
<td>Finding Ellipses using a Hough Circle Detector</td>
<td>115</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Hough Parameter Space for Ellipses</td>
<td>115</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Noise Tolerance for Circles</td>
<td>116</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Using Ellipse Signatures for Detection</td>
<td>117</td>
</tr>
<tr>
<td>5.5</td>
<td>Conclusion</td>
<td>121</td>
</tr>
</tbody>
</table>
## Progressive Connectivity

9.1 Progressive Connectivity Concept ........................................ 162
9.2 Progressive Connectivity for Lines ...................................... 163
9.3 An Algorithm for Progressive Connectivity for Lines .............. 165
  9.3.1 A Linear Example ...................................................... 166
9.4 Progressive Connectivity for Circles ................................... 168
  9.4.1 Feature Selection ...................................................... 168
  9.4.2 Region Shape .......................................................... 169
  9.4.3 Example Remaining Significant Points ............................... 170
  9.4.4 Fitting to Points ...................................................... 171
9.5 Results ............................................................................ 173
9.6 Other Considerations ....................................................... 174
  9.6.1 Deterministic .............................................................. 174
  9.6.2 Efficiency ................................................................. 174
9.7 Conclusion ........................................................................ 174

## A Degree of Confidence Metric for Outdoor Pupils

10.1 Purpose ........................................................................... 175
10.2 Position in processing chain .............................................. 175
10.3 Biometrics ........................................................................ 176
  10.3.1 Survey of Biometrics ..................................................... 176
  10.3.2 Borrowing from Biometrics ............................................ 180
10.4 Our Degree of Confidence [DOC] ....................................... 180
  10.4.1 Component calculations ................................................ 180
  10.4.2 Contrast ....................................................................... 181
  10.4.3 Incoherence ................................................................. 181
  10.4.4 Separation .................................................................... 182
  10.4.5 Relative edge strength .................................................. 182
  10.4.6 Ensemble combination experiment ................................. 182
10.5 Final selection for DOC ..................................................... 183

## Comparison of Human Analysts

11.1 Intra-Analyst Comparison ................................................ 185
11.2 Inter-Analyst Comparison ................................................ 187

## Results

12.1 Olema ............................................................................. 196
12.2 Olema 2012 Subject 6 ........................................................ 196
12.3 Lone Pine Data Analysis ................................................... 197
  12.3.1 LonePine 2013 Subject 3 ............................................... 197
  12.3.2 LonePine 2013 Subject 5 ............................................... 198
12.3.3 LonePine 2013 Subject 8 ................................................. 198
12.4 Death Valley ................................................................. 199
  12.4.1 Death Valley 2013 Subject 3 ......................................... 199
  12.4.2 Death Valley 2013 Subject 7 ......................................... 200
12.5 Initial Results .............................................................. 203
  12.5.1 Lone Pine 2013 without and with SER ................................ 203

13 Future Work ................................................................. 205
  13.1 Tracking ................................................................. 205
  13.2 Best Mixture of Characteristics for a Degree of Confidence ........ 206
  13.3 Edge Estimation ........................................................ 206
  13.4 Discussion .............................................................. 207

14 Conclusions and Key Findings ........................................... 208
  14.1 Contributions of this Thesis .......................................... 209
    14.1.1 Multiple Sampling Passes ...................................... 209
    14.1.2 Eyelid Removal .................................................. 209
    14.1.3 Adaptive Noise Cleaning ...................................... 210
    14.1.4 Focused Hough Transform ..................................... 210
    14.1.5 Unwrapping the Eye ............................................ 210
    14.1.6 Strange Ellipse Rejection ..................................... 211
    14.1.7 Benefits of Combining Multiple Features .................... 211
    14.1.8 Pupil Contour Extraction ...................................... 211
    14.1.9 Degree of Confidence Metric .................................. 211
    14.1.10 Results .......................................................... 211
  14.2 Closing ................................................................. 212

A Publications and Presentations ......................................... 213

B Misperceptions and Challenges ......................................... 215
  B.1 Humans Detect Eyes From Infancy ................................... 215
  B.2 The Pupil Illusions ................................................... 216
  B.3 Resolution and Quality Challenges .................................. 216
  B.4 A List of Challenges .................................................. 217
  B.5 Summary ............................................................... 219

C Irradiance Computation .................................................... 220
  C.1 Estimating the Amount of Light Reaching the Eye ................. 220
  C.2 Relative Amount of Uncontrolled Light ............................ 223
D Cornea as a Window: 224

E Hough Transform Timeline 227
   E.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 227
   E.2 Important Dates in Hough Transform Development . . . . . . . . . . . . 227

F Simulating the Image of the Eye 229
   F.1 Graphical Users Interface . . . . . . . . . . . . . . . . . . . . . . . . . . . . 230
   F.2 Effect of Refraction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 231
   F.3 Refraction of Pupils . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 234
   F.4 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 234

G Testing for Bias in Results 236
   G.1 Comparison between Program and Analysts . . . . . . . . . . . . . . . . . . 236
   G.2 Comparison between Analysts . . . . . . . . . . . . . . . . . . . . . . . . . 239
   G.3 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 242

H System Hardware and Software 244

Bibliography 245
List of Figures

1.1 Main Stages of Processing ................................................. 30
2.1 Work flow overview ...................................................... 34
2.2 Computing statistics for frames in first pass ......................... 37
2.3 Classifying blink frames ................................................ 38
2.4 Easy pass .............................................................. 40
2.5 User Validation .......................................................... 41
2.6 Automatic Pupil Detection Pass ....................................... 44
3.1 Example of eyelids detected ............................................ 47
3.2 Finding Lower Eyelid ...................................................... 51
3.3 Finding Upper Eyelid ....................................................... 53
3.4 An example of eyelid detection ........................................ 54
3.5 Second example of eyelid detection ................................... 54
3.6 Third example of eyelid detection ..................................... 55
3.7 Fourth example of eyelid detection .................................... 55
3.8 Eyelid on an low contrast image ...................................... 55
3.9 Eyelid on an ugly image ................................................ 56
3.10 Eyelid on an image of the eye looking away from camera ......... 56
3.11 Eyelid on an image of the eye looking away from camera ......... 56
3.12 Two frames from the test suite that failed the eyelid detection .. 57
4.1 Example of Adaptive Noise Cleaning .................................. 60
4.2 Example non-circular pupil ............................................. 62
4.3 Example image with compression noise ............................... 66
4.4 Kuwahara filtering on a $3 \times 3$ region .............................. 73
4.5 Kuwahara filtering in one dimension ................................ 77
4.6 Shapes of regions used for adaptive noise removal ............... 80
4.7 The Adaptive Noise Reduction Algorithm ............................ 83
4.8 Re-weights for variance of regions by size. .......................... 84
LIST OF FIGURES

4.9 Failure of adaptive noise smoothing ........................................... 85
4.10 Statistics from Bilateral Filter .................................................. 97
4.11 Statistics from Adaptive Noise Cleaning ...................................... 98

5.1 Three points to fit a circle to ...................................................... 101
5.2 Three points and one circle ....................................................... 102
5.3 Three points to fit a circle to with 3 circles ................................. 102
5.4 Three points and resulting center ............................................... 103
5.5 Three points with three voting circles of radius 90 pixels ............. 104
5.6 Three points with three voting circles of radius 130 pixels ........... 104
5.7 Ideal circle located ................................................................. 105
5.8 Effect of noise edges on the Hough transform ......................... 105
5.9 Ideal circle to fit to ................................................................. 107
5.10 Ideal circle with one edge ....................................................... 108
5.11 Ideal circle with one edge, multiple radii ................................. 108
5.12 Ideal circle with multiple edges, multiple radii ......................... 109
5.13 Eight of the compass angles contributing to the voting ............. 109
5.14 Combined edges, multiple radii ............................................... 110
5.15 Example edges detected in an eye image .................................. 111
5.16 Hough space for radii from 10 to 18 pixels ............................. 113
5.17 Quantized Gradient Directions ............................................... 114
5.18 Ellipse with eccentricity of 1.04 .............................................. 117
5.19 Ellipse with eccentricity of 1.33 ............................................. 118
5.20 Ellipse with eccentricity of 1.5 ............................................... 119
5.21 Ellipse with eccentricity of 2.0 ............................................... 119
5.22 Three dimensional view of previous accumulator space ............ 120
5.23 Ellipse with eccentricity of 2.0 ............................................... 121

6.1 Figure from Dollár, et. al., showing that the eye can only take on a limited number of appearances .................. 123
6.2 Face motions on a manifold ...................................................... 124
6.3 An Unwrapping Example ......................................................... 126
6.4 Example of Unwrapping With Grid Lines .................................. 127
6.5 The Circle Illusion ................................................................. 128
6.6 Schematic for understanding unwrapping ................................. 130
6.7 Schematic with variables used for understanding unwrapping ........ 131
6.8 Minor axis projection to find center of expansion ..................... 133
6.9 Out of plane rotation by $\theta$ degrees ..................................... 135

7.1 Some Strange Ellipses ............................................................ 139
7.2 Finding Region A – pupils that are invalid because they are highly eccentric 143
LIST OF FIGURES

7.3 Example of a region A found .................................................. 144
7.4 Example of regions B and C found ............................................. 146
7.5 Algorithm for finding region of pupils in parameter space of which are tilted wrong ......................................................... 148

8.1 For one feature, 10 pizzas (represented by each box) span all possible amounts of sauce in ounces. Ten instances fill the entire data space ........................................ 151
8.2 With three separate ten value features for a hypothetical pizza recipe, it would require 1,000 pizzas to try all possible combinations. Ten instances of pizza now only spans only one hundredth the full data space .................................................. 152
8.3 Human Vision versus Computer Vision ........................................... 155
8.4 Pixel Selection Algorithm ......................................................... 157
8.5 Input eye image ................................................................. 158
8.6 Candidate pupil edge pixels ..................................................... 158
8.7 Candidate pupil edge pixels in 3D .............................................. 159
8.8 “Best” pupil edge points ......................................................... 160

9.1 The progressive connectivity algorithm ........................................ 163
9.2 An image of the Erie Canal in Fairport, NY .................................... 164
9.3 The edge magnitude computed at each point on the green channel of Figure 9.2. This demonstrates how dramatically the image changes focus from left to right across the image. The darker the pixel, the stronger the local edge gradient is ......................................................... 165
9.4 Progressive connectivity example for two regions .............................. 166
9.5 The edges that are mutually strongest edges for two overlapping regions at another point in the image ......................................................... 167
9.6 Results of repeating the process across the entire edge from left to right across the entire image ......................................................... 168
9.7 Progressive Connectivity - Radial Annular .................................... 169
9.8 Progressive Connectivity - Radial Point Selections ............................ 170
9.9 The original set of possible edge points (hollow yellow squares), remaining significant edge points (solid yellow squares), and the circle formed by a direct least-squares fit to all of these points ......................................................... 171
9.10 The final selected circle fit, based on the best semi-circle that has the most significant edge points come within a set tolerance (nominally one pixel) of the fit circle ......................................................... 172
9.11 The final pupil circle shown on the original input image .................... 173

11.1 Video A vs the other two ....................................................... 186
11.2 Video B vs the other two ....................................................... 186
11.3 Video C vs the other two ....................................................... 187
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.4</td>
<td>Comparison between analysts</td>
<td>188</td>
</tr>
<tr>
<td>11.5</td>
<td>Analyst 1 vs 2 X values</td>
<td>189</td>
</tr>
<tr>
<td>11.6</td>
<td>Analyst 1 vs 2 Y values</td>
<td>190</td>
</tr>
<tr>
<td>11.7</td>
<td>Analyst 1 vs 3 X values</td>
<td>191</td>
</tr>
<tr>
<td>11.8</td>
<td>Analyst 1 vs 3 Y values</td>
<td>192</td>
</tr>
<tr>
<td>11.9</td>
<td>Analyst 2 vs 3 X values</td>
<td>193</td>
</tr>
<tr>
<td>11.10</td>
<td>Analyst 2 vs 3 Y values</td>
<td>194</td>
</tr>
<tr>
<td>12.1</td>
<td>Selected frame 8315 from Death Valley, 2013, Subject 7. This shows that the eye camera is looking up over the bottom of the bottom eyelid. The red circle is the initial location that the program suggested as the first pupil location. For this example the program was correct.</td>
<td>201</td>
</tr>
<tr>
<td>12.2</td>
<td>A pupil detection even in the presence of lens flare.</td>
<td>202</td>
</tr>
<tr>
<td>B.1</td>
<td>Example of eye camera problem</td>
<td>217</td>
</tr>
<tr>
<td>C.1</td>
<td>Fraction of Planckian black-body radiator that could be detected by our silicon based IR camera. About 0.25.</td>
<td>221</td>
</tr>
<tr>
<td>C.2</td>
<td>Reflection curve for an area near Arizona’s Remote Sensing Group. For 740 to 1100 nm, the typical reflectance is about 0.38. From [1].</td>
<td>222</td>
</tr>
<tr>
<td>D.1</td>
<td>Fresnel Reflection off the eye surface</td>
<td>225</td>
</tr>
<tr>
<td>F.1</td>
<td>Eye Simulation GUI</td>
<td>230</td>
</tr>
<tr>
<td>F.2</td>
<td>Eye simulation showing effect of refraction</td>
<td>231</td>
</tr>
<tr>
<td>F.3</td>
<td>Eye simulation showing effect of refraction as a function of rotation</td>
<td>232</td>
</tr>
<tr>
<td>F.4</td>
<td>Pupil appearance without and with refraction</td>
<td>232</td>
</tr>
<tr>
<td>F.5</td>
<td>Impact of refraction on Position and Eccentricity.</td>
<td>233</td>
</tr>
<tr>
<td>G.1</td>
<td>Bias diagram LP 2013 SN 03</td>
<td>238</td>
</tr>
<tr>
<td>G.2</td>
<td>Bias diagram LP 2013 SN 07</td>
<td>239</td>
</tr>
<tr>
<td>G.3</td>
<td>Bias diagram bet. Analysts 1 and 3</td>
<td>241</td>
</tr>
<tr>
<td>G.4</td>
<td>Bias diagram bet. Analysts 2 and 3</td>
<td>242</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Edge detector used for eyelid edge detection. ...................................... 49
4.1 Comparison images of Adaptive Noise Cleaning ................................. 70
4.2 Images ANC contrast set 1 ............................................................. 88
4.3 Images ANC contrast set 2 ............................................................. 89
4.4 Images ANC contrast set 3 ............................................................. 90
4.5 Images ANC contrast set 4 ............................................................. 91
4.6 ANC on images with exposure issues ................................................ 92
4.7 ANC on images of motion blur ....................................................... 93
4.8 ANC on images with other issues .................................................... 93
4.9 ANC on images with different pupil sizes ........................................ 94
4.10 ANC on first surface reflections ..................................................... 95
6.1 Results for first pass only. .............................................................. 122
9.1 Progressive Connectivity Results ..................................................... 173
11.1 Comparison of blink classification within analysts ............................ 185
12.1 South of Olema 2012 SN06 .......................................................... 197
12.2 Results for Lone Pine 2013 SN03 .................................................. 197
12.3 Results for Lone Pine 2013 SN05 .................................................. 198
12.4 LonePine 2013 SN08 ................................................................. 199
12.5 Data for Death Valley, SN03 ........................................................ 200
12.6 Data for Death Valley, SN07 ........................................................ 202
12.7 Lone Pine, Subject SN03, without and with SER ............................ 203
12.8 Lone Pine, Subject SN05, without and with SER ............................ 204
F.1 Table of displacements and eccentricities ......................................... 234
G.1 Table of differences between system results and analyst results for Lone Pine 2013 subjects 03 and 07. .................................................. 237
G.2 Table of differences between analysts for South of Olema, 2012, subject 06. 240
Vision is part of complex human behavior and gaze is a precognitive indicator of a subject’s attention [2, 3]. A subject’s fixation patterns indicate their cognitive strategies, but are not consciously available to a subject. Even though not cognisant of their actions, people use the world as an external storage for information [2]. In order to extract information from the world, people must look at the objects in the world and fixate the world on their retinas. The patterns that subjects use to access that information can reveal important insights into their problem solving strategies, even if the subject is not consciously aware of these strategies [3].

The process of determining where a person is looking in the world requires a recording of where the eye is looking, a frame of reference for the world around them, and a correspondence formed between the two. To form a correspondence between the pose of the eye and the location of attention in the scene, the first step is to determine the pose of the eye for every frame of recorded eye motion.

This eye pose, or pupil location, must be located regardless of other noise present in the scene, such as reflections, shadows across the eye, deep shadows of the corners of the eye, and partially occluded pupils by the eyelids.

Commercial fully-automated eye-tracking systems are available for processing indoor eye-tracking tasks. For outdoor eye-tracking the fully-automated system become unreliable. To compensate for that unreliability a labor intensive, completely manual system was developed including custom software. An analyst used the software by looking at each frame and locating the pupil in each individual frame of the eye video. The fully-manual system allowed an analyst to process videos at a rate of two to three frames per minute.

A previous method for assisting in the coding of these videos was developed and patented [4]. This method was instantiated in a program named *SemantiCode* [5]. *SemantiCode* was subsequently commercialized and licensed. While helpful, the program still requires that every frame in the input video have the eye location correctly identified it
This work provides analysts with a tool that automatically locates pupils quickly, even under challenging conditions. This work was developed to locate the pupil in videos of the eye that are captured under challenging conditions at a faster rate than manually labeling every single frame of the video.

The larger goal is to analyze observer behavior in natural outdoor conditions that include being surrounded by high contrast daylight, sudden changes in illumination, and first surface reflections off the cornea of the eye [6,7].

1.1 Problem Statement

In a portable eye tracking study, a subject’s eye is followed with a head mounted camera. This records the subject’s eye pose throughout the study, which then requires that the subject’s pupil be located in every frame of the video.

This pupil location is then used with a calibration sequence to form a mapping from where the subject’s eye pose is in the eye video, to where the subject is looking in the real world (the scene).

Fully automated video based eye-tracking systems work for indoor tasks when the lighting is controlled and the user is constrained to a limited range of motion. Previous work at MVRL developed and used pupil detectors based on Loy and Zelinski’s fast radial circle detector [8], [9]. That approach worked well for indoor eye tracking, but when the pupil became elliptical the circle detector located an end of the pupil ellipse instead of the center of the pupil ellipse. Each end of an ellipse has a focal point, and circle detectors tend to find these.

Our goal is to analyze observer behavior in natural outdoor conditions that include being surrounded by high contrast daylight, sudden changes in illumination, and first surface reflections off the cornea of the eye [6,7].

All of these issues arise when eye-tracking subjects are taken out of doors. Scientists are increasingly taking eye-tracking studies outside in uncontrolled lighting. Even when the eyes are shaded from direct sunlight using large hats, some of the previously listed issues still remain.

The Rochester Institute of Technology’s Multidisciplinary Vision Research Lab (MVRL), in collaboration with the University of Rochester, is involved in a multi-year NSF study tracking the eyes and gaze patterns of geology students and experts [6]. Since multiple students were eye-tracked for multiple hours over multiple days, and since the study was a multi-year study, a large amount of data is generated.

During the study, the subjects’ eyes were shielded from direct sunlight using large hats. Nevertheless many of the listed problems associated with reflection of the world in the eye, uncontrolled lighting, and tightly constricted pupils remained. Once let loose in the natural world, the eye-tracking task becomes challenging, and automated processing
becomes intractable. Processing of these videos requires having analysts manually locate the pupil in every single frame of the video.

Using custom developed software an analyst can processes two to three frames per minute. Each analyst must be carefully trained, and the analyst is susceptible to fatigue from hours of sitting and inspecting video frames.

The goal in this work is to increase the data processing rate – the rate at which we can identify the center with certainty the pupil location in a video of a subject’s eye as they are being eye-tracked.

The end result is the location of the center of the pupil in each frame, which is subsequently used to discover where the subject was looking in the scene.

The rest of this chapter discusses some of the challenges we faced in trying to reduce the amount of human-computer interaction while still maintaining pupil location accurately.

1.1.1 System Requirements

Mobile eye tracking uses two head-mounted video cameras on human subjects to record:

- the position of the subject’s eye in their head (recorded in the “eye video”), and
- the scene in front of the subject (recorded in the “scene video”).

These videos are recorded continuously throughout the study. These two videos (eye and scene) must be kept synchronized for later calibration. The latest capture system synchronizes these eye and scene frames automatically.

1.1.2 Calibration

During data collection, we ask the subjects to look at known points in the scene and move their heads in fixed incremental amounts. In this way their gaze traverses the full range of eye motion. The points from this calibration sequence are then used to provide the points required to form a calibration.

By forming a correspondence between the two videos we are able to estimate where the subject was looking in the scene at every point in time with the temporal resolution limited by the frame rate of the video.

The correspondence between the location of the eye-in-head to the location of the gaze-in-world can only be formed if the location of the pupil in the eye is correctly identified in every frame of the video of the eye.

To form this correspondence, it is necessary to determine the center of the pupil in every frame of the video in which the eye is opened.
1.1.3 Mobile Eye Tracking Requirements

For indoor eye tracking where the illumination is stable, and when the light incident on the cornea of the eye is controlled by a voltage regulator, eye-tracking is a solved problem. One can purchase or make an eye tracker that correctly tracks the pupil location [10–12].

One requirement for commercial eye trackers is that the lighting (illumination) is controlled. This requirement breaks down in many situations, such as:

1. when the subject walks up to a window,
2. if there are moving lights in the scene surrounding the eye (such as when driving), or
3. if the subject goes outdoors in bright sunlight.

Bright sunlight is problematic outdoors even if the subject is wearing a large hat to shade their face, because as the subject’s head rotates from facing towards the sun to looking away from the sun the amount of light reflected onto the face can change by several orders of magnitude.

The autoexposure algorithm in the video camera attempts to compensate for changes in the amount of light, but this algorithm always lags the changes. Furthermore, as the camera’s exposure algorithm adjusts the exposure it also adjusts gain and contrast controls which in turn changes the global histogram for the entire image of the eye on a frame-by-frame basis.

Unfortunately, the system records no meta-data about how much contrast or exposure compensation was applied. We cannot invert the contrast adjustment or the exposure compensation that was applied to each frame.

1.2 Data Inundation Problems and Issues

The most common technique these days for eye-tracking is video-based recording of the eye motions [13]. Yet, this advancement in the ability to record the data leads to all the issues associated with large amounts of data.

The study becomes inundated with data and must use data reduction techniques to reduce the data to a manageable amount.

The following subsections discuss some of these issues, and the need to efficiently process data.

1.2.1 Data Inundation in Frames

As previously stated, the Multidisciplinary Vision Research Lab (MVRL), in collaboration with the University of Rochester, ran a multi-year NSF study tracking the eyes and gaze
patterns of geology students and experts. During this study 12 subjects are eye tracked for about 2 hours a day, over 10 days (2011) [6]. The trip returns with roughly half a terabyte of data to analyze.

Over the course of the ten days, the number of video frames collected can be estimated by making some rough assumptions:

- 2 videos per subject,
- times 29.97 frames per second per video,
- times 60 seconds per minute,
- times 60 minutes per hour,
- times 2 hours per day,
- times 12 subjects tracked per day,
- times 10 days of tracking

The product of these factors is about $5.2 \times 10^7$ video frames each year. The study spans multiple years, so the maximum number of video frames is potentially higher.

### 1.2.2 Data Inundation in Pixels

Section 1.2.1 discussed the number of frames that must be analyzed by a human looking at each frame of the eye video. Here we consider this number of frames in terms of data.

If each frame was only 240 rows and 320 columns and monochrome, that would be 76,800 pixels per frame. In terms of uncompressed data, for the 26 million frames, this amounts to 3.7 terabytes of data that must be processed for the eye videos alone.

This is a tremendous amount of data to contend with, both from the point of view of the analyst looking at the videos, and from the point of view of a computer that must read, process, and annotate the data.

### 1.2.3 Yarbus Software

In our eye tracking work flow, the process of locating the subject’s eye in every video of the eye is usually handled by the Positive Science® eye-tracking software called Yarbus®. This software works well for standard, uniform, office lighting situations. However, it breaks down when the subject takes the mobile eye-tracking headgear outside.

The Yarbus software encounters problems if:
CH. 1. INTRODUCTION

- The subject goes outdoors, where the illumination on the eye is not controlled only by the illumination from the Positive Science® eye tracker. Generally, an infrared LED (IRED) is used by the eye tracker to illuminate the eye. A subsequent section will compute the amount of illumination caused by reflected background lighting when eye tracking outdoors in bright sunlight (See Appendix C).

- The illumination of the eye changes quickly, as when the subject rotates his/her head quickly, or when driving in dim light in the face of on-coming car headlights.

- The subject’s eyelashes go in front of the pupil. Eyelashes in front of the pupil change the local photometric value of that region of the image, and occlude the features of the pupil. The local pixel values become either lighter or darker.

1.2.4 Data Reduction Techniques for Analysis

While each observer might be eye tracked for two hours per day, it is often the case that a few critical minutes of the video data can be identified as most important to process. Once the critical data has been identified, those sections of eye-and-scene video data are the only sections that need to have the eye movements and fixations precisely analyzed. The analyst’s job is simplified by first identifying and isolating these sections of the eye tracking videos.

The identification of the sequences of interest is one method of data reduction. Typically for each subject the critical frames include a sequence that is used for calibration, and another sequence during which the subject is asked to do a visual search or visual analysis of the scene.

As another data reduction technique, the subject’s fixations are computed. These fixations are the times at which the eye is stable with respect to the world. At these times the image of the world around the subject is stable on the retina. However, fixation analysis cannot take place until the pupil is located in each frame.

The data reduction techniques mentioned here are procedural techniques, used by analysts processing the video to reduce the number of frames that need to be processed. Other techniques are used in the algorithms to reduce the amount of memory necessary.

1.2.5 Pupil Location

Before calibration can occur the pupil center must be located in the eye video.

For indoor videos this is done using automated software and an algorithm that has several controls for the analyst. This software includes histogram analysis, an optional feature selection step, and the heuristic that pixels below a set threshold should be considered pupil pixels. Videos that come from outdoor eye tracking studies may not work with the threshold assumed.
However, current pupil location systems fail when used under challenging lighting conditions, which occur as soon as the subject steps up to a bright window. In such situations, the standard assumptions fail. For example, the pupil is not the darkest location in the eye image, the corners of the eyes are. As a result, pupil location techniques that use only the pixel value fail.

Other techniques do not use the pixel value itself, but the relative pixel value in the region. An easy example to consider is the locally darker edges that indicate a pupil. However, edges themselves are unreliable because edges detected in the eye video are not always caused by the eye itself. Edges can be caused by the world being reflected off of the cornea of eye.

The current work-flow for locating pupils in each frame of a subject that has been outside involves having students step through the video frame-by-frame, using custom software, and identify the pupil location and size in each frame.

Analysts can do this task because the human visual system (HVS) automatically performs background subtraction. A trained human vision system automatically ignores things that are not relevant to the task. This top-down aspect to the HVS lets us ignore skin surrounding the eyes, sclera (white of the eyes), eyebrows, the corners of the eyes, eyelids, eyelashes, stray hairs in front of the camera, and reflections off of the eye.

In this thesis we will mimic several of these aspects of the HVS in order to better recognize and fit the pupil locations. We draw inspiration from how the human vision system works. The intention is to use computer analysis to increase processing throughput, decrease uncertainty in the data, and reduce the data analysis effort specifically for natural outdoor images.

1.3 Processing Stages

1.3.1 Automated Approach

From the start, it was known that detecting the pupils of subjects under these circumstances would be difficult.

The initial design was to use several passes over the video streams to capture the pupils. The idea was to have an early pass identify the most reliable and stable pupils automatically. The knowledge of where these reliable pupils were would be used to learn the parameters of the subject’s motion, and bootstrap the detection of the rest of the pupils in a subsequent pass. As a final pass, the user would validate the pupils.

The original sequence of processing steps included:

1. Statistical collection on all frames, to identify stable pupils for detection.

2. An initial pupil detection on the most stable frames.
3. Parameter learning of the subject’s range of eye motion, and variations across the video sequence.

4. Subsequent processing of the frames to identify the less stable pupils.

5. Final user validation of the pupils.

This was implemented, and shown to work for several subjects. Unfortunately subject-to-subject variations caused the automated methods to work for some subjects but fail for others.

Since the subject-to-subject variations were too large for a fully automated solution to work reliably on all subjects, a method to determine the pertinent parameters for each subject was needed. We achieved this through the use of semi-supervised feedback.

1.3.2 Use of Semi-Supervised Learning

To collect ground truth data from the first pupil detection stage, a validation pass was created. This validation allowed the initial set of pupils to be used as ground truth for discovering the parameters and range of variation for subsequent pupil detection.

The decision to validate the first set of pupils was pivotal in getting the program to work reliably for multiple subjects. The pupils from the most stable frames provided subject-specific information about the changes in edge strength, pupil darkness, surrounding iris brightness, as well as the angle of rotation and eccentricity of the pupil ellipses as a function of position in the image.

Using the validated ground-truth pupils from the first pass, the parameters of the eye model could be found independently instead of using an optimization method to try to solve for both simultaneously. The availability of this ground truth also provided several opportunities for machine learning and parameter modeling.

Figure 1.1 shows the main stages for identifying the pupil. These stages involve:

1. Statistical collection on all frames. These statistics are used to identify stable pupils for detection. The stable frames serve as a sampling of the full video, in such a way as to avoid closed eyes and eyes in motion.

2. An initial pupil detection on these most stable frames. This identifies the most likely pupil locations in each of these frames.

3. Validation of these detected pupils. The user checks the located pupils to assure that they were located correctly.

4. Construction of a model of the subject’s range of eye motion and variations across the video sequence, using the validated frames. This model is used to search for pupils in the remaining video frames.
5. Final user validation of the pupils. In this phase the user has the chance to examine all of the detected pupil position.

![Diagram of processing stages]

Figure 1.1: The main stages of processing for pupil location.

### 1.4 Innovations

The following chapters catalog the methods developed, and advancements that were enabled by using a semi-supervised approach. We briefly list them here.
The core routine used to identify and locate the pupil is a circular Hough Transform, a circle detector. The Hough Transform was optimized for pupil detection as described in Ch. 5.

However, the Hough Transform relies on correctly detecting the pupil-to-iris edges. Especially when trying to detect small pupils, it is thrown off by other edges in the image. Any techniques that can remove non-pupil-to-iris edges improve the accuracy and detection rate for the Hough Transform.

To improve the reliability of the Hough Transform, noise cleaning and image smoothing are used. This noise cleaning includes removing the eyelids around the eye, using a similar technique similar to that of Masek [14]. The details of eyelid removal are described in Ch. 3.

The presence of other edges that reflect off the first surface of the cornea of the eye cause additional false positives in the Hough detector. Noise from compression artifacts is also an issue in these images. To remove this noise an adaptive noise removal technique was developed which takes advantage of the structure of the eye.

This pixel-by-pixel noise cleaning technique involves an edge-preserving noise smoothing algorithm, based on Kuwahara’s work [15]. The developed algorithm is several times faster than a bilateral filter, and has one quarter the false alarms as the bilateral filter. This process is described in Ch. 4.

The initially detected pupils are used to develop a model of the subject’s eye. This model is used to apply a non-linear subject-specific transformation to the eye image which converts elliptical pupils into circles. This technique, called unwrapping the eye, is described in Ch. 6.

That same model of the subject’s eye allows the program to identify pupil ellipses that would never occur on a real human, and automatically reject them. Some pupil ellipses are tilted the wrong way, and could not possibly be correct pupil detections of the subject’s eye. Using the model, these strange ellipses can be identified and rejected in a process called strange ellipse rejection, as described in Ch. 7. Strange ellipse rejection improves the accuracy and reliability of the entire program.

Using multiple features creates a higher dimensional space for analysis. The benefit for this is that the data becomes sparse. This sparsity can be taken advantage of by isolating the target points in feature space. The method for feature combination is described in Ch. 8.

Once the pupil location has been detected, an adaptive edge following scheme is used to follow the contours of the pupil. This technique, called progressive connectivity, is described in Ch. 9.

The program automatically identifies multiple candidate pupils per frame. Each candidate pupil is assigned a degree of confidence, a metric for how strong and clear the pupil is. This allows the pupils to be sorted automatically by the program. Of all the candidates, only the three with the top degree of confidence are retained. Furthermore,
the degree of confidence is used to sort the highest confidence pupils to the first position in the list of possible pupils. The degree of confidence is described in Ch. 10.

The overall system enables an order of magnitude improvement in processing speed when identifying pupils in outdoor images. Results are discussed in chapter 12, future work is given in Ch. 13, and conclusions are given in Ch. 14.
Ch. 2

Workflow and Primary Passes

This chapter provides a high level overview of the primary passes used to locate and detect pupils in the eye videos.

Figure 2.1 shows a block diagram of the main stages, or passes, of the work flow. Each of these passes are described in the following sections.
2.1 Overview of Primary Processing Passes

When asked to find pupils in a frame of a video, a natural question is, “Why don’t you just use a circle detector?”

Circle detectors work very well in situations when the images are of coins taken on axis of the coin, with a simple contrasting background. Such images exist in assembly lines where the lighting is controlled and the background is a uniform contrasting color designed to simplify the computer vision task.

This real-world problem includes a considerable amount of pre-processing and noise removal to increase the accuracy of the Hough circle detector. Following the circle detection, post-processing is used to clean up the detected circles.
The program makes several passes over the data. In each pass select video frames are read in, examined, and processed. Not every frame of the video is read during every pass.

The following passes are used to locate the pupil in the frames of the eye video:

1. **Statistical collection pass.**
   An initial statistics collection pass to detect sudden changes (such as blinks) and regions of the video frame that can be ignored throughout the entire video.

2. **Sampling pass**
   An initial easy-pass samples the video frames that are likely to contain strong circular pupils without motion blur. This easy-pass includes masking off invariant pixels (Section 2.2.1), eyelid detection (Chapter 3), circle detection (Chapter 5), and edge refinement (Chapter 9). The result is an initial guess at the pupil locations for approximately \( \frac{1}{30} \) of the frames in the video.

   The Hough detector for this pass is fast and not as complicated as later stages. Pupils that are highly eccentric because the eye is looking away from the camera are missed. The expectation here is that the frames analyzed will be stable, with little motion, and the pupils will be close to on-axis.

3. **User validation pass**
   The user confirms or fixes the pupils detected in step 2. User validation creates known ground truth frames that the program can then build models from.

4. **Pupil edge modeling**
   The pupils that were confirmed to be valid are used to form models of the edge strength as a function of position.

5. **Geometric modeling**
   The program uses information from the pupils found previously to determine the eye size and center of rotation. A geometric model of the subject’s eye results. This determines how to unwrap the eye image so that the elliptical pupils in the eye image become more circular. Unwrapping is a non-linear image transformation that converts eccentric pupils into more circular pupils that can be detected by the circle detector.

