1. INTRODUCTION

1.1 MOTIVATION

The motivation for this research stemmed from a need to detect characteristic eye motion and location of the eye's gaze. Uses for such detection span many markets—from hands-free mouse interfaces for the disabled to pilot cognitive activities during flight for military applications. Advertisement agencies use such devices to study what people focus on when they observe the advertisers' products. Other important applications have developed to maintain or monitor operator alertness in nuclear power plants as well as in trucks and other vehicles. Such research relies on the eye as a simple communicator of human interest or awareness through which a computer can understand and respond to a person's needs. In addition, the data formulated by a computer during a subject's use of this equipment can prove valuable for studies into the human mind by using the insight that eye motion can provide.

Our particular motivation for this project was to provide a prototype of an eye tracking system optimized for detecting a slow rolling motion of a human eye--an indicator of drowsiness. With such an early warning system in place, it could be possible to outfit shift operators of machinery, power stations, or motor vehicles with eye tracking safety devices. In the case of motor vehicle operation, since "fatigue contributed to one third of all fatal truck accidents"[1] and "200,000 auto accidents a year may stem from driver fatigue"[1], our system might provide future relief to a prevalent problem.

Since we optimized our system for detection of a type of motion in real time, pinpoint accuracy of the pupil location was not needed although speed was. Other optimizations of our system included ease of use and comfort of the user during use. We researched many of the present eye tracking products on the market to evaluate their performance for these criteria and to prepare for how we would approach their same problems with novel solutions.

1.2 SURVEY OF EYE TRACKING DEVICES

We surveyed commercial as well as experimental eye trackers presently available to demonstrate what work had been conducted on the subject prior to this thesis. Since the majority of the devices were based on computer vision techniques, we devoted most of our investigation to those topics. However, some work on eye tracking has been conducted using other methods that deserved attention. This survey attempts to provide a condensed background of the subject.

Most eye tracking devices are based on computer vision imaging systems, yet some are based on other detection means. Take, for instance, a technique based on fixed items such as tiny mirrors engraved on contact lens surfaces; the reflections of light from these mirrors serve as detectable points for a tracker—a CCD camera or even a single photodetector. Other items such as induction coils have been embedded within contact lenses to give a signal when the user is exposed to a high electromagnetic field. Another method detects the changes in the electrical potential of the skin around the eye, since an electrostatic field rotates along with the eye.

The similarity with all of the above methods of detection lies in the difficulty of use for a person; the application of contact lenses or electrodes to one's eye for the simple purpose of mousing could lead an uncomfortable user to doubt the benefits of such a system. The eye tracking researchers seemed to reach this consensus when they concentrated their efforts on imaging systems that did little to interfere with their subjects. Indeed, work is being done to eliminate headgear altogether so that the effect felt by the user is nonexistent.

Many imaging practices have been developed based on reflections of light from various portions of the eye. Some of these methods detect reflections off of the surface of the eye, where changes in gray-level value are regarded as the markers by which tracking is made possible. Other techniques rely on bright reflections off of the retina, cornea, and lens compared to each other in order to track the eye's gaze. Although the use of cameras

to form these images allows a user to attach fewer cumbersome devices to himself, it greatly increases the complexity of the system. The processing and computations involved in working with digitized images usually press the limits of the most sophisticated computers.

Some vision techniques used to work with reflections off of the eye surface involve simple detection of changes in gray-level value (light to dark). The detection of the boundary between the white sclera and the dark iris is termed limbus tracking[2] (the boundary is called the limbus), and simply involves that particular boundary. The other boundary worth noting is between the pupil and the iris; it defines the marker for pupil tracking techniques.

Other reflection images can be produced by an infrared light and an infrared sensitive camera. These reflections are termed the Purkinje images[3]—the two important reflections being the glint off of the cornea surface and the bright reflection of the retina through the pupil (see Figure 1.1). When these two reflections are detected by a camera, image processing is done to ascertain the relative position of one to the other. In this manner, a vector pointing in the direction of the eye's line of sight is formed.

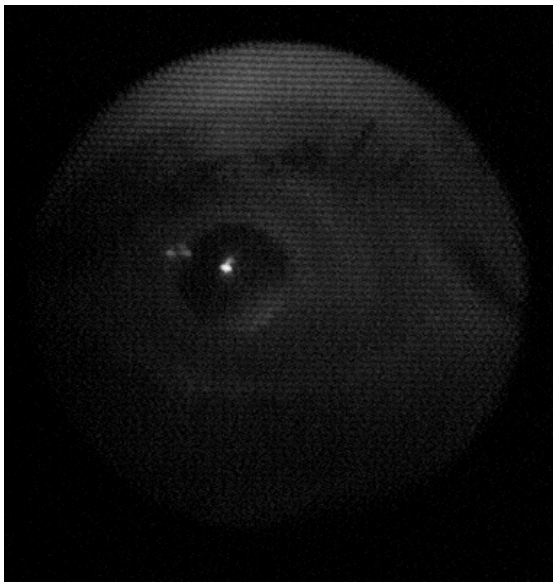

Figure 1.1 Purkinje images—note reflection off of retina through pupil as well as off cornea

Since we decided to approach our problem using a head-mounted tracker, we devoted more of our background study to the same kind of developments. Research has been conducted, however, in tracking a subject's eyes while the camera was situated at some distance away. For instance, a system was developed for monitoring a driver's blink rates using a camera mounted in a vehicle dashboard[6]. The vision concerns were amplified by the large area over which the processor was forced to hunt for eyes. Since our focus was to design a portable unit with rapid speed of computation, we avoided this method of detection. Most eye trackers on the market today are of this head-mounted type; they avoid the complexity of discerning eyes from an entire face image.