6. **Automatic pupil detection pass**
   Using the models for how to unwrap the eye, an automatic pass is run over the video to find pupils in the frames that do not have pupils located in them yet. Complex processing, using the models developed during the previous steps, This results in three potential pupils per frame, which are sorted by decreasing degree of confidence.
7. **Final user validation**
   A final user-validation pass over the pupils allows the user to inspect and select from the program’s three candidate pupils. The three candidate pupils can be selected by the analyst with the push of a single button. The analyst can also refine the pupil location by manually clicking on five or more edge points around the pupil edge. A pupil ellipse is then fit through those points.

### 2.2 First Pass and Statistical Collection

The first pass over the data collects statistics about the frames of the video being processed. These statistics are used for several three reasons: 1. background removal, 2. blink detection, and 3. stable frame identification.

#### 2.2.1 Background Removal Statistics

During the first pass, two important statistics are collected over the entire video range specified. The first is an average frame over frames of the video being processed. This is an average value for every single pixel in that range of frames. The second statistic is the standard deviation of each pixel over that same range.

Pixels that have very low standard deviation are ignored. Pixels that do not change over the course of the video cannot contain information relevant to this task.

The result is a mask of pixels, outside of the eye, that can be ignored. The mask is specific to that subject in that video, how the eye camera is positioned for that trial, and how that subject moves his or her eyes.
for each frame do
    // Find the center region of the image, which gives the best signal: ;
    $V_{vt} =$ vector of center half of vertical pixels ;
    $V_{hz} =$ vector of middle 5/7 of horizontal pixels ;
    $C_{frame} = frame(V_{vt}, V_{hz}) ;$
    // Find the average and std of the center of the image itself: ;
    $A_{frame} = mean(C_{frame}) ;$
    $S_{frame} = std(C_{frame}) ;$
    // Find the magnitude of spatial edge differences: ;
    $E_{spatial} = abs(sobel\ edges) ;$
    // Find the temporal image difference: ;
    $E_{time} = A_{frame}(k) - A_{frame}(k - 1) ;$
    // Record the statistics on the edge strengths: ;
    $A_{e_{spatial}} = mean(E_{spatial}) ;$
    $A_{e_{time}} = mean(E_{time}) ;$
    $S_{e_{spatial}} = std(E_{spatial}) ;$
    $S_{e_{time}} = std(E_{time}) ;$
    // Compute histogram of these temporal differences: ;
    $H_{time} = Histogram(E_{time}) ;$
    // Find the absolute value of the change in the histogram: ;
    $H_{change} = sum(abs(H_{time}(k) - H_{time}(k - 1))) ;$
end

Figure 2.2: Computing statistics for frames in first pass

2.2.2 Blink Detection Statistics and Algorithm

During the same first pass, frame-by-frame statistics are collected. These statistics are used to predict which frames are blinks, and to find frames that are very stable.

During the blink of an eye, the closing of the eyelid causes the global brightness of the video frame to change suddenly. Eyelids are brighter than the pupil. When the eye is closed the eyelid hides the crevices, pupil, and shadows of the open eye from the view of the camera. When the eye is closed, the video frame contains fewer edges. The statistics collected to detect blinks are all measures related to the image brightness and the number and strength of the edges present.

Figure 2.2 gives the algorithm used to collect per-frame statistics. Spatial edges are computed, and statistics are collected on them. A temporal edge feature is also computed by computing a frame-to-frame difference, and statistics are computed on that.

The global illumination of the frame can change over the video. A sudden change in overall brightness could be mistaken for an eye closure.
The following per-frame processing is performed, and the resulting per-frame statistics are recorded.

Figure 2.3 gives the algorithm for blink detection. The frame brightnesses are normalized to a range of $[0, 1]$, so that video-to-video changes can still use the same algorithm. This normalization also compensates for exposure condition. The intuition here is that a frame is a blink if it is more than 0.20 above the running average over 15 frames. This means that the frame is suddenly much brighter than the local average. If the frame is in the top 70% of the brightness range, then it is probably a blink as well.

```plaintext
1 // Find the min and max brightnesses from average frame illumination:  
2 $B_{\text{min}} = \text{min}(A_{\text{frame}})$;  
3 $B_{\text{max}} = \text{max}(A_{\text{frame}})$;  
4 // Compute the normalized brightnesses. Values are in range $[0, 1]$:  
5 $B_{\text{norm}} = (A_{\text{frame}} - B_{\text{min}})/(B_{\text{max}} - B_{\text{min}})$;  
6 // Compute running avg over 15 frames: (half second)  
7 $B_{\text{avg}} = \text{sum}(B_{\text{norm}}(k-7, k+7))/15$;  
8 // Classify blinks:  
9 $\text{Blink} = (B_{\text{norm}} - B_{\text{avg}}) > 0.20 \cup (B_{\text{norm}} > 0.70)$;
```

Figure 2.3: Classifying blink frames

### 2.2.3 Stable Frame Identification

Along with the differences from the frames, the data that is collected is used to identify the most stable frames. The first pass uses these most stable frames as the best frames to try to detect pupils in because they are least likely to have fast eye motions.

The statistics from the initial pass (see Figure 2.2) are used to derive a measure of how much a frame differs from the norm for the video. We assume a multivariate Gaussian distribution for these variables, perform principal components analysis (PCA), and use the resulting eigenvectors as input to a Karhunen-Loève Transform (KLT) [16, p. 266] The KLT subtracts off the mean values and rotates the data to axially align it. The Mahalanobis distance from the norm is then computed [16, p. 25].

The result is a distance from the norm for every frame. This distance from the norm is used in the following passes to identify pupils that are most likely to contain an open eye. The more normal the frame is, the more stable it is.

Frames are processed in this pass if they are stable, and have not been identified as a blink using the blink detection algorithm of Section 2.2.2.
2.3 Easy-Pupil Detection Pass

Once the stable frames have been identified, they are analyzed to find the initial pupils. This easy-pass examines only a sample of the frames to identify likely strong circular pupils. Figure 2.4 diagrams the processing used during this pass.

The statistics collected are used to form a mask for removing pixels from consideration. The pixels with the lowest standard deviations are used to establish a mask of pixels that do not change throughout the video. This is called an invariant pixel mask. This mask is constant throughout and is used for every frame of the video.

In addition to the invariant pixel mask, an eyelid detector is used to find the upper and lower eyelids, as described in Chapter 3. The regions above the upper eyelid and below the lower eyelid are masked off and ignored.

Adaptive noise cleaning is used to remove edges that do not match the model of structured edges for pupils, described in Chapter 4.

All of this pre-processing is done to reduce the amount of work needed for the Hough circle detector, described in Chapter 5. The Hough circle detector for this pass is faster and not as complicated as later stages. It does not try to compensate for elliptical pupils. Pupils that are highly eccentric because the eye is looking away from the camera are missed. The expectation for the easy-pass is that the frames analyzed will be stable, with little motion, and that the pupils will not be far off axis.

An edge refinement and contour following scheme is used to refine the pupils detected, as described in Chapter 9.

From this the candidate pupils detected in each frame are ranked, based on a degree of confidence metric, as described in Chapter 10. The top three candidate pupils, with the highest degree of confidence are retained for consideration by the user.

The result of this easy-pass is the three strongest candidate pupils for approximately 1/30th of the frames in the video. This sampling selects roughly 3% of the video frames, or approximately one frame every second.
2.4 First User Validation Pass

Given the pupils from the initial pass, the user can then inspect the pupils found to assure that they are correct. The result of this validation is labeled ground truth pupils.

The initial goal of this pass was to find the good pupils to use for building the geometric models of the eye.

In addition, labeling of the data also creates a two-class problem that can be used to automatically reject some incorrect pupils in later passes. (See Figure 2.5.)
2.4.1 Establishing Ground Truth Pupils

Each pupil has a degree of confidence (DOC) associated with it. The user can set a threshold for the range of DOC he or she is interested in. Because of this, pupils with a very high DOC can be automatically ignored and not examined. The user can elect to examine only those frames for which the pupils have a very low DOC.

During this first validation pass the top three pupils for each frame are presented to the user. The user can then select from these top three with the push of a single button. The user can also refine the pupil location to get a more accurate ellipse for the pupil position. Alternatively, the user could reclassify the frame as a blink.

Once the easy-pass has been run, and the user has validated the pupils in these frames, the program now has ground truth data to build models for the pupils in that video. These ground truth frames are used to develop models for edge strength and eye geometry.
2.4.2 Establishing Ground False Pupils

There is another benefit of this first user validation that was not initially obvious – it establishes known bad pupils.

Each stable frame that was analyzed retains the top three candidates for pupils that were found. Since only one of them was selected by the validation pass, the others two candidates are known to be wrong. The user validation pass establishes both correct and incorrect pupil locations. It labels the candidate pupils as being good or bad. This provides the data needed for a two class classification problem, one class is known to be good, and the other class is known to be bad.

When the process of validating the pupils from the easy-pass was started, it was not obvious that by not selecting the wrong pupils the incorrect class of pupils was being labeled implicitly.

Using the set of candidate pupils that were rejected, we convert the problem from only matching the best pupils, into the task of matching the best pupils and simultaneously avoiding the wrong pupils. This knowledge is used to automatically reject pupils that are the wrong size for their location in the frame, or that are rotated incorrectly for pupils in the eye frame. These pupils that can be identified as the wrong size or the wrong rotation are termed strange pupils. The process of strange pupil rejection is described in Chapter 7.

2.5 Modeling

2.5.1 Edge Strength Modeling

Once the edges of the ground truth pupils are identified, the minimum and maximum edge strengths can be modeled. The ground truth pupil locations provide known pupil edge locations, and from these the minimum and maximum edge strengths can be extracted for each position, across all the video frames that contain the ground truth information. Using these known minima and and maxima, maps are created for the weakest and strongest edges within 100 pixels of any location in the frame.

The resulting edge strength maps are used in the automatic pass, to automatically remove edges that too weak or too strong to be associated with a pupil.

This automatically removes some edges, such as those caused by the subject wearing glasses, because the edges are too weak to be pupil edges. The edge strength models are also used to remove edges that are too strong, such as those caused by strong first surface reflections off of the cornea.

2.5.2 Eye Geometry Modeling

The ground truth pupils are also used to find the center of rotation of the subject’s eye, and the relative size of the subject’s eye. The unwrapping transformation is described in
Chapter 6.

The two key parameters for the model is the \((x, y)\) location of the center of rotation of the subject’s eye in the video frame, and the approximate size of the subject’s eye. These parameters are derived from the ground truth pupils collected from the first user validation pass.

2.6 Automatic Pupil Detection Pass

A block diagram of the automatic pupil pass is shown in Figure 2.6.

Once the parameters of the eye have been discovered, this second pass over the video is used to fill in the pupils for frames that were not previously analyzed.

The first pupil detection pass samples only one 1 of 30 pupil frames, or roughly one frame per second. This automatic pass is responsible for finding pupils in 29 out of 30 frames, or approximately 97% of the frames.

Since most of the frames are analyzed by this automatic pass, extra steps are used to try to assure robust automatic processing. This pass takes advantage of the previous models of the subject’s eye geometry, and the parameters that are used to identify and automatically reject strange pupil ellipses.

As with the easy-pass, this pass uses adaptive noise cleaning, and contour following to refine the pupil edges. Candidate pupil ellipses found are assigned a degree of confidence (DOC) measure.

However, before the pupil is located using a circle detector, the image of the eye is first unwrapped. A Hough circle detector is then used to locate the pupil in the unwrapped image. Because of the importance of this automatic pass, the Hough circle used modified to allow it to detect more eccentric circles in addition to nearly circular pupils. This adds an additional second per frame for detection, and adds additional false positives, but assures that the pupil is not missed.

Once located, the candidate pupils are then re-wrapped into ellipse locations in the original image. Chapter 6 discusses unwrapping and re-wrapping of the eye image further.
2.7 Final Inspection

After any pass, the user can inspect and modify the pupils located. This task is very similar to the routine for user verification described in Section 2.4. For the user, this inspection is the majority of the work. The previous pass over the data automatically identified three pupil candidates in every frame. The goal of that work was to locate the pupil correctly. In this final inspection, the user is responsible for verifying that the correct pupil of the three was selected automatically. If the program selects the correct pupil as one of the top three candidates, then the user can hit a button to select it automatically, and proceed to the next frame.

Alternatively, if the program found three incorrect pupil candidates, the user can manually identify the pupil by entering at least five points on the border of the pupil. Using these points, the program fits the pupil ellipse.

2.8 Summary

This chapter describes the primary passes that occur over the video data at a high level.
The initial pass collects statistics that are used to guide the rest of the processing passes, and to ignore pixels in the video which do not change over the course of the video.

The initial statistics are also used to identify frames that have little frame-to-frame change, the frames that are deemed most stable.

An initial easy-pass examines these most stable frames to find pupil candidates. These frames constitute about three percent of the frames in the video. The processing used for the initial pass is simpler, and more error prone, than the final automatic pass because it does not know the eye geometry of the subject. The user must inspect the output of this easy-pass, and correct any mistakes.

The frames that result from validating the results of the easy-pass are used to build subsequent models of the eye geometry, and to establish the limits for the maximum and minimum pupil edge strengths as a function of pixel location throughout the entire video.

The automatic pass then analyzes the remaining 97 percent of the video frames, using those models. This automatic pass includes a non-linear geometric transformation that converts the ellipse of the pupil into a circle so that a circle detector can be used to find the pupil ellipse. This operation avoids the need to use a complicated ellipse detector. Compared to an ellipse detector, the circle detector uses less memory and greatly reduces the number of parameter combinations that must be considered.

A final user inspection is performed, during which the user can inspect pupils that had a low degree of confidence. During this inspection the user can select any of the three pupil ellipses using a single key press, or enter an entirely new pupil ellipse if necessary.

Though this process, the user has some manual interaction upfront, during the validation of the first three percent of the frames that come out of the easy-pass. The knowledge of these pupils and their geometry is used to form models that allow the program to then more accurately find eccentric pupil ellipses that occur throughout the entire video. The automatic pupil detection pass runs in batch mode and analyzes the vast majority of the frames. Finally, the user inspects the final automatically located pupils for quality assurance.
3.1 Overview

In this chapter we describe the process of finding the eyelids and removing the uninteresting image regions from consideration by the pupil detector. These regions include the image above the top eyelid and below the bottom eyelid.

Regions outside of the two detected eyelids are masked off, and removed from consideration by the pupil detector.

Figure 3.1 shows an example of the eyelid detection and masking results. With the upper and lower eyelids detected, a mask is formed to remove pixels above the upper eyelid and below the lower eyelids. This removes additional eyebrow hairs and eyelashes that could be a distraction for the Hough circle detector.
Figure 3.1: An example showing the upper and lower eyelids that were detected. The cyan and red lines on the left of Figure 3.1(b) show the intermediary calculations used to find the eyelids. The cyan and red horizontal lines are the rows of the lower and upper eyelids that were detected.

Removing the region of the image outside of the eyelids has two benefits. First, it removes sections of the image that include eyebrows, and eyelashes, and thus reduces regions of the image that could cause false alarms when looking for pupils. Second, it reduces the amount of the image that needs to be processed by the downstream processing.

We use an eyelid detection approach inspired by Masek’s description [14], but developed independently, to accommodate the fact that our subject’s eyes are not always opened. Masek had cooperative subjects, who held their eyes open and steady so that they could be recognized by the iris recognition system. We have subjects whose eyes are in motion, ideally not thinking about the fact that their eyes are being imaged, and whose eyes are often squinting in bright light and are sometimes closed during blinks.

It is important that the process be conservative so that it does not err on the side of accidentally masking off a region where the pupil could be located.

The approach consists of an edge detector, histogram projection, lower eyelid detection, upper eyelid detection, and use of some conservative association rules.

We cannot assume that the eye will be fully opened, and that the lower eyelid will be concave upwards. In situations where the light is bright outside, subjects squint their eyes. In situations where the camera is located below a subject’s eye, the camera is looking up at the eye and the lower eyelid is concave downwards.

We find the best horizontal line for both eyelids. Regions of the image outside of these two lines are masked off.
3.2 Edge Detection for Eyelid Location

For the purposes of eyelid detection, we seek to detect a relatively large edge transition from dark-to-light or from light-to-dark.

The shadows that form between the eyelids and the surface of the eye are used to detect the eyelid.

We use a $5 \times 5$ edge detector. Table 3.1 gives the filter coefficients used for detecting edges for eyelids. The coefficients shown are used to detect horizontal edges, and the transpose of this table is used to detect vertical edges. The edge detector shown in Table 3.1 computes two local averages over five horizontal rows. One of these averages is two pixels above the center pixel, and the other is two pixels below a central pixel. The difference of these two averages is used to estimate the local image gradient.

For this application, the images are roughly 320 pixels wide, and 240 pixels high. Since the eyelids in these images are relatively large, the standard Sobel edge detector provides edge detection on too small a scale for this application. Instead, a different edge detector is used to detect the edges of eyelids. This edge detector was developed to emphasize larger transitions between objects.

The usual Sobel edge detector is only a $3 \times 3$ detector. It finds skin wrinkles, eyelashes, and the MJPEG (image compression) block artifacts that occur in the image. These edges are distractions from the true eyelid edges we hope to isolate and remove.

This wider filter has stronger support for edge detection because it is computed over a larger region. This means that it is less likely to detect small edges (such as MJPEG artifacts). The edges we seek are the large edges of the eyelids. For a fixed image gradient (such as a ramp) the image will change more over the longer distance of the five pixels than the three pixel Sobel edge detector. The $5 \times 5$ filter is working on a stronger input signal than a Sobel edge detector would be. The larger filter is more likely to detect the eyelid-to-eye transition, while being less sensitive to changes that occur over only a few pixels.

Using a larger filter is sufficient for detecting these edges, at the cost of decreased edge localization. In terms of localization, the $5 \times 5$ filter results in a step response over four pixels while a Sobel edge filter gives a step response over only two pixels. So the edges detected are less specific to a given area.

We compensate for this decreased localization later on by using a peak detector after edge projection. The projected edges are averaging over a region of the image that is the length of the filter in each direction, plus one for a center pixel. This gives a value of 11 (5 pixels in each direction plus 1) for smoothing the response.
Edges are detected both horizontally and vertically, and the magnitude and angle of these edges is estimated for the pixels in the center half of the input image. Weak edges that have almost no edge strength and can occur in almost any direction. To compensate, only edges whose magnitude is in the top quartile are used for further computations. This assures that when considering angles, only strong edge angles are considered.

A problem when trying to detect eyelids is horizontal reflections off of the cornea. In some cases the horizontal boom of the eye-camera is detected as a strong horizontal edge, which can be confused as either eyelid. To help avoid false-positives for eyelid detection, a set of association rules is used to reduce the chance that the eyelid detectors accidentally masks off too much of the image.

3.3 Lower Eyelid Detection

To detect the lower eyelid we seek a horizontal row of the image that has a peak in the number of strong edges whose gradient is angled upwards. The following procedure is used.

Weak edges that have almost no edge strength can occur in almost any direction. Again, when detecting edges for the eyelid detection, only those edges whose magnitude is in the top quartile are used. All other edges are set to zero and ignored. This assures that when considering angles, only strong edge angles are considered – those edges most likely to be caused by eyelids.

For every line of the image, the total edge strength of edges whose gradient is angled between $60^\circ$ and $120^\circ$ is computed, for each and every line of the image. This collects edges over a span of $60^\circ$ that are angled at $90^\circ \pm 30^\circ$.

To ignore noise spikes, the local average over 11 lines of image is computed, and the lines that have the local maximum after smoothing are found. Each of these peaks found is considered a potential location for the lower eyelid.

The value of 11 is twice the filter width plus one to assure that the smoothing filter has a definitive center.

We require that the lower eyelid be at least 30% of the way down the eye image. Peaks
that do not satisfy this requirement are removed from consideration. This imposes the heuristic constraint that the lower eyelid may not be located in the top third of the image.

We also require that the number of edges found in the projected histogram reflect at least 7.5 percent of the image width. Image lines that have fewer than this amount of edges were found to be spurious, and are not strong enough for consideration as an eyelid. In terms of signal processing, this imposes a noise floor that both the lower lid and upper lid must exceed to be considered valid lids.

Of the remaining peaks the peak furthest down the image is used as the lower eyelid. If no peak is found, the bottom of the image is used as the location of the lower eyelid in order to assure that a region of the image is not accidentally removed.

The procedure for finding the lower eyelid is summarized in Figure 3.2.
Figure 3.2: Finding Lower Eyelid

1. Find the edges at every pixel, using the filter in table 3.1.
2. Find the associated angle at each pixel.
3. \( \text{edgeMinStr} \leftarrow 75 \text{ percent of range of edge strengths detected.} \)
4. All edge strengths under edgeMinStr are removed.
5. // Remove edges that are tilted the wrong way to be a lower lid edge:
6. Remove all edges that are angled over 120°.
7. Remove all edges that are angled under 60°.
8. // Form a horizontal projection by adding up all the edge strengths on each line.
9. \( \text{rowSum} = \sum (\text{edges in each row}) \)
10. // Find the peaks in this total – the lines with the most horizontal edges over any 9 rows:
11. \((\text{allMaxs allRows}) \leftarrow \text{findpeaks}(\text{rowSum}, \text{MINPEAKDISTANCE'}, 5)\)
12. // Reject all peak values that account for less than 7.5% of the width of the image.
13. // This is a lower bound, below which a peak is considered noise.
14. \(\text{rejectList} \leftarrow \text{allMaxs} \leq \text{imageWidth} \times 0.075\)
15. \(\text{allMaxs(} \text{rejectList} ) \leftarrow 0;\)
16. \(\text{allRows(} \text{rejectList} ) \leftarrow 0;\)
17. // Reject any peak values for the lower eyelid that are more than 30% of the way up the eye.
18. \(\text{rejectList} \leftarrow \text{allRows} \leq \text{imageHeight} \times 0.30\)
19. \(\text{allMaxs(} \text{rejectList} ) \leftarrow 0;\)
20. \(\text{allRows(} \text{rejectList} ) \leftarrow 0;\)
21. // best remaining row is tentatively the largest total of these edges:
22. // this is the line of the image with the most and strongest horizontal edges:
23. \(\text{bestRow} \leftarrow \text{row associated with maximum of allMaxs:}\)
24. // Guard against strong horizontal edges going through the image, such as the image of a horizon reflected off of the eye:
25. \(\text{if there is another peak value that is more than 50% of this peak value below this edge then}\)
26. \(\text{bestRow} \leftarrow \text{this lower row as a precaution}\)
27. // The image below the bestRow is considered the bottom eyelid, and removed.

### 3.4 Upper Eyelid Detection

Detection of the upper eyelid is accomplished using a similar approach, but leveraging the prior knowledge that the lower eyelid has already been detected.

Again, only edges in the top quartile of edge strength are used for eyelid detection.

The upper eyelid is more arched than the lower eyelid, so the more variation in the edge angle is allowed. For each line of the image, a count of the number of edges that are
angled between $-45^\circ$ and $-135^\circ$ is computed. This collects edges over a larger span of $90^\circ$ whose gradients are angled at $-90^\circ \pm 45^\circ$.

As with the lower eyelid, peaks of the projected values are computed. These peaks must correspond to a minimum amount of 7.5 percent of the image width to be considered a possible upper eyelid. If no peaks matching the criterion are detected, then the top of the image is used as the top eyelid by default.

Otherwise, for all peak values over the minimum threshold of edges found, the strongest peak is used as the initial position for the top eyelid.

We wish to move the peak as far down as reasonable to remove as much of the image as possible. After the initial strongest peak is found, other peaks that correspond to at least 75 percent of the maximum value are identified as contenders. Of these contenders, the one furthest down the image that is at least a minimum amount above the lower eyelid is used.

For subjects with an epicanthic fold (additional crease in the top eyelid) or who are squinting, the first location found for the upper eyelid corresponds to too much of the image. The process of using a lower contender helps eliminate an additional portion of the image which could contain eyelids or eyelashes.

To avoid situations caused by severe squinting we require a minimum opening between the eyelids. In this case that the distance from the lower eyelid to the upper eyelid be at least 20% of the height of the image. If the two eyelids are too close together then the subject is squinting, and we cannot be certain where to locate the upper eyelid, so the top of the image is used for the upper eyelid to be conservative.
The pseudocode for this algorithm is provided in Figure 3.3.

Figure 3.3: Finding Upper Eyelid

1. Find the edges at every pixel, using the filter in table 3.1.
2. Find the associated angle at each pixel.
3. \( \text{edgeMinStr} \leftarrow 75 \text{ percent of range of edge strengths detected.} \)
4. All edge strengths under edgeMinStr are removed.
5. // Remove edges that are tilted the wrong way to be an upper lid edge:
6. Remove all edges that are angled over \(-135^\circ\).
7. Remove all edges that are angled under \(-45^\circ\).
8. // Form a horizontal projection by adding up all the edge strengths on each line.
9. \( \text{rowSum} = \sum(\text{edges in each row}) \)
10. // Find the peaks in this total – the lines with the most horizontal edges over any 9 rows:
11. \( (\text{allMaxs allRows}) \leftarrow \text{findpeaks}(\text{rowSum}, '\text{MINPEAKDISTANCE}', 5) \)
12. // Reject all peak values that account for less than 7.5% of the width of the image.
13. // This is a lower bound, below which a peak is considered noise.
14. \( \text{rejectList} \leftarrow (\text{allMaxs} \leq \text{imageWidth} \times 0.075) || (\text{allRows} \geq \text{lowerLid} + 0.2 \times \text{imageHeight}) \)
15. \( \text{allMaxs( rejectList )} \leftarrow 0; \)
16. \( \text{allRows( rejectList )} \leftarrow 0; \)
17. // Find the initial largest collection of horizontal lines on any row:
18. \( (\text{bestMax bestRow}) \leftarrow \text{max of allMaxs and allRows.} \)
19. // Identify any other contenders for a lower location of the upper eyelid:
20. \( \text{contenders} \leftarrow \text{allMaxes} \geq \text{bestMax} \times 0.75 \text{ if any contender row} \geq \text{bestRow} \text{ then} \)
21. \( \quad \text{bestRow} \leftarrow \text{this lower row} \)
3.5 Example Results

Figure 3.4: An example showing the eye detection process. Figure 3.4(a) shows the input image.
Figure 3.4(b) The peaks of the number of edges on each line are graphed on the left hand side, showing the number of upper eyelid edges detected (red on the left), and the final position of upper eyelid detected (horizontal line, again in red).
It also shows the peaks in the lower eyelid detection (in cyan) and the position of the lower eyelid (horizontal line, also in cyan).
Figure 3.4(c) shows the input image with the upper and lower lids masked off.

Figure 3.5: Shows a second example of the eye detection process. Multiple peaks are seen for possible lower eyelid positions. The lowest peak is selected.
Figure 3.6: A third example showing the eye detection process. In this case there are two possible peaks to use for the top eyelid position. Since the lower peak has more edges accumulated, and has a larger peak, the lower of the two candidate peaks is used as the top eyelid position.

Figure 3.7: A fourth example showing the eye detection process. As the eye blinks, the method still works.

Figure 3.8: An example showing eyelid detection on a image with low contrast.
Figure 3.9: An example showing eyelid detection on an image with lens flare (an ugly image).

Figure 3.10: An example showing eyelid detection on an image when the eye is looking to the side.

Figure 3.11: An example showing eyelid detection on an image when the eye is looking to the side.
3.6 Numerical Results

The test set contains 1,140 frames selected from different subjects, different videos, and different studies. Some of the frames were selected by a random number generator. Others were included in the test suite because they represented cases that caused problems on previous versions of the eyelid detector.

The frames were manually inspected individually for problematic cases in which too much of the image is rejected because the eyelids were located in the part of the eye we wished to retain.

Of the 1,140 frames in the test suite the eyelid detector accidentally masks off image data from two cases. Two out of 1,140 is a 0.175 percent failure rate.

The two problematic frames are shown in Figure 3.12.

![Failure 01](image1.png)  ![Failure 02](image2.png)

(a) Failure 01 – LonePine 2013 S08 Frame 0264  (b) Failure 02 – LonePine 2013 S08 Frame 0265

Figure 3.12: Figure showing two frames from the test suite that failed the eyelid detection test. Close inspection shows the the lower eyelid was located unusually low, which allows the peaks from the top eyelid to shift to a position further down the image.

3.7 Conclusions about Eyelids

A method has been described for finding the lower, and then the upper, eyelids in an image of the eye for our subjects. The successful strategy was to locate the lower eyelid before locating the upper eyelid. If the lower eyelid is located too low, then the upper eyelid may also be located too low.

For our test suite, we found that only two frames in 1,140, or 0.175 percent, were problematic. The method can be thrown off by strong shadows across the eye, but selects conservatively to avoid accidentally masking off more of the image than desired.

This technique reduces the search region by removing regions outside of the two detected eyelids are removed from consideration. This removes eyebrows, creases in
the upper eyelid, and eyelashes. By removing these regions of the image the amount of
nuisance and noise edges that the Hough detector must contend is reduced.
Ch. 4

Adaptive Edge Preserving Noise Reduction for Pupils

4.1 Introduction

The image of the pupil is corrupted by first surface reflections off the cornea. These reflections cause edges that impede the process of detecting the pupil edges. The following method is used to reduce the gradient of distracting edges of reflections and small structures such as eyelashes, eyebrows, iris textures, and compression artifacts. The method simultaneously enhances the boundary between the pupil and the iris.

This processing is performed prior to using an edge-sensitive Hough circle detector. The approach can easily be adapted for other domains where image segmentation is desirable, or inverted for use when sub-pixel interpolation is desirable, such as: enhancing iris patterns and finger prints for biometric identification, following blood vessels and catheters in bioinformatics images, and even geospatial remote sensing. The fundamental idea is to examine regions around each pixel in terms of consistency, and to adaptively replace the pixel based on the most consistent region. This processing attenuates small first surface reflections and noise edges that are caused by fixed pattern noise, sensor noise, and compression artifacts.

4.2 Adaptive Processing

The Hough circle detector relies on edge information to find the pupil. It presumes that any edge provided to it is potentially a pupil edge. To improve the accuracy of the Hough circle detector, the adaptive noise removal process is used to remove or remove nuisance edges. Nuisance edges include eyelashes, reflections of objects in the eye, and the detailed structure of the edges surrounding the iris.
This adaptive processing runs on a per-pixel basis and removes edges that do not match the structure of a pupil edge. The technique was inspired by Kuwahara [15,17,18]. A dictionary of different shaped regions around each pixel are examined. The variance over each region is computed. The region with the lowest variance is interpreted as the region that is most similar to the pixel being considered.

Once the best region for interpolation is determined, the pixel is replaced with the average over that region.

The important principle here is that the regions are designed to either fit inside a pupil or outside a pupil, but not inside distracting objects. Distracting objects include eyelashes that are typically only a few pixels across, and reflections of objects from the first surface reflection of the eye. Objects reflected off of the eye are shrunk because of the geometry of the eye (the cornea is convex).

This adaptive noise removals noise and unwanted edges before the circle detector is run. Figure 4.1 shows an example of an input and adaptively processed image.

![Figure 4.1: One of many examples of the adaptive noise cleaning being used to remove unwanted edges. For clarity of illustration, a front facing pupil was selected here. Notice that the texture in the iris and the eyelashes are removed, while the edges of the pupil remain.](image)

4.3 Background Importance of Data

Mobile eye tracking is used during psycho-physical studies to gain insights into what subjects are attending to. This works because vision is a demand-pull system. In order to find out information about what something looks like or where it is, people have to look at it.

The eye tracking apparatus uses two video cameras: one records the scene in front of the subject, while the other video camera records the subject’s eye position. A correspon-
dence is formed between the two videos, frame by frame. Using this we can relate the position of the eye in a frame of the eye video to where the subject is looking in the scene. Knowledge of where people look and how they look can be helpful for training and education. It can be used to determine what subjects see, or fail to see.

When viewing a pitched baseball, novice baseball players fixate on the pitcher while experienced baseball players fix their eyes on the anticipated release point of the ball [19]. Similarly, when novice drivers start driving they drive slowly and look just past the front of the car. Experienced drivers make anticipatory fixations 0.75 to 1 second into the future to predict where the car needs to be in the lane, and use fixations that are closer (0.5 seconds) to stabilize the position of the vehicle on the road [20–22]. Hayhoe et. al.’s findings from a driving simulation indicate that viewers learn active search patterns to maintain attention and control of the vehicle [20, 23].

Kasarskis et. al. found that under visual flight rules (VFR) expert pilots moved their eyes more frequently, and had more fixations and shorter fixations compared to novice pilots [24]. Expert pilots had a stronger and more well-defined scan pattern, which was more efficient for maintaining airspeed and accurate landings.

For our study we wished to study how geology students learn about the San Andreas Fault, running from the Pacific Ocean to Death Valley in the Mohave dessert [6]. Because of the amount of light on the dessert, the world is reflected off the first surface of the eye, the cornea.

In their work, Theory of Edge Detection, Marr and Hildreth [25], describe edges as resulting from either: 1) changes in illumination, such as shadows and light sources, 2) changes in surface reflectance, or 3) changes in orientation of a surface. The first surface reflections of the world off the cornea cause edges that are a combination of these, but primarily the first two.

The edges of the pupil are really edges between the iris of the eye, and the opening to the rest of the eye. The pupil is not an object that absorbs light, it is the opening that allows light through the iris into the inside of the eye – the optics and photoreceptive cells.

Here we describe an algorithm to find pupil edges in uncontrolled lighting, in the presence of first surface reflections of the world off the cornea.

Appendix C computes an approximation to the amount of ambient light present in full daylight during our study. Appendix D discusses the reflective properties of the cornea of the eye.

At this time, having collected data from 2010 to 2013, we have accumulated a large existing database of videos captured at low spatial and temporal sampling rates from which we wish to extract the location of subjects’ pupils in these videos as automatically as possible.
4.3.1 Pupil Detection Approaches

Initially we tried the obvious approaches using a blob tracker to locate the darkest region in the image. Unfortunately, the pupil is not the darkest region of the eye when working outside; the lacrimal caruncle (the main tear duct on the nasal side of the eye) is very often the darkest region. In other frames, the outside corner of the eye is the darkest region.

Additionally, the pupil is not a perfect circle. When constricted, the pupil shape is determined by the stress in the sphincter muscle that causes it to constrict. As a result the pupil can become non-circular and may not even be convex in places around the pupil perimeter (see fig. 4.2).

![Figure 4.2: Due to the muscles around the pupil, a tightly constricted pupil can appear non-circular and may not even be entirely convex. (From our data set, 2010, subject number 11 SFC, frame 747.)](image)

Even if the uneven stress was not an issue, the pupil would only appear circular in the eye camera if the pupil were looking straight at the camera. Since the eye camera is positioned below the line of sight of the subject, this is rarely the case.

Some techniques optimize the pupil location with the goal of every pixel in the pupil, even for highly irregular pupils [26]. For our purposes we wish to find the the center of the pupil ellipse, and slight irregularities in the edge are not problematic as long as the center that is found is consistent frame to frame.

A straightforward Hough circle detector works well for situations when the eye is looking close to the optical axis of the camera because the pupil is nearly circular.
However, to compensate for the issues of pupils that appear elliptical, we developed a modified Hough circle detector which is more tolerant of eccentric and imperfect circles, and procedures for removing extraneous unwanted edges which pupil detection.

Nevertheless, experiments using just a Hough circle detector still miss approximately one in five pupils. In an experiment the computer was told to try to find a pupil in every frame using the easy-pass. This found only 792 pupils out of 1,001. The missed pupils were mostly too eccentric to be detected.

4.4 A Taxonomy of Edges

Understanding edges is important in this context because we want to enhance some types while obliterating other types. We classify edges into three basic categories, based on the objects in the scene that cause them. In some cases, the difference between these is context or resolution dependent.

1. Boundaries.
   These are edges caused by transitions from one object to another. Boundaries are characterized by a large consistent transition between two levels: darker to lighter, lighter to darker, or consistent color transitions in color imaging.

2. Lines.
   These are caused by a long thin object that is in front of another object. In our case, eyelashes that are in front of pupils are lines. The radial striations of the iris of the eye is another source of lines.

3. Peaks and Valleys.
   These are caused by textures of a single object. For example the skin texture has ridges where the skin is higher and valleys where it is lower. These can cause locally high or low pixel values in images, and depends on the direction of illumination.

In some cases, the difference between these is context or resolution dependent. For remote sensed images, telephone wires and power lines are lines, as are the white lines on the edges of roads. At closer distances or higher resolutions, those same objects are imaged as boundaries.

The consequence of this is that no single standard edge detection method applies to all situations. For this application, the most important type of edge is the boundary between the pupil and the iris. We wish to develop an algorithm to enhance the boundary edges for the pupil/iris transition, while attenuating the edge strength of all other edge types.
4.5 The Resolution Challenge

A general rule of computer vision is to use the highest quality, highest resolution camera possible, so that the signal to noise ratio is as high as possible. Another rule is to control the background that you are trying to differentiate from the foreground as much as possible – keeping the background simple and free of distracting objects. Furthermore, it is beneficial to control the amount, direction, and spectral content of all light sources so that they consistently maximize the contrast between the foreground object and the background.

Unfortunately, for portable eye tracking studies, the subjects should be free to explore the world and the eye tracking gear should inhibit them as little as possible. This means using the smallest, lightest camera possible so that it does not interfere with a subject’s field of view. Small size is also important because it does not add any weight that the subject is aware of, facilitating rapid habituation to the equipment. Subjects in mobile eye-tracking studies must be able to move around the world, ride elevators, and climb stairs. The importance of image quality is secondary to the importance of small size for eye tracking.

For our data a fifth generation Positive Science® eye tracker was used. The resolution of the eye camera was limited to 240 images lines and 320 pixels per line. The eye camera is set to quarter VGA, with interlaced fields.

4.6 Noise Sources

Whenever attempting to remove noise from an image, it is important to have an understanding of the nature of the noise sources. It is also beneficial to remember how the output of the noise cleaning stage will be used. In our case the edges of the resulting process will be used in the Hough detector.

It is common in some literature to see salt-and-pepper noise (also known as impulse noise or shot noise) artificially added to images and then show that it can be removed [27]. Salt-and-pepper noise consists of pixels that take on the maximum or minimum pixel value. This was problematic when information was exchanged using analog radios between satellites. It is no longer as prevalent with the use of fiber optics cables, digital transmission, and error correcting codes. It does occur on sensors either due to dead pixels (always dark) or hot pixels (always bright), which hardly occur in our data. Noise removal techniques developed to remove salt-and-pepper noise are not appropriate for our application.
4.6.1 Structured Noise

Structured noise is any correlated noise. It may be correlated with itself spatially, or with the signal level.

In situations when the autoexposure algorithms cause the eye image to become underexposed (See Section 4.6.5), fixed-pattern sensor noise becomes apparent in the image. This noise forms a pattern aligned with the rows and/or columns of the image and exhibits strong spatial correlation.

The eye is illuminated by an infrared emitting diode (IRED). On a bright day in Death Valley, the spectral content of the ambient illumination that reflects off the scenery and onto the first surface of the eye exceeds the amount of illumination from the IRED. We estimate this excess to be roughly nine times the strength of the IRED (See Section C.)

Any reflections off of the air/cornea interface contains edges from the scene. The Hough detector has to sort through these edges to find the edges that are associated with the pupil.

For pupil detection, any non-pupil edges are noise. These include: the limbus of the eye, the iris texture, eye lashes, eyebrow hairs and compression artifacts.

4.6.2 Compression Artifacts

By the time we capture the videos in our system, each frame has been through a an image compression process. This causes a loss of image fidelity and some quantization artifacts.

Each frame is compressed using a spatial compression technique, similar to MPEG, which introduces blocking artifacts in the image. In some situations these blocking artifacts cause strong enough edges that the ellipse detection routine would use the edges of the blocks as input.

Figure 4.3 shows an example eye image that has been enhanced so that the noise artifacts are evident to the reader. The image was put through saw-tooth contrast enhancement to emphasize the contrast across pixel values. Specifically, the image pixel values were multiplied by four, and then values over one were mapped back into the range of [0 to 1]. The image size was magnified by a factor of eight using pixel replication to preserve visible edges. This processing emphasizes the compression and blocking artifact so that they are visible here, otherwise the display and print versions make it difficult to immediately see the compression artifacts.

Compression artifacts are rarely visible to the eye, yet are easily detected by standard edge detectors. Consequently, compression artifacts are a source of noise to the Hough detector.
4.6.3 Variations Across Subjects

When the eye tracking headgear is set up on the subject, an attempt is made to focus the eye camera on the subject’s eye. The sharpness of this focus changes from subject to subject and from experiment to experiment, even when using the same subject and camera. This is attributed to accidental changes in the distance between the camera and the subject’s eye. In addition to focus changes from subject to subject, and from day to day, the rate that the focus changes across the image differs from subject to subject if the eye-to-sensor distance changes due to the fixed depth of focus of the camera.

Some subjects wear contact lenses, which have a nominal index of refraction of 1.42, compared to 1.377 for the cornea [28], and being particularly smooth, cause sharper reflections of the world off the first air/lens surface. Contact lenses have an edge around the iris, which causes an additional unwanted edge.

Subjects that wear glasses also cause subject-to-subject variations. To track subjects with glasses on, the eye camera is adjusted to a lower angle so that the camera views the eye from a low angle underneath the glasses. A section of the glasses is cut away to allow the camera a low view of the eye, but the low camera angle makes the pupil more
eccentric, and the pupil is more often hidden behind the lower eyelid.

Some subjects have a very dark limbus around their iris, while others do not. This dark limbus causes edges that must be separated from valid pupil edges.

Our novice geologists are drawn from college students, graduate students, post-docs, professional geologists, and professors. Younger subjects can constrict their pupils more than older subjects, so that even for the same amount of light at the same location there is subject-to-subject size variation in pupil size [29].

Pires [30] notes that the pupil appears in the camera as an ellipse, but that it is a circle on the “surface” of the eye. In bright daylight the severely constricted pupil does not constrict uniformly [31], (see fig. 4.2).

All of these variations in the size and shape of pupils and changes in image quality are issues.

4.6.4 Within Video Variations

Within a single video from our study, the geographic location and subject is constant, but variations still remain. Daylight is strong enough that it reflects off of the skin surrounding the eye, causing unwanted specular reflections.

When the subject changes the direction of their head, sudden changes in the brightness of the skin surrounding the eye causes a lag in the auto-exposure and auto-contrast algorithms in the camera. The pupil size also changes due to changes in environmental brightness when a subject simply turns towards or away from the sun. Strong shadows can be cast onto the surface of the eye, causing unwanted edges.

The corneal refraction also causes the pupil to change shape depending on the angle the eye is viewed. The more oblique the viewing angle, the more eccentric the pupil becomes.

4.6.5 Sudden Changes in Illumination

The camera includes a basic auto calibration algorithm for: auto exposure, auto contrast, and auto color adjustment (“3A”). These algorithms adjust to slow changes, but are thrown off by sudden changes in illumination. For example, when subjects blink, the amount of light reflecting off the eye region goes up because of the skin of the upper eyelid. The auto-exposure compensates by collecting less light, and boosting the contrast. When the subject then re-opens their eyes, the pupil becomes visible again, but the camera’s auto-exposure is under-exposing the pupil, and the camera’s auto-contrast is boosting the contrast so much that it is difficult to discriminate the pupil from the iris – because the high contrast makes both the pupil and the iris dark.
4.6.6 Additional Noise Edge Sources

For the Hough pupil detector, the texture of the iris itself is noise. This texture includes the radial striations of the iris, as well as the crypts, pigmented spots, radial furrows, ciliary zone, concentric furrows, and other structures of the iris such as the collarette [32].

The eye is constantly in motion. Motion blur helps remove some textured noise from the iris, but also smears the edges of the pupil in the direction of motion.

4.7 Standard Noise Removal Methods

Traditional noise removal considers “noise” as either a high frequency effect (sudden fluctuations), or impulse noise, such as salt-and-pepper noise. Standard noise techniques are limited to a region of spectral frequencies, or values in a local region of the image.

Some noise sources come with the application, while others are per-sensor and must be detected on each fabricated sensor for appropriate removal. This detection is usually done before a device is shipped, such as a camera or satellite sensor.

Before describing the details of the proposed technique, we discuss some standard noise cleaning techniques. Here we discuss Gaussian filtering, median filtering, and compare these two techniques to the method used.

4.7.1 Gaussian Filtering

The most traditional method for noise removal is Gaussian filtering [33]. Gaussian filtering is a center weighted averaging filter, and involves setting two parameters: the values for the relative Gaussian weights as a function of distance, and a maximum distance over which the averaging is to be performed.

Gaussian filtering attenuates noise as each pixel is replaced by the center weighted local average of the surrounding pixels. This smoothing reduces outlying values, but at the cost of attenuating the signal as well as the noise.

In addition to attenuating the signal, it has the undesirable effect of spreading the noise over a local neighborhood. Bright specular reflections that appear in the pupil region raise the pixel values of any other pixels within the region neighboring them.

Gaussian filtering can also be considered a low-pass filter – a method for reducing the highest frequencies from an image. The amount that the high frequencies are attenuated depends on the parameter settings.

The result of Gaussian filtering is that the dynamic range of the pupil-to-iris edge is reduced which does not help with edge detection and localization for the Hough detector.

4.7.2 Median Filtering

Another typical approach to noise removal is median filtering [33].
A median filter also works on a local neighborhood around a pixel, considering each pixel in turn. All of the pixels in a neighborhood are sorted in order, and the center value from the sorted list is used to replace the current pixel. If a pixel is unusually bright, or unusually dark, it is replaced with a more moderate value that is drawn from the center value of the neighborhood.

This process removes unusually bright or dark pixels from each neighborhood region of the image.

The disadvantage of this is that if the border edge of the pupil iris boundary is frequently a transition from a locally darkest pixel to a locally brightest pixel. The effect of the median filter is to reduce the contrast of the pupil-to-iris border by replacing to darkest pixel by brighter pixels, and replacing the brightest pixels by darker pixels.

The result of median filtering is that the dynamic range of the pupil-to-iris edge is reduced which does not help with edge detection and localization for the Hough detector.

In some cases when the pupil is highly constricted, the median filter also reduces the size of the pupil in the resulting image. For pupil detection we wish to have as many pixels on target as possible.

### 4.7.3 Comparison to the Method Used

By comparison, the proposed method retains the border, and enhances the contrast across the iris-to-pupil edge.

The Table 4.1 of images shows columns of images. Each column consists of the original image, the result of Gaussian filtering image, the result of median filtering, and the adaptive noise removal method.

### 4.7.4 Morphological Noise Removal

Spatial noise is often handled using a filter of fixed spatial compute support, such as a median filter across a $N \times M$ neighborhood of the image. Here $N$ and $M$ are parameters that must be determined, tuned, and optimized. This is appropriate for finding “hot” or “dead” pixels.

Spatial noise can also be caused by artifacts of the sensor such as dead lines or dead columns. These cause vertical or horizontal lines through the entire image that must be detected and then handled.

Morphological processing can be used to remove some noise for which the spatial extent and statistics are fixed across the video. These techniques include: erosion, dilation, erosion then dilation, dilation then erosion, median filtering, the full range of ordinal filtering, and various permutations of combinations.

Like many disciplines, morphological processing has developed its own terminology. **Erosion** examines a range of pixels across a region around that pixel, and then replaces the pixel with the minimum pixel value in that region. In terms of **ordinal filtering** erosion
Table 4.1: Original image, Gaussian Blurred, and Median Filtered versions, and the method used.

<table>
<thead>
<tr>
<th>Input</th>
<th>Gaussian</th>
<th>Median</th>
<th>Method Used</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Image" /></td>
<td><img src="image2" alt="Gaussian Blurred" /></td>
<td><img src="image3" alt="Median Filtered" /></td>
<td><img src="image4" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image5" alt="Input Image" /></td>
<td><img src="image6" alt="Gaussian Blurred" /></td>
<td><img src="image7" alt="Median Filtered" /></td>
<td><img src="image8" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image9" alt="Input Image" /></td>
<td><img src="image10" alt="Gaussian Blurred" /></td>
<td><img src="image11" alt="Median Filtered" /></td>
<td><img src="image12" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image13" alt="Input Image" /></td>
<td><img src="image14" alt="Gaussian Blurred" /></td>
<td><img src="image15" alt="Median Filtered" /></td>
<td><img src="image16" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image17" alt="Input Image" /></td>
<td><img src="image18" alt="Gaussian Blurred" /></td>
<td><img src="image19" alt="Median Filtered" /></td>
<td><img src="image20" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image21" alt="Input Image" /></td>
<td><img src="image22" alt="Gaussian Blurred" /></td>
<td><img src="image23" alt="Median Filtered" /></td>
<td><img src="image24" alt="Method Used" /></td>
</tr>
<tr>
<td><img src="image25" alt="Input Image" /></td>
<td><img src="image26" alt="Gaussian Blurred" /></td>
<td><img src="image27" alt="Median Filtered" /></td>
<td><img src="image28" alt="Method Used" /></td>
</tr>
</tbody>
</table>
is replacing the pixel with the first sorted ordinal value, when the pixels in the region are sorted in ascending order. Dilation is similar, but replaces the pixel with the maximum of the sorted pixels, or the last ordinal in the region.

Erosion and dilation suffers if the lowest or highest pixel in a region is noise because that pixel can be propagated to surrounding regions. (Examples are shown in Section 4.9.2.)

As mentioned in Section 4.7.2 median filtering tries to avoid this issue by replacing the pixel with the middle pixel in a sorted region. This avoids sudden changes in pixel values, but also degrades the edge sharpness of the pupil edge that we are trying to detect.

### 4.7.5 Other Methods

Many techniques have been developed for noise cleaning images. Other techniques, such as unsharp masking [33, p. 162] amplify all edges and noise, regardless of structure. Having the noise amplified is detrimental. Unsharp masking also requires careful parameter tuning and selection to assure appropriate processing for the given lighting and contrast conditions of the image.

Bilateral filtering [34–36] is another technique for adaptive noise removal. Bilateral filtering includes a Gaussian distance weight to favor pixels that are closer to the pixel of interest. Bilateral filtering also includes a Gaussian level weight to favor pixels that are of similar pixel value. The combined weightings tend to favor averaging across pixels of similar value and location, while preserving edges by not averaging pixels of distant location or different pixel values.

As with unsharp masking, bilateral filtering requires careful parameter selection. Incorrect parameter selection will either remove the sought pupil edges or enhance the noise edges.

Bilateral filtering requires computing the weight normalization factor for all the weights on a pixel-by-pixel basis. This per-pixel processing significantly increases runtime, requiring over 20 seconds per frame for our images.

By comparison, with the same software and hardware system runs the following adaptive noise removal technique in runs in 1.2 seconds per frame, and requires no parameter tuning. The tuning is already included in the region shapes selected for the algorithm. Section 4.10 describes the performance optimization that makes this possible.
4.8 An Adaptive Noise Removal For Edge Detection

4.8.1 Leveraging Duality of Segmentation and Edge Detection

Duality

In image processing, segmentation and edge detection can be considered roughly dual problems. If you can find the best segmentation, you have the edges. If you can find the best edges, you can invert it to get an approximate segmentation. For our algorithm, we find the best segmentation for every single pixel, compute the segmented noise-cleaned image, and then use the edges to find the pupil.

4.9 An Adaptive Edge Preserving Filter

This technique was discovered in chapter five of Advanced Biomedical Image Analysis by Haidekker [18]. Haidekker cites Kuwahara et. al.’s 1976 paper [15] as the source of the method. Kuwahara et al. in turn attributes the idea to three other sources: Rosenfeld [37, 38], Newman and Dirilten [39], and Davis et al. [40].

Similar work on edge preserving filtering has been done by Nagao and Matsuyama [17]. The work has been extended by Yamashita et al. [41], Foo et al. [42], and Kyprianidis and Kang [43]. The specific algorithm that Haidekker described as Kuwahara filtering is also repeated in other publications [43–46].

The basic approach is to find the region around a pixel that is the most self-consistent and replace the pixel with a representative value from that region.

Haidekker’s diagram shows regions that do not include the pixel itself. Because we want the pixel to be as representative of the region as possible, we include the pixel in our calculations. We also improve the method specifically for pupil border enhancement by explicitly using region shapes that will only match the object we seek to enhance (pupils) while tending not to match any region that would match any distracting objects (eyelashes).

There are two possible positions for the pixel in a region. In one situation, the pixel is contained in the region considered, and a version that excludes the pixel in the region considered. In our version, we include the pixel in the region when evaluating the self-consistency.

We also use different sized regions, which introduces some complexity when deciding on the most self-consistent region. We handle this with a penalty function based on the number of elements in the region, as described in Section 4.10.3.

The following sub-sections describe our algorithm by explaining the traditional approach over only eight small regions, a discussion of possible measures of self-consistency, a discussion of possible measures of central tendency, a comparison to some morphological operations using a one dimensional signal, some important considerations when
selecting shapes, how the algorithm was actually implemented in an efficient manner, and the surprising unexpected side effect of this method that yields the most likely edges between the pupil/iris and iris/sclera.

4.9.1 Example Algorithm Using Eight Regions

We presume that every pixel consists of a mixture of signal and noise. To obtain an approximate measure of the true signal, we examine a spatial region around the pixel and use the average value of the region as the true signal.

If we just used a circular or Gaussian weighted region around the pixel, we would be performing block averaging or Gaussian weighting. Neither of these methods account for the presence of edges, and both methods attenuate edge strengths.

Here we examine several regions around that pixel, and choose the one that is least likely to contain an edge. Figure 4.4 shows a center pixel, marked with an ‘x’ and eight different regions surrounding it.

![Figure 4.4: Kuwahara filtering on a small 3×3 region. The center pixel is replace by a representative version of one of the surrounding regions being considered. Note that these are not the regions we use for our implementation and are only shown here for instructional purposes. After Kuwahara et al. [15] These regions have the nice feature that each region includes the same number of pixels. Our implementation has different numbers of pixels per region.](image)

The basic algorithm is to retrieve all of the pixels associated with each region, and compute a measure of self-consistency for each region. Once all the regions have been considered the most self-consistent region is used to obtain an estimate of the true signal.

Clearly, other region criteria could also be used for regions. In our case the regions include the pixel in question. In other applications such as color filter array interpolation or sub-pixel resolution it would make sense to exclude the pixel. The concept could be extended to three and four dimensional analysis.
The crux of the algorithm requires: a measure of self-consistency and a measure of central tendency. It is worth considering other options for these measures.

**Measures of self-consistency**

The premise of this technique is that if the most consistent region including the pixel is most likely to belong to that region.

For this work the variance is used to indicate inconsistency, so minimizing this inconsistency maximizes consistency.

**Measures of Central Tendency**

Several measures of central tendency exist and the optimal one for a given application depends on the application. These include the mean, the mode, the median, the mean after the outliers have been removed, and so on.

The key to selecting the best one is to remember that the measure of central tendency is used to find a pixel value to represent that region. The algorithm uses the mean value, with the understanding that the region averaged over has been selected to be as consistent as possible over that region.

The use of the mean values agrees with Kuwahara’s 1976 technique [15] for using non-linear filtering to recover cardiac segments from radio isotope images.

**Variations on this Algorithm**

Peyman Milanfar has shown that subtle changes in an algorithm can generate many different papers for publication, and new algorithms to claim, but the generalized mathematics is often the same [47]. The same philosophy we use here generates the same basic algorithm regardless of the measure of dispersion, the measure of central tendency, or the normalizing constant(s) used.

For example, the same math, with a change in variables gives noise smoothing across space, volumes, or space-time volumes. Inverting the results gives edge detection, or motion detection. These subtle changes serve to generate an impressive number of papers published and algorithms named, but the framework of the approach is the same.

By the same argument, the framework of the algorithm described here for noise cleaning pupil images here could easily apply to other disciplines, situations, and fields of study.

**4.9.2 Comparison in One Dimension**

Here we compare several algorithms on a one dimensional example. Figure 4.5 compares five different methods of noise remove from an input signal.
In Figure 4.5 the methods are listed on the left side of each graph. The hollow white circles at each point are the input signal to each method, and are the same in all five cases. The solid shapes at each point are the output of each method.

For each algorithm, the input is examined in an input buffer, and the results are written to a separate output buffer.

For all these methods, two 9-point regions were examined – one to the left of the point, and one to the right of the point.

The goal in each case is to recover the sharp boundary transition that occurs near the value of 180 (center of the domain) where there is a transition from a high value to a low value. An ideal response would be a unit step from $+1$ to $-1$.

The following five subsections all refer to Figure 4.5.

**The local average**

The first method replaces each pixel in the output buffer with the average value of the nine input points in each moving region. This is referred to as a *moving average filter*.

Impulse noise causes a change in the entire region containing the impulse. On the left, for $x$ values from 160 to 175, all the values are pulled down by the impulse spike at 162 and 171. The boundary transition from $x$ of 790 to 182, where the signal drops from $+1$ to $-1$ at the value of 180 is spread out to $x$ values of 176 to 184, and the slope of the transition region is attenuated.

This moving average filter blurs both the location and slope of the transition region, and is pulled off by outlying signal values.

**The Median**

For the second method, the median value of the 9 points in the region is used as the output. The process of computing the median requires an additional sorting of the values in the region, to find the middle value. This median value is used as the output value.

This removes impulse noise, and does a fairly good job at maintaining edge fidelity in one dimension. However, the slope of the transition region is still attenuated because the values near the transition are replaced with less extreme values.

The full dynamic range of the response is lost by median filtering.

**Erosion followed by dilation**

The third method uses morphological erosion followed by dilation. It can be seen on the left that outlying impulse noise causes the erosion operator to remove a significant region to the left of the edge boundary.
Dilation followed by erosion
The fourth method uses morphological dilation followed by erosion, experiencing the opposite problems.

Kuwahara 1D example
The fifth method used is a one dimensional 9-point Kuwahara region, and requires additional description.

At each point of the input, the input point and the 8-points region to the left of the point are evaluated for self-consistency. The same is done with the 8-points to the right of the point. If the left side is more consistent than the right side, the average value of the left neighborhood is used for the output value. Alternatively, if the right side is more consistent, then the average value of the right neighborhood is used as the output value.

In the sub-figure, a downward magenta triangle is used to signify that the algorithm used the left mean value, while an upward blue triangle is used if the right mean value is used.

In the region of the transition, this method enhances the boundary transition. This is because just to the left of the boundary the region to the left is more consistent, so the first few values in the transition region are replaced with the average from the left. Further through the transition, at 181, the region to the right is most consistent with the value at 181, and an average from the right side of the boundary is used.

The Kuwahara filter nicely removes the high frequency components from the relatively flat regions and thus attenuates noise. At the same time, it enhances the boundary at the transition. Both of these properties are beneficial.

This method also maintains the location of the transition region. The output signal also has an edge transition right at 180, the desired location.

4.9.3 The Secret of the Shapes
While a one dimensional example is better for instructional purposes, two dimensional regions are used for images. The two dimensional shapes shown in Figure 4.4 were for also for conceptual understanding, and are too small to do any significant noise cleaning. We use a larger variety of shapes, and larger shapes. Here we describe the shapes we use, and how and why the shapes were selected.

Region Rules
The two dimensional regions used were carefully designed and arranged in a specific order so that one algorithm would enhance pupil edges, attenuate undesirable edges in
Figure 4.5: A one dimensional comparison of filtering types. Types shown include: mean, median, erosion/dilation, dilation/erosion, and the Kuwahara.
uniform regions, and intentionally blur unwanted thin lines to reduce the significance of eyelashes and skin creases.

The following general rules were used:

1. The region with the lowest variance will be the region selected. Low variance is considered a measure of self-consistency.

2. In the case that two regions both have the same minimum variance, the first one of the two is used. This is how the algorithm breaks a tie.

3. The largest regions are given first, so that in the case of two regions having the same minimum variance, the region that uses an average over the greatest number of pixels will be used.

4. Shapes are designed so that they will fit either inside a pupil, or outside a pupil. The shapes should either be able to follow the outside contour of the pupil or follow the inside contour of the pupil, but not cross a pupil boundary. In general, this means the the center pixel of the filter will be on the boundary of any shape.

5. In general, shapes are selected so that subsequent ones are sub-sets of previous shapes. This is an outcome of the decreasing size rule.

6. Shapes are designed so that they cannot fit inside a vertical line that is three pixels wide, the nominal width of an eyelash in our system.

The nature of any search algorithm is that it will always return the closest match that it can find, even if the item sought is not in the dictionary of shapes. Since one of the shapes must be selected, and none of them can fit inside a vertical line that is three pixels wide, a vertical eyelash in our eye videos cannot be perfectly reproduced. This guarantees that the pixels inside eyelashes will be smeared at least in some direction outside of the eyelash.

This approach is somewhat analogous to taking a jig-saw puzzle piece that does not fit at a location, and forcing it into place. Just as the puzzle piece becomes damaged when forced into place, the edges of eyelashes are damaged because none of the regions have shapes that support reproducing thin straight lines.

The region shapes are selected to be an intentionally incomplete basis set for reproducing eyelashes and other thin lines.

**Region Generation**

Figure 4.6 shows the set of regions used. Each region is 11x11 pixels. The center pixel is the pixel that is being noise cleaned. The gray region is the set of pixels included in the calculation. The black pixels are not included in the calculations.
Not all of these regions are generated explicitly. Instead a template for the region is used, which is then mirrored, vertically and horizontally, and rotated in order to generate the others versions.

Considering the two rows of figure 4.4. It can be seen that each row is one shape that is rotated to produce all the other shapes in that row. The combination of rotation and mirroring can generate duplicate shaped regions so duplicate regions are detected and removed.

The first region is a disk with a diameter of 11 pixels. This assures that all other variance computations will be compared to a relatively large region. If no other variances are less than this, then the pixel is replaced by an average of that 11 pixel diameter region. In other words, if the most consistent region is this large region, then a representative mean value of that is used.

The thought behind the small curved arcs is that in some situations there is a strong image reflected off the front surface of the pupil, such as the hat of another subject from the study. The small, tight curved arcs fit inside the edge of the pupil, and avoid replacing a pixel on the inside of the pupil with a representation that includes a relatively white hat, or other distracting object.

To evaluate the technique 180 problematic eye images were selected for testing. The proposed technique improves pupil-versus-iris discrimination in 163 of the 180 images. In the other 17 eye images, the eye was closed (blink), or the image was completely over or under exposed so that the pupil was not discernible.

One can see that using a minor variation of this, such as a set of lines that go through the center pixel, would be a technique for edge enhancement along lines through the image. However, such a technique would fit eyelashes and which would be counter-productive to finding the pupil edges.
Figure 4.6: Shapes of regions used for adaptive noise removal. The shape number is given in the upper left. The number of times each shape was used for noise smoothing on an image in the test suite is tracked. This frequency of matching was used during development to be sure that one shape was not dominating over another, which can happen if a region is much smaller than comparable regions.

This image is a snapshot of the results for one image captured during the development process. It is included here to display the shapes that are used and how the shapes are reflections and rotations of each other.
4.10 Performance Optimization

Performance optimization involves decreasing the time that an algorithm takes to run, or optimizing the quality of the result for a given amount of time. This section discusses changes to the brute force algorithm to decrease the run time, failure modes of the algorithm, and steps taken to avoid failures.

The straightforward implementation of our algorithm required 15 minutes per frame. For 62 different region shapes, 62 passes over the image were required, fetching the associated pixels from the image for each pixel. Even for offline processing 15 minutes per frame was unacceptably long.

Here we describe the steps taken to get the timing down to roughly a second per frame.

It goes without saying that when designing the regions, the fewer regions used, the faster the code will run. Consequently, the first method used to reduce timing was to decrease the number of regions used.

A test suite of example eye images was generated, and the algorithm was run against them. The frequency-of-use for each region was tracked, and those which were not used often were removed. Using just 16 regions dropped the time down to about two minutes (compared to 15), which was still too long to spend in the noise cleaning pass over the image. More importantly, using too few regions reduced the edge sharpness of the pupil/iris edge noticeably.

The best practice philosophy for performance optimizing software algorithms is to get the algorithm working soon, and then make it work fast. Getting it working soon assures that the algorithm is correct and precious development time is not wasted, while making it work fast can take eighty to ninety percent of the development time.

The following subsection, 4.10.1, discusses a performance optimization used to compute the variance faster.

4.10.1 Fast Variance Computation

By definition, the variance is the average of the squared deviation from the average value of a distribution, as shown in Eqn. 4.1, with $\mu_x$ the expected value of $x$ and $E[\quad]$ is the expectation operator (the mean value).

$$\text{Var}(x) \equiv E[(x - \mu_x)^2]$$

(4.1)

This expands and converts as follows:

$$\text{Var}(x) = E[(x^2 - 2x\mu_x + \mu_x^2)]$$

(4.2)

$$\text{Var}(x) = E[x^2] + E[-2x\mu_x] + E[\mu_x^2]$$

(4.3)
Var(x) = E[x^2] - 2E[x]E[μ_x] + E[μ_x^2] \tag{4.4}

But \( μ_x ≡ E[x] \) by definition, and \( E[μ_x^2] = μ_x^2 \). Substituting we get:

Var(x) = E[x^2] + μ_x^2 - 2μ_x^2 + E[μ_x^2] \tag{4.5}

Combining terms we get:

Var(x) = E[x^2] - E[μ_x]^2 \tag{4.6}

This is the faster way to compute the variance for a given region. We convert each region to a filter of 1's inside the region, and zeros outside the region. Then compute the average for the region. The entire image is squared once, pixel by pixel, and the average of the squared values in each region is found. Similarly, one average is computed over each region. The variance for each region is found by subtracting the squared average of the region, from the average of the squared value.

The square of the image values is maintained while the variances of all shapes are computed, saving time, memory, and cache utility.

The process of squaring the image values is a fixed cost per image. Consequently, the variance for each separate region is computed using two filtering operations and one subtraction. This method decreases the entire noise smoothing operation to 1.2 secs.

A significant decrease from the original compute time of 15 minutes (900 seconds) by any measure of significance.

4.10.2 Algorithm in Pseudocode

Algorithm 4.7 shows the optimized version of the algorithm. This explains how the time consuming process of repeatedly fetching each pixel in a given region can be replaced
with several filtering operations. One filter operation is used for each region considered.

```
1 Compute the unweighted filters to use for each region shape ;
2 Compute the weighted filters to use for each region shape ;
3 Let im be the input image ;
4 Compute im^2, the squared value of all pixels in the image ;
5 for each region shape do
  6 % Compute several layers of information ;
  7 Compute a_n, the average of im at each location ;
  8 Compute a2w, the average of im^2 at each location ;
  9 Compute vw, the variance of im at each location ;
end
11 % Find the region id yielding minimum variance for the every pixel of the image,
   using one vector operation ;
12 (minvar, minloc) = min(vw) ;
13 Find the associated average value for each layer, and return this as an image ;
14 imclean = a_n(minloc) ;
```

Algorithm 4.7: The Adaptive Noise Reduction Algorithm

4.10.3 Re-weighting the Variance

The regions do not all contain the same number of pixels.

Given two random samples of a population, the smaller sample size would naturally
have less variance (i.e. be more self-consistent) simply because there are fewer elements
in the set.

However we do not necessarily want smaller regions selected because they are small.
We wish instead to compensate for this size effect.

From another point of view, consider a situation in which two regions had the same
variance. Which region should be used for determining a representative pixel value?
Intuitively we would want the larger region because it has a larger, more reliable sample
size. However, there is nothing in our algorithm to accommodate this intuition. We wish
to have the algorithm incorporate a tie breaking strategy that would reduce the variance
of larger regions slightly, so that they are more likely to be selected than smaller regions
with the same variance.

Because of this phenomenon, we re-weight all of the computed variances by a function
of the number of elements in the region. See eqn. 4.9, where m_s is the matrix size of the
entire region, and n is the number of pixels in the region. For this implementation, m_s is
11, and \( n \) differs for each region.

\[
w = \frac{m_s^2}{\sqrt{m_s^2 + n}}
\]  (4.9)

Figure (4.8) shows a graph of the function.

![Graph showing re-weighting of variances](image)

Figure 4.8: Re-weighting the variances of each region, as a function of the number of elements in the region \( m_s \), to avoid favoring smaller regions. Data points relevant to the regions used are shown.

The regions we use have sizes of: 27, 30, 31, 33, 36, 38, 39, 40, 46, 46, 46, 54, 56, 61, and 89 elements. This results in relative weightings of: 0.76, 0.82, 0.83, 0.83, 0.85, 0.85, 0.85, 0.87, 0.87, 0.87, 0.88, 0.89, 0.90, and 0.90 respectively.

The re-weighting function was determined empirically using a test suite of 703 images. The algorithm was run on the images, the frequency of use for each region was used was computed and the results were inspected for clarity.

The value of \( m_s^2 \) was used so that any shaped region can be used in the future, having any number of elements up to a value of \( m_s^2 \). Our test suite of 703 eye frames showed
improved results, and the frequency of matching to various regions shifted to be more uniformly distributed across region sizes.

### 4.10.4 Failure Modes

Figure 4.9 shows a situation where the adaptive noise removal can fail. In this situation the pyramid shaped region has low variance, because it mostly extends to the region on the left. However an edge separates the pixel to clean (marked with an x) from the region on the left and a separate region on the right. If the variance of the region on the left was lower than the variance of any region that fits in the light blue region on the right, the pixel would be replaced with a representative region from the left.

![Figure 4.9: A case where the adaptive noise smoothing could fail.](image)

### 4.11 Additional Benefits

There are two additional benefits of using this technique. Out of this anisotropic filtering we automatically can extract an indication of how difficult it is to fit a pixel at a given point.