Many companies offer head mounted eye trackers for commercial use. Consider the GazeTracker offered by Mooj & Associates (Figure 1.2)[5]. Notice the head mounted infrared sensitive camera and hot mirror (IR reflective) below the eye used to direct the image of the eye into a digitizer. Infrared light is used for imaging so that the light incident on the eye and the eye image reflected off the mirror are not observable by the user. An infrared LED to illuminate the eye is enclosed on the helmet near the camera. An illustration of the system setup is given in Figure 1.3 below. The device uses the same kind of eye detection techniques as mentioned previously—most probably using Purkinje images. The company states that its product can calculate angle of view in real time.

Figure 1.2 GazeTracker by Mooj & Associates[5]

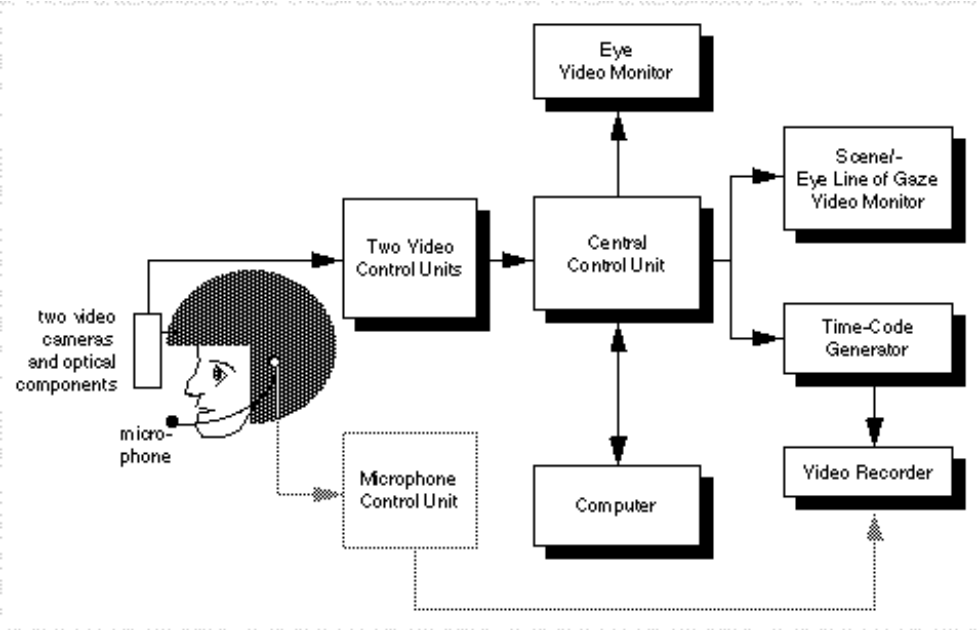

Figure 1.3 GazeTracker system layout[5]

At The University of Illinois at Urbana-Champaign's Beckman Institute, an eye-tracking facility provides many such devices for use by researchers in its Human

Perception and Performance and Cognitive Neuroscience groups. The Institute has purchased 5 eyetrackers for research purposes—most of them Purkinje image detectors. All of these sophisticated trackers are able to track in real time (video rate of 60 frames per second) or better (up to 1000 frames per second). Figure 1.4 shows one of the systems owned by the institute.

Figure 1.4 Beckman Institute eye tracker[6]

The companies represented in the Institute's choice of eye trackers for the facility are Applied Science Laboratories, Stoelting, and Eyelink[6].

As we can see, these commercial products have the technology capable of handling real-time gaze tracking, but require a great deal of headgear for a user to wear. In addition, these systems require powerful computing systems in order to handle the large quantity of image data. The cost of such systems, which is largely contributed to by imaging and processing power as well as infrared sensitive cameras (in most systems) ranges from $20,000 to $100,000. This cost could be a considerable drawback, since the aim of some detectors would be to monitor the performance of plant operators or vehicle operators—where many such devices would be purchased. These limitations will be overcome in the next generation of eyetrackers, when more lightweight headgear or even

independent camera systems will be developed and unique processing approaches to the eye tracking problem will be created.

1.3 OBJECTIVE

This thesis work developed as a solution to some of the problems presented by current eye tracking products and research. To overcome the limitations demonstrated above, several factors of a new prototype eye tracker were to be considered. The most apparent unappealing factor about the current eye trackers was their head-mounted gear. We would aim to alter the approach of the head-mounted optics with the introduction of a fiber optic bundle to help decrease the load of the apparatus. The second factor which we would want to study was the image processing algorithms. We hoped to understand the difficulties involved in eye tracking as well as propose a novel approach to finding the approximate pupil center in real time.

Of course, many advantages offered by the current systems were retained in our proposed device. These advantages include the non-intrusive illumination of the eye that allows a user to see normally even with light incident on the eye and reflected from a mirror in the eye's line of sight. This advantage is provided by the use of infrared light and a (visible light transparent) hot mirror instead of visible light and a silvered mirror. The illumination allows the vision system to detect the eye under near constant lighting conditions—without which the system would need constant calibration to deal with shadows and ambient light. The other advantage of the head-mounted systems is that the camera remains relatively stable with respect to the eye; the image of the eye rarely moves with respect to the camera's line of sight. An image processing system would thus be unencumbered by the task of finding the eye in the first place. We chose to use a head mounted system to similarly decrease the load on our image processing system.

The eye tracking device we intended to design would include an improvement of the current systems along with an optimization of a system for our specific needs. A study

of current systems provided us with background tools from which we could draw valuable hypotheses concerning our task at hand, as well as ideas to make the device more practical for operator use. Our task—detecting driver drowsiness using slow rolling eye motion as our indicator—required us to follow specific guidelines for comfort of use, for speed of detection, and for cost effectiveness. In summary, our objective was to provide a simple, lightweight, and inexpensive prototype for the express purpose of detecting a slow rolling motion of the eye for early warning of driver drowsiness.