Additionally, while this work does not take advantage of it, we potentially also obtain the approximate edge direction.

#### 4.11.1 Edge Enhancement in Transition Regions

Referring to the set of shapes used, shown in Fig. 4.6 we can see that in almost all of the regions, the pixel location is on the edge of the region.
There are two exceptions to this that measure the variance in circular regions, for which the pixel is located in the center of the circle. These two exceptions are included in cases where the pixel is located in a relatively smooth region without edges, and the best representative value for the pixel is the average across that smooth region.

The measure of the residual (minimum) variance at a pixel reveals those pixels which were hardest to fit to any of the shape regions. These pixels occur in edge boundaries, such as the edges of the sclera/eyelids, the sclera/iris, and the pupil/iris. (See the tables of figures 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, and 4.10 for examples.)

This occurs because a boundary between two objects is not a single transition, it is a transition region. Those pixels inside the transition region will not be very consistent with the regions on either side of the transition.

The fact that the residual variance corresponds well with the pupil/iris is used to give us an importance map of which edges should be considered most important to the downstream Focused Hough Ellipse Detector (FHED).

### 4.11.2 Edge Direction

Again referring to the set of shapes used, (figure 4.6 on page 80), we can see that at each pixel location where the pixel’s variance is measured across all of the shapes, the pixel is located on the edge of a shape of the region.

While not used in this work, once the most consistent region is determined we could use the location of the point on that shape to obtain the tangent to the curve. Effectively the best fit to the region gives us a direct measure of the most likely edge direction at every point in the entire image. This local edge direction could be combined with an edge strength to find the most probable edge directions at each location. For example, if the strongest edge was associated with an edge at 33°, this could be used as an indication of the best local edge orientation.

Other methods of using this information present themselves as well. The local region’s edge directions could be combined with edge strengths to form a histogram of oriented edge gradients, effectively a histogram of oriented gradients (HOG) descriptor at a given scale size [48–51]. Alternatively, in future work this information could be considered a shape context [52] at a given scale.

### 4.12 Results – Before and After Images

Tables 4.2 to 4.10 on pages 88 to 95 demonstrate some of this processing. In each table of figures, the input images in the left column. The second column of images is the cleaned versions of these images (the output of the adaptive noise cleaning). The third column shows the residual variance, left after the cleaning process. As mentioned, this is used as
an indication of the edges that segment the pupil from the iris, the iris from the sclera, and the boundaries around the sclera.

For the minimum residual variance images, pixels with a remaining variance in the top 20% of the range are shown in white. Notice that the residual variance detects edges even when the pupil moves into relatively dark regions.
Table 4.2: Examples with various contrast, set 1

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Image" /></td>
<td><img src="image2" alt="Cleaned Image" /></td>
<td><img src="image3" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image4" alt="Input Image" /></td>
<td><img src="image5" alt="Cleaned Image" /></td>
<td><img src="image6" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image7" alt="Input Image" /></td>
<td><img src="image8" alt="Cleaned Image" /></td>
<td><img src="image9" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image10" alt="Input Image" /></td>
<td><img src="image11" alt="Cleaned Image" /></td>
<td><img src="image12" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image13" alt="Input Image" /></td>
<td><img src="image14" alt="Cleaned Image" /></td>
<td><img src="image15" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image16" alt="Input Image" /></td>
<td><img src="image17" alt="Cleaned Image" /></td>
<td><img src="image18" alt="Residual Variance" /></td>
</tr>
</tbody>
</table>
Table 4.3: Examples with various contrast, set 2

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Input Image 1]</td>
<td>![Cleaned Image 1]</td>
<td>![Residual Variance 1]</td>
</tr>
<tr>
<td>![Input Image 2]</td>
<td>![Cleaned Image 2]</td>
<td>![Residual Variance 2]</td>
</tr>
<tr>
<td>![Input Image 3]</td>
<td>![Cleaned Image 3]</td>
<td>![Residual Variance 3]</td>
</tr>
<tr>
<td>![Input Image 5]</td>
<td>![Cleaned Image 5]</td>
<td>![Residual Variance 5]</td>
</tr>
</tbody>
</table>
Table 4.4: Examples with various contrast, set 3

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Image" /></td>
<td><img src="image2" alt="Cleaned Image" /></td>
<td><img src="image3" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image4" alt="Input Image" /></td>
<td><img src="image5" alt="Cleaned Image" /></td>
<td><img src="image6" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image7" alt="Input Image" /></td>
<td><img src="image8" alt="Cleaned Image" /></td>
<td><img src="image9" alt="Residual Variance" /></td>
</tr>
</tbody>
</table>
Table 4.5: Examples with various contrast, set 4

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Image 1" /></td>
<td><img src="image2" alt="Cleaned Image 1" /></td>
<td><img src="image3" alt="Residual Variance Image 1" /></td>
</tr>
<tr>
<td><img src="image4" alt="Input Image 2" /></td>
<td><img src="image5" alt="Cleaned Image 2" /></td>
<td><img src="image6" alt="Residual Variance Image 2" /></td>
</tr>
<tr>
<td><img src="image7" alt="Input Image 3" /></td>
<td><img src="image8" alt="Cleaned Image 3" /></td>
<td><img src="image9" alt="Residual Variance Image 3" /></td>
</tr>
</tbody>
</table>
Table 4.6: Examples of ANC on images with exposure issues.

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
<td><img src="image3" alt="Image" /></td>
</tr>
<tr>
<td><img src="image4" alt="Image" /></td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
<tr>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
<td><img src="image9" alt="Image" /></td>
</tr>
<tr>
<td><img src="image10" alt="Image" /></td>
<td><img src="image11" alt="Image" /></td>
<td><img src="image12" alt="Image" /></td>
</tr>
<tr>
<td><img src="image13" alt="Image" /></td>
<td><img src="image14" alt="Image" /></td>
<td><img src="image15" alt="Image" /></td>
</tr>
</tbody>
</table>
Table 4.7: Examples of ANC on images with motion blur

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Input Image" /></td>
<td><img src="image2" alt="Cleaned Image" /></td>
<td><img src="image3" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image4" alt="Input Image" /></td>
<td><img src="image5" alt="Cleaned Image" /></td>
<td><img src="image6" alt="Residual Variance" /></td>
</tr>
</tbody>
</table>

Table 4.8: Examples of ANC on images with other issues.

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image7" alt="Input Image" /></td>
<td><img src="image8" alt="Cleaned Image" /></td>
<td><img src="image9" alt="Residual Variance" /></td>
</tr>
<tr>
<td><img src="image10" alt="Input Image" /></td>
<td><img src="image11" alt="Cleaned Image" /></td>
<td><img src="image12" alt="Residual Variance" /></td>
</tr>
</tbody>
</table>
Table 4.9: Examples of ANC on images with smallest and largest pupil sizes.

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
<td><img src="image3" alt="Image" /></td>
</tr>
<tr>
<td><img src="image4" alt="Image" /></td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
<tr>
<td><img src="image7" alt="Image" /></td>
<td><img src="image8" alt="Image" /></td>
<td><img src="image9" alt="Image" /></td>
</tr>
<tr>
<td><img src="image10" alt="Image" /></td>
<td><img src="image11" alt="Image" /></td>
<td><img src="image12" alt="Image" /></td>
</tr>
</tbody>
</table>
Table 4.10: Examples of ANC on images with first surface reflections and squinting.

<table>
<thead>
<tr>
<th>Input</th>
<th>Cleaned</th>
<th>Residual Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Image 1]</td>
<td>![Image 2]</td>
<td>![Image 3]</td>
</tr>
<tr>
<td>![Image 7]</td>
<td>![Image 8]</td>
<td>![Image 9]</td>
</tr>
<tr>
<td>![Image 10]</td>
<td>![Image 11]</td>
<td>![Image 12]</td>
</tr>
<tr>
<td>![Image 13]</td>
<td>![Image 14]</td>
<td>![Image 15]</td>
</tr>
<tr>
<td>![Image 16]</td>
<td>![Image 17]</td>
<td>![Image 18]</td>
</tr>
<tr>
<td>![Image 19]</td>
<td>![Image 20]</td>
<td>![Image 21]</td>
</tr>
<tr>
<td>![Image 22]</td>
<td>![Image 23]</td>
<td>![Image 24]</td>
</tr>
<tr>
<td>![Image 25]</td>
<td>![Image 26]</td>
<td>![Image 27]</td>
</tr>
<tr>
<td>![Image 28]</td>
<td>![Image 29]</td>
<td>![Image 30]</td>
</tr>
</tbody>
</table>
4.13 Comparison to Other Methods

This adaptive noise removal technique maintains and enhances pupil edges, while attenuating nuisance edges.

4.13.1 Unsharp Masking

Other techniques, such as unsharp masking [33, p. 162] amplify all edges and noise, regardless of structure. In particular, it amplifies the edges on the first surface of the cornea, of the subject’s eyes, complicating the pupil detection task.

Unsharp masking also requires careful parameter selection to assure appropriate processing for the given lighting and contrast conditions of the image.

4.13.2 Bilateral Edge Detection

Discussion of the bilateral filter originated with the SUSAN feature detector in 1997 [53], and was explicitly written about by Tomasi in 1998 [34].

Bilateral edge filtering detection methods work well only for situations where the statistics are stable and fixed across the entire video and across all subjects. It requires the correct parameter settings for proper behavior. Incorrect parameter settings degrade the results by blurring the edges of the pupil.

Bilateral filtering includes a Gaussian distance weight to favor pixels that are closer to the pixel of interest. Bilateral filtering also includes a Gaussian level weight to favor pixels that are of similar pixel value. The combined weightings tend to favor averaging across pixels of similar value and location, while preserving edges by not averaging pixels of distant location or different intensity values.

As with unsharp masking, bilateral filtering also requires careful parameter selection. Incorrect parameter selection can either remove the pupil edges or enhance the noise edges.

We tested bilateral filtering with a range of parameters, and selected an $11 \times 11$ window size. For a standard deviation for the pixel level, the value of $8/256$ or eight pixel levels was used. For a standard deviation for distance, the value of two pixels different was used.

Bilateral filtering is slow because after all of the weights must be computed for every pixel individually. Then, those per-pixel values must be added and all of the weights normalized. This per-pixel processing significantly increases run-time, requiring over 20 seconds per frame for images of our size.

By contrast, the described adaptive noise removal process runs in 1.2 seconds per frame, and requires no additional parameter selection.

Figure 4.10 shows a histogram of the number of candidate pupils per frame that were found by the Hough circle detector when the frame was first noise cleaned using a bilateral
filter, excluding the spike of zeros for visibility. The mode was 55 candidate pupils. Each of the detected pupils would then need to be sorted through by the downstream processing to find the correct pupil in the set of candidates. (Lone Pine, Subject 3, 2013.)

Figure 4.10: Histogram of the number of candidate pupils found per frame by the Hough circle detector when the image first cleaned using a bilateral filter. The mode is 55 candidate pupils per video frame.

By contrast, Figure 4.11 shows a histogram of the number of candidate pupils per frame that were found by the Hough circle detector when the frame was first noise cleaned using our adaptive noise cleaning technique. The mode dropped from 55 candidate pupils down to only 15 pixels. (Again the zeros were excluded for visibility.) This decreases the number of candidate pupils that would then need to be sorted through by the downstream processing to find the correct pupil. (Lone Pine, Subject 3, 2013.)
Figure 4.11: Histogram of the number of candidate pupils found per frame by the Hough circle detector when the image first cleaned this new adaptive noise cleaning. The mode is reduced from 55 down to only 15 candidate pupils per video frame.

When used as a pre-filter for the Hough circle detector, the adaptive noise removal process generates many fewer pupil candidates from the Hough circle detector than when using a bilateral transform. The adaptive noise filter runs faster, and removes more of the noise that confuses the Hough circle detector.

4.14 Conclusions

This chapter describes a non-linear method for filtering images that enhances the boundaries in edge transition between pupils and the surrounding iris, while attenuating and reducing the edge strength of smaller distracting objects and reflections. The method considers multiple sized overlapping regions, and uses the the region with the minimum weighted variance as the optimum region to interpolate the pixel value over.
By using an intentionally incomplete set of basis regions as features, we developed a filter that detects and enhances pupil edges, but intentionally cannot protect smaller distracting textures such as: eyelashes, skin, iris patterns and compression artifacts. The set of regions are designed so that the method enhances the edge sharpness of the pupil/iris transition by decreasing the width of the transition region. The method consequently blurs the edges of eyelashes, skin textures, iris textures, and other small edges such as those caused by first surface reflections of objects on the cornea.

The algorithm is faster than bilateral filtering, and robust to changes in lighting and contrast.

A novelty of this method is that it achieves pupil edge enhancement and eyelash edge blurring without needing to explicitly find, identify, and then remove eyelashes and hairs. The algorithm does not rely on a prior belief that eyelashes are darker than the skin, and works with blonde or light colored eyelashes.

The algorithm would work with other regions that are specifically designed to enhance edges, such as the Nagao-Matsumyama regions [17].

Obvious extensions would include applications in higher dimensions, such as voxels of image volumes, hyper-spectral, and temporal volumes. The method has potential applications from image enhancement of medical and consumer imaging. Modified region shapes could be used to enhance lines instead of edge regions.
Ch. 5

The Hough Transform

5.1 Overview

Variations of the Hough transform are used in many applications. Surveys of the Hough Transform are performed every few years. Of particular note, there was a survey by Illingworth and Kittler in 1988 [54], Levers did a survey in 1993 [55], and most recently, Mukhopadhyay et. al. surveyed the technology in 2015 [56].

The Hough circle detector has been used for decades in applications spanning: bubble chambers for detecting sub-atomic particles [57], video inspection on assembly lines [58], inspection of submarine cables [59], robotics [60, 61], surgery [62, 63], MRI [64–66], and radio astronomy for detecting Super Novae [67].

In November of 2009, a search for the term “Hough Transform” on Google Scholar [68] estimated that there were over 22,000 publications [69]. At the time of this writing (May 2015) the same search estimated that there are currently over 74,000 publications. Appendix E gives a brief timeline of the most relevant events in the development of the Hough transform.

We describe the basics of the Hough transform, and how it is used for circle detection in section 5.2. This views the transform as a voting algorithm and discusses the size of the parameter space required to capture all the votes.

Section 5.3 discusses the issue of noise, and how using the gradient of the edge we are able to vote for the center of the pupil in a reduced number of directions. This reduces the total number of votes in accumulator space and increases the validity of the votes cast.

The Hough circle detector does not produce one local maximum for an ellipse, instead it produces several candidates to select from. We show in section 5.4 how the circle detector can be extended to detect ellipses, without needing to dramatically increase the size of the accumulator space.
5.2 The Hough Transform Basics

Typically, the Hough Transform is used for finding edges of objects that can be described by a mathematical parameterization, such as lines, circles, or ellipses. It relies on starting with a valid segmentation of objects in an image. Each point on these edges then “votes” for all possible parameter combinations that could possibly explain that edge point [70]. The combination of parameters that have the most votes is selected as the best choice for the line, circle, or ellipse.

Because of the reliance on proper edge detection, the Hough transform is usually used to find objects in images where the object is against a simple high contrast background (such as objects on a conveyor belt), objects that have a high contrast edge on them, or objects that are well segmented if the correct edge segmentation parameters are selected.

Most situations have grayscale images for which the object could be segmented using a thresholding technique such as Otsu’s method [71]. This thresholding converts the images to black and white, and the edges are extracted and then used in a Hough Transform.

We know from basic geometry that three points on the perimeter of a circle define the circle. If there were just three edge points, known to be on a circle, the task of determining the location of the circle would reduce to using a direct least-squares fit, as is done in Gander et. al. [72]. The Hough Transform is needed when there are not only three points, when the points are not all on the circle, when there are multiple circular objects, or when the object is not perfectly circular.

5.2.1 The Hough Transform for Circle Detection

Suppose we had three points believed to be on the edge of a circle, as the x’s are in figure 5.1.

![Figure 5.1: Three points on the edge of a circle, for which the underlying circle should be recovered.](image)

The goal of the Hough transform is to recover the three parameters of the circle that caused the edge points. These three parameters are: the x location of the center, the y
location of the center, and the radius of the circle.

The Hough transform is most easily understood as a voting algorithm [70]. For example, suppose we knew that the radius of the underlying circle in Figure 5.1 was 100 pixels. We could then create a grid of points (one per pixel) and vote for every pixel that was that 100 pixels away from the edge points as a possible center.

For instructional purposes, it might be beneficial to look ahead at the exact circle these points come from, shown later in figure 5.7. However, in computer vision, we do not know the ideal solution ahead of time.

In Figure 5.2 a circle of radius 100 pixels is drawn around one of the potential edge pixels. Each pixel that fell under that circle would receive one vote for a potential center.

Figure 5.2: Three points on the edge of a circle, for which the underlying circle should be recovered. When searching for a circle of radius 100 pixels, the point on the right would vote for every point under the circle of 100 pixels around that point.

In Figure 5.3 we see three different edge points, each of which votes for pixels at a radius of 100 pixels around it.

Figure 5.3: The three edge points, with three circles of radius 100 pixels drawn around each. Each edge point votes for potential circle centers in all directions around them.
A circle of radius 100 pixels has a circumference of approximately $2\pi r$ edge pixels (approximate because pixels are discrete). If all the edge pixels could be perfectly detected, there would be 628 edges detected. If all of these edges voted for all pixels that were 100 pixels away from themselves, we would eventually have one pixel in the center that had 628 votes.

In Figure 5.3 we see one point where the three separate circles overlap. In this ideal situation, that would be the only pixel that received three votes. Since we are using votes from only three edges, three is the largest peak that could possibly be detected.

Figure 5.4 has a black square drawn around the point where the three circles overlap, to identify where the pixel with the most votes for the center is.

![Figure 5.4: The three edge points, with three circles of radius 100 pixels, one drawn around each. The one location that has three votes has a black box around it to show the location where the three circles intersect.](image)

In the computer these votes are accumulated in memory locations called accumulators. For our implementation we use one accumulator for every possible $(x, y)$ center of a circle. We have a two-dimensional array of accumulators of votes for all possible $(x, y)$ center locations.

The example shown is the ideal. In reality, we do not know the exact radius for the underlying circle. The radius could be smaller or larger.

We need to try all possible radii for the underlying circle in the specified range to search for. This means that if we are considering 30 different radii, there must be 30 different two-dimensional arrays of accumulators. The result is a three-dimensional array of accumulators, which we refer to as Hough space.

Every edge point votes for all the points around it at each different radius being considered. After all votes are cast we search the three-dimensional array of accumulators across all possible votes, for all possible radii, and find all local maxima as candidate pupil centers.

In Figure 5.5 we see the same three points which vote for all pixels that are a radius of 90 pixels away. Because the radius of 90 pixels is too small, they do not meet at one
point and the center of the underlying circle that we are trying to recover is not a local maximum of votes.

![Figure 5.5: The same three points, with three circles of radius 90 pixels drawn around each.](image)

Similarly, in Figure 5.6 we see the same three points which vote for all pixels that are a radius of 130 pixels away. Again, because the radius of 130 is wrong. The circles do not meet at one point and the center of the underlying circle that we are trying to recover is not found.

![Figure 5.6: The same three points, with three circles of radius 130 pixels drawn around each.](image)

If everything were perfect, the circle shown in Figure 5.7 would be recovered.
Of course, if everything were perfect, and we knew that all edges belonged to the circle in question we would not need to use a voting algorithm, we would use a least-squares fit to the edge points because only the edge points would be detected. In actual applications edge points are detected that are not associated with the desired circle. These are noise edge points, and it only takes a few of them to create local maxima that are not at the center of the desired circle.

Figure 5.8 shows the same three circles with a few nuisance (noise) edge points included. We can see that the correct center is no longer the only pixel that receives three votes.

In addition to unwanted nuisance edges, we also have additional issues that cause difficulty recovering the circle of the pupil. Most images that use the Hough transform use images for which the objects are easily recovered so that they can show the efficiency of the algorithm in a concise publication. For our application the pupil edge strength is not known a priori, the edge strength varies, the pupil is not a perfect circle (see Figure 4.2), and the pupil is projected onto the sensor as an ellipse.

Figure 5.8 also depicts another important aspect of the Hough transform. A half dozen edge points cause an inordinate number of votes when using a plain Hough transform. In this case, six points generate votes at a disproportionately large number of other locations in the image. The Focused Hough Transform, described later in Section 5.3, reduces the
amount of clutter and mitigates the number of nuisance votes.

### 5.2.2 Ideal versus Actual Votes

For an analytical solution, in which an infinite number of edge points on the circumference of the pupil contributed votes for the best center, the votes at the center would become a delta function that goes to infinity. The local maximum, the local peak, would be easy to identify.

Since images are pixelated, edges are detected only at discrete locations, so only a finite number of votes contribute to the accumulator at the center of the circle. For a digital solution, the number of votes at the center would be the number of edge pixels actually detected on the circumference of the circle. Nevertheless, in a perfect system, the number of votes would be the number of edge pixels that were detected on the circle, and would still be easily identified as a local maximum.

Of course, the edges in an image are not perfectly detected, nor is the pupil perfectly circular, so that the local maximum detector needs to integrate over a range of possible locations and a range of local radii. This integration is achieved using a 3D convolution, and the peak values of the convolution are used as candidate pupil centers.

In the system used for this dissertation the pupil can be relatively small, around 8 pixels in radius when constricted. The Hough transform is trying to find a local peak of $8 \times 2 \times \pi$ (around 50) votes in an accumulator. In the presence of noise, this can be similar to trying to find a needle in a haystack without a magnet.

### 5.2.3 Hough Parameter Space for Circles

The size of memory required for accumulating votes has one dimension per parameter. When detecting lines, which have two parameters, Hough parameter space has two dimensions. When using a Hough transform to detect circles, there are three parameters: the x location of the center, the y location of the center, and the circle radius.

The size of the parameter space is one possible accumulator per pixel in the image, times one set of these arrays for every possible radius sought. For our images, the number of radii sought is controlled by the user. When searching for pupils of radius 7 to 24, in images that are 320 by 240, this amounts to $18 \times 320 \times 240$ or 1,382,400 required accumulators.

Anything that can be done to increase the likelihood of success is beneficial. Especially considering that many values could be caused by noise edges, and that we are looking for only a few hundred votes in accumulator space. In the next section we describe a focusing technique used to reduce the amount of unwanted noise in accumulator space.

The accumulator for votes is called either Hough space or sometimes Hough parameter space.
5.3 The Focused Hough Circle Detector

The typical approach to Hough circle detection considers all edge segments as potentially being on the circumference of the circle sought and votes in all directions in an unconstrained manner. Then the algorithm then seeks the combination of parameters with the most votes to identify the circle center and radius.

Since pupils are always darker than their surroundings, this knowledge can be used to reduce the number of mistaken votes in Hough space by casting votes only in the direction of the edge that is darker. Remember that the votes are voting for the center of the circle.

5.3.1 Example Incorporating Dark Edges

An additional example of circle detection is given here. This example incorporates the knowledge that votes for the center of the pupil should only be cast on the dark side of an edge.

Figure 5.9 depicts a perfect circle to use for explaining the voting process.

Figure 5.9: An image of an ideal circle to find using a Hough circle detector.

Figure 5.10 shows the case where one point on the perimeter of the circle has been detected. Using the gradient, the algorithm knows to vote for the center of the circle on the darker side of the edge.
Figure 5.10: An image of an ideal circle, with one edge point and gradient detected.

Figure 5.11 depicts that since circles of many different radii could have caused that edge, multiple radii need to be voted for.

Figure 5.11: Multiple possible radii could have caused the detected edge, so multiple radii are given votes.

Figure 5.12 shows that multiple edges contribute to the voting process. Here the two edges cast votes from different directions, towards the darker sides of the edges.
There are more edges that contribute to finding the center of the circle. Figures 5.13(a) and 5.13(b) show eight of the edge directions. In each of these, the edges that are detected vote for multiple radii, and for a small arc of 90° around the possible circle center.

Figure 5.14 shows eight of the many edges that contribute to the voting process. Note that any detected edge votes for all possible circle radii and for centers that are in a 90° arc around the quantized direction. Only eight of the detected edges are shown voting in this figure. However, all of the edges on the circle perimeter would vote.
Each detected edge’s direction is estimated, quantized to the nearest 45°, and then an arc at that angle ±45° is voted for at all possible radii. Together the votes from these edges combine to cause one region in Hough space with a maximum number of votes.

5.3.2 Traditional Approach is Susceptible to Noise

Even for well separated hand drawn images, the traditional Hough Transform is highly susceptible to noise [70,73–75]. Anything that can be done to decrease the noise in the Hough space helps the algorithm find the correct circle instead of a wrong circle.

For example, if trying to find truck tires in the presence of falling snow, the snow would create a number of edge segments that are not related to the perimeter of any circle. When used for pupil detection, curved eyebrow hairs and curved eyelashes cause the traditional approach to generate incorrect results.

Another subtle problem with the traditional approach in the presence of noise is that noise accumulates in a circle center from all the points that are at a distance of the radius. Consider for a moment a fixed (X,Y) point for the center of a circle in parameter space in the presence of noise. That (X,Y) location would accumulate votes from several different radii. For circles of radius 5, it would accumulate votes from pixels of roughly $5 \pm \frac{1}{2}$ pixels, or an annulus of area $\pi \times (5.5 - 4.5)$ – approximately 31 square pixels. For the next larger radius, at that same circle center, the accumulator collects noise from $\pi \times (6.5 - 5.5)$, or roughly 38 square pixels. The extra noise from larger circles creates a bias that makes the traditional approach over-estimate the radius of a given circle, because larger candidate circles accumulate proportionately more noise.
To compensate for this bias towards larger circles in the presence of noise, our implementation compensates for the circumference of the circle. This compensation is achieved by normalizing each point in accumulator space by the circumference of the circle, or $2\pi r$ for the respective radius.

### 5.3.3 Example on an Eye Image

As discussed in Section 5.2.2 a small number of detected edges cause a large number of votes, as illustrated in Figure 5.8.

Figure 5.15 shows some ideal edges detected on an eye image. The following discussion shows the votes that are cast in Hough space from these edge locations using the Hough Circle detector.

Figure 5.15: Ideal example edges detected in an eye image. These edges are used as input to the Hough Circle detector.

Hough space includes the full range of all possible $(x,y)$ locations, and all possible radii. The possible $(x,y)$ locations come from the image itself, and the radii that the user indicates they want to find. In some implementations, curved arcs in an image can be caused by circles centered outside of the image. For this implementation we restrict the center of the image to points inside the image because we expect the usable pupils will be found inside the image.

The minimum and maximum radii are set by the user as a control parameter. These radii are reflected in Hough space as layers of possible $(x,y)$ locations.

The following figures show the votes that are cast in the different radii of the Hough detector, at different radii of Hough space. These figures were extracted from the running code from an actual eye image. Figure 5.15 was created later to estimate where the edges were that caused the votes.
Figure 5.16 shows nine different sets of votes from the same edges, edges which are depicted in Figure 5.15. The use of the Focused Hough Circle Detector reduces the amount of stray votes used, since votes are only cast in a $90^\circ$ arc around the quantized edge angle.

For the example given, the largest number of votes, the global maximum across all of the possible radii, occurs for the radius of 13. So, in this case, a circle of radius 13 would be detected with a center at the location of that peak maximum.
Figure 5.16: An example of slices through Hough space for different radii. Figure 5.16(a) shows the votes that result from the edges shown in Figure 5.15 at a radius of 10. Figure 5.16(i) shows the votes that result from the same set of edges, but at a radius of 18. Note that votes are not cast in all directions around an edge, but only in a 90° arc. This focused voting technique reduces the amount of stray votes.

5.3.4 Novel Contributions

We use the direction of the gradient of the edge to vote in roughly ±45 degrees of the desired direction.

The algorithm uses a focused voting scheme, to only vote for potential circle centers in one direction, based on the gradient direction. To compensate for inaccuracy of the gradient direction, the direction is quantized to one of 8 octants, by quantizing the gradient
direction to the nearest 45 degrees. This quantization would still make the direction susceptible to aliasing artifacts caused by pixilation. To mitigate this, instead of just voting for pixels in the 45 degree range indicated by the gradient direction, a total of 90 degrees are voted for, to provide additional tolerance to quantization errors.

Figure 5.17 shows the eight different directions considered.

Figure 5.17: Gradient directions are quantized to a subset of possible angles for the same central edge point. These sub-figures are distributed at angles of 45° to indicate the direction of the quantized edge angle.

The votes are cast in the direction of decreasing digital count, from the bright side to the dark side of an edge, because the pupil should be darker than the surrounding iris.

Whether votes are cast towards the brighter side of an edge or towards the darker side of an edge is application dependent. For wheels on cars, it would be related to the contrast between the metal wheel and hub cap, and the surrounding rubber tire.
orange soccer balls, it could be increasing brightness compared to the surrounding grass.

Kimme et. al. also restrict the direction that votes are cast in to a limited number of
degrees [76]. They presume that the edge gradient direction is correct and cast votes only
in a 45 degree arc. In their work the positions of the accumulators were computed for
every edge detected, and limiting the arc length for voting reduced their processing time
by a factor of six because they only had to compute one eighth the number of sine and
cosine functions.

In this implementation the gradient is not assumed to be exact at each given point,
nor does it assume that the detected edge direction points exactly towards the center of
the pupil to be detected. This approach quantizes the gradient direction to the nearest
45 degrees, and then votes in a 90 degree arc centered on that direction. This technique
provides noise tolerance for the case of poorly determined edge directions.

For speed optimization, the location of votes in accumulator space are not computed
for each detected edge. Instead the relative locations are pre-computed for all eight
octants and for all possible radii sought by the user. Then during the voting process we
use one vector operation to vote for all positions in the accumulator for each radius. This
reduces the size of the inner loop for processing when casting votes to a single vector
operation, and reduces the time required to accumulate votes.

5.4 Finding Ellipses using a Hough Circle Detector

During the automatic pupil detection pass, a non-linear image warping is used to unwrap
the image and make the pupil ellipses less circular. However, this technique cannot
be used until the unwrapping parameters are first determined. Even then, the result
of unwrapping is not a perfect circle because the unwrapping transformation is never
exactly perfect. It is beneficial to have the Hough transform able to tolerate more elliptical
pupils that occur in the image.

5.4.1 Hough Parameter Space for Ellipses

A natural question would be, “If the pupil is always projected onto the sensor as an
ellipse, why not use an ellipse detector?” We explore this question here in this section.

The Hough parameter space for circles spans three parameters: the x location of the
center, the y location of the center, and the radius of the circle.

If searching for ellipses, there would be five parameters. Ellipses are defined by the x
and y locations of their centers, the minor axis, the major axis, and the rotation of the major
axis. If the minor and major axes could range from 7 to 24, and the rotation was desired to
the nearest degree, then the accumulator space for a straight ellipse detector would require
$18 \times 18 \times 360 \times 320 \times 240$ or, $8.96 \times 10^9$ accumulators. Matlab is fastest when working
with double precision, which requires 8 bytes per variable, so that a straightforward
implementation of an ellipse detector would require 71,663,616,000 bytes of memory. Since a straightforward implementation of a Hough ellipse detector would require more than a 70 gigabytes of memory for voting we did not pursue this approach further.

The task of searching for an ellipse in an image is a fundamental task in computer vision. Ellipses occur very frequently because, as in this case, circles are usually presented to the sensor as an ellipse. There are other methods that reduce the search space for ellipses if the ellipse is high contrast and well segmented from the background, and there are enough pixels on the ellipse that the tangent to the ellipse can be accurately determined on the edge of the ellipse [77–79].

However, in this situation there is a small pupil, with noisy edges, and the pupil is not a perfect ellipse. Furthermore, the edge contrast on the pupil varies from frame to frame, and the edges of the pupil are confounded with eyelashes and other edges that are sometimes in front of the pupil edge.

The following subsections discusses how the Hough transform is modified to tolerate more elliptical pupils.

5.4.2 Noise Tolerance for Circles

Perfectly round circles accumulate votes in the Hough space at only one point for the correct radius. For a circle with a radius of 13 pixels, the local peak in accumulator space happens for the slice of accumulator space associated with a radius of 13.

In the other radii of accumulator space, that same circle of a radius of 13 casts votes that are diluted around in the shape of a circle – both for the slices of accumulator space that have smaller radii, and the slices that have larger radii. These “shadows” are created at all other possible radii. For perfect circles none of these votes create local maximum at any other place in accumulator space.

For example, looking back to Figure 5.16, the accumulator in radius 13 is a distinct sharp point. However, that same circle causes a known pattern of votes in the slices of accumulator space around it.

Looking at Subfigure 5.16(a), there is an open circle of votes in the slice of Hough space for circles of radius 10. Similarly, looking at Subfigure 5.16(i), there is another open circle of votes in the slice of Hough space for circles of radius 18. In fact, a circle casts a deterministic pattern, a signature, at all the different radii of Hough space. By accumulating this evidence, the circles of radius 13 could be detected by looking at the other radii in Hough space.

To accommodate noise in the edges of the circle, our Hough detector integrates a range of locations around each point. Additionally, it integrates votes in accumulator space for the radii above and below the radii being considered. This helps to boost the signal strength because some edges vote for the wrong place due to noise. By integrating over a slight range of \((x, y)\) locations and a small range of radii in accumulator space the detector recovers circles even in the presence of noise.
5.4.3 Using Ellipse Signatures for Detection

Just as the circles cast “shadow” votes in other slices of the Hough accumulator space, so too do ellipses.

However, instead of casting circular patterns in accumulator space, they cast different patterns particular to an ellipse. The following figures show different patterns in accumulator spaces with a major axis of 24 pixels, and various minor axis lengths.

Figure 5.18 shows an ellipse with a major axis of 24 pixels and a minor axis of 23 pixels. Overlaid on top of the ellipse is a top view of Hough accumulator space, looking down through all radii. The red circles mark the locations in Hough accumulator space where the ends of the major axis (with tighter curvature) cause votes to accumulate for radii that are under 24 pixels.

By contrast, the edges of the ellipse near the minor axis have less curvature. These sections of the ellipse edges have radii larger than 24 pixels, and cause votes to accumulate at radii higher than the 24 pixel radius. These votes are shown in the blue circles.

![Figure 5.18: Ellipse with major axis of 24 pixels, a minor axis of 23 pixels, and an eccentricity of 1.04](image)

The effect is seen more clearly as the eccentricity of the ellipse increases. (See figures 5.19 to 5.21.)

Of particular interest is how quickly a small eccentricity causes the votes in the accumulator space to not coincide with one location. Even an ellipse with an eccentricity of only 1.04 and a major axis of 24 pixels causes the votes to not coincide at the same $(x, y)$ location in Hough accumulator space. Further, these votes are not even cast at the same
radius – some are above the nominal radius and some are below the nominal radius.