2. THE EYE TRACKING SYSTEM COMPONENTS—INTRODUCTION

2.1 880 NM LED

For our infrared source we chose an 880 nm LED from Gilway Technical Lamp. The device operates with a 1.3 V input to give up to 35 mW light output depending on the supply current. From the evidence of the LED spectrum shown in Figure 2.1, this device obviously has a broad spectral width. The width is not broad enough for the spectrum to be visible to the human eye, but a standard CCD camera should detect the light very well. A typical CCD spectrum is given later to show why we believe a CCD would detect our 880 nm LED well: the response of the camera is at a maximum in that region.

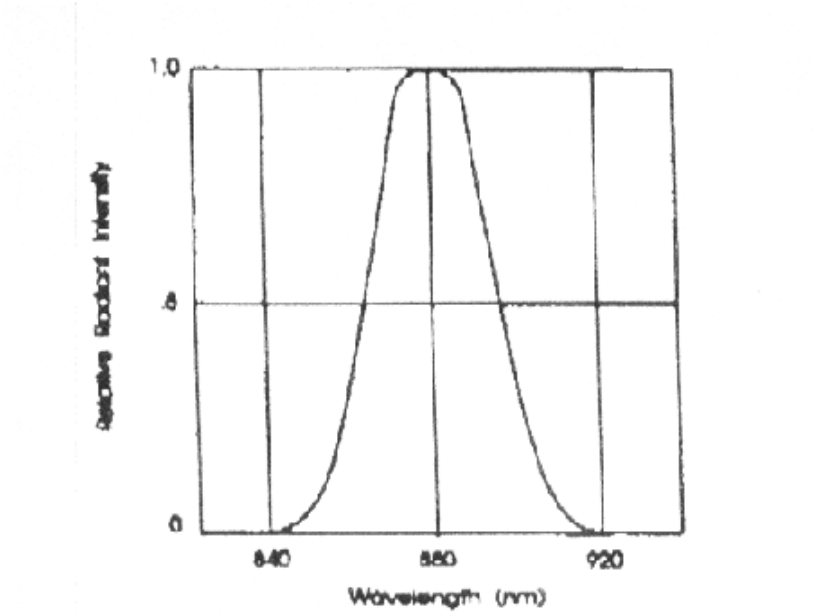

Figure 2.1 Spectral Response of Gilway Technical Lamp LED[7]

The source we chose fit all of the requirements determined to meet our needs, in addition to being extremely low-cost. The light incident on the eye would be unobservable

by any user, but easily seen by our CCD camera. The LED would require little power and add negligible weight to our devised system. Although the incident light provided by our LED was low power, we would have desired that the power incident on the eye was below the medical recommendation of < 1 mW per cm^2 as long as enough light was illuminating the eye. We wanted to prove that the LED would meet those criteria.

2.2 HOT MIRROR

Another required component of our device would be optics to direct the eye's image into a CCD camera without obstructing the view of the user. To facilitate this objective, we employed the use of a hot mirror—a filter that reflects infrared light and passes light in the visible range. Our particular hot mirror was made at ACT in Radford, VA.

These filters are primarily used in projection and illumination applications where heat is a concern. The hot mirror will reflect harmful infrared radiation from lamps before the light strikes other objects that might be heat-sensitive. Mirrors for this purpose have been in use for many years, so the technology to produce them is well formulated; these mirrors and films are generally not expensive.

These heat mirrors are dielectric multilayer interference filters. They are produced by evaporating or sputtering many layers with precisely known thickness and optical constants onto a glass substrate. When incident white light enters the multilayer structure it undergoes constructive and destructive interference which differs as a function of wavelength. The multilayer structure is optimized to produce the desired result, in this case, the transmission of visible light and the reflection of " heat" from the near IR.

The heat mirror is comprised of the film structure: Air | 1.1(L/2 H L/2) (L/2 H L/2)^5 1.125 (L/2 H L/2) 0.57(LMHML)^8 L/2 | Glass. Here L = quarter wave optical thickness (QWOT) of low index material, M = QWOT of intermediate index optical material, and H = QWOT of high index optical material. The QWOT center is at 0.860

microns The calculated reflection of this structure known as a double stack heat reflector appears in Figure 2.2, plotted from 0.3 to 2.2 microns below.

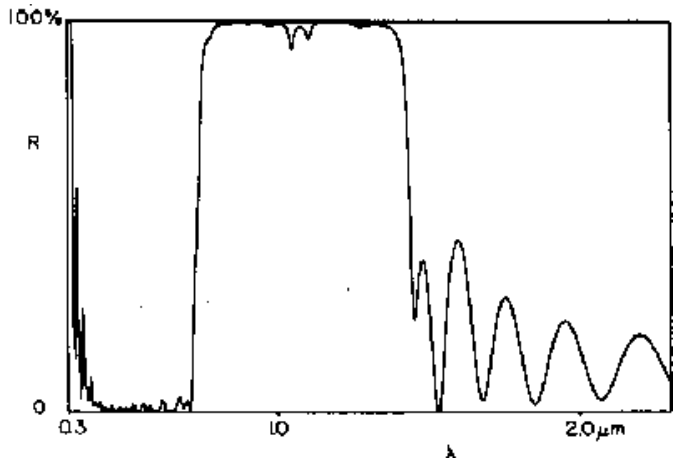

Figure 2.2 Reflection curve of the double stack heat mirror.

2.3 FIBER OPTIC BUNDLE

Fiber optics are used in most instances for their signal speed and large bandwidth for communication purposes. Another general property of fiber optics—simply their light guiding property—has become useful for industry in the form of fiber optic bundles. Fiber bundles are used for delivering and receiving light from specific locations that may be difficult to fit with bulk optics.

Illumination sources employ fiber bundles to deliver light to specific or hard to reach places. UV epoxy cure guns inject UV light through a fiber bundle to cure epoxy on orthodontic braces or any other small area. Illumination for some microscopes is provided by fiber optic bundles connected to lamp housings, since the close proximity of a lamp power supply and a lamp causes detrimental heating and vibration near the point of measurement. Also, lighting a sample through microscope objectives forces readjustment of the light under high magnifications and zoom applications. Fiber optic bundles can

alleviate these illumination problems by providing light at a desired point without added heat, vibration, or adjustment.

Fiber optic bundles made in precise patterns that are aligned the same on both ends are used for receiving or sending images through the light guide. These devices have been used for many years in imaging applications like image intensifier tubes. An image tube generally uses fiber optics to minimize degradation of the resolution of the edge of the image and to provide a good coupling surface for other image tubes or for photographic film[8].

Boroscopes have recently popularized the use of fiber optics for medical imaging applications. These devices usually are a coherent (both ends aligned) fiber bundle fitted with optics so that the image can be focused onto an eyepiece or camera. The obvious advantage of the device is that it can provide an image at a location where a camera or microscope would be an intrusion.

A boroscope seemed ideal for the application of the eye tracker because of its beneficial guiding properties. The advantage of using a boroscope in our system was that the camera would not rest in the head mounted gear, but could be located at some distance away. The fiber bundle could then carry the image of the eye to the camera. We chose to purchase a boroscope from Edmund Scientific and test its appropriateness in our prototype as our image delivery mechanism.

2.4 CCD CAMERA

Commercially available CCD cameras are fairly standard in their response and output. A typical spectral response peaks at about 900 nm, although color cameras usually include an IR filter to cut off the higher wavelengths. The output of most cameras is standard RS-170—directly compatible with most monitors and framegrabbers. These standard features make camera choice easy for users desiring normal operation video.

CCD cameras also have become extremely lightweight and inexpensive while preserving high resolution. The low cost camera purchased from Edmund Scientific for this thesis work weighed less than 25 g. This low weight and compact size afforded us the opportunity to reduce the size and weight of equipment accompanying the head gear for our prototype design. The number of pixel elements available on this unit was 512 $\times$ 492, which would provide excellent resolution for our application.

2.5 FRAMEGRABBER AND PC

In order to capture an image from our CCD camera and process the pixel information as digital information in memory, we employed the use of a computer board framegrabber. These devices take analog composite video information and digitize frames for further processing. This digitization is accomplished through an analog to digital converter and synchronization circuitry, making pixel information available to the computer in the form of memory locations. Framegrabbers generally use the computer's memory, but the more expensive types have on-board memory of one to several megabytes.

Our particular framegrabber, the Imaging Technology IC-PCI board, offered on-board memory and high speed. The on-board memory option was chosen so that the memory of our computer could be used for other processing while a frame was being captured. We hypothesized that this advantage would help in increasing the speed of our system. The board itself was high speed—having clock rates of up to 40 MHz, which transfers into about 150 frames per second for our image size (512 $\times$ 492).

The software used to capture these images and drive the framegrabber was written for use with Microsoft Visual C++. This programming package was purchased to be installed with the framegrabber software on a Gateway G6-200 running a Microsoft NT 4.0 operating system. The framegrabber board would be installed into an available PCI

slot in the computer and driven with the board's accompanying software in the Microsoft Visual C++ programming environment.

Programming for the imaging system that would track the eye would be written in Visual C++ using a variety of computer vision techniques. In order to familiarize the reader with the proposed algorithm for tracking the eye, some background information regarding computer vision is in order. The following section introduces the topic as well as provides sufficient preparation for the discussion of the prototype algorithm in later sections.

3. COMPUTER VISION

Visual information is obviously of extraordinary importance to obtaining information about our environment. The mass of data we humans process, about 10 megabytes of information per second[9], allows us to recognize, track, and judge parameters of objects around us. We can predict, say, the motion of a ball that we want to catch. We can reach out to grasp our favorite fruit from a bowl of many different kinds. We can read letters on a page. All of these tasks involve not only the capture of an image, but understanding of that image as well. It is this type of understanding that is required for computer vision challenges.

The importance of solving these computer vision problems has been realized by many markets. In an industrial setting, the significance of vision is apparent for the operation of machines or automatic devices that need to locate and recognize objects, assemble objects, or inspect objects. Machine vision has recently enabled industry to cure the blindness of these tasks by employing vision capable machines that can pick items from a bin for assembly, that can inspect circuit cards, or that can guide a welder along a seam. The consumer market has also seen the explosion of applications for computer vision. A hands-free mouse that is guided by eye movements is based on vision techniques close to those represented by this thesis. Also, optical character recognition (OCR) is employed to transfer paper copy text to a computerized format for file recall and storage. OCR is also used in the surveillance market for the inspection of license plate numbers on vehicles.

More surveillance applications lie in future investigations into face recognition and suspicious motion recognition embedded into new CCTV systems.

Since many applications and challenges for computer vision have recently been uncovered, so have the solutions to these problems. All of these solutions involve image manipulation and image understanding to accomplish their goals. We will briefly cover some of the techniques that are used to accomplish image understanding, and especially those that apply to the particular problem presented in this thesis.

3.1 IMAGES AS 3-D ARRAYS

A digitized image that appears on a flat screen can be represented by a two dimensional array of quantized image elements called pixels. If we consider the simple case of an eight bit gray-scale (rather than a color) image, each one of these image pixels appears to be a value from dark to light and is thus assigned a value from hex 0 (decimal 0) , the color black, to hex FF (decimal 255), the color white, respectively. If we were to plot our array using gray-scale as a third dimension as in Figure 3.1, we would see the topological surface that is generated by an image—indicating how the features change across the image and how we might glean understanding from these changes.