Figure 5.19 shows an ellipse with a smaller minor axis. As the minor axis decreases, the edges of the ellipse near the ends of the minor axis take on a larger effective radius. When the center from these larger radii are located in \((x, y)\) values of accumulator space, they are further from the true center of the ellipse. Remember that the true center of the ellipse is the center of the pupil which we are trying to recover.

![Figure 5.19: Ellipse with major axis of 24 pixels, a minor axis of 18 pixels, and an eccentricity of 1.33](image)

Figure 5.20 shows a more eccentric ellipse. As the ellipse becomes more and more eccentric, the centers the edges project to a further distance from the true center of the ellipse that we are trying to recover.
As the ellipse becomes more and more eccentric, the local maxima caused by the major and minor axes move further from the true center of the ellipse that we are trying to recover. Figure 5.21 shows an even more eccentric ellipse. Again, the edges with a larger effective radius of curvature project to a center that is further from the true center we wish to recover.
This is a problem for a typical Hough circle detector because the detector finds the two local maxima on each side of the ellipse and considers them as centers of circles that are much larger.

Clearly, the circular Hough detector is sensitive to subtle changes in eccentricity.

Worse, for large eccentricities, the ellipse of the pupil causes the circle detector to find two larger circles, neither of which are correct. To compensate for this sensitivity for the detector, we accumulate votes over a region of locations in Hough accumulator space.

By computing the locations of where circle centers accumulate across different radii for ellipses with eccentricities of 1.0 to 2.0, and finding the convex hulls of those loci above and below the nominal radii, we are able to compute two regions of integration that help us detect the center of ellipses. One region is a region in accumulator space that is above the nominal radius, and the other is the region below the nominal radius.

Once the accumulator is formed, and all the votes are cast from all the edges, we use these regions of integration to locate ellipses. For each radius, we add up the accumulators for radii below that radius, and integrate over the regions associated with smaller radii. We do the same for the larger radii. These two scores are added to form an integral vote for an ellipse at each location, which is able to recover ellipses whose votes are spread out in accumulator space.

Figure 5.23 shows the convex hulls that are computed using ellipses from an eccentricity of 1.0 up to 2.0. The solid blue lines indicate the region of integration for radii larger than the expected radius, on only one side of the ellipse center. The other side is also used in the integration process for larger radii. Similarly, the solid red line indicates the region of integration for radii smaller than the expected radius. The mirror image of this region is also used when integrating.
a = 24.0
b = 12.0
ECC = 2.0

Figure 5.23: Ellipse with major axis of 24 pixels, a minor axis of 12 pixels, and an eccentricity of 2.00

For ideal images, that are noise free, the technique of adding up the Hough accumulators for all the radii below a given radius, and all the radii above the given radius, would detect ellipses with eccentricities up to whatever range was desired. However, in implementation, the larger these convex hulls of Figure 5.23 become, the more noise is accumulated. For implementation considerations experiments showed that integrating up to an eccentricity of 1.3 correctly captured nominally eccentric pupils without causing too many false detections.

5.5 Conclusion

The basic Hough circle detector works as a voting algorithm by using each edge presented to it as a potential edge on the side of a candidate circle.

However, a plane Hough circle detector would cast votes for 360° all around every edge point detected. Voting in all directions can cause false detections in Hough space.

The Focused Hough Circle Detector reduces the sensitivity to noise by only voting for pupil centers that are on one side of the edge. This approach is similar to how Kimme, Ballard and Sklansky improved the processing by using the local gradient [76].

The size of the accumulator space grows geometrically with the number of parameters for the object sought. A straightforward implementation of an ellipse detector would require a large amount of memory to be accessible simultaneously. The Hough circle detector is made more robust to eccentric pupils by integrating over several regions of Hough space in order to find the signatures of eccentric pupils.
6.1 Motivation

In Chapter 5 we discussed modifications to a Hough circle detector so that it can detect more eccentric pupils. However, the range of eccentricities for these elliptical pupils is limited. Trying to get the Hough circle detector to detect ellipses that are too eccentric causes the false alarm rate to rise.

By forcing the program to attempt to find a pupil in every frame of several videos using the initial easy-pass only we can determine how much improvement is needed. Table 6.1 shows the results for three videos. For these three videos the number of frames in which pupil candidates were found ranged from 66 percent to 77 percent, missing from 23 to 34 percent of the frames. A method for generating candidate pupil locations in more videos was sought. We seek a way to improve the likelihood that elliptical pupils will be correctly detected, even if they are more eccentric.

Table 6.1: Results of trying to find a pupil in every frame using only the initial processing pupil detection pass.

<table>
<thead>
<tr>
<th>Video</th>
<th>Number of frames with candidates found</th>
<th>Total number of frames</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lone Pine 2013 SN03</td>
<td>1350</td>
<td>1839</td>
<td>73 %</td>
</tr>
<tr>
<td>Death Valley 2013 SN03</td>
<td>2017</td>
<td>2611</td>
<td>77 %</td>
</tr>
<tr>
<td>Olema 2012 SN 06</td>
<td>2381</td>
<td>3599</td>
<td>66 %</td>
</tr>
</tbody>
</table>

The elliptical shape of the pupil is described by five parameters: the x and y location of the center of the pupil, the length of the major axis, the length of the minor axis, and the angle of rotation of the major ellipse axis with respect to the x axis. However, in our situation, for a given subject wearing a head-mounted eye-tracker, the camera geometry...
stays nearly constant. Consequently, these five parameters are tied to the two parameters of the location in the image.

This concept is important and warrants emphasis. As long as the eye camera and the subject’s eye are in a fixed geometry, then when the eye is centered on any given (x,y) point in image, the appearance of the pupil at that location will be the same. Given the location of the eye, the pupil ellipse should have the same major axis, the same minor axis, and the same tilt.

In mathematics, a manifold is a locally Euclidean space that simplifies the task of working with high numbers of dimensions by mapping them to a representation in a lower dimensional space. Manifold learning is learning how to map a higher dimensional parameter space to a lower dimensional parameter space in order to achieve dimensional reduction.

On June 21st, 2015 a search on the internet search engine www.bing.com for “manifold learning” returned the following image as the first result (see Figure 6.1). Figure 6.1 is Figure 6(a) from, Learning to traverse image manifolds, by Piotr Dollàar, et. al. [80], from the University of California at San Diego Center for Visual Computing.
Figure 6.2 shows another example of manifold learning. Here Halevy [81] produced a manifold of facial motions using a piecewise continuous locally preserving projections [82]. In situations, where the physiology of the body limits the range of motion a manifold exists. Manifold learning techniques are now so popular that they have now been incorporated into common software toolkits, such as scikit-learn [83].

Figure 6.2: Facial motions mapped to a lower dimensional manifold. From Halevy [81].

For more recent papers, manifolds that are learned use piecewise techniques to learn the nuances throughout an anisotropic domain. Consequently, most recently the idea of learning a manifold presumes that that relationship learned is complex and must be learned in a piecewise manor. For our data, as with Dollár et. al.’s work [80], the manifold of the eye is an isotropic transformation – a continuous mapping for dimensional reduction.

To avoid confusion with these higher order techniques, we refer to our mapping as an *unwrapping* of the eye image. Regardless of the vernacular, the key insight is that a pupil at a given \((x,y)\) position should have the other three parameters close to what would be expected, and that these observed parameters are a manifestation of the kinematic motion of the eye.

At a given \((x, y)\) location, a pupil ellipse that has unusual values for the other three parameters we term a *strange ellipse*. A pupil whose ellipse is too *strange* can be eliminated
from consideration as a pupil because the limitations of physiology preclude the pupil from taking on that set of parameters. This idea is discussed in Chapter 7, on Strange Ellipse Rejection.

In this chapter we seek a way to take advantage of the fact that the appearance of the pupil takes on only a limited set of parameters, based on its location in the image. Specifically, we develop a method to stretch the original eye image so that the off-center pupils become less elliptical, but the central (circular) pupils remain circular. This assists the circle detector so that it is more likely to detect off-center pupils correctly.

To achieve this, we modify the original eye image by *unwrapping it* so that the elliptical pupils will appear circular.

### 6.2 Results

To assist the reader with understanding the following discussion, we first give example results.

The image in Figure 6.3(a) has an elliptical pupil with the major axis angled at over $60^\circ$. The unwrapping transformation stretches the image in a non-linear transformation, so that the left side of the pupil is moved further out than the right side of the pupil, thus removing the eccentricity. The result, shown in Figure 6.3(b), is that same pupil, but more circular than it was in the original input frame.

On first inspection, the pupil shown on the left figure, Figure 6.3(a) seems circular because human perception knows that a pupil on the surface of an eye is a circle in the world. This illusion is extremely powerful. The human perception system converts the 2D image presented into a 3D understanding of what it expects. (This will be discussed in the next section.)
Figure 6.3: An example unwrapped image. The ellipse of the off-center pupil is converted to a circle for easier detection by the circle detector.

The following example was created to help mitigate perceptual effects. Figure 6.4(a), a different eye image, has had grid lines artificially added to the image before it is run through the unwrapping transformation.

The resulting unwrapped image is shown in Figure 6.4(b). Here the pupil can clearly be seen as less eccentric and more circular. The transformation also shows that thin lines near the center of the image stay thin, while thin lines at the outside of the input image become thicker.
6.3 The Circle Illusion and Near Circularity

Novice developers when asked to find pupils in an eye-image ask the question, “Why don’t you just use a circle detector?” The answer is what we call the circle illusion. As mentioned previously, Figure 6.3(a) illustrates this problem that people perceive circles when shown an ellipse in a picture, especially when they know that that ellipse represents something in a three dimensional space that is circular if seen on axis.

When people see an image of the pupil, they perceive it as a circle. They map the 2D image of the ellipse into a 3D understanding of the pupil as a circle in 3D space. We call this the circle illusion, and it is a trap which novice developers can easily fall into.

The pupil is almost never a perfect circle. The pupil is almost always an ellipse, and this must be kept firmly in mind.

In order for the pupil to be a perfect circle, several conditions would have to hold:

1. The eye camera would need to be pointed so that it was oriented straight at the center of the center of rotation of the eye.
2. The eye would need to be looking straight at the center of the eye camera.
3. The camera would need infinite resolution so that there was no pixelation or digitization artifacts.
4. The sensor on the back of the camera would need to be mounted so it was exactly orthogonal to the optical axis of the camera.

5. The lens(es) of the eye camera would need to be mounted so that their optical axes were all perfectly aligned with each other, and orthogonal to the sensor.

6. The pupil of the subject would need to be perfectly circular.

7. The cornea of the eye would need to have no astigmatism or imperfections that cause aberrations of the pupil.

While the pupil is almost never a perfect circle, it is almost always perceived as a circle because human vision involves the human brain, prior knowledge, and an understanding that we live in a three dimensional world. The Figure 6.5 illustrates how the human vision system understands the world as a three dimensional space, and illustrates the circle illusion. The picture of the dark compact disk was taken from an oblique angle so that the disk is projected onto the camera sensor as an ellipse with a eccentricity of 1.4. Nevertheless, the disk is perceived by the human viewer as a circle.

![Figure 6.5: A dark compact disk on a gray mouse pad with rounded corners.](image)

This illustrates the challenges of trying to find the elliptical pupil in Figure 6.3(a) using a circle detector.
6.4 Transformation

Section 6.2 demonstrated that a transformation could make the elliptical pupil more circular, and thus easier for a computer to find. The transformation we use assumes that the eye camera is located at infinity, and that the eye image is an orthographic (i.e. non-perspective) projection. Since there is no perspective transformation, the image that the eye was projected onto could be at infinity, or touching the eye.

6.4.1 Eye model for transformation

We model the pupil as a dark circle, rotating around on a round sphere of radius \( r \). The other parameters are the \( x \) and \( y \) locations of the center of the eye as it would project onto the eye image. This \((x, y)\) location of the center of the eye we call the center of expansion.

This is a convenient approximation in many ways. The eye is not a perfect sphere. The model does not include the cornea. The model does not include the refraction of the cornea.

The shape of the human eye is better described as as an three dimensional ellipsoid [84, 2011]. Ellipsoids have three different axis lengths, which adds unnecessary complexity to the task of describing the the interrelations of parameters of the pupil ellipse.

Pirri discusses these issues in his work in which he used four high resolution cameras to build a gaze machine [84, 2011]. Pirri used sophisticated cameras and more complicated modeling to solve for the quadric of the eye [84].

Tsukada used a similar model as ours, but tracked the iris (not the pupil), so he did not need to worry about the refraction of the cornea. Tsukada neglected to give how many frames processed or subjects used in his results [85]. More recently Pires used similar mathematical models as ours to unwrap the eye for iris tracking [30]. Pires reported results on two sequences of video of 123 and 480 frames, for a single subject.
6.4.2 Conceptual Understanding

Our model is elegant in that it requires the determination of only three parameters: the radius of the eye, and the \((x, y)\) location of the center of the eye (the center of expansion), as projected onto the eye image. The \((x, y)\) location of the center of the eye is most easily understood as the need to know how much the eye image is shifted with respect to the camera, or it could be considered where the camera was aimed with respect to the eye.

The radius of the eye is also required so that we know the extent of the input image to apply the transformation to.

Figure 6.6 provides a schematic for understanding this unwrapping process. Conceptually, if the sphere was a ball wrapped in rubber, we are effectively unwrapping the rubber and stretching it until it is a flat plane. The outside perimeter of the rubber sheet is stretched more than the inner regions of the image.

Once the three parameters are determined, the procedure for transforming the image follows. Figure 6.7 shows some of the intermediate values calculated during this procedure.

1. Given the pixel position in the input eye image, we solve for the Euclidean distance (in pixels) to the center of expansion, in input image space, \(d\). We also solve for an angle of rotation around that center of expansion, \(\phi\) from the front view of the image (not shown).

2. Given that distance, and knowing the radius of the eye, we solve for the angle subtended by an arc for a right triangle of hypotenuse \(r_s\) and opposite side \(d\). This gives us the result that \(\omega = \arcsin(d/r_s)\).
3. From the arc length, we find the distance around the circumference of the eye from which that pixel would have come, if the eye were a perfect sphere, $c$. This is a circumference arc length.

4. Using this circumference distance, $c$, and the angle, $\phi$ from the front view (not shown), we compute the new pixel position in unwrapped space as if the surface of the eye were unwrapped into a flat surface.

![Image of an eye with annotations](figure6_7.png)

Figure 6.7: Schematic with some additional variables added to explain the process of computing the unwrapping process.

### 6.4.3 Optimized transformation

Conceptually, we have described the mathematical transformation.

For the sake of efficiency, when the first image transform for a subject is computed, the conversions are cached. Subsequently, the entire image is transformed using a single vector transformation from the original space to unwrapped space.

The resulting image is padded appropriately as the unwrapped image is larger than the original image. One of the issues we had to solve was how much extra padding needed to be added to the image during the transformation process.
6.4.4 The inverse transform is also required

It goes without saying that once a circle is found in the unwrapped space, the inverse transformation must also be available.

Once a circle is found in the unwrapped space, we find the ellipse in the original space using the inverse transformation of the center of the circle, and four points on the circle: the point furthers from the center of expansion, the point closest to the center of expansion, and two points halfway between them.

Invertability is a requirement of a homeomorphic transformation (for any manifold) [86]. However, for us this is a practical requirement because once we locate the circular pupil in the unwrapped space, we need to identify the ellipse in the original image that it came from.

6.5 Finding the Transformation Parameters

In the past we tried using a gradient descent procedure to find the unwrapping parameters. The gradient descent procedure iteratively solved for both the center of expansion and the size of the eye simultaneously. However, this was used before we had user-validated pupils coming in from the first pass.

With the availability of user validated ground truth pupils from the first pass, we could use the pupil ellipses to find the center of expansion without the radius of the eye by projecting the minor axis of the ellipses to a point of convergence. We then use the center of expansion and the eccentricity of the given ellipses to estimate the radius of the eye sphere.

6.5.1 Finding the center of expansion

To find the center of expansion, we first compute the eccentricity of each ellipse found. Ellipses that are nearly round can have a significant amount of noise in their pointing accuracy because a nearly round ellipse can nearly fit a pupil even if the pointing accuracy has significant error. In an extreme case, a round pupil has no pointing accuracy in any direction, but will fit the pupil perfectly.

To be sure to use pupils that have a high signal to noise ratio in terms of pointing accuracy, we only use those pupil ellipses that have an eccentricity of at least 1.4. Again, here eccentricity is the ratio of the major axis to the minor axis.

Experience teaches that those ellipses that have an eccentricity of 1.2 or less are likely to include noise in the pointing accuracy. By using a cutoff value of 1.4 we are sure to have sufficient tolerance in case the user made some errors hand inputting the pupils during the first pass.
Using this sub-set of pupils, we project lines from the minor axes across a voting space, where there is one bin per pixel of the image. Each bin within one pixel of the line is given a vote.

Figure 6.8 shows three ellipses with lines projecting out of their minor axes to meet at one point, the center of expansion.

There can be an issue with too many ellipse on one side of the image, simply because that is how the subject was looking, or how the subject tended to hold their head. To avoid having a cluster of ellipses over-weight the voting bins, only one vote is cast per pixel. Pupils that recur due to the natural motion of the subject do not get two votes for the same image positions.

Once the votes have been cast, the weighted local average over all the bins is computed. The position with the most votes is judged to be the best center of expansion.

A circular disk, fifteen pixels in radius, is used to find the local average of votes across all voting bins.

![Figure 6.8: Using the minor axis of the ellipse to find the center of expansion.](image)

### 6.5.2 Estimating the radius of the eye

Using the prior knowledge from our eye simulation that the major axis of the pupil is relatively constant, and approximately the true radius of the pupil.
This means that the radius of the pupil, $r$, is approximately the same as the semi-major axis, $a$ of the measured ellipse associated with that pupil.
Figure 6.9: Side view of eye showing a rotation of $\theta$ degrees.
(See Figure 6.9 for a schematic of this situation.)

Foreshortening caused by an out-of-plane rotation by \( \theta \) causes the semi-minor axis of the ellipse to decrease by \( \cos(\theta) \). Thus we know that \( 2 \times b = 2 \times a \times \cos(\theta) \), or \( b = a \times \cos(\theta) \), or \( \cos(\theta) = \frac{b}{a} \).

We remember that we define the eccentricity \( e \equiv \frac{a}{b} \). Substituting we find that \( \theta = \acos(\frac{b}{a}) \) becomes \( \theta = \acos(\frac{1}{e}) \). Thus the eccentricity can be used to get an estimate of the amount of out-of-plane rotation (neglecting the effects of refraction).

Remembering that from Subsection 6.5.1 we have an estimate of the location of the center of expansion, or the center of the eye sphere in image coordinates.

For each pupil ellipse, we can compute the distance to the center of this center of expansion because we know the \((x, y)\) center of each ellipse. This distance to each ellipse, \( d_{\text{ep}} \) is related to the radius of the eye sphere, \( r_s \), by the equation \( d_{\text{ep}} = r_s \times \sin(\theta) \).

Yet we have an approximation for \( \theta \) already as \( \theta = \acos(\frac{1}{e}) \). From this we can solve for \( r_s \) and find:

\[
rs \approx \frac{d_{\text{ep}}}{\sin(\acos(\frac{1}{e}))} \tag{6.1}
\]

Using Equation 6.1 we then compute the approximate radius of the eye sphere for all the pupil ellipses that have an eccentricity over 1.4 (again to assure that the signal is strong and to increase the likelihood of getting a valid estimation). The average of the estimates is taken and used as the radius for unwrapping the eye.

Future implementations might wish to take advantage of the fact that Equation 6.1 reduces to \( \sqrt{1 - \frac{1}{e^2}}, \sqrt{\frac{e^2 - 1}{e^2}} \), or \( \frac{\sqrt{e^2 - 1}}{e} \). For our implementation we allow the computer to evaluate the trigonometry.

Remembering that the input ellipses that this is based on were from the most stable pupils, and that the calculation is an approximation, a 15% tolerance is added onto this to avoid a situation where a pupil is at the edge of the radius of the eye, and becomes cut off the image.

### 6.6 Unexpected Benefits

The process of unwrapping the eye was performed to increase chances that the circle detector will work and detect the pupil in each frame.

The input frame is read in, and all the pupils within \( r_s \), the radius of the eye, are unwrapped to an expanded location.

The unintended consequences of this is that any pixels outside of the radius of the eye are cropped out of the transformation.
Just as the process of finding eyelids helps us avoid processing and accidentally detecting false alarm pupils in some regions of the input image, unwrapping the eye discounts any region of the eye that is outside the radius of the eye.

In addition, we can also use the resulting geometry model to exclude candidate pupil ellipses that are inconsistent with these modeling parameters. This \textit{strange ellipse rejection} is discussed in the following chapter.

\section*{6.7 Conclusions}

Because we have hand-validated pupil ellipses from one of the early passes of processing, we know that the locations, sizes, and eccentricities of these pupils are correct.

Using the five measured parameters of these ellipses $x, y, a, b, \tau$ we can directly estimate the three parameters required for the unwrapping transformation. These three parameters are the $(x, y)$ location for the center of expansion in the image, and the radius of the eye.

Whether we call this \textit{manifold learning}, \textit{unwrapping}, or \textit{a non-linear distortion}, the process reduces the eccentricity of the ellipse pupils by converting them from ellipses to circles for easier detection by the Hough circle detector.

The ability to directly compute these parameters avoids the need to use an iterative process and gradient descent. This completely avoids previous issues with gradient descent – such as getting trapped in a local minimum instead of finding a global optimum.

Once the unwrapping transformation is computed, it is performed using a single vector operation to avoid re-computing all the unwrapping math for every single input frame. The inverse transformation is performed on an as-needed basis to convert the circles detected in the unwrapping space back into ellipses in the original image space.

The unwrapping transformation has the unexpected benefit that some regions of the image are automatically excluded from consideration because they are outside the radius of the eye. This precludes any false alarms that would be caused by moles, dark spots, or eyebrows in those regions.

The following chapter discusses that once these parameters are known, there is an additional benefit that inconsistent candidate ellipses can be identified and automatically rejected as part of the process of identifying the correct pupil ellipse.
Ch. 7

Strange Ellipse Rejection

7.1 Motivation

The purpose of the upstream processing is to detect any possible pupil in the input image. The input stage generates as many candidate pupils as possible. From those candidate pupils ellipses detected, the next step is to identify the real pupil in the video frame as reliably as possible.

Based on the expected kinematic motions of the eye, there are some ellipses that are inconsistent with the range of appearances of a pupil for the given eye video. Some of the candidate ellipses cannot possibly be a pupil.

Here we describe a process we call strange ellipse rejection. The goal is to identify those candidate pupils that are so egregiously inconsistent that they can be removed from consideration using some simple, reasonable, and robust heuristics.

Figure 7.1 depicts some candidate pupils that were found before the implementation of strange ellipse rejection. The figure shows three different candidate pupils, none of which are reasonable for this subject. Consider the two candidates that are near the center of the eye, shown in red and green ellipses. If these were real pupils near the center of the eye, the would not be this eccentric. The third candidate in the figure is shown with a blue ellipse. A pupil this far from the center of the eye would have a minor axis that pointed towards the center of the eye, unlike the ellipse depicted.

We can learn some rules for rejecting all of these candidates, and instead have the program find a better choice. While learning rules for rejecting these strange candidate pupils ellipses, tolerance is included to avoid false rejections.
CH. 7. STRANGE ELLIPSE REJECTION

Figure 7.1: Example candidate pupils captured before the strange ellipse rejection stage. The pupil was inadvertently split into two ellipses, both of which are too eccentric to be true pupils that close to the center of the eye. Another ellipse in the corner of the eye can be rejected because its minor axis points too far from the center of expansion.

From our simulation of the refraction of the cornea of the eye we find some expected parameter interactions. These are captured in the resulting table of expected eccentricities (Table F.1). For each input eye video, we discover some parameters of the subject’s eye, such as the center of expansion, as described in Chapter 6.

From the simulation results in Appendix F we know that the major axis of the pupil does not change size extensively as a function of rotation or position. We use this to create some fundamental rules to reject candidate pupils based on their major axis. Some of these rules relate to the absolute size of the pupil. Other rules relate to the combinations of parameters for the proposed candidate pupil ellipse.

Considering that we have a QVGA image (320 × 240) and referring to table F.1 (which was based on VGA sized images) we can write down some rules to identify egregious candidate pupil ellipses. These candidates can be eliminated from consideration automatically.
7.2 Positive and negative examples

A key insight was that the ellipses found during the first pass should be validated by the user. This validation gives us solid positive examples to use for finding the center of eccentricity and the radius of the eye to use for unwrapping.

For learning the parameters for unwrapping, it is sufficient to use only the positive examples. We use the parameters for unwrapping that are best for the given set of validated pupil ellipses.

In addition we have another rich source of information from the first pass over the video frames. Each frame that was analyzed on the first pass finds and suggests to the user the three best pupil ellipses.

The user then takes one of three possible actions. Firstly, the frame could be labeled as a blink, which does not pertain to the topic at hand.

Secondly, the user could select one of the three recommended pupils as the best pupil to use. By default, during the validation phase, the user is certifying pupils as valid. Even if the user takes the default action of advancing to the next frame, the user is certifying that the first recommended pupil ellipse is the correct pupil. In this case the first pupil is marked as having a degree of confidence of 1.0. A DOC of 1.0 is a signal to later passes that this pupil, in this frame, has been validated by the user as the correct pupil.

If the user selects one of the other two pupils, then the selected pupil is moved into the first position and marked with a DOC of 1.0. The pupil that was recommended as the first choice is moved into either the second or third choice, respectively, and the new first choice is marked as having a DOC of 1.0. The other two pupil ellipses are carefully retained, and the program knows that for this frame in this video, they are not the best choices for the pupil ellipses to use.

Thirdly, the user could hand enter an alternative choice for the correct pupil ellipse. In this situation the new ellipse entered is put into the first position and given a DOC of 1.0. The original first and second choices are moved into the second and third choices, and carefully retained.

The outcome of this process is that the program not only has valid first choices for pupil ellipses, it also has valid selections of pupils that should not be used. These certified invalid pupil ellipses provide valuable training data to use. They indicate which pupil ellipses should not be used based on their position, orientation, and eccentricity.

For machine learning, these provide vitally important negative examples. These negative examples are not just arbitrary ellipses that are not pupil ellipses, these are pupil ellipses that came out of the early recommender stage for pupil detection. These are the mistakes that were made by the first stages of detection that must be rejected manually or automatically in preference for a better pupil.

The subtle importance that the following stages need to consider only the cases from the previous stages comes from a deep reading of the works of Dr. Paul Viola and
Dr. Michael Jones [87–89] and personal communications from both of these authors [90]. Viola and Jones’ development of their fast rejection cascade was one of their key developments. Each decision point in their fast rejection cascade only needs to correctly classify those candidates that pass through the earlier decision points.

The result of our user validation is two sets of pupil ellipses from our earlier stages: those that should be selected, and those that should be rejected.

The two sets of user-supervised ellipses that have been selected and rejected form a valuable database of information for later decision making and machine learning.

7.3 Rules for ellipse rejection

Since the major axis of the pupil does not change extensively as a function of position in the eye frame, we can impose the provided user parameters to reject some of the candidate pupils. Furthermore we can learn additional rules that help select the correct candidate pupils as we go.

7.3.1 User controlled size rules

The user provides a minimum and maximum expected pupil radius to search for. From this observation we can convert the given restrictions to rules as follows:

1. Any candidate pupil whose semi-major axis exceeds the user specified maximum radius can be rejected. These candidate pupils are too large for consideration.

2. Any candidate pupil whose semi-minor axis exceeds the user specified maximum radius can be rejected. These candidates are much too large for consideration if the minor axis exceeds the maximum expected pupil size.

3. Any candidate pupil whose semi-major axis is less than the user specified minimum radius can be rejected. These candidates are too small for consideration as a valid pupil.

These rules come from the learning and findings from the simulation, and are reasonable considering that the user has specified a maximum and minimum expected radius for the pupil.

7.3.2 User controlled eccentricity limitations

The user can also specify a maximum eccentricity for a pupil. From our simulations we know that beyond a certain eccentricity the pupil becomes too thin to reasonably be detect. Figure F.5 shows that the pupil can have a maximum eccentricity of three to four.
The pupils in our outdoor study typically have an actual semi-major axis of 12 pixels. A pupil with a semi-major axis of 12 and an eccentricity of 4 would have a semi-minor axis of only 3 pixels. It would be unlikely that the system should be able to have detected such a pupil.

A semi-minor axis of three pixels is a minor axis of six pixels, and is not enough pixels on target to expect to be detected. For a basis of comparison, consider that out-of-focus eyelashes can cause a line three to four pixels wide. Consequently, we can add to the previous list the rule that if the candidate ellipse has an eccentricity over three, then it can be rejected out of hand as not likely to be a correctly detected pupil ellipse.

4. Any candidate pupil whose eccentricity exceeds the value of three is either incorrect or it is unreasonable to expect that it was actually detected, and can be rejected.

### 7.3.3 Strange Ellipse Regions

It follows from unwrapping that only certain combinations of elliptical parameters are possible. Using the user validated pupils from the early pass we can find parameters that define certain regions. We can describe several regions of pupil ellipses and the rules that go with them.

A Extremely eccentric ellipses.

Candidate pupil ellipses that have more than a threshold of eccentricity and are under a certain distance from the center of expansion of the eye can be rejected. These candidate pupils are too eccentric for this distance from the unwrapping center to be valid.

B Highly circular region.

Valid pupils that are very central should not have more than a certain threshold of eccentricity. Region B address those pupil candidates that are near the center of expansion. We know from simulations that the refraction of the cornea keeps the pupil circular for forty to fifty pixels from the center of expansion.

Pupils that are too eccentric, too close in, can be rejected.

C Increasing eccentricity with distance.

Valid pupils that are far from the center can be characterized as having at least a minimum threshold of eccentricity as a function of distance from the center. Furthermore, the rate at which eccentricity increases with distance is bounded.

Pupils over a certain distance must have at least a corresponding amount of eccentricity. Otherwise, the pupils can be rejected as not eccentric enough at that distance.
Region D Correctly angled ellipses.

Valid pupils that are eccentric should be angled so that the minor axis points towards the center of expansion, within some threshold of degrees.

Otherwise, the pupils can be rejected because they are tilted the wrong way.

7.4 Learning central pupil limitations

We wish to learn some rules to characterize each of the regions discussed in Section 7.3.3.

We know from finding the parameters for unwrapping the eye the approximate location that the eye rotates around – the \textit{center of expansion}. We also have been given two data sets of certified valid and certified invalid frames.

From this information we wish to find additional parameters so that \textit{strange ellipses} can be rejected while minimizing the chance of accidentally rejecting a true pupil.

7.4.1 Learning Region A

To learn how to identify strange ellipses that are over a given eccentricity and less than a given distance, we try all eccentricities using the algorithm HighEccentricity, listed in Figure 7.2.

```plaintext
// Initialize the parameter settings to fail safe values:
bestEcc ← inf
bestDist ← 0
bestClass ← − inf

for ecc ∈ eccentricities do
    goodPts ← all valid pupils above this ecc
    badPts ← all invalid pupils over this ecc
    minDist ← minimum distance to a valid pupil
    thisClassification ← classification rate
    if thisClassification > bestClass then
        bestEcc ← eccentricity
        bestDist ← minDist
        bestClass ← classification
end

HighEccentricity 7.2: Finding Region A – pupils that are invalid because they are highly eccentric.
```
The result of algorithm HighEccentricity is that a rule can be made in the subsequent pupil candidate sorting routine that states: if a candidate pupil has an eccentricity over bestEcc and a distance from the center of expansion under bestDist, then it is strange and can be rejected.

Figure 7.3 shows an example of a region like this found for one of the data sets. In this figure $\Delta X$ and $\Delta Y$ are the x and y distances from the center of expansion. If the eccentricity of a pupil is over 1.8 and it is within the distance $bestDist$, then the pupil will be rejected. (The best distance is not marked.)

Figure 7.3: Example of a region A found for strange ellipse rejection rules.

Algorithm 7.2 was used to find Region A, a region of rejection automatically.
7.4.2 Learning Regions B and C

As described in Subsection 7.3.3 Region B might be called “highly normal” ellipses for pupils very close to the center of expansions, and Region C might be called “far out” for pupils very far from it.

A machine learning technique was developed that worked for a few videos, but was sensitive to the location of the center of expansion. Instead of continuing a routine that was susceptible to input noise, we use a technique that would alert the user to the issues and allows the user to decide how much tolerance to use in the placement of the rejection regions. This is achieved by plotting the data and letting the user decide where to draw the boundaries that determine the regions.

Figure 7.4 shows a plot of all valid and invalid points from the Death Valley subject number 3, 2013 video for the validation frames, plotted as eccentricity versus distance from the center of expansion (the center of unwrapping). The hollow red circles are the valid pupil ellipses and the solid blue points are the invalid pupils.
Figure 7.4: Example of regions B and C found for strange ellipse rejection rules. Valid pupils are shown in red hollow circles, while invalid ones are shown in solid blue circles.

Notice that there is significant overlap between the two data sets of valid and invalid pupils. This is caused because often the second-best pupil is very close to the first choice pupil. Given only the two attributes of eccentricity and distance, the valid and invalid data sets overlap frequently.

It is rewarding to compare the shape of the valid pupils (hollow red circles) to the predicted pupil eccentricity as a function of distance from the simulation of the eye, as shown in Figure F.5. Both plots have a similar shape.

**Region B – highly circular**

Looking at Figure 7.4, we see two solid red lines superimposed onto the data on the left side. These solid red lines denotes Region B – the region for which the pupils should be very round. The lines indicate that any pupil that is within 55 pixels of the center of
expansion must have an eccentricity of less than 1.6. Again, since these lines are under user control, the user could have reduced Region B to pupils that are less than 40 pixels from the center of expansion and less than an eccentricity of 1.3.

Regardless of where the region lies, candidate ellipses that are in Region B are rejected as inconsistent or strange ellipses. Such strange ellipses can be caused when valid pupils are bisected by reflections going through the center. The resulting ellipses are too eccentric at a distance too close to the center of the eye.

Region C – increasing eccentricity with distance

Again, looking at Figure 7.4, we see a dotted purple line added to the data. This line denotes Region C – the region for which the eccentricity is increasing quickly with distance from the center of expansion. Pupils below this line are rejected because they are not eccentric enough for pupils at the given distance from the center. Again, the user decides where to draw the line, so the user can decide how much tolerance to use to avoid accidentally rejecting a valid pupil.

7.4.3 Learning Region D

Eyelashes around the eye tend to form ellipses tilted in the wrong direction. Valid pupil ellipses over a set eccentricity should have their minor axis angled toward the center of expansion, i.e. the center of the eye. The corners of the eyes sometimes cause candidate pupil ellipses that are angled with the major axis pointing more towards the center of expansion. If we can detect these situations, and reject them, we increase the probability that the program will identify the correct pupil. Algorithm StrangeTilt, shown in Figure 7.5 describes this process.