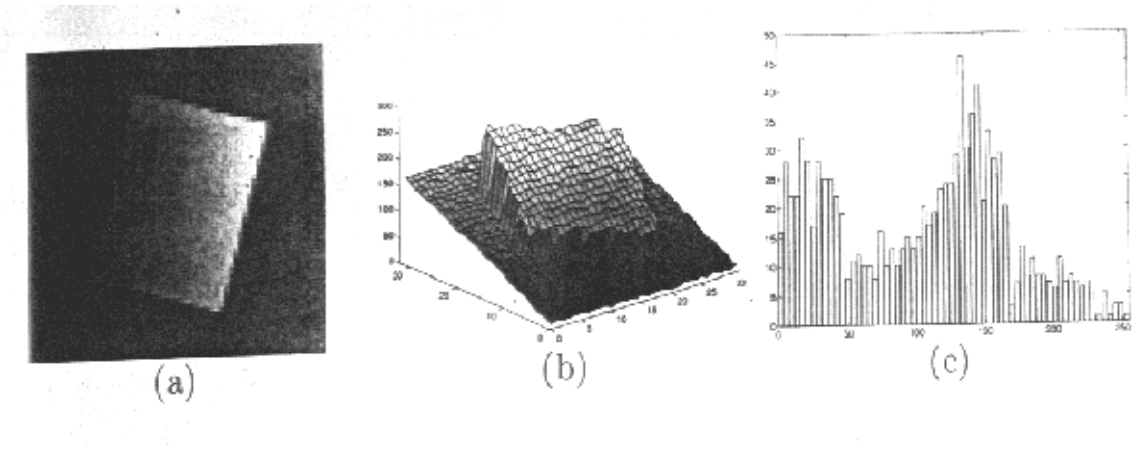

Figure 3.1 (a) original image (b) 3-D plot including gray-scale (c) histogram[10]

3.2 THRESHOLDING, HISTOGRAMS

Features that are usually of interest in an image are wide areas of uniform height—like a plateau or hill in the or gray-level topology. We may want to segment out such features in order to filter out unnecessary information and only use that portion of the image in further processing. A technique used often in computer vision to simplify an image relies on regions with gray levels that are above a certain value and into regions with gray levels that are below. This technique, called thresholding, usually binarizes the image by turning "on" pixels (setting pixel value to hex FF) or by turning "off" pixels (setting pixel value to hex 0) based on a certain criterion. An example of thresholding is given in Figure 3.2.

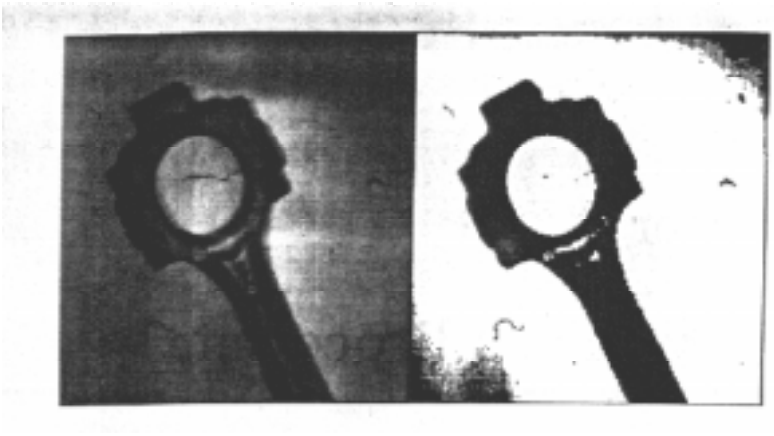

Figure 3.2 Thresholded image[10]

A simple way to threshold an image is to establish a minimum pixel value that a given image pixel must be greater than in order to be turned "on" in the thresholded image. As we saw in Figure 3.2, a minimum threshold level was set at a particular value, so the thresholded image has pixels only where the corresponding original image pixels exceeded that value.

Other ways of thresholding an image lie in obtaining first a histogram of the image. For each gray-level value, hex 0 to hex FF, we establish an array element whose assigned

value is incremented according to how many original image pixels have that gray-level value. For example, if the original image has 2000 pixels of value hex 80 (decimal 128), then array element 128 is assigned the value 2000. If there are 4051 pixels with value hex FF (decimal 255), then the histogram array element 255 would be assigned value 4051 and so on. We compute the histogram array over the entire image and observe the trends as in Figure 3.3. Usually a thresholding scheme will calculate from this histogram hills and valleys that could possibly segment the image into distinct uniform parts. In Figure 3.3 a threshold placed where the line is drawn between the two maxima would prove a good choice for segmenting the image.

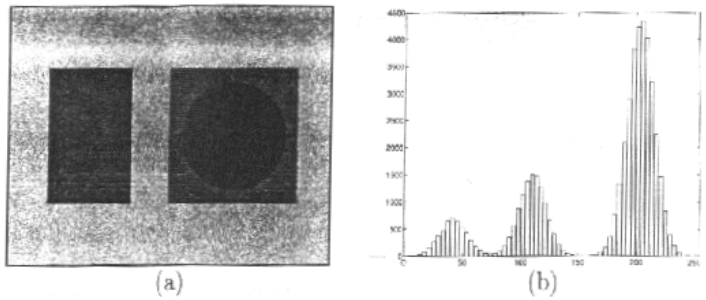

Figure 3.3 Choosing a threshold[10]

Other methods of segmenting an image into useful parts not only rely on the application of simple thresholds, but also specialized filters that amplify the desired qualities of the image. Where thresholds can simplify the image into parts of uniform gray level value, they do little to enhance certain shapes, textures or transitions in the image. Filters amplify or suppress desired parameters in the image to lend some understanding into what significant objects are in the image.

3.3 FILTERS AND CONVOLUTION

In image operations, filtering is a regular task that can redefine images to a more preferable form. The process of convolving an image with a preferred mask is simple and

usually involves straight computation and no decision-making. Thus, a mask applied to an image will mathematically glean from the image a desired effect that we can then use for further interpretations.

Consider the problem of noise in an image. Just like noise in an electrical signal, we can think of solutions using noise filtering techniques and convolution with various filters that would reduce this noise. For electrical signals we have filters defined in the frequency domain that can effectively reduce noise, whereas in image processing we use spatial frequency domain filters. One such mask developed for noisy images is based on the averaging effects of a Gaussian filter. The Gaussian plot in two dimensions is defined as

$$h(x,y) = \exp(-(x^2+y^2)/2\sigma^2),$$

where σ =standard deviation.

A typical 7 X 7 mask used in imaging applications is[10]

1	1	2	2	2	1	1
1	2	2	4	2	2	1
2	2	4	8	4	2	2
2	4	8	16	8	4	2
2	2	4	8	4	2	2
1	2	2	4	2	2	1
1	1	2	2	2	1	1

We can pass this mask of 7 X 7 pixel size over our image, thereby convolving the image with our filter in the fashion illustrated below in Figure 3.4.

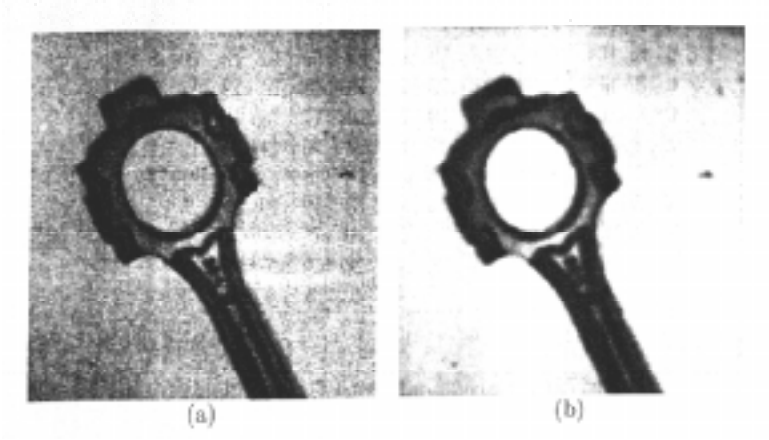

Figure 3.4 Convolution with Gaussian[10]

The resultant image from passing a Gaussian filter over an image looks like the example. The effects of the filter are a general blurring and smoothing of the original image. This appearance is due to the averaging computations that occur over several pixels while the mask is passed over the image.

3.4 EDGE DETECTION

Masks can also be used to locate features such as edges in an image. We can think of how to generate such a mask by determining first what an edge is: a distinct, usually abrupt, transition in gray-level value. Thus, if we are looking for the edges in the image, we are also looking for the large changes in brightness values over small spatial changes, or (calling I intensity or gray-level value), dI/dpixel. This relation is a familiar gradient of intensity values, or the first derivative of the value I with respect to position. Since we have only digital dx or dy positions, we say that

$$dI/dr=((dI/dy)^2+(dI/dx)^2)^{1/2}$$

and we take the 0-255 value normalized results of this relation in order to determine the magnitude of the true gradient for a given set of pixels.

Many well-known masks have been developed for use as differential operators in edge detection. A list and description is given below of some in Figure 3.5. The Robert's operator is also shown below in Figure 3.6 as it is convolved with an image. Note that as the operator approaches and crosses the edge how the edge is enhanced in comparison to its surrounding pixels.

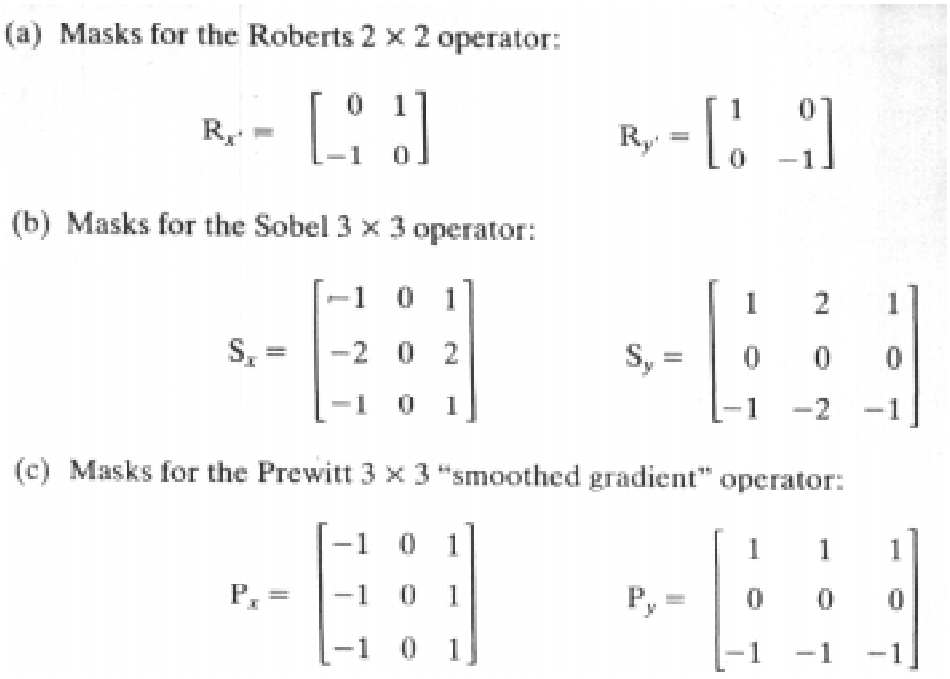

Figure 3.5 Various operators for edge detection[9]