The result of this is that we can write a strange ellipse rejection rule in the pupil selection stage for region D which states, "if a candidate pupil has an eccentricity over bestEcc, and the difference in angle between the angle of the minor axis and the angle to
the center of expansion is over bestAngle, then it is strange and can be rejected.”

```plaintext
1 // Initialize the parameter settings to fail safe values:
2 bestEcc ← inf
3 bestAngle ← inf
4 bestClass ← inf
5 allAngleDiffs ← angle to all ellipse centers
6 allDists ← distance to all ellipse centers
7 allEccs ← eccentricity of all ellipses
8 // Find an angle, ecc boundary between the two populations:
9 for angleDiff ∈ allAngleDiffs do
10     for ecc ∈ allEccs do
11         goodPts ← all valid pupils above this ecc and angleDiff
12         badPts ← all invalid pupils above this ecc and angleDiff
13         thisClassification ← classification rate for ecc and angleDiff
14         if thisClassification > bestClass then
15             bestEcc ← ecc
16             bestAngle ← angleDiff
17             bestClass ← classification
18     end
19 end

StrangeTilt 7.5: Algorithm for finding region of pupils in parameter space of which are tilted wrong.
```

7.5 The Final Rule

We use one final rule to invoke when eliminating strange pupils.

5. If the process of strange ellipse rejection removes so many candidate pupil ellipses that there are less than three remaining, some are restored for consideration.

If so many pupils are rejected that less than three remain, we restore those that have the highest degree of confidence one at a time until there are at least three candidates. This assures that there will be at least three candidates presented to the user to select between. This effectively enforces the prior knowledge that the pupil is probably in the image someplace, and it is advantageous for the program to make at least three attempts at pupil location.
7.6 Conclusions about strange ellipse rejection

We can apply *reasonableness rules* for rejecting some candidate pupils which do not match the constraints provided by the user. We can automatically eliminate some pupil candidates that are too large, too small, or too eccentric for the constraints provided by the user.

The user validates the pupils found by the first pass of the algorithm. This validation provides a valuable source of information for machine learning and for semi-supervised learning.

In two cases the machine can automatically locate some characteristics of ellipses that are pure regions to eliminate: ellipses that are either highly eccentric but too close to the center of the eye, or tilted too far from the direction we would expect them to be angled.

We take advantage of the fact that we have a human operator and incorporate semi-supervised learning to let the user decide which pupils are too close to the center and too eccentric to be valid. Future candidates with these characteristics are eliminated. We also use semi-supervised learning to let the user specify the rate at which valid pupils should be allowed to become eccentric as they get further from the center of expansion.

Section 12.5.1 presents the results of analyzing videos without and with the strange ellipse rejection rules. These rules all help winnow out candidate pupils that are inconsistent with the geometry of the subject and eye-camera. The result is that the true pupil is more likely to be selected from the set of presented candidate pupil ellipses.
The Benefit of High Dimensionality

We know that when studying the effects of multiple features, feature space grows exponentially with the number of features. For a large number of features, it becomes impractical to collect enough data to completely fill the feature space. Richard Bellman coined the phrase, “The Curse of Dimensionality” in 1969 [91] in regard to the limitations of trying to study complicated data spaces in multiple dimensions and identify the important interactions of the features.

Nevertheless, when the important features are known beforehand, there is a benefit to having a large number of features and many dimensions to data space. In some situations, when the individual impact of each feature is known independently, and when we know to either maximize or minimize the feature, we can combine features to create good initial choices for the best sub-set of the data to use.

An illustration of this approach is given here using an easily understood analogy. The approach is then related to the problem of finding the best pixels to use for determining the pupil edge.

8.1 The Problem of too Many Features to Explore

As the term curse of dimensionality implies, having more dimensions is a problem in situations when one is trying to learn the impact of all possible features, or trying to learn all possible interactions between features.

For educational purposes, we illustrate the problem by analogy to the hypothetical problem of trying to determine the best possible recipe for making pizza.

8.1.1 The Successively Adding Features

Many factors contribute to enjoyable pizza, one of them is the amount of sauce. Some people like almost no sauce, while others prefer the pizza drenched in sauce.
Figure 8.1: For one feature, 10 pizzas (represented by each box) span all possible amounts of sauce in ounces. Ten instances fill the entire data space.

For the hypothetical pizza recipe being tested, Figure 8.1 shows 10 different data points that span the full range of amounts of pizza sauce from 1 to 10 ounces. Each box represents a pizza that would be made for testing, so that all possible amounts are covered. To figure out the best amount of sauce, all other factors being equal, only 10 pizzas are required, and a study could easily be done to determine the most preferred amount. Testing 10 levels of sauce would involve creating only 10 instances of pizza, each with the desired amount of the independent ingredient and testing the results.

The amount of pizza sauce is not the only feature that contributes to the success of a pizza recipe, however. A second feature might be the amount of cheese (in ounces) applied to the pizza, in the range from 1 to 10 ounces.

This second feature causes 10 possible amounts of cheese and 10 possible amounts of pizza sauce to explore in data space. To fill the full data space would require creating 100 pizzas to test all possible combination of the two ingredients.

There are complications because features interact with each other, they are not independent.

For example, consider crust thickness as a third feature. Given too much sauce and too much cheese, the crust might not be able to maintain the weight of the ingredients. The amount of crust required to support the pizza could also be a factor. To perform a proper customer satisfaction survey, we might need 10 different crust thicknesses, from 1 to 10 mm.

The addition of this third parameter increases the data space by an additional factor of 10. At this point feature space grows to $10^3$ or 1,000. Ten samples no longer spans the full data range, but only occupies one one hundredth of the space. (See Figure 8.2.)
Figure 8.2: With three separate ten value features for a hypothetical pizza recipe, it would require 1,000 pizzas to try all possible combinations. Ten instances of pizza now only span only one hundredth the full data space.

If it took a half an hour to make a pizza, then 1,000 pizzas would require over 22 days of straight cooking to make them all.

As the number of dimensions increase, the space of possibilities increases exponentially until it quickly becomes impossible to try all possible combinations. In high dimensions it is not possible to collect enough data to fill the space, and due to feature interactions, only a small subset of these might ever actually occur.

8.2 The Benefit of Multiple Features

This curse of dimensionality limits how much and what we can learn about various features and their interactions, because it requires us to obtain an enormous amount of data to span the full feature space. However, there is a benefit to having multiple known features.

Consider again the hypothetical case of studying the best features for pizza. The curse of dimensionality complicates the process of selecting the single best combination of features for an entire population of people. By contrast, it simplifies the process of
selecting the best pizza for an individual person from a population of pizzas if the person already knows the combination of features that they seek.

Given a situation with pre-set data, and known desirable features, the opportunity to determine the best possible solution is simplified by combining the desirable features.

The curse of dimensionality can be stated as, “If there are many interacting features, and the benefits of each are unknown, then finding the best feature combination requires a large amount of data, and is complicated.”

The contrapositive of this can be stated, “If there is a fixed amount of data to choose from, with many independent features, and the desirability of those features is already known, then selecting the best subset of the given data is simplified.”

The following sections expound on this as it applies to pupil detection.

8.3 Application to Pupil Fitting

The output of the Focused Hough Circle Detector (Section 5.3), is a good starting location for the center of the pupil. The process of noise cleaning the data smooths out the data, but potentially shifts edges. The initial pupil location is determined from noise cleaned image data, which gives an approximate location which should be refined to fit the original input image.

Before we start the refinement process we can state what desirable features we want in pupil edges. Good pupil edges have:

1. high edge strength,

2. an edge pointed towards the center of the candidate circle, and

3. dark pixels inside the circle edge

We can compute the gradient of the edges at all locations in the image, and can measure all of these features for a pixel that is potentially a pupil pixel. The use of Adaptive Noise Reduction (ANR) (Ch. 4) and unwrapping the eye image (Ch. 6), combined with the Focused Hough Circle Detector (Chapter 5) results in an approximate location for the pupil in the eye image.

Instead of using the noise cleaned image we consider the original frame of the eye image, and the pupil location is refined to better fit the original eye image.

To illustrate the issues involved here, we use frame 23629 from subject number 05, from the eye video “LonePine WhitneyPortalRoadOffroad RockBedLonePineCreek” sequence, year 2013. In this particular frame, the top of the pupil is partially occluded by the upper eyelid.
8.4 The Pupil Fitting Problem

Figures 8.3(a) to 8.3(d) illustrate some of the issues of fitting the best circle (or ellipse) to a pupil.

An exact edge does not fall on a single pixel boundary. The edge between the iris and the pupil is not a unit step function, but instead a transition region that extends over several pixels. The exact pupil boundary location is complicated by: aliasing artifacts, pixelation, edge smear, field capture, and other noise artifacts. Figure 8.3 illustrates the difference between human perception of an image and how computer vision measures the image.
Figure 8.3: Human Vision versus Computer Vision. The four sub-images show: an enlarged pupil region (8.3(a)), contour lines and edges that are not visible to a human (8.3(b)), a sub-section of the input image (8.3(c)), and a relief plot with pseudo-color of the same sub-section. This last sub-figure shows that selecting which edge is the absolute best edge is not simply a matter of choosing a threshold. The maximum edge gradient on the left is different from that on the right.

Figure 8.3(a) shows an enlarged region of a pupil in one frame of an eye video. Figure 8.3(b) shows a contour plot of that same image. The contour plot reveals edges that are detected by the edge detector. Some of these edges go straight through the center of the pupil – even if they are not apparent to the unaided human eye. These edges were reduced through the adaptive noise cleaning algorithm, but now the computer must
detect and reject them as unimportant to the pupil fitting process.

Figure 8.3(c) shows a subsection of the image, which is also shown in relief in Figure 8.3(d). Pseudocolor is added to the image to help reveal the change in pixel values across the region. Clearly, the transition between the iris region that surrounds the pupil, and the inside of the pupil, is not a sudden drop over a single edge. Instead, it is a transition over several pixels.

8.4.1 Combining Features to Isolate Good Pupil Edge Points

As previously stated, having too many features whose impact is unknown complicates the process of finding the “best” over-all combination of features, or even the “best” individual features. Yet, because the impact of these features is known and the data is a fixed set of data, we can use this prior knowledge about the features to spread the data out in feature space, making it locally sparse, and quickly isolate good candidate data points.

Here we discuss an algorithm used to help us identify some of the “best” pupil edges. We start our refinement with a tentative understanding of the local pupil size and location. The Adaptive Noise Cleaning technique (Ch. 4) could have shifted the pupil edge in the process of removing noise and enhancing the edge. To accommodate this potential shift we add some tolerance for the edge location.

For a concrete example, suppose we had a tentative pupil circle with a radius of 16 pixels. Suppose we set a tolerance of ±5 pixels for an edge tolerance (which is half the size of the filters used in the adaptive noise cleaning process). This means that the pupil edge pixels could be any of the pixels in a range from 16 ± 5 pixels, or 11 pixels out to 21 pixels. This forms an annulus with an area of $\pi \times (21^2 - 11^2)$ pixels, or roughly 1,005 pixels.

To narrow down these 1,005 pixels to the most plausible pixels, that have the darkest inside pixel value, with edges that point closest to the center of the tentative circle.
algorithm SelectBestPixelEdges is used. SelectBestPixelEdges is shown in Figure 8.4.

```plaintext
1 for all points on the region do
2   compute the edge gradient (magnitude and direction)
3 end
4
5 for all points on the region do
6   compute the angle to the center of the proposed pupil,
7   and compute the absolute difference between those two angles.
8 end
9 for all points on the region do
10  find the pixel value on the darkest side of the edge.
11 end
12 Sort all pixels by each individual feature.
13 // Form a desired target subset of the pixels:
14 Iteratively decrease the fraction of the range of each of the features used from 100%
   down towards 0% until the desired target subset size is achieved.

SelectBestPixelEdges 8.4: Pixel selection algorithm for pupil edge points.
```

The process of using angles avoids converting gradient directions into unit vectors, computing dot products, and then converting dot products back to angles. It also avoids division-by-zero issues, but introduces the complication that the difference between $-\pi$ radians and $\pi$ radians is zero.

A subroutine was written to compute the absolute value of the difference of two angles, based on which quadrants the two angles are in.

### 8.4.2 Example

Figure 8.5 shows an eye image for which the approximate pupil center and radius has been determined using prior processing. The image has been “unwrapped” so the elliptic pupil becomes more circular. However, the pupil circle still needs refinement.
Using a tolerance of ±5 pixels around the tentative pupil location, we establish a range of pixels that could be on the pupil edges. This generates a set of candidate edge pixels shown in Figure 8.6.

Figure 8.6 shows these candidate pupil edge pixels, highlighted around the tentative pupil.
Figure 8.7: Candidate points near the pupil perimeter, plotting in terms of three features: edge strength, inside pixel brightness, and angle of the edge from the tentative center of the pupil circle.

Those exact same points from Figure 8.6 are plotted in three dimensional feature space in Figure 8.7.

Since we know we want edges with a high edge strength, an edge angle that points nearly towards the center of the circle, and a dark pixel inside the pupil, we apply Algorithm 8.4 to restrict the values of each feature until only a fraction of the data points are selected. The resulting points are shown in Figure 8.8.
Figure 8.8: A sub-set of points (from Figure 8.7) that are likely to be the best candidates for the edge of the pupil, based only on the given features.

8.5 Discussion

The strategy of combining multiple features to limit choices applies to many fields of study. The intersection of the set of independent features for which the desirable settings are known limits choices to those most desirable.

This method is so general that it can be used to help find a new vehicle car. At the time of this writing, the website cars.com has 1,682,562 used vehicles available for purchase. We can explore the database by limiting by features independently, and then using the joint combination of features.

Simply limiting the choices to vehicles with four doors drops the choices to 1,406,345.
Only limiting to those that have an automatic transmission drops the choices to 1,351,310. Only limiting to those that are not black or brown, drops the choices to 1,152,384. Only limiting to hatchback or sedan drops the choices to 734,569. Only limiting to those with under 30,000 miles on the odometer drops the choices to 519,686. Only limiting to those made in 2011 or 2012, drops the choices to 418,184. Only limiting to those with under 15,000 US dollars, drops the choices to 313,453. Only selecting those within 100 miles limits the choices to 25,434. Yet, the joint combination of these features limits the number of possible cars to only 82.

This demonstrates that in situations where there are many features of the data, and the desirability of these features is known, the joint combination of features can reduce the number of data instances from a large data space to a smaller, more tractable subset.

We use the fact that we have known desirable features for a pupil to limit the pixels around a candidate pupil to a smaller subset of pixels that have the highest magnitudes, the darkest dark sides, and are most radially arranged around the candidate center.

Algorithm SelectBestPixelEdges, shown in Figure 8.4 describes how we combine features to reduce the sub-set of edge pixels to consider fitting the pupil circle to.

The algorithm is straightforward, using only a linear search by shrinking the number of pixels considered until the desired percent is found. For our implementation we have the routine isolate the pixels to the top 20 percent of pixels to consider.

Chapter 9 discusses a progressive connectivity technique for selecting points from this set that are consistent with a good pupil edge in a manner that is similar to how humans view images.
Ch. 9

Progressive Connectivity

In chapter 4 we discussed how we use locally adaptive edge preserving noise reduction to remove unwanted edges, retain the strongest edges, and enhance edge transitions for better pupil detection. Section 4.10.4 described the problem that in some cases the edge enhancement might be overly aggressive at removing variance, and cross a pupil/iris boundary.

The output of this noise cleaning image is used by the Focused Hough Circle Detector which is described in (Section 5.3).

Since we are interested in having the pupil location as accurate as possible, we wish to refine the edge location to compensate for minor shifting that might have happened during the noise cleaning process.

Chapter 8 discussed a method for rejecting many potential pupil pixels based on the joint combination of their features.

To the human observer, it is obvious where the pupil edges are. Here we examine a phenomena of the vision system that works well, and develop a computer algorithm to emulate this to solve the computer vision problem of identifying good pupil edges.

The vision system is especially good at seeing through noise. In particular, the vision system seems to be able to quickly determine when lines are contiguous and consistently connected.

We develop an algorithm that works well for refining the size and extent of pupils, and emulates the accuracy of a human image analyst. Accuracy is measured in terms of the distance between the located pupil center and the pupil center located by a human image analyst.

9.1 Progressive Connectivity Concept

The human vision easily adapts and adjusts to changes in local features. Consider a picture of a long sidewalk. In this picture, the edge of the sidewalk is in sharp focus on
one end, but has blurry focus at the other end. The vision system does not decide that the edge is two edges because the focus changes from one end to the other.

Instead, the vision system notes that the sharpness of the focus changes gradually along the length of the sidewalk. The entire sidewalk is seen as one sidewalk because the focus changes progressively from one end to the other.

The general gist of progressive connectivity is given in Figure 9.1. This is used to find the best points around a candidate pupil to fit to a model of the pupil.

Given an image
Divide the image into overlapping regions.
for each region do
  find the pixels in that region that have the most desirable combination of features.
end
for every pair of adjacent overlapping regions do
  keep only those pixels that are common to both regions.
end

Figure 9.1: The progressive connectivity algorithm

The algorithm allows the values of the features to change from region to region around the pupil. However, for any two adjacent regions only those pixels that are most desirable in the two adjacent regions are retained.

The pupil is sub-divided into 16 overlapping regions around the pupil. The most desirable feature are those pixels that have a strong edge gradient pointed towards the putative center of the pupil.

The result is a set of pixel edge gradients that are consistent around the perimeter of the pupil. The following sections give examples for lines and circles to better explain the algorithm.

9.2 Progressive Connectivity for Lines

When people view a scene the entire vision system works together to automatically perform an initial segmentation of the scene. This gist of the scene provides an immediate understanding of approximately which objects are connected to which others, and how the scene is segmented [92, 93].

Looking at Figure 9.2, the viewer immediately understands that there is an edge that runs from left to right across the entire image.
The human vision system makes the connections that if a region of the edge is connected to the next region of the edge, then these edges must all be connected. In terms of logic, if object A is connected to object B, and object B is connected to object C, then by deductive reasoning object A must be connected to object C.

This happens very quickly in the vision system, as if the regions in the vision field are progressively making the connection between adjacent regions. For this reason the term *Progressive Connectivity* was coined.

Progressive connectivity seems to happen regardless of whether the edge is all in focus or not. Figure 9.2 was carefully designed so that the edge sharpness changes from left to right across the image, having its sharpest focus near the life preserver ring, about ¼ of the way from the right side of the image.

Figure 9.3 shows the edge magnitude computed on the green channel of the input figure. The Laplacian of Gaussian operator was used to compute the edge gradient, consequently dark pixel values here represent strong edges. By examining the edge strength of the edge of the canal and the sidewalk, the change in edge strength as a function of horizontal location can be seen.
9.3 An Algorithm for Progressive Connectivity for Lines

The Progressive Connectivity algorithm starts by computing the edge magnitude for local regions. For each local region, the edges that have the edge strength in the top five percent of the range are determined. Overlapping local regions are compared, and only the edge points that are consistently in the top five percent of the edge points of both regions are retained. Other edge points are excluded.

Given a set of edge points to choose from, the strongest edge points are selected as the most significant for each region. Those points that are significant in two overlapping regions are considered mutually significant, and consistent across multiple regions.
(a) Left hand region, showing strongest few percent of edges in the region.

(b) Right hand region, showing the strongest edges in this region.

(c) Overlapping region, showing the mutually strongest edges.

(d) Excluded edges between the two regions.

Figure 9.4: Progressive Connectivity - left and right regions that overlap, and the mutually strongest edges. The strongest edges here are those in the top fraction of the edge strength for the region. For example the top \( \frac{1}{20} \) of the range of edge strengths we term the top five percent. All other edges are excluded. This demonstrates the power of this method to select locally consistent edges and reject inconsistent edges.

9.3.1 A Linear Example

By way of a simple linear example consider a few circular receptive fields, or circular regions, from Figure 9.2.

If we use algorithm 8.4 on one circular region from the image, and tune it to find the horizontal edges (with a vertical gradient) that are the top five in terms of edge strength alone, we arrive at Figure 9.4(a). The same computations on a circular field to the right of
that result in Figure 9.4(b).

The pixels edges that are mutually selected in both figures are shown in Figure 9.4(c). This process automatically excludes many of the edges, some of which are strongest in either region, but not in both regions. These strong edges that are excluded are depicted in Figure 9.4(d).

The process of finding mutually best edges can be repeated across the image. Figure 9.5 shows the results of the process further to the left of the image. Notice that the strongest edge is still selected, even though the local edge sharpness on the left is significantly reduced from maximum edge sharpness. Comparing Figures 9.5 and 9.4(c), the change in edge sharpness is apparent.

Figure 9.5: The edges that are mutually strongest edges for two overlapping regions at another point in the image.

Figure 9.6 shows the results of repeatedly running the processing all the way across the image on overlapping circular regions of the image near the edge of the sidewalk. It shows that the process finds the locally strongest edges which are also consistent from
region to region.

![Figure 9.6: Results of repeating the process across the entire edge from left to right across the entire image.](image)

This section has explained finding strong edge points for line fitting, but pupils are circular. The next section discusses extending this idea for circles.

### 9.4 Progressive Connectivity for Circles

#### 9.4.1 Feature Selection

In section 9.3 we discussed finding horizontal edges (with strong vertical gradients) that can be used to consistently fit nearly horizontal lines though images. For horizontal edges, strong consistent vertical gradients are used as the criteria for selecting the significant points.
The features used for selecting the significant points for pupil fitting are different. As a reminder, at the time this routine is called, we already have an initial guess at the location and size of the pupil from a Hough detector that ran on a noise cleaned version of the image. The task here is to refine the boundary of the pupil.

As mentioned in Section 8.3 the features selected for pupil fitting are intended to find the best pupil edges that they are locally consistent. The values of the features themselves change around the circumference of the pupil, but generally tend to be:

- strong edges (large gradient values),
- edges whose gradient direction points close to the tentative pupil center, and
- edges for which the inside pixel value is uniformly dark.

### 9.4.2 Region Shape

It makes sense that the shape of the regions used would affect the results. In Section 9.2 for the initial explanation of the Progressive Connectivity algorithm circular regions were used because they are similar to the circular receptive fields found in the human vision system [94].

Here we are searching for a circle that denotes the edges of a circular pupil. As the potential radius of the pupil increases, the search region along the circumference also increases.

We know the approximate radius of the noise-cleaned pupil. To this nominal radius we add ±5 pixels for the radius, and sub-divide the circumference into arcs. The result is that we use regions that are sections of an annulus to search for pupil edges.

![Figure 9.7: Progressive Connectivity - Using sections of an annulus.](image)

(a) Example of an annular section  (b) A second annular section  (c) A third annular section

Figure 9.7 shows some examples of these regions. Notice that there is a fifty percent overlap for each pair of regions. Because of this overlap, any point is a member of two regions that are used to make decisions about it.
For each separate region, the points in that region are run through algorithm 8.4 to select the most significant five percent of the points in each region. Then the algorithm described in Section 9.3 is used to identify the points that are mutually significant in overlapping regions. The resulting mutually significant points are more selective, but are consistent from region to region.

The significance of the points is determined using the features described in Section 9.4.1: gradient magnitude, gradient angle compared to the tentative center of the pupil, and the darkness of the inside of the edge point.

At points where the pupil is occluded by the upper eyelid, when the regions change from covering a pupil to covering an eyelid, the significant points of each region are no longer consistent. The number of mutually significant points drops, which reduces the number of distracting edge points the circle might be fit to. The criterion that the points must be consistent from region to region excludes extra points where the regions around the pupil change from being pupil-to-iris boundaries to being pupil-to-eyelid boundaries.

### 9.4.3 Example Remaining Significant Points

This process greatly reduces the number of points to a much smaller and more consistent set of edge points to use for fitting the circle of the pupil to. The resulting edge points are more representative of the edge points a human would select.
Figure 9.9 shows the remaining sub-set of points selected (in solid yellow). The figure also shows the circle that would result from a direct least squares (DLS) fit to these remaining points. The circle fit is still thrown off by the points on the upper eyelid. For pupils without obstructions, the DLS fit works well for many pupils. However, because of the possibility of the presence of obstructions we use a modified fitting technique.

### 9.4.4 Fitting to Points

To account for the possibility of obstructions, the mutually significant points that resulted from the previous fits are used to fit a semi-circle. Using the significant points from half of the circle at a time, sequentially around the circle. Using this process many different semi-circles are fit. The semi-circle that explains the largest number of significant points is used as the final fit.

Figure 9.10 shows the final fit selected in this case. Remember that there are fewer consistent points in the region where the pupil-to-eyelid boundary occurs.
Figure 9.10: The final selected circle fit, based on the best semi-circle that has the most significant edge points come within a set tolerance (nominally one pixel) of the fit circle.
9.5 Results

The results for subjects from year 2013, from the MVRL NSF study, at LonePine WhitneyPortalRoadOffroad RockBedLonePineCreek. Table 9.1 gives the preliminary results from some selected sections of video, for frames where both the analyst and the program found pupils in the frame. Frames that were classified as blinks by either the analyst or the program were excluded from the calculations.

Table 9.1: Preliminary Results for Progressive Connectivity.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Start Frame</th>
<th>End Frame</th>
<th>Percent of Pupils within 1.25 pixels of Human Analyst</th>
</tr>
</thead>
<tbody>
<tr>
<td>S03</td>
<td>28012</td>
<td>30618</td>
<td>98</td>
</tr>
<tr>
<td>S05</td>
<td>23170</td>
<td>25827</td>
<td>95</td>
</tr>
<tr>
<td>S08</td>
<td>21929</td>
<td>24574</td>
<td>98</td>
</tr>
</tbody>
</table>
9.6 Other Considerations

9.6.1 Deterministic

Previously when fitting ellipses with colleague Ariana Hong on for finding mirrors in her eye-tracking equipment [95, 96] we used a RANSAC algorithm [97]. Due to the number of parameters involved, and the extra noise, the processing was set to use 7,000 iterations to assure of finding the ellipse that explained the most edge points. The time required was roughly four minutes per frame.

The Progressive Connectivity algorithm is not random, and is completely deterministic. This reduces the compute time to a quarter of a second, which is roughly three orders of magnitude speed-up.

9.6.2 Efficiency

The Progressive Connectivity algorithm requires computing three global features, and then making several tests on regional samples of those features. The features involve computing the edge gradients, and testing the directions of those gradients. The most complicated operation is an arctan() operation used to find the angle of the gradient. All other operations are simple, and are either computed on a 3x3 neighborhood, or are computations involving the difference of two angles.

9.7 Conclusion

This technique we term progressive connectivity is a simple and fast technique that is useful for finding a good pupil edge fit, even in the presence of changes in focus and changes in lighting across the image.

By using a combination of three features, we find good candidate points to use to fit the pupil to. Progressive connectivity is a new algorithm inspired by how the human vision system quickly detects contiguous edges and performs segmentation. Using it, we identify consistent points across spatial, and exclude distracting points that might otherwise be used in the fitting process.

In the example given here, the pupil is partially occluded by the upper eyelid. To achieve robustness to partial occlusion, a semi-circle is fit to successive points around the pupil, and the circle that fits the most edge points is selected.

Progressive connectivity is a fast, simple algorithm that can easily be converted to parallel implementations. Results for three subjects are locate 95% to 98% of pupil centers within 1.25 pixels of where a human analyst would locate them.
Ch. 10

A Degree of Confidence Metric for Outdoor Pupils

10.1 Purpose

The upstream processing generates a large number of candidate pupils for consideration. The user sets thresholds for rejection from the Hough detector, but if these thresholds are set low, the Hough detector could generate dozens of candidate pupils. The program must select the best pupil from these candidates. We want a measure that is proportional to pupil quality to help the program make that selection. We call such a measure a degree-of-confidence [DOC] measure.

There are two types of degree-of-confidence [DOC] measures that could be used. One type, selects the best measure from one frame to the next. These frame-to-frame DOC’s try to estimate the quality of the pupil in one frame compared to the next frame. In biometric identity applications a video sequence of a subject’s eye could be used, and the best pupil from those frames would be used to generate a biometric iris code.

By contrast, we need a within-frame degree-of-confidence measure to select the best candidate pupil from all those available every single frame. We require a within-frame DOC, not a frame-to-frame DOC.

A beneficial DOC would be a quality metric that will let the computer algorithm automatically differentiate the true pupil of the eye from eyebrow hairs, reflections, the dark corners of the eyes, and the reflection of the eye camera.

10.2 Position in processing chain

The DOC is positioned at the end of the processing chain. All of the steps that come before it serve to filter the selections that are presented to it.
In particular, the Focused Hough Ellipse Detector collects votes based on light-to-dark edges and the direction that they are pointing. The complication here is that since eyelashes are curved, creating nice circular shapes that cause local peaks in the Hough detector.

As a consequence, most of the candidate pupils that are presented for consideration are either: the correct pupil, dark eyebrow hairs, dark eyelashes, corners of the eyes that collect all circle sizes, circles reflected by the first surface of the cornea, dark shadows reflected off the surface of the eye, the dark silhouette of the eye camera, or dark corners of the eye. It is these things that our DOC should be able to discriminate between.

We need some sort of center-versus-surround classifier to select the pupils over the other candidates.

10.3 Biometrics

Biometrics is another field where the pupil of the eye must be located in an image.

It is a much different application than eye tracking. In particular the subjects that use a biometric system want to be identified so that they can gain access to some protected resources. People using a biometric application are fully conscious of the fact that they are using such an application, and will hold still so that the system can positively identify them. If the system fails to identify them the first time, they will remain still until the system correctly locates their iris, creates a code for their iris, and authenticates them as who they say they are.

In some biometric systems, each video frame is checked to see how well focused it is, and poorly focused frames are rejected, the camera is refocused, or the subjects are cued to adjust their position until the camera is in focus [32].

Biometrics has control of both the illumination light source, and the wavelength that the sensor is sensitive to. For example, Camus used a 880nm light source (infra-red), and used a matched filter over the detector [31]. By contrast, we only control the filter over our sensor. During the course of a study, the illumination level can change by several orders of magnitude based on the location and orientation of the subject. Similarly, the spectral content of the illumination will change, based on the location of the subject: full sun, clouds, or shade.

10.3.1 Survey of Biometrics

In Biometrics a major resource is Libor Masek’s 2003 thesis on Recognition of Human Iris Patterns for Biometric Identification [14]. Masek’s work comes with the benefit of an open source code base for experimentation.

As a result, Masek’s work is cited by many different publications, sometimes with the reference right in the title, such as Improved Masek approach for iris localization [98]. Masek’s
work has been cited more than 66 times.

At the time of Masek’s writing the only commercial biometric identification system was patented by John Daugman [99]. Daugman later wrote other overviews of how his system works [32, 100, 101], and worked with López on hardware acceleration of the algorithms [102]. However, Masek’s overview and source code still remain the best available starting point for fully understanding how a completely integrated biometric identification system works.

Daugman’s Work

Daugman described an operator referred to as the Daugman Circular Integro-Differential operator (DCID) for finding irides and pupils. This is an energy function that favors: circular regions with a strong image gradient in the shape of a circle [99]. Daugman’s DCID is described by the equation:

\[
\max_{r, x_0, y_0} \left| G_\sigma(r) \ast \frac{\partial}{\partial r} \int_{r, x_0, y_0} I(x, y) \frac{1}{2\pi r} \, ds \right| \tag{10.1}
\]

In this equation \( G_\sigma(r) \) is a Gaussian smoothing function with a standard deviation of \( \sigma \), and the \( * \) operation denotes convolution [99, p 1149]. Effectively, this is a circle detector that finds the maximum total change in a circular gradient as a function of position \((x_0, y_0)\) and radius, \( r \). The closed contour line integral, \( \oint \), is evaluated iteratively for a range of locations and radii. The peak value found then identifies the best values of location and radius.

Notice in particular that Daugman’s equation has a normalization factor of \( 2\pi r \). This normalizes the energy by the circumference, in order to avoid favoring larger circles over smaller ones.

Normalizing by the perimeter of the circle is an important concept, and key to the success of this operator. (We incorporate this in our Hough detector as well.)

The DCID does not use the gradient direction around the circumference. The gradient direction around the circumference should be chosen so that the image region is darker on the inside than the outside. This is true both for the pupil/iris transition and for the limbus/sclera transition under infrared light.

Wildes’s Work

In 1997, Wildes wrote “Iris recognition: an emerging biometric technology” [103], that describes his system for iris recognition [104]. Wildes used a 330mm lens located 15mm to 40mm from the subject’s eyes. Wildes’s other works describe a method for fitting a parabola to the upper and lower eyelid for eyelid removal [105].

Wildes mentions that biometric systems expect to find the entire pupil in the image, and that the image gradient on edges of the pupil should be relatively uniform around the
circumference of the pupil. “In practice, [fine tuning] of the image information has proven to be critical for accurate localization.” [103, p 1355]. Perturbations in tuning parameters cause the system to instead find other structures in the image, such as eyelashes, eyelids, and eyebrow hairs.

Wildes [104] also describes using tuned filters to get mostly horizontal or mostly vertical edges. This is helpful for finding the left and right side of the limbus of the iris, and for finding the top and bottom eyelids. Wildes does not include our idea of using the directed edges gradients to isolate only downward edges for the upper eyelid, or only upward edges for the lower eyelid.

In [103], Wildes mentions depth of field as an important consideration when collecting eye data. Wildes felt that video rate capture was sufficient to avoid blur issues, especially because in his systems the subjects are consciously trying to keep their eyes steady.

Wildes implies that Daugman uses gradient ascent to fit the iris region. Others that implement the Daugman method suggest that it is an exhaustive process, and compare their performance optimizations to the this process [106]. In gradient ascent, parameters are modified based on the estimated derivatives to “zero in” on the location of the iris or pupil.

Wildes notes that parameter tuning is critical when using derivatives to find the iris/pupil: “In practice, this fine tuning of the image information has proven to be critical for accurate localization [103, p 1351](1997).”

**Camus’s Improvements of the DCID**

In 2002 Camus and Wildes wrote a paper titled “Reliable and Fast Eye Finding” [31]. In it, they made three observations:

1. At the circumference of either the pupil or the iris, there should be a local maximum in the radial derivative as a function of radius. This local maximum is because there is a sum over all angles.

2. Since the pupil, iris and sclera are relatively uniform, this local maximum should be nearly uniform at each angle.

3. The inside of the pupil-iris boundary and the inside of the iris-sclera boundary should be darker on the interior than on the exterior.

To model these ideas mathematically, Camus’s detector uses three separate terms:

1. A term that computes the weighted summed strength of the gradients over the boundary.

2. A second term for the uniformity of the gradients.
3. And a last term that causes the over-all degree-of-confidence to prefer darker interiors.

\[
\Sigma_{\theta=1}^{n} \left( (n - 1)\|g_{\theta,r}\| - \left( \Sigma_{\phi=\theta+1}^{n} \|g_{\theta,r} - g_{\phi,r}\| \right) - \frac{g_{\theta,r}}{8} \right)
\]  

(10.2)

This is Camus’s equation (2) [31], with \(n\) is the number of quantized values of the polar variable \(\theta\) used. Camus used \(n = 8\), which explains the 8 explicitly used in the equation. The term \(g_{\phi,r}\) is the directional derivative of the image intensity in the radial direction [31, p391].

For iris recognition Camus unwraps the image to a polar representation. In this representation, the pupil or iris boundary becomes a straight line at the radius of the circle. We note here that computing the gradient in the radial direction is straightforward for \(n = 8\), but becomes more complicated for higher values of \(n\).

Camus does not model the pupil as a pure circle, but allows for deformations due to pupil dilation and constriction process.

Camus computes his Degree of Confidence, Equation (10.2), for both the pupil and the iris, adds the two together, and then adds two constraints. First, the pupil is constrained to have an expected radius that is a certain ratio of the iris radius. In particular, if the iris-to-pupil radii ratio is greater than 4.0 or less than 1.75, then the quality measure is scaled by a factor of \((4.0/\text{ratio})^{3}\) or \((\text{ratio}/1.75)^{3}\), respectively. Secondly, the average DOC for the iris, \(DOC_{\text{iris}}\), should be greater than the DOC for the pupil, \(DOC_{\text{pupil}}\). When this fails, the total DOC is multiplied by \((DOC_{\text{iris}} - DOC_{\text{pupil}})^{2}\) to keep the pupil DOC from favoring only a very bright region, such as a specular reflection.

**Other Modifications of the DCID**

Other efforts have made modification to the DCID:

- Optimizing the algorithm [107](2008) by discarding values of the possible matches if any pixel on the pupil boundary is above a threshold. (Hebaishy also mentions issues with illumination in the pupil causing problems.)


- Making the computation faster by using hierarchical decomposition to the the darkest spot, and only computing the DCID for those locations [106] (2009).

- To improve the segmentation [109,110] (2012).

In their paper, “Weighted Adaptive Hough and Ellipsopolar Transforms for Real-time Iris Segmentation”, Uhl and Wild describe an approach where multiple passes are
used to segment the iris [110](2012). They use an adaptive Hough transform at multiple resolutions to estimate the center of the iris, and use downstream processing to find the ellipse of the limbus.

Daugman also describes the DCID in “How Iris Recognition Works” [32](2004), and gives some additional constraints, such as that he expects the pupil diameter to be 0.1 to 0.8 of the iris diameter.

In their Java implementation of Iris recognition, Roy et al. also used multiple passes over the image [111](2007), but assumes that the pupil is in the center of the image, which is not true for our data. Fu also uses the DCID in a multi-pass approach for iris segmentation, but assumes that the pupil is the darkest location in the image [112](2010), which is not true for our data.

10.3.2 Borrowing from Biometrics

As mentioned the field of biometrics has some significant differences from eye tracking systems. Some of the constraints on the biometric systems simplify the task, and as such are well justified for those applications.

In biometrics, if the iris/pupil is not identified in a few seconds, the subject can present their pupil again. In eye tracking, the image of the eye is ephemeral - changing quickly and needs to be captured and located as it is. If the image cannot be correctly analyzed, the pupil location for that frame will be lost.

Biometric systems often presume that the pupil is in the center of the image, and rely on it being a near circle, which is not necessarily true for our data.

10.4 Our Degree of Confidence [DOC]

Because we want a degree-of-confidence metric that works when the image is over exposed and under exposed we base the DOC on local statistics derived from the image itself.

We generate an over-all degree of confidence metric for pupils from component parts related to how uniform the pupil is, how much darker the pupil is than the region around it, and the edge strength.

10.4.1 Component calculations

In preparation for the calculation we find three regions of pixels: those inside the pixel ellipse, those outside the pixel ellipse, and the combination of these two regions.

We then compute:
μ₀ the average value of the pixels of the outside region
μᵢ the average value of the pixels of the inside region
μᵥ the average value of the pixels of two combined regions

σ₀ the standard deviation of the pixels of the outside region
σᵢ the standard deviation of the pixels of the inside region
σᵥ the standard deviation of the pixels of the combined region

We then also compute the average standard deviation of the inside and outside regions:

\[ σ_{\text{avg}} = \left( σᵢ + σ₀ \right) / 2 \]  \hspace{1cm} (10.3)

10.4.2 Contrast

Ideally the pupil inside would be much darker than the iris around the pupil. To model this mathematically as something that is proportional to a good pupil, we define a term called contrast, or C.

\[ C = μ₀ − μᵢ \]  \hspace{1cm} (10.4)

This separation term relates to how well separated the pupil is from its surrounding. Intuitively, a high contrast value is good because the iris is brighter than the pupil. Unfortunately, having the inside of a region darker than the outside does not guarantee that the region is a pupil.

10.4.3 Incoherence

We then define a term that relates to how incoherent the entire region is. Here we define incoherence or K as:

\[ K = σᵥ / σ_{\text{avg}} \]  \hspace{1cm} (10.5)

We interpret the value of the incoherence by cases as follows:

Case 1: If the area is uniform and the same inside as outside, then the standard deviation of the entire region is the same as the average of the inside and outside standard deviations. In this situation term goes to 1.

Case 2: If the pupil is well defined, and uniform on the inside and uniform on the outside, but the inside is a dark pupil and the outside is a brighter iris, then the standard deviation of the entire region will be considerably higher than the average of the inside and the outside. This will cause the value of the incoherence term to go above 1.

Intuitively, the larger this term is, the better the pupil is.
10.4.4 Separation

Separation, or $S$, is similar to contrast, but is normalized by the standard deviation of the pixels in the region:

$$S = \left( \frac{\mu_o - \mu_i}{\sigma_c} \right) \quad (10.6)$$

Some regions are dark, due to eyebrow hairs or first-surface reflections off the cornea. If the standard deviation in the region increases due to variations in surface texture, the value of the separation will drop.

For bad pupils, and partial pupils, this term decreases quickly. The separation especially drops for dark regions that are caused by eyebrow hairs because the standard deviation of hairs is higher than nearly uniform skin.

10.4.5 Relative edge strength

We also compute the average edge magnitude for the entire frame. From this we can compute the z-score for edge strength.

The pixels in the region near the edge of the ellipse should have a higher than average edge strength. Similar to Daugman’s work, the purpose of this is to reward strong edges around the pupil.

We find the pixels in the image that are within one pixel of the edge of the proposed pupil. For these pixels we find the fraction that are above a z-score of 1.5. The value of 1.5 was selected and validated such that compared to all the other edges in the frame, edges above this value are more likely to make a contribution to a pupil edge than not.

We define the symbol $E_f$ as this fraction of the candidate pupil edge that is well above the average edge strength for the image.

10.4.6 Ensemble combination experiment

The relative weights of the terms that contribute to the DOC are important. We want terms to contribute, but we do not want one term to dominate the others.

Intuitively, since we want to detect dark pupils, we would want to use a large amount of contrast in the ensemble mixture. However, using just contrast, or too much contrast, favors situations where the darkness of the inside is caused by the wrong phenomenon. If too much contrast is used then other factors, such as the edge strength, are not considered.

The separation value of blobs of dark eyebrows decreases faster than the incoherence term. Because of this, the ratio $S/K$ can be used to help differentiate eyebrow hairs from stronger pupil.

The final mixture was determined using sequence from a subject for which eyebrows were frequently confused with the pupil.
We used frames 17400 to 18404 for subject 3 from the 2013 video in Death Valley, skipping nine frames between samples. The code was temporarily modified to only use every tenth frame, and a breakpoint in the debugger was used to stop execution. This provided 100 frames for a difficult subject to hand inspect.

The default values for the program were used (in particular, the minimum Hough score was 0.2). The number of times that the DOC score did not select the pupil, and the number of times it selected eyebrows was recorded. If the DOC caused an eyebrow hair to be selected, and caused the pupil to be missed, then both errors occurred, and that was counted separately.

The following data resulted:

<table>
<thead>
<tr>
<th>DOC Vers.</th>
<th>Misses Pupil</th>
<th>Eyebrow hits</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>934</td>
<td>0</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>935</td>
<td>1</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>936</td>
<td>0</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>937</td>
<td>0</td>
<td>48</td>
<td>0</td>
</tr>
<tr>
<td>971</td>
<td>0</td>
<td>31</td>
<td>0</td>
</tr>
<tr>
<td>972</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

Experiments showed that the components interacted. Some of the discoveries were counter-intuitive. The original formulation for version 934 included factors such as the number of pixels outside the candidate pupil that were above the average value in the pupil. Prior experimentation had found these beneficial, in selecting the pupil from the corner of the eyes. For the final DOC the most important aspect is differentiating between the pupil from the eyebrows because the corners of the eyes are differentiated by through strange ellipse detection.

10.5 Final selection for DOC

The final formulation used as the DOC metric is:

\[
DOC = \left(2 \times C + \frac{S}{K} + \frac{2 \times E_f - 1}{3}\right) \times \frac{2}{7}
\] (10.7)

The ensemble mixture starts with twice the contrast. It was found experimentally that increasing this factor above two caused the term to dominate and other terms became insignificant. Without enough contrast non-pupils were selected.

The term \(S/K\) is the separation over the incoherence. As mentioned, this term prefers solid objects over textured objects. This is the primary factor that causes pupils to be favored over eyebrows.
The term involving the edge fraction, $E_f$, helps favor strong pupil edges. If more than half the edge pixels have a strong edge strength, then this term goes above one. Otherwise the term is below one.

While the edge fraction term is helpful, it is scaled down by a factor of three because it is possible to have strong edges that are not pupil edges. For example, eyebrow hairs have strong edges, but those edges are not beneficial to differentiating them from pupils. Without scaling this factor down the strong edges in the region of curled eyebrow hairs and eyelashes could cause them to be favored over the pupil.

The entire DOC is scaled down by a factor of $\frac{3}{7}$ to avoid having the value go to 1.

A DOC of 1 is significant within the program as it is used to indicate that the pupils were hand-coded by the operator, or were hand-inspected by the operator during the manual validation phase.

Unlike Daugman’s work, in which the DCID is used to find the pupil in the image, we have an upstream process that selects pupil candidates. The purpose of our degree of confidence metric is to select the correct candidate as frequently as possible.

Overall, this DOC favors pupils over distractions, and results in most valid pupils having a DOC in the range of 0.45 to 0.55.
Ch. 11

Comparison of Human Analysts

11.1 Intra-Analyst Comparison

Before comparing the program to an analyst, we wish to determine how consistent an analyst can be.

To determine this, we asked an expert analyst to code one video three different times. The analyses were performed at least a week apart.

The video was Lone Pine, 2013, subject number 05. The frames processed were 23,170 to 25,829.

Of the 2,660 frames, 412 were coded as blinks in at least one file. This is a nearly 15% blink rate for the video, and is a fairly high rejection rate. (See Table 11.1.)

Table 11.1: Comparison of blink classification within one analyst from run to run.

<table>
<thead>
<tr>
<th>Analyses being compared</th>
<th>Number of blink frames that agree</th>
<th>Number of blink frames that disagree</th>
</tr>
</thead>
<tbody>
<tr>
<td>A and B:</td>
<td>397</td>
<td>8</td>
</tr>
<tr>
<td>A and C:</td>
<td>397</td>
<td>14</td>
</tr>
<tr>
<td>B and C:</td>
<td>397</td>
<td>10</td>
</tr>
</tbody>
</table>

Figures 11.1 to 11.3 show the distance from the pupil location in each video to the average of the other two videos. The only frames that were used for comparison were those that were not coded as a blink in any of the three coding runs.
Figure 11.1: Distances for Video A to the average of other two video analyses.

Figure 11.2: Distances for Video B to the average of other two video analyses.
Figure 11.3: Distances for Video C to the average of other two video analyses.

This demonstrates that one analyst can be very self-consistent in this situation. A particular analyst seems to process a given video the same way, regardless of the time that elapses between analysis runs. It may be important to note that the analyst for this comparison effectively served in the role of lab manager, took the task seriously, and was a very professional analyst.

11.2 Inter-Analyst Comparison

Given that one analyst is self-consistent, the next natural question that arises is, “What is the error between human analysts?”

To answer this question we had one eye video coded by three different analysts. At least two of these analysts were undergraduate student workers.

For the three different analysts, Figure 11.4 shows a histogram of the analysts that agree the most, the median distance between the three analysts, and the worst-case distance between any of the three analysts. This makes it more apparent that there is variation between analysts.
CH. 11. COMPARISON OF HUMAN ANALYSTS

At first glance, these analysts seemed to also be fairly self-consistent. However, the initial analysis did not include blinks that differed from analyst to analyst.

The following figures plot all of the x values for analyst 1 versus the x values for analyst 2, y values for analyst 1 versus the y values for analyst 2, x values for analyst 1 versus the x values for analyst 3, y values for analyst 1 versus the y values for analyst 3, x values for analyst 2 versus the x values for analyst 3, and y values for analyst 2 versus the y values for analyst 3.

For blinks, the x and y values are set to a negative value. Because there is no jitter added to the points, the point at (-25,-25) is a repeated point. These show that there was significant disagreement between the three analysts as to which frames constituted a blink.

Figure 11.4: Figure showing how well Analysts Agree
Figure 11.5: Analyst 1 vs 2 X values

Analyst 01 coded 569 blinks, Analyst 02 coded 414 blinks.
Analyst 01 coded 569 blinks, Analyst 02 coded 414 blinks.

Figure 11.6: Analyst 1 vs 2 Y values
Analyst 01 coded 569 blinks, Analyst 03 coded 557 blinks.

Figure 11.7: Analyst 1 vs 3 X values
Analyst 01 coded 569 blinks, Analyst 03 coded 557 blinks.

Figure 11.8: Analyst 1 vs 3 Y values
Analyst 02 coded 414 blinks, Analyst 03 coded 557 blinks.

Figure 11.9: Analyst 2 vs 3 X values
Analyst 02 coded 414 blinks, Analyst 03 coded 557 blinks.

Figure 11.10: Analyst 2 vs 3 Y values
Ch. 12

Results

The program allows the user to hand inspect and refine the results as much as desired. It is entirely possible to generate perfect results given an infinite amount of time because the location of the pupils would default to hand locating every single pupil in every single frame of the video. Clearly, this is not the intent of this work.

Our goal is to present to the user three good candidate pupils for the user to select from, and have one of them be the correct pupil as often as possible. Given three pupils to select from the user can simply choose the best one, and immediately go on to the next frame.

The ultimate goal is to have the first choice presented be the best choice for the pupil as often as possible. In this situation the user just pushes the right arrow, and goes on to inspect the next frame of the video.

Unfortunately there are situations where it is not possible to automatically choose between the silhouette of the eye camera and the real pupil. In such situations it is sufficient to present the three choices as options to select from, and the user can then click one of three keys on the keyboard to immediately indicate the correct pupil choice, and proceed to the next frame of video. This aims to minimize the amount of hand motion on the part of the user.

Lacking a sufficiently good choice, the user can over-ride the presented choices and hand enter a better location by clicking on the perimeter of the pupil in five or more points. This does require that the user switch from using the keyboard to using the mouse. In many user interfaces having the user switch from keyboard to mouse slows the processing considerably, and as such is usually avoided. In reality, this is remedied by having the user use both hands.

The results that follow are processed to various degrees of processing.

In some cases the results were collected before some of the more recent innovations were implemented. As such, in each situation we describe the state of the program at the time the data was collected. In some cases this serves to demonstrate the incremental
benefit that the innovations provided.

12.1 Olema

12.2 Olema 2012 Subject 6

Comparing results for our program to analysts involves trade-offs of time and accuracy. Our program allows the user to set a threshold for frame inspection, based on the degree of confidence of the candidate pupils found. By setting the frames to inspect to only those with a degree of confidence below this threshold only a sub-set of the total frames need to be inspected.

The frames considered for South of Olema 2012 Subject 6, were frames 18,157 to 21,755. We found experimentally that when the DOC is over 0.425, the pupil is usually valid. In this case, by setting the program so that only frames with a degree of confidence under 0.425 were inspected, only 171 frames needed to be checked.

These 171 frames could be visually inspected and corrected in 14 minutes. In this case the user was inspecting and correcting roughly 12 frames per minute. However, since this 14 minutes was sampled from 3,599 frames of video the effective processing rate is 257 frames per minute.

After this 14 minutes of inspection, 96.4% of the frames were within 2 pixels of a human analyst. In this case the threshold degree of confidence of 0.45 selected less than 5 percent of the frames. For studies requiring more accuracy the program could be set to show all frames by using a higher threshold degree of confidence for inspection. This higher threshold would present more than 171 frames to the user for validation, and would result in a better accuracy but at a slower time.

Setting the threshold for the degree of confidence to 1.0 would cause every frame to be inspected by the user. This mode of operation would be almost the same as the custom build software described for manual inspection at the start of Section 1. The difference however, is that our system presents three candidate pupil choices for the user to select from using the push of a single button.

Table 12.1 shows a comparison to the same subject, after 14 minutes of hand inspection. The frames considered were frames 18,157 to 21,755.
Table 12.1: Results for South of Olema 2012, Subject Number 06 after 14 minutes of hand inspection and user interaction.

<table>
<thead>
<tr>
<th>Distance in pixels</th>
<th>Approximate Visual Degrees</th>
<th>Percent with this distance After 14 minutes of Adjustment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>88.0</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>92.3</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1.0</td>
<td>96.4</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>1.7</td>
</tr>
</tbody>
</table>

12.3 Lone Pine Data Analysis

12.3.1 LonePine 2013 Subject 3

The following results were obtained for subject number 3, from 2013 at the Lone Pine site. For Frames 28,000 to 30,620 were analyzed.

In terms of open eyes versus blinks, the program identified 2,615 true positives, 4 true negatives, 1 false positive, and 1 false negative. This is a precision $(TP/(TP + FP))$ of 1.00, and a recall $(TP/(TP + FN))$ of 0.999.

Table 12.2 show the results. The initial program generated 87.7 of the first choice locations within 2 pixels (approximately 1 degree) of the ground truth location. Over 2.5 percent of the pupils were over 5 pixels (approximately 2.5 degree) away from the ground truth location.

Table 12.2: Results for Lone Pine 2013, Subject Number 03

<table>
<thead>
<tr>
<th>Approximate Degrees</th>
<th>For First Initial Pupil Choice</th>
<th>After 81 minutes of human interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>77.8</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>80.6</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1.0</td>
<td>87.7</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>5.2</td>
</tr>
</tbody>
</table>

The entire video section was hand examined in 81 minutes (45+36 with a seven minute break). This is a processing rate of approximately 32 frames per minute, a rate only possible because the first choice was frequently correct.

After hand examination, 94.7 percent of the pupils were located within half a degree of the ground truth location, and 98.5 percent of the pupils were located within one degree of the ground truth location.
12.3.2 LonePine 2013 Subject 5

Table 12.3 gives the results for subject number 5, from 2013 at the Lone Pine site in 2013. For this video, frames 23,170 to 25,829 were analyzed.

Table 12.3: Results for Lone Pine 2013, Subject Number 05

<table>
<thead>
<tr>
<th>Approximate Degrees</th>
<th>For First Initial Pupil Choice</th>
<th>After 90 minutes of human interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>77.4</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>83.4</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1</td>
<td>92.3</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>1.8</td>
</tr>
</tbody>
</table>

At the time this data was collected, the program was using the advanced noise cleaning, and progressive connectivity to find the pupil. This data did not include the use of strange pupil elimination, because for this subject that extra advancement was not imperative.

Compared to the analysts, this 2,659 frame sequence generated roughly 141 false alarms – situations where the program thought that the eye was open, but the analyst thought that it was a blink. This seemed excessively high. To investigate this, we ran the inspection routine, stopped it in the debugger, and reset it to look at all frames where the program thought there was an open eye, but the analyst thought it was a blink. Of these 141 frames, 121 were found to have an open eye looking out. The previous chapter on the accuracy of the analysts did not reveal this issue, but there is certainly an opportunity for some analysts to accidentally label open eyes as blinks and vice versa.

12.3.3 LonePine 2013 Subject 8

Subject number 8 has an epicanthic fold, or an extra fold of skin in the upper eyelid.

During early work, this extra fold caused some challenges. For example, when trying to find the upper eyelid by fitting a parabola to the strongest edges found, the fitting routine would sometimes find the wrong curve. This issue was resolved by fitting a straight line instead of a parabola when finding eyelids. Table 12.4 summarizes the results for this subject.

For Frames 21929 to 24574 the program correctly classified 94.8 percent of the frames as either open eyes or blinks. The number of true positives was 2,372. The number of true negatives was 137. There were 135 false positives and 2 false negatives. This resulted in a precision of 0.946 and a recall of 0.999.
Table 12.4: Results for Lone Pine 2013, Subject Number 08.

<table>
<thead>
<tr>
<th></th>
<th>Approximate Degrees</th>
<th>First Initial Results</th>
<th>After 44 Mins of human interaction</th>
<th>After 67 Mins of human interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>86.8</td>
<td>91.3</td>
<td>94.8</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>90.6</td>
<td>94.3</td>
<td>98.0</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1.0</td>
<td>95.0</td>
<td>96.1</td>
<td>99.4</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>2.6</td>
<td>2.6</td>
<td>0.10</td>
</tr>
</tbody>
</table>

12.4 Death Valley

The Death Valley National Park, and the eye-tracking data collected at Badwater Road Mounds Roadside (in 2013) was especially problematic. After several attempts this was considered impossible to process, and was set aside as too difficult of a challenge.

The lighting was very harsh at this site. Subjects that moved their heads caused sudden changes in illumination that rendered the image either too dark or too bright.

With the development of the Strange Ellipse Rejection method, and a way to set the parameters for it, these video sequences become more tractable for automatic processing.

For brevity, in the following tables we use the abbreviation SER for Strange Ellipse Rejection.

To provide a range of examples of the processing of eye-tracking videos, both easy and difficult, a video from the Death Valley was selected. As will be shown, with the development of the Strange Ellipse Rejection algorithm it was necessary to find an even more difficult example of an eye-tracking video.

12.4.1 Death Valley 2013 Subject 3

Frames 15,797 to 18,407 were processed. The blink detector classified 96.7 percent of the frames correctly.

Referring to Table 12.5, we can see that without Strange Ellipse Rejection the program was only finding 47.8 percent of the pupils within 2 pixels of ground truth. Furthermore, it was effectively missing more than 30 percent of the pupils.

With the advent of the Strange Ellipse Rejection method, many of the distracting eyebrow hairs were removed. This caused the fraction of pupils found as a first recommendation within 2 pixels of ground truth to jump to 72.0 percent, up from 47.8 percent.
Table 12.5: Processing accuracy for Death Valley, Subject Number 03, 2013, with SER.

<table>
<thead>
<tr>
<th></th>
<th>Approx. Degrees</th>
<th>First Pupil w/o SER</th>
<th>First Pupil with SER</th>
<th>Closest Pupil with SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>28.8</td>
<td>43.2</td>
<td>67.1</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>35.2</td>
<td>52.7</td>
<td>76.3</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1.0</td>
<td>47.8</td>
<td>72.0</td>
<td>90.0</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>30.8</td>
<td>6.2</td>
<td>2.1</td>
</tr>
</tbody>
</table>

The first recommended pupil is important because if it is reliably on target, the user can quickly scan through the video and verify that the correct pupil was found without having to move a single finger from the keyboard.

Almost as efficient as having the first suggestion correct, is having any of the first three recommended pupils be correct because the user can rest three fingers on the key board, and push one of the three keys to select the correct candidate pupil. This interface was developed by Pelz and Pontillo [5]. The interface was reused here because of its success and efficiency.

We must acknowledge that comparing the first suggested pupil location to any of the closest of the first three pupil locations is not a commensurate comparison. However, when we consider that this was intended to be the difficult video for comparison to the other data, the improved use of the Strange Ellipse Rejection algorithm is apparent.

### 12.4.2 Death Valley 2013 Subject 7

To provide a particularly difficult example, we use subject 7 from the Death Valley 2013 site.

Subject number 7 from the Death Valley site was difficult due to lighting conditions, and because the subject was squinting significantly. Furthermore, subject 7 wore prescription glasses. To accommodate these glasses a cut was made in the glasses in the lower right side, and the eye camera was mounted looking up at the pupil, under the glasses. This meant that the eye camera was trying to observe the pupil over the lower eyelid, under the bottom of the glasses, and in between squinting eyelids.

Subject 7 also had dark eyelashes, and wrinkles in the skin (causing additional edges).
Figure 12.1: Selected frame 8315 from Death Valley, 2013, Subject 7. This shows that the eye camera is looking up over the bottom of the bottom eyelid. The red circle is the initial location that the program suggested as the first pupil location. For this example the program was correct.

Figure 12.1 shows an example frame from this video. Frames 8,000 to 8,600 were processed for these results.

Within this range, for frames 8,000 to 8,300 the subject was squinting too hard for the algorithm to automatically lock onto a pupil. Instead, the program tended to fit circles between opposing eyelids.

Frames 8,300 to 8,650 could be processed automatically and generated better results for this sub-range. Table 12.6 gives some of the values found.

When considering Table 12.6, it is important to keep in mind that the last column of data is a sub-set of the second-to-last column.

Perhaps the most revealing number from Table 12.6 is the portion of the frames in which closest pixel is more than 2.5 degree away from the desired location. Notice that this drops significantly for the sub-range.
Table 12.6: Processing accuracy for Death Valley, Subject Number 07, 2013 with SER.

<table>
<thead>
<tr>
<th>Percent within Degrees</th>
<th>Closest Pupil for Frames 8,000-8,299</th>
<th>Closest Pupil for Frames 8,300-8,600</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00 pixels</td>
<td>0.5</td>
<td>12.8</td>
</tr>
<tr>
<td>1.25 pixels</td>
<td>0.6</td>
<td>13.8</td>
</tr>
<tr>
<td>2.00 pixels</td>
<td>1.0</td>
<td>18.1</td>
</tr>
<tr>
<td>over 5.00 pixels</td>
<td>2.5</td>
<td>70.2</td>
</tr>
</tbody>
</table>

These results were included as examples of especially challenging video sequences. However, the numbers alone do not demonstrate some of the more dramatic results. There were still some frames in which the automated process found the pupil in spite of dramatic lens flare, as shown in Figure 12.2.

Figure 12.2: Selected frame 8551 from Death Valley, 2013, Subject 7, showing a correct pupil detection in spite of significant lens flare.
12.5 Initial Results

With the development of *strange ellipse rejection*, the initial results were improved. Here we report the most recent results for two subjects, results that incorporate the most recent algorithm improvement.

For each frame the program seeks two minimize two goals. Firstly, it seeks to minimizing the distance between the *first* proposed candidate pupil – because it improved the accuracy of the first choice.

Secondly, it seeks to minimize the distance to any of the first three candidate pupils – because it is easy for the user to select one of these three so this decreases manual inspection and processing time. When reporting this second measure, we call this the *closest* choice.

The following subsections demonstrate these improvements.

### 12.5.1 Lone Pine 2013 without and with SER

Table 12.7 gives the accuracy for Subject 3 (SN03) without and with strange ellipse rejection. Similarly Table 12.8 gives the accuracy for Subject 5 (SN05) without and with strange ellipse rejection.

The comparison between the first initial pupil choice without and with SER is a fair comparison. Again, comparing the first initial pupil choice without SER to the closest initial pupil choice with SER is not a fair comparison.

Comparing percent correct has the problem that the number is not a linear relationship. A student that scores a 90 percent on an exam missed one question out of ten. A student that scores a 95 percent on an exam missed one question out of twenty. In this example, the false alarm rate is halved, but the score only improves five percent.

<table>
<thead>
<tr>
<th>Approximate Visual Degrees</th>
<th>For First Initial Pupil Choice Without SER</th>
<th>For First Initial Pupil Choice With SER</th>
<th>For Closest Initial Pupil Choice With SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>77.8</td>
<td>80.3</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>80.6</td>
<td>84.5</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1.0</td>
<td>87.7</td>
<td>91.5</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>5.2</td>
<td>4.0</td>
</tr>
</tbody>
</table>
Table 12.8: Processing accuracy for Lone Pine, Subject Number 05, 2013 without and with Strange Ellipse Rejection [SER].

<table>
<thead>
<tr>
<th>Approximate Visual Degrees</th>
<th>For First Initial Pupil Choice Without SER</th>
<th>For First Initial Pupil Choice With SER</th>
<th>For Closest Initial Pupil Choice With SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent within 1.00 pixels</td>
<td>0.5</td>
<td>77.4</td>
<td>84.6</td>
</tr>
<tr>
<td>Percent within 1.25 pixels</td>
<td>0.6</td>
<td>83.4</td>
<td>88.9</td>
</tr>
<tr>
<td>Percent within 2.00 pixels</td>
<td>1</td>
<td>92.3</td>
<td>96.4</td>
</tr>
<tr>
<td>Percent over 5.00 pixels</td>
<td>2.5</td>
<td>2.5</td>
<td>0.4</td>
</tr>
</tbody>
</table>
Ch. 13

Future Work

The critical insight to our development was that the user should validate the candidate ellipses that come out of the first pass of sampling the most stable frames. These frames are roughly one frame every thirty frames. Unfortunately, this critical insight did not occur until a late in our development.

Once this critical insight occurred, and we had ground truth data for this subject in this video many opportunities became available that were not previously available to a fully automated pupil identification method. Ideas for innovation occurred faster than they could be implemented and tested.

We capture some of these ideas in the following sections.

13.1 Tracking

Previously, we used tracking to follow the pupil for several seconds worth of video data [113]. This work used model generation and an offset-correlation detector to locate the component parts of the object (i.e. pupil) being tracked. The offset-correlation detector is based on the Fourier shift theorem that a spatial translation by a fixed amount causes a linear phase shift by a relative amount [114, p287].

However, tracking added several issues that we did not have solutions for at the time. Any time a tracking is used, the program needs to be ready to handle situation when the tracking loses track and starts tracking the wrong object. For example, if the tracking algorithm switched to following the eye camera instead of the pupil. In such situations a program needs a way to detect when it has lost track, and a way to recover from the loss by re-acquiring the object. At the time we lacked both of these.

Now, with the availability of pupils that have been validated by the user from the first detection pass, we would know the pupil location, shape, and ellipse parameters at a future frame. This information could be used to detect when the tracking algorithm had lost track. Additionally, the validated pupils would allow the program to re-initiate the
track from the next valid pupil, and possibly even track the pupil backwards sequentially through the frames.

Furthermore, we had a version of the Hough detector that preferentially weighted the pupils that were of the same size and edge strength as the pupil in the last frame. This improved the accuracy of the pupil detector. However this also suffered from tracking loss, and at the time we had no way of recovering and resetting the expected edge strength of the pupil edge.

Now, as a result of our simulations, we realize that the major axis of the pupil ellipse is relatively constant from frame to frame, even in spite of refraction. This information could be incorporated into an improved degree of confidence measure that used the knowledge that at the major axis should remain relatively constant.

The frame number of the validated pupils could also be used in a degree of confidence measure. A degree of confidence measure could be developed such that it was very sure of the pupil parameters at the points where the pupils had been validated, but had relaxed confidence for frames further from these validated pupils.

13.2 Best Mixture of Characteristics for a Degree of Confidence

The development of our degree of confidence was inspired by other works derived from Daugman’s work for locating pupils for biometric identification [99]. Chapter 10 discusses the components of the degree of confidence metric that we use to judge the quality of the candidate pupils in each frame of the video.

During development of this degree of confidence we lacked ground truth information concerning the exact ellipse center and the other ellipse parameters. We had only the pupil location and approximate pupil size from the analyst. The relative importance of the components in the ensemble degree of confidence were developed using the method of adjustments. Now that we have validated ground truth data for multiple videos of different observers at different sites, we could use that data with the positive and negative examples that have been collected, to develop a better ensemble mixture of those components.

Given that we now have hand validated videos, we could certainly envision developing a better degree of confidence for pupil selection based on an improved weighting of the features, or improved features.

13.3 Edge Estimation

The present Hough detector uses the standard Sobel edge detector for estimating edge strength and direction. This works on a $3 \times 3$ region of the image, and is particularly sensitive to noise. It would be interesting to explore using the same the $5 \times 5$ edge
detector that was developed for eyelid detection, in Section 3.2, but for pupil detection. Changing the edge detector would impact other aspects of the imaging chain. A thorough testing of the impact of changing edge detectors is left to future work.

13.4 Discussion

While this work solved some issues with pupil detection, many other potential solutions were revealed. In addition, the solutions presented here could all be improved by increasing accuracy and decreasing processing speed. Converting from an interpreted language to a compiled language would immediately help improve processing speed.

The availability of ground truth pupil locations that are roughly every thirty frames in the video also opens up the possibility of using parallel processing. The video could be distributed in small segments across multiple servers and processed using parallel concurrent processing.
Ch. 14

Conclusions and Key Findings

This thesis has explored techniques to improve the speed of automatically detecting pupils in eye-tracking videos that are recorded outdoors in the presence of full sunlight. The presence of full sunlight constricts the pupil and complicates the task of automatically locating the pupil using standard techniques.

These improvements include:

- a framework for using multiple passes over an eye video so that most of the processing can be performed offline,
- a technique for masking off the invariant regions of pixels throughout the video,
- a method for identifying the eyelids in the image of the eye video,
- a technique for adaptive noise cleaning of the image that preserves and enhances the pupil-to-iris edge,
- a version of the Hough transform that is optimized for detecting eye pupils,
- a method for building a model of each subject’s eye so that the image of the eye can be unwrapped from an ellipse into a circle and then detected as a circle,
- a method for following the contour of the detected pupil to fit an ellipse,
- and a degree of confidence metric for automatically ranking the detected pupils to identify the most likely pupil.
14.1 Contributions of this Thesis

14.1.1 Multiple Sampling Passes

In Chapter 2 we described a multiple pass system for processing the video frames of the eye. An initial pass collects statistics that are then used by all the subsequent passes. Additional passes include user interaction to validate intermediate results. Validated intermediate results allow the system to learn a model of the subject’s eye geometry and rules to automatically exclude some candidate pupils based on their ellipse parameters. Subsequent pupil detection benefits from this model and these rules to better determine three candidate pupils per frame.

A final user inspection pass allows the user to inspect a subset of the frames to validate, based on the degree of confidence metric assigned to the candidate pupils in the frame. The top three suggested pupils are presented to the user, and one of these can be chosen with a single key press. Alternatively, during this final inspection the user may override the program’s suggestions and manually input a pupil location.

Section 2.2.1 discusses a method that uses the initial statistics collected in the first statistics pass to identify regions of each frame that are stable throughout the video sequence. A bit mask is developed to remove these invariant pixels throughout the video. This masking reduces the amount of edges that must be considered by the Hough circle detector.

Section 2.2.2 describes how the initial statistics of the video sequence are used to identify blinks. Additionally, the video statistics are also used to determine which frames are most likely to contain stable open eyes.