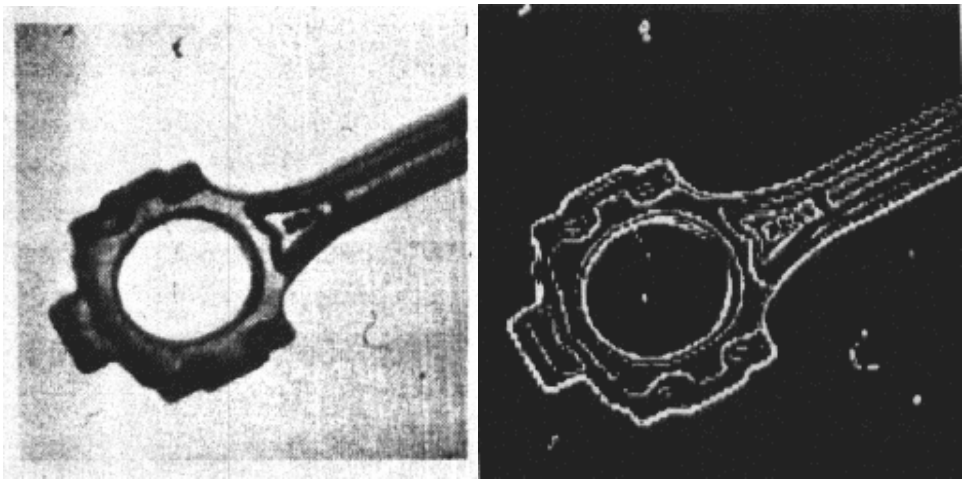

Figure 3.6 Original image on left edge-detected on right with Robert's operator[10]

A usual edge detector of choice (due to its simplicity) is the Sobel operator. The application finds the magnitude of an edge by the square root the sum of squares of the results of convolution with S_x and S_y . That is, for every pixel, we convolve S_x and S_y with the original image, then calculate $(S_x^2 + S_y^2)^{1/2}$. The result becomes the new value for the pixel in the edge-detected image. Next, each pixel is normalized to a value between 0 and 255. Finally, a threshold of these values reassigns 255 to the thresholded "on" pixels in the edge-detected image (most likely to be edges), and reassigns a value of 0 to those pixels which are most likely not edge pixels.

For further discussion later in this thesis, we provide a brief introduction to interesting information given by the separate x-dependent and y-dependent gradient operators. The very nature of the strict x or y dependency allows us to formulate gradient direction simply by taking the arctangent of the derivatives. That is, in the Sobel case we can say that

$$\text{Gradient direction} = \theta = \arctan (S_y / S_x).$$

This information will come in handy when we want to determine shapes of edges without calculating the exact curves from the edge points.

3.5 SHAPE DETECTION AND IMAGE RECOGNITION

A major portion of image understanding is based on information about the general shapes present in an image. To distinguish between a ball or a cube in an image we must first be able to recognize the differences between squares and circles, or even lines and curves. Even more complex objects such as faces or vehicles must be recognized by some telltale shape that we have either seen before (we provide a template) or that we can deduce to be a certain complex object based on the simple shapes (lines, circles, boxes) making up the object.

The first method of shape detection relies on a database of templates that are compared to the image for correlation. A template's size, location, and orientation is varied and compared to the image for correlation. Correlation is usually computed by the sum of the squares of the differences between the image pixels and the template pixels. When this sum is at a minimum, a match has occured.

This method of shape detection is usually not preferred because of the amount of computations involved in comparing the two images. Say, for instance, we wanted to find up to a 50×50 pixel template in a 256×256 image. 50×50 size permutations must be compared over 256×256 locations, as well as 360 degrees of rotation. In all, 5.89×10^{10} comparisons must be made[10], which could take hours on a 1 MIPS processor. Intelligent limitations on the template application (say, we knew the object we wanted to find was usually in a certain place, or was a certain gray-level value, etc.) could reduce time of processing, but also could limit the range of conditions under which the image was taken.

Some shape detection methods do not rely on templates but on votes accumulated according to how many object parameters have been met. The usual method of parameterization of an image space is called a Hough transform. To illustrate a Hough transform of an image, we consider a straight line in an image. For every point (x, y) in this line, there exists the relation

$$y = mx + c \, .$$

Now, in order to transform this image from value (x, y) space to parameter (m, c) space, we rewrite the above relation as

$$c = y - mx .$$

Figure 3.7 illustrates the transformation we have just made. Every value (x, y) on the original line represents a line in the new parameter space. All those lines intersect in the parameter space at the point (m, c) of the original line equation. Thus, if we accumulate votes in an array of size m × c wherever a line lies in parameter space, our maximum will be the intersection of those lines. By this voting scheme, we have located a line with slope m and intercept c in our original image.

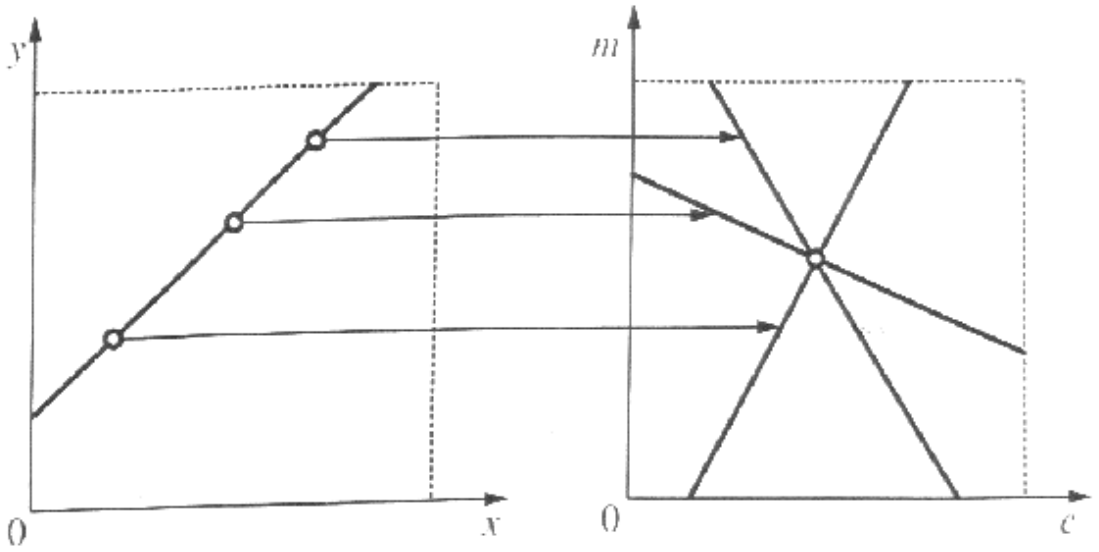

Figure 3.7 Hough transform parameterization[10]