A subsequent pass takes advantage of knowing which frames are stable frames, and attempts to locate candidate pupils in each frame. Once verified, these pupil ellipse locations are used to build models of the subject’s eye geometry, expected pupil edge strengths, and rules about pupil ellipse orientations.

14.1.2 Eyelid Removal

Chapter 2 describes an eyelid location and removal technique that further reduces the amount of the image that must be searched for a pupil. This algorithm uses a more aggressive $5 \times 5$ edge detector, projected histograms of horizontal edges, and a set of heuristics for finding the eyelids and rejecting the region outside of the eyelids.

The eyelid detector had trouble with only 0.175 percent of the eyes in the eyelid test suite. The eyelid detector helps reduce false pupil candidates by removing regions containing eyebrow hairs. Again, this can be seen as a technique for noise removal.
14.1.3 Adaptive Noise Cleaning

Chapter 4 describes an adaptive noise cleaning technique, inspired by Kuwahara [15]. The technique reduces small edges in the image, and simultaneously enhances the magnitude of the pupil-to-iris edges.

The technique uses an intentionally incomplete set of matching functions, and so intentionally causes blurring of eyelashes and small objects reflected off the first-surface of the cornea. This helps remove edges caused by compression artifacts, eyelashes, and first-surface reflections.

The results of this adaptive noise cleaning routine was compared to standard techniques such as median filtering and Gaussian blurring; both of these standard noise reduction techniques also reduce the contrast of a pupil-to-iris edge, whereas our technique enhances the pupil-to-iris edge.

This noise cleaning is used to reduce or remove nuisance edges before the image is run through the Hough circle detector.

As discussed in Section 4.13, replacing this adaptive noise cleaning with a bilateral filter caused the number of false pupil candidates coming out of the Hough circle detector to jump from a mode of 15 candidates to a mode of 55 candidates. Each of these pupil candidates that come out of the Hough transform must be considered by the downstream processing and ranked by the degree of confidence measure. The adaptive noise removal technique generates fewer candidates to process.

Furthermore, the adaptive noise removal technique runs in 1.2 seconds, while a bilateral transform requires over 20 seconds per frame due to the per-point computations required for the bilateral transform.

14.1.4 Focused Hough Transform

Chapter 5 described a version of the Hough transform that is optimized for detecting eye pupils. It uses the direction of the edge gradient to reduce the amount of noise in the Hough accumulator space. As described in Section 5.4, a novelty is that the Hough detector incorporates tolerance so that elliptical pupils are more likely to be detected.

14.1.5 Unwrapping the Eye

Chapter 6 describes a method for using the sampled pupils collected and verified during the early passes to build a model of the subject’s eye. This model enables the image of the eye to be “unwrapped” so that off-axis pupils change from an ellipse shape to a circle shape, while on-axis pupils remain circular.

This non-linear transformation causes elliptical pupils to be detectable as a circle, using a Hough circle detector.
The contribution here is a different method for calibration. Instead of using calibration targets, we use samples of the subject’s own pupils, from the eye video itself, to determine an unwrapping model. This allowed us to generate an unwrapping model, years after the original eye tracking data was collected.

14.1.6 Strange Ellipse Rejection

Chapter 7 discusses a side benefit of having a model of the subject’s eye.

Using the pupils that were rejected by the user, we can form classifiers that allow us to automatically reject pupils that do not conform to the model of the subject’s eye. This process is termed “strange ellipse rejection.” This help sort through the candidate pupils ellipses that come out of the Hough detection process and automatically eliminate candidate pupils that could not have come from the subject.

Section 12.5 shows the improvements generated through the use of this “strange ellipse rejection” technique. For some subjects this increases the portion of the pupils that are within one degree of the analysts’s center for the pupil to 95%.

14.1.7 Benefits of Combining Multiple Features

Chapter 8 discusses the benefits of being able to combine multiple features to select the sub-set of pixels that are most likely on the edge of the pupil. This advantage is used during the process of refining the edge of the pupil-to-iris boundary.

14.1.8 Pupil Contour Extraction

Chapter 9 discussed a technique we termed progressive connectivity which is the method for refining the pupil-to-iris boundary. It allows the edge strength to change locally, as when the pupil is hidden behind first surface reflections, but still allows the pupil’s perimeter to be recovered.

14.1.9 Degree of Confidence Metric

Chapter 10 described a degree of confidence metric for automatically ranking the detected pupils to identify the most likely pupil. This degree of confidence metric was developed to work in outdoor eye tracking applications, where the image of the pupil is corrupted by noise and first surface reflections.

14.1.10 Results

Experiments using just the initial first Hough circle detection pass on all frames only found pupils in 66 to 77 percent of frames (Section 6.1). The multi-pass system, with unwrapping, identified at least one candidate pupil ellipse in every non-blink frame.
Chapter 11 compares the differences between analysts. Intra-analyst results for one analyst were very consistent. Inter-analyst results were also consistent in position, but inconsistent about which frames should be classified as blinks.

Chapter 12 gives the results compared to analysts. The system generated more reliable blink classification, while simultaneously delivering over a 10 times reduction in analyst processing time.

14.2 Closing

The problem of identifying pupils in videos of eyes for eye-tracking has been approached using an ensemble of techniques that work together to minimize classification error and user interaction time.

Unlike manual techniques, each pupil located has an associated degree of confidence that serves as an independent measure for the quality of the pupil detection. This allows the user to selectively inspect only those pupils which are likely to be incorrect.

The developed user interface allows the user to quickly select one of three candidate choices presented, or over-ride those three suggestions and manually enter an alternative pupil location. This enables the rapid selection of the correct pupil choice, usually with the single keystroke. Since the user can refine these choices, the user can improve the pupil location accuracy until it is arbitrarily close to those achieved using fully manual pupil location.

The implementation of these techniques provides the user with an alternative to either fully manual or fully automated pupil location for outdoor images of eye-tracking subjects in natural light. The results demonstrate comparable pupil location accuracy at a frame rate that is at least ten times faster than the previous rate.
Appendix A

Publications and Presentations

“Progressive Connectivity and Using the Curse of Dimensionality in your Favor”, Kinsman, Thomas B., RIT Graduate Symposium, April 18, 2014. (Oral) Received award for Best Overall Presentation.


“Location by Parts Model Generation and Feature Fusion for Mobile Eye Pupil Tracking Under Challenging Lighting, Kinsman, T., and Pelz, J., Pervasive Eye Tracking and Mobile Eye-Based Interaction at the 14th International Conference on Ubiquitous Computing, Pittsburgh, PA, 2012 (Oral)

“Efficient tagging of visual fixations from mobile eye trackers using hierarchical

“Hierarchical image clustering for analyzing eye tracking videos,” Kinsman, T.; Bajorski, P.; Pelz, J.B., IEEE Western New York Image Processing Workshop (WNYIPW), Nov. 2010 (Poster)


“Hierarchical image clustering for analyzing eye tracking videos,” Kinsman, T.; Bajorski, P.; Pelz, J.B., IEEE Western New York Image Processing Workshop (WNYIPW), Nov. 2010 (Poster)


Appendix B

Misperceptions and Challenges

One of the most frustrating aspects of building a computer vision task to detect pupils under challenging situations is that to a human observer this is an easy task.

It is challenging to convince the reader that the task is not simple. These misperceptions must be overcome.

The challenges and misperceptions are described here.

B.1 Humans Detect Eyes From Infancy

In 2001 Taylor et. al. [115] presented images of eyes, faces, and inverted faces to 128 subjects in seven different age groups (4-15 and adult) and studied the neurological response of the brain through event related potential (ERP). Their study demonstrated that people develop the ability to visually detect eyes before they develop the ability to detect full faces. Taylor et. al. conclude that there is a specific part of the brain that is used to detect eyes.

Eye detection is important for human social interactions to the extent that most people can infer another person’s mental state by looking at their eyes [116]. Human infants, as young as three months old will attend to an adult’s eyes [117]. Itier et. al. also found that in the task of face recognition, the eyes are detected separately before the full face is detected [118].

The task of locating eyes and and human gaze direction is easy for human observers because they develop eye recognition at an early age and have a section of the brain specifically dedicated to recognizing eyes. Not only is eye recognition easy for people, it is easy from a very early age.

When people look at an image and easily see an eye, we project this ease onto a computer and assume that it should be easy for the computer to detect eyes, without consideration for the fact that there is a specialized eye detector built into the human brain.
By contrast, if the computer vision task were to detect bicycles, cars, or airplanes, people are more likely to understand the challenges involved.

### B.2 The Pupil Illusions

The human visual system (HVS) is very good at detecting pupils. When a person looks at an image of a pupil, there are illusions of perception which occur. There are a surprising number of them that contribute to the misconception that teaching a computer to find pupils should be easy.

1. Because people grow up looking at other people’s eyes, they know what a pupil looks like. People will perceive where the pupil is located, even if it is obscured by eyelashes because other clues (such as the iris around the pupil) help communicate to a person where the pupil is.

2. Because people know that pupils are “circular”, they interpret the pupil as a circle, even if seen from far off axis and fairly elliptical in the image.

3. People’s brains can automatically ignore the first surface reflections off of the cornea and ignore them. For people, this is as simple as “window shopping” in which the object behind the window is perceived while the first surface reflection is discounted. Computer vision systems do not automatically unmix mixed images.

4. People’s brains also automatically discount the illuminant and fall-off of illumination in the camera. Viewers naturally assume that the pupil is the darkest object in the image. In reality the darkest spots in the image are either: the corners of the eye, the side of the face, or the main tear-duct near the corner of the nose.

### B.3 Resolution and Quality Challenges

Logically, when designing a computer vision project it would be best to use the highest quality camera available with the highest signal-to-noise ratio per pixel and with as many pixels of resolution as possible. The camera should record all the parameters used to collect every frame. It would also make sense to use a very simple background to simplify the object detection task and carefully control the lighting so that changes in lighting do not cause apparent changes in the objects.

The system used to collect the videos for this project violate all of these guidelines. The camera is instead selected to be as small as possible, so that it is as unobtrusive as possible in the field of view of the subject. The cameras are low quality. The lenses are molded plastic and not high quality ground glass. The sensor records only QVGA resolution, 240 by 320 pixels of resolution per frame. The camera does not record any
meta-data about the contrast or exposure conditions used to capture each frame. Every frame also has unwanted compression artifacts from the process of compressing the video.

![Figure B.1: An example of the reflection of the eye camera being detected as a possible pupil. In this case it was detected as the third possible option, as indicated by the thin blue ellipse around the eye camera. In this case the pupil was identified as the first choice, as indicated by the thin red ellipse around the perimeter of the pupil. (The pupil options are presented to the user in the order: red, green, blue.)](image)

### B.4 A List of Challenges

The videos of the subjects’ eyes that are being eye-tracked in the field are challenging due to several issues. It is helpful to have a list to refer to.

1. As mentioned, the camera was selected to be small and unobtrusive, not high quality.

2. For the videos of interest, the subjects are outside. Subjects wear a hat to try to reduce direct illumination, but the amount of light reflecting off the ground into
their eyes exceeds the amount of light from the infrared LED (IRED) illuminant. (Computations for this are given in Appendix C.)

3. The lighting for the eye is uncontrolled. The subject is free to rotate their head in any direction, causing drastic changes in illumination.

4. The camera includes electronic measures for auto-exposure and auto-contrast. This meta-data is not available to us, making it difficult to interpret the content of the video.

5. The camera’s automatic electronics has a phase lag in it. The auto-exposure is set to the correct exposure over the last few frames, not the current frame. Sudden changes in exposure cause the current video frame to be incorrectly exposed.

6. The images of the subject’s eyes include a reflection of the world off the first surface of the eye. These reflections contain the structure of the objects from the world. The world introduces edges and things that look like circles, which often cause the eye detector to fail. The camera is looking at the pupil through the window of the cornea, and there are reflections off the front of that window. For the computer vision algorithm, the edges of the world camouflage the edges of the pupil. (Computations for this are given in Appendix D.)

7. For outdoor subjects in sunlight the pupil is constricted, sometimes maximally constricted, making it a smaller target to detect. There are fewer pixels on target.

8. The camera records a maximum of 29.97 frames per second. When the eye is in motion, motion blur is recorded, the edge strength of the pupil is reduced, and the pupil’s shape is distorted.

9. Each video frame is spatially compressed. This introduces motion-JPEG blocking artifacts.

10. The processing system includes temporal video compression, which limits the quality of the video stream.

11. Subjects in outdoor sunlight blink more frequently than normal.

12. Subjects in the outdoors close their eyes more, causing eyelashes to obscure the pupil. In some cases the subjects are squinting as much as possible to protect their eyes.

13. Circle detectors match the corners of eyes very well, because dark corners contribute edges that all point to a line of centers.
14. The eye camera itself is a dark object reflected off the cornea. The size of the eye camera is comparable to the size of the pupil and is periodically mis-identified as the pupil. (See Figure B.1.)

B.5 Summary

The task of finding pupils is easy for humans. It is so easy that people think that it should be easy for a computer to do this. This misperception must be overcome in order to explain why a computer has difficulty finding the pupil.

The quality of the video capture system was compromised in order to allow the camera to be small and unobtrusive for the subjects of the eye tracking studies. The quality is not comparable to other video capture systems.
Appendix C

Irradiance Computation

The Positive Science headgear for the mobile eye tracking unit illuminates the eye with an IRED that produces approximately $0.8\text{mW/cm}^2$ at 940 nm [9, p 49]. This is sufficient in a standard office environment to produce enough contrast on the eye to discriminate the pupil from the iris, thus enabling pupil tracking.

However, when taken outdoors in uncontrolled lighting, the outside lighting dominates the IRED.

C.1 Estimating the Amount of Light Reaching the Eye

Here we perform an estimate of the amount of irradiance onto the subject’s eye, for an NSF eye tracking study of geology subjects. We presume that the subjects are all wearing wide brimmed hats to avoid direct sunlight on the eye, and compute the amount of irradiance that would reflect off the surrounding earth and sky into one eye.

The irradiance from the sun, at the outside of the Earth’s atmosphere is roughly $1,390\ \text{W/m}^2$ [119, p 51].

We assume that the subjects are in the Mohave desert and that the air is fairly clear, having an overall transmission of roughly 0.9. This transmission factor includes absorption effects as a function of wavelength.

The eye camera is sensitive to IR exposure in the range of 740 to 1100 nm. This is the wavelength range above where the Wratten 87c filter starts to pass infrared, to where silicon tends to have very little sensitivity [9, p 49]. So, if we approximate the sun as a 5,800 degree Planckian black body, and integrate the power from 740 nm to 1100 nm, we find the portion of the incident irradiance that is of concern to us. (Please see Figure C.1.)
Figure C.1: Fraction of Planckian black-body radiator that could be detected by our silicon based IR camera. About 0.25.

For a surface reflectivity, we use the reflectivity of the Railroad Valley Playa, a remote sending derived from the Remote Sensing Group of the University of Arizona [1].
Figure C.2: Reflection curve for an area near Arizona’s Remote Sensing Group. For 740 to 1100 nm, the typical reflectance is about 0.38. From [1].

This gives the surface reflectance of roughly 0.38 in the wavelength range being considered.

In this range of wavelengths, there is very little Rayleigh scattering, so there is no need to include irradiance from atmospheric scattering, and we can limit ourselves to the irradiance from the sun reflected off the surface of the Earth.

Reflection creates a factor of $1/\pi$, referred to as a “magic pi” due to the spherical geometry.

Being conservative, if we presume that the nose blocks sunlight from the left half of the head, the hat blocks sunlight from above the head, and the head blocks sunlight to the eye from behind the head, the eye collects light from $1/8$th of a sphere. Since there are $4\pi$ steridians in a sphere, we get a collection factor of $1/8 \times 4\pi = \pi/2$ steridians.

Since we assume the sun is a Planckian radiator, we can use Planck’s equation to find the relative fraction of incident irradiation that is of interest to the sensor. Referring to the fraction of Figure C.1 that is shaded in gray, we find that this is about 0.25 of all the power incident. The fraction 0.25 was computed using a Matlab integration over the Planckian curve.

Combining values we find a very conservative approximation for the irradiation onto the eye is:
\[ N_p = 1390 \text{ (W/m}^2\text{)} \times 0.9 \times 0.38 \times 1/\pi \times \pi/2 \times 0.25 = 59.4 \text{ (W/m}^2\text{);} \quad \text{(C.1)}\]

Which works out to about 59.4 (W/m\(^2\)) reflected onto the eye from the surrounding world. This irradiation is uncontrolled light.

In our eye trackers, the eye is irradiated with controlled lighting from an IRED onto the eye, with an irradiance of 0.8 mW/cm\(^2\), which converts to 8 W/m\(^2\) for comparison purposes.

### C.2 Relative Amount of Uncontrolled Light

Using the previous values, we can compute the ratio of controlled to uncontrolled lighting as:

\[
8 \text{ (W/m}^2\text{)}/59 \text{ (W/m}^2\text{)}} = 0.14 \quad \text{(C.2)}
\]

The amount of controlled infrared light supplied by the IRED is considerably less than the amount of infrared light supplied by the ambient illumination, even using conservative estimates.
Appendix D

Cornea as a Window:

The cornea of the eye acts as a window through which we look to see the iris and pupil. People look back out through this window, via the pupil, to see the world.

Any time light crosses an interface between two different media, a fraction of the light is transmitted, and a fraction of the light is reflected. The fractions add up to 100%. If the light is coming from air, the relative amounts of light reflected or transmitted depends on a number of factors, including the index of refraction of the material the light is going into $n_2$. For windows in air, glass acts as a 4% reflector, which is about the same as the superficial oily layer of the surface of the eye.

How reflective a window appears depends on how much light is on each side of the glass, as compared to the side the viewer is on. At night with all the lights in a house are off, and the house is dark, a person on the inside of the house can easily see outside. Looking out the resident can only see light coming in from the outside, such as moonlight, or starlight.

For the same house and the same person in the house if indoor lamps are turned, 4% of the houselight is reflected back towards the viewer. Four percent of the light from lamps is much stronger, and has much higher contrast than the moonlight and starlight that is outside. So, with the lights on inside at night, a viewer cannot see dim lights outside through the windows.

This is precisely the challenge we face in outdoor eye tracking when trying to see the pupil by looking in through the cornea. During the daytime the amount of light outside is significant. When four percent of it is reflected off of the cornea, it creates a strong, high contrast signal through which to find the pupil.

(The amount of ambient irradiance is estimated in section C.)

How light behaves when it interacts with a substance is modeled by the Fresnel equations (equation D.1 and D.2). These equations depend on the angle at which a light ray strikes the substance, $\theta_i$, the angle at which a light ray is transmitted into the substance, $\theta_t$, the index of refraction that the light ray is coming from, $n_1$, and the index
APPENDIX D. CORNEA AS A WINDOW:

of refraction that the light ray is going into, \( n_2 \). Equations D.1 and D.2 give the amount of light reflected by light polarized parallel to the surface, and light polarized transverse to the surface [120, p114].

\[
R_\parallel = R_{\text{parallel}} = \frac{n_1 \cos \theta_i - n_2 \cos \theta_t}{n_1 \cos \theta_t + n_2 \cos \theta_i} \quad (D.1)
\]

\[
R_\perp = R_{\text{transverse}} = \frac{n_1 \cos \theta_t - n_2 \cos \theta_i}{n_1 \cos \theta_t + n_2 \cos \theta_i} \quad (D.2)
\]

If the outdoor air has \( n_1 \approx 1.0 \), and the superficial oily layer of the surface of the eye \( n_2 \approx 1.377 \), then we can generate a graph of the angular dependence of the reflected power. Figure D.1 shows a graph of the dependence on the angle of incidence.

![Reflection Coefficient of the Eye, as function of angle](image)

Figure D.1: Computation of the Fresnel reflection off the cornea.
Considering figure D.1, the average reflection coefficient is about 0.04, or 4%, perpendicular to the surface. The reflection coefficient does not raise to 10% until the light is about 65° from the perpendicular. The fact that it is fairly flat out to this point is beneficial for the eye tracking system when it is indoors because almost no light can strike the eye from 65° off axis and then reflect into the eye camera, because of the geometry of the eye.

When taken outside, the average reflection coefficient is 4% from almost any angle into the eye camera (see figure D.1). This means that the high amount of light bouncing around the environment creates a high contrast reflection of the scene, making it difficult to locate the edges of the pupil.

Environmental light can also cause flare light, if it goes directly into the eye camera. This explains why the Positive Science® eye tracking system works in an office environment, but not when the subject is outdoors surrounded by objects bathed in bright daylight.
Appendix E

Hough Transform Timeline

E.1 Introduction

Variations of the Hough transform are used in many imaging science and signal detection applications. Surveys of the Hough Transform are performed every few years. There was a survey by Illingworth and Kittler in 1988 [54], Levers did a survey in 1993 [55], and most recently, Mukhopadhyay et. al. surveyed the technology in 2015 [56]. Here we provide only a timeline of most relevant events in the development of the Hough transform.

E.2 Important Dates in Hough Transform Development

1960 Hough conceives of the idea as a way to solve the issue of bubbles generated in a bubble chamber [69].

1962 P.V.C. Hough receives patent for Hough Transform [57].

1968 Rosenfeld publishes, “Picture Processing by Computer.” [37]. This describes mentions one form of the equation of a line as the tangent to a circle, which is a sinusoid in parameter space. There are five standard forms for the equation of a line, and \(xcos\theta + ysin\theta - p = 0\) called the normal form of a line [121].

1972 Hart reads Rosenfeld’s publication [69] and realizes that this representation convert points in (x,y) space on lines in an unbounded space of slopes and intercepts into a finite bounded space of distances from the origin and angles [122]. This allows the storage of votes for a line by representing in a finite space that can fit into a finite computer memory.
1975 Kimme, Ballard and Sklansky used the Hough Transform to improve the process of finding circles using the local gradient information [76, 123]. They also limited angle that each edge point votes to ±23.5° of the direction of the gradient.

1975 Merlin and Farber showed that the Hough Transform could be applied to arbitrary shapes at a given orientation and scale (Merlin and Farber, 1975) [124].

1981 In the Generalized Hough Transform (Ballard, 1981) extended it to an arbitrary shape at an arbitrary orientation and arbitrary scale. [125].

1990 The Randomized Hough Transform (Xu et al. 1990 and 1993) [123, 126] created what is effectively an early version of RANSAC (Fischler and Bowles, 1981) [97]. This was important because it sampled the image space for multiple points and used them to fit a desired object based on a parametric model. This maps multiple points in image space to a single point in all of a quantized parameter space, resulting in a data reduction.

1991 The Probabilistic Hough Transform (Kiryati 1991) [127]. This sampled the points based on the probability of them being important to the detection of the object. This technique is now called “importance sampling.”
Appendix F

Simulating the Image of the Eye

By observing the motion of the eye as it moves through the recorded videos subject’s eye motions we notice that the appearance and position of the pupil is influenced by refraction as light from the pupil leaves the cornea of the eye. The cornea is a lens and deforms light the light passes out of it.

The question posed was, “How much does this refraction change the appearance and location of the pupil?”

To determine the answer to this question, a computer simulator incorporating a model of the eye was developed to simulate the refraction and reflections at the optical surfaces of eye. The simulation included analytical models of the cornea, aqueous, pupil, Fresnel reflection, and associated indices of refraction.

Here we paraphrase the software described in Kinsman and Pelz [128]. For a detailed description of parameters and attributes refer to in the proceedings of the ACM in 2014 ETRA-2014 conference paper. We include mention of it here to show that the development of this simulation allowed us to answer some of the questions facing us. Some of the resulting curves are very similar to the curves used to reject strange pupils in Chapter chSED.

The computer simulation allows us to study the eye motion with (and without) unwanted scene reflections.

Resulting videos have smooth eye surfaces without the faceting that can occur in other applications, such as Open-GL. Arbitrarily high camera resolutions can be simulated by setting the resolution of the simulated camera. Simulation generates output videos and text files to provide frames with known pupil locations. Most pertinent here is that the computer simulation allows us to artificially study the effects of corneal refraction on the pupil position, shape, and size.
F.1 Graphical Users Interface

Additionally, a graphical user interface was developed to control the parameters for the simulator (see Figure F.1).

When run, the simulator creates several output data products. It creates an output video of the eye, with first-surface reflections off of the cornea. I also programmed to create an output pixel map of all pixels in the pupil. It also produces an ascii file with the pixel location that is most central to the center of the pupil.

By changing the index of refraction of the cornea and aqueous, we can remove the effect of refraction of the cornea and aqueous.
Figure F.2: Figure showing the simulator run with different parameter settings. The output resolution is set to 640 × 480 pixels.
Figure F.2(a) shows a simulation of the eye without any refraction. This is accomplished by setting the index of refraction of the cornea and aqueous to 1.0 and using a blank scene in front of the eye.
By using the appropriate values for the index of refraction for the aqueous and the cornea, refraction of the pupil location occurs. This can be seen in Figure F.2(b).
Finally Figure F.2(c) shows the simulation with a scene in front of the eye.

F.2 Effect of Refraction

To find the answer to these questions, we configure the simulator so that the camera is looking straight into the eye from the front, and have the eye move in 10° increments to the right.
To observe the effect of refraction, refraction is turned on and off by changing the index of refraction of the aqueous and cornea (see Figure F.2).
Figure F.3: Figure showing the eye rotated 10 degrees, 50 degrees, and 80 degrees away from the camera. At 80 degrees the pupil would not normally be visible. However, due to the refraction of the cornea it is still visible to the camera.

Figure F.3 shows the resulting simulated eye images at three selected angles of rotation. Angles from 0 to 90 degrees were used.

Figure F.4: This figure shows the pupil off axis at 10 degree increments, starting at 0 degrees. The top sub-figure is without refraction. The bottom subfigure is with refraction.

Figure F.4 shows dark circles where the pupils would be for 10 degree increments of rotation, starting at 0 degrees. The top sub-figure is for pupils without refraction in the cornea. The bottom sub-figure is with refraction simulated.

The difference of these two series of pupil silhouettes shows the effect of refraction. The top series of pupils shows that without refraction, the pupil becomes eccentric fairly quickly, and that the pupil becomes nearly a slit at 70 degrees. The bottom series shows that with the refraction of the cornea the pupil remains relatively circular out to around 30 degrees, and can be seen all the way to the point where the eye is rotated 90 degrees away from the camera.

Additionally, Figure F.4, shows two other effects. Notice that the pupils in the lower series, with refraction, are slightly larger than those of the top series, without refraction.
Thus the refraction of the cornea makes the pupil appear slightly larger than it would without refraction.

The last thing to note about this figure is that the height of the pupil remains nearly constant as it is rotated away from the camera. This is important, because it means that when the circular pupil is transformed the largest axis of the ellipse does not diminish.

This fact is used to reject candidate pupils that the circle detector finds, that are too small to fit expectations.

With the refraction of the cornea, the pupil appears more circular (i.e. less eccentric), for a longer distance off axis.

If we find the centers of these pupils and plot them on a graph including their eccentricity, we obtain the following figure, Figure F.5.

Figure F.5 shows the position of the pupils plotted in terms of number of pixels off
Table F.1: Displacements and eccentricities as a function of rotation.

<table>
<thead>
<tr>
<th>Eye Rotation (deg)</th>
<th>Displacement (vga pixels)</th>
<th>Displacement (qvga pixels)</th>
<th>Eccentricity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.0000</td>
</tr>
<tr>
<td>10</td>
<td>56</td>
<td>28</td>
<td>1.0000</td>
</tr>
<tr>
<td>20</td>
<td>108</td>
<td>54</td>
<td>1.0806</td>
</tr>
<tr>
<td>30</td>
<td>156</td>
<td>78</td>
<td>1.2182</td>
</tr>
<tr>
<td>40</td>
<td>196</td>
<td>98</td>
<td>1.3830</td>
</tr>
<tr>
<td>50</td>
<td>229</td>
<td>114.5</td>
<td>1.7105</td>
</tr>
<tr>
<td>60</td>
<td>252</td>
<td>126</td>
<td>2.1000</td>
</tr>
<tr>
<td>70</td>
<td>267</td>
<td>133.5</td>
<td>2.8636</td>
</tr>
<tr>
<td>80</td>
<td>275</td>
<td>137.5</td>
<td>4.2000</td>
</tr>
<tr>
<td>90</td>
<td>276</td>
<td>138</td>
<td>9.2857</td>
</tr>
</tbody>
</table>

center on a 640 × 480 image as a result of 10 degree increments. The vertical axis is how eccentric the pupils are at that rotation.

The dark data circles on the graph show the data points without corneal refraction. The hollow circles on the graph, show the same data points with the refraction of the cornea – this is as they appear in reality.

Arrows are drawn from the data point without refraction to the corresponding data point with refraction. Not all data points with refraction have a corresponding data point without refraction, because without refraction, the pupil is not visible past 70 degrees.

F.3 Refraction of Pupils

From the simulation we can find the following table of displacements from the center and eccentricities as a function of eye rotation angle.

The displacements are in pixels on a 640 × 480 image, so must be scaled down by a factor of two for our system.

F.4 Summary

The fact that the pupil appears more circular when it goes off axis infers that the refraction of the cornea helps animals with corneas see the world at a consistent size, further off axis than if there was no refraction. This is because if the pupil appears the same relative size and less eccentric up to 30 degrees, then the objects in the world are seen through a pupil that is relatively the same size, and with astigmatism than they would be without the refraction of the cornea.
By simulating the appearance of the pupil we find the following:

1. For any angle off axis, the corneal refraction makes the pupil appear less eccentric than it would without refraction.

2. The corneal refraction makes the pupil visible to the camera further off axis than it would be without refraction.

3. The maximum size of the pupil (major axis of the ellipse) remains nearly constant as the pupil looks away from the eye-camera.

Again, for a detailed description of the computer simulation, see the proceedings of the ACM in 2014 [128].
Appendix G

Testing for Bias in Results

Previous results showed a one degree accuracy could be achieved quickly using the developed system. A natural question might be, “Is there bias in that one degree of accuracy?” For example, the system might always estimate high by half a degree.

G.1 Comparison between Program and Analysts

To explore the possibility of bias between the system and the analysts, we reloaded the data for the Lone Pine site, from 2013, for two subjects and the corresponding files from the human analysts. We computed a $\Delta X$ as the value found by the program minus the value reported by the analyst. The same computation was done for $Y$ the values. Frames that contained blinks were identified and excluded from the analysis.

Figure G.1 shows the results for subject 03, while Figure G.2 shows the results for subject 07. A large red plus sign is put over the mean value in $(X, Y)$, and in both cases there is a noticeable shift in the data. For neither of these cases are the average errors centered exactly on zero.

Looking at the $\Delta Y$ component, we see that the sign of the error differs from one case to the other. Sometimes the program underestimates a value, and sometimes it overestimates a value. Table G.1 tabulates these differences.
Table G.1: Table of differences between system results and analyst results for Lone Pine 2013 subjects 03 and 07.

<table>
<thead>
<tr>
<th>Video</th>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lone Pine 2013</td>
<td>Mean Δ X</td>
<td>+0.24</td>
</tr>
<tr>
<td>Subject 03</td>
<td>Mean Δ Y</td>
<td>+0.31</td>
</tr>
<tr>
<td></td>
<td>σ₇</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>σᵧ</td>
<td>0.20</td>
</tr>
<tr>
<td>Lone Pine 2013</td>
<td>Mean Δ X</td>
<td>+0.13</td>
</tr>
<tr>
<td>Subject 07</td>
<td>Mean Δ Y</td>
<td>-0.17</td>
</tr>
<tr>
<td></td>
<td>σ₇</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td>σᵧ</td>
<td>0.42</td>
</tr>
</tbody>
</table>
Figure G.1: Bias diagram showing the program computed value minus the analyst’s entered values for Lone Pine site, 2013 subject 03. Blinks were excluded.
APPENDIX G. TESTING FOR BIAS IN RESULTS

Figure G.2: Bias diagram showing the program computed value minus the analyst's entered values for Lone Pine site, 2013 subject 07. Blinks were excluded.

G.2 Comparison between Analysts

To compare the differences between analysts, we examine the location data from the South of Olema site 2012 subject number 6, which was coded by three different analysts. The same bias computations were made, in the same way.

Figure G.3 and Figure G.4 show the differences between analysts 1 and 3, and between analysts 2 and 3 respectively. Table G.2 summarizes the measurements for these two sets of analysts.
Table G.2: Table of differences between analysts for South of Olema, 2012, subject 06.

<table>
<thead>
<tr>
<th>Analyst Pair</th>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 and 2</td>
<td>Mean ΔX</td>
<td>-0.22</td>
</tr>
<tr>
<td></td>
<td>Mean ΔY</td>
<td>-0.03</td>
</tr>
<tr>
<td></td>
<td>σₓ</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td>σᵧ</td>
<td>0.13</td>
</tr>
<tr>
<td>2 and 3</td>
<td>Mean ΔX</td>
<td>-0.25</td>
</tr>
<tr>
<td></td>
<td>Mean ΔY</td>
<td>-0.01</td>
</tr>
<tr>
<td></td>
<td>σₓ</td>
<td>0.43</td>
</tr>
<tr>
<td></td>
<td>σᵧ</td>
<td>0.13</td>
</tr>
</tbody>
</table>
Figure G.3: Bias diagram showing difference between analysts for South of Olema, 2012 subject 06. Blinks were excluded. This shows the differences between analysts one and three.
Figure G.4: Bias diagram showing difference between analysts for South of Olema, 2012 subject 06. Blinks were excluded. This shows the differences between analysts 2 and 3.

G.3 Discussion

We notice a slight bias between the system results, and the pupil locations generated by the analysts. The bias does not appear to be in a consistent amount or a consistent direction.

The maximum difference between the system and the analysts is a value of 0.31 in the ΔY direction for the difference between the system and the analyst for subject 03 (See Table G.1, Lone Pine 2013 video subject 03). This compares favorably with a maximum difference of a ΔX of −0.25 between analysts two and three (See Table G.2, South of Olema 2012 video subject 06). The magnitude of the maximum differences amounts to less than
half a degree in each case.

In some cases the standard deviations between analysts are larger than the standard deviations between the system and an analyst. No trend can be drawn from such a small sample size, however we can see that the absolute values of the bias between system and analyst, compared to the bias between analysts, are similar to each other. Additional investigation into sources of error are left for future work.
Appendix H

System Hardware and Software

The software was written in Matlab® Version 7.14.0.739 (R2012a).

The system was implemented on a Apple Macintosh MacPro5.1 12-Core workstation computer running MAC OS X version 10.8.4. The system had dual 2.4 GHz 6-Core Intel Xeon processor with 48 GB memory. It is equipped with a ATI Radeon HD 5870 graphics display processor.

A copy of the software is warehoused in the Multidisciplinary Vision Research Labs or the Center for Imaging Science, at the Rochester Institute of Technology in Rochester, NY.

For software questions please either contact the author directly (thomaskinsman@gmail.com), or Jeff Pelz (jeffpelz@gmail.com).
Bibliography


[90] Paul Viola and Michael J Jones. Personal correspondence to the authors., Apr 2013.