Hough transforms are used not only for location of lines within an image, but also for more advanced shapes likes arcs, circles, and ellipses—the kinds of shapes that we will want to detect in an eye. As the order of complexity increases, however, so does the amount of parameters that need to be considered. An ellipse, for instance, has the equation

$$x^2 + 2\,Hxy + By^2 + 2Gx + 2Fy + C = 0 ,$$

which has 5 parameters. With so many parameters to consider we have fabricated a method of shape detection even more time consuming than template matching.

In order to limit the processing time of such transformations, several novel approaches have been taken to exploit the useful geometry of shapes in images. Consider again the example of an ellipse. Tsuji and Matsumoto[11] introduced a method of diameter bisection in order to determine the center of an ellipse. This method makes use of the ellipse geometry; every 2 edge points with exactly antiparallel edge orientations on an ellipse are situated at the same distance from the ellipse center. Thus, if a line is drawn from the one antiparallel edge point across the ellipse to the other, the midpoint is at the center of the ellipse as in Figure 3.8. This method is simple to implement using edge detection and a voting scheme as described previously.

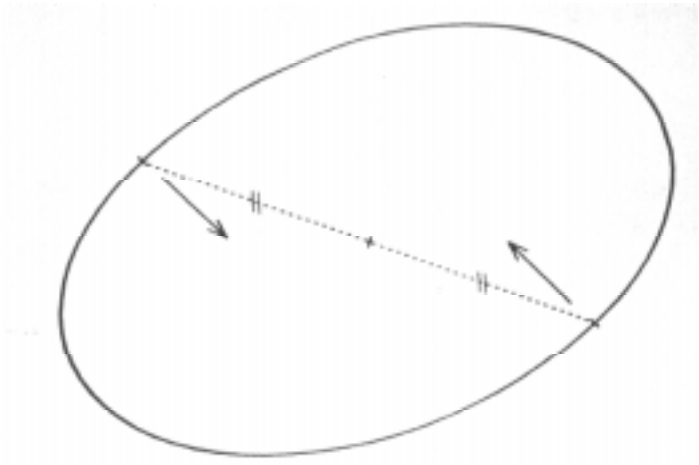

Figure 3.8 Diameter bisection method[9]

Other modified versions of Hough transforms have been applied to images to find ellipses, as well as different techniques such as least-square methods. Watanabe and Shibata[12] describe a robust approach to finding ellipses where the noise is severe and the ellipse patterns are discontinuous in images. Their method uses an ellipse template which is passed over an image in order to accumulate votes where contours of the template overlay the broken pattern of a possible ellipse in the image. The method is adaptive in

that it applies the templates at first using a very low resolution variation of the template parameters (size, rotation, translation) and formulates a list of possible ellipses for later application of more stringent template matching. The final ellipses are filled in by applying the least-squared method on the edge points matching a particular ellipse template.

3.6 OUR ALGORITHM

In these pattern matching techniques we have seen different ways of approaching the same problem with obvious advantages and disadvantages for both methods. The older ellipse detection technique is simple but highly inaccurate in its loose description of an ellipse; not only would ellipses be detected using this technique, but any highly symmetric pattern (square, rectangle, oval, or circle) would also be detected. Tsuji and Matsumoto[11] establish a test based on another geometric property of ellipses to overcome this flaw. With this test, however, comes additional time and logic for determination of true ellipses. The second technique described above is significantly more robust than the first in its treatment of ellipses, but requires much more computation and time.

Our algorithm differs from both of these methods in that the requirements for our eye detection are not based on accuracy but on time. Although some accuracy is required, finding an exact center of the iris is not as important as detecting movement and rate of movement of the iris across the eye in real time. Since real time processing for images is difficult even for most high level processors, the formulation of a detection technique was forced to limit computation to preserve speed. In addition, our algorithm used the limits established by the conditions to its advantage; only a window of the image area was used, knowledge of the relative value levels in an eye was employed, and a simplified transform

based on the eye characteristics was developed. Through these conditions we were able to formulate a working algorithm that processes images at speeds approaching real time.

The simplest way to limit the amount of pixels processed is to limit the amount of pixels counted as an original image. This windowing process can only be carried out if the system will truly never need to examine areas outside the window. In our case, since the camera's view of the eye was to be held firmly in place so that little relative movement between the camera and the eye perimeter would be expected, windowing was a viable opportunity. The smaller the window, the faster our speed became, so we could use extra time to do more computations within our small window.

The detection of the center of the pupil or the iris reduces to the problem of finding an ellipse center in the eye's two dimensional image space. We have seen several methods of ellipse detection—all of them accurate but time intensive. For our purposes, a simpler, less time consuming method of detection was appropriate. In addition, we decided to employ the generalized values (dark pupil, white sclera) as markers in our detection scheme.

The algorithm's process has four steps: edge detection, transformation, vote accumulation, then estimated guess at center of ellipse or circle. The edge detection within the chosen window is accomplished through a Sobel operator, and the edge points are binarized through simple thresholding. The gradient direction at each edge pixel is preserved for the next processing step, where a line is drawn in the opposite direction of the gradient, starting its corresponding edge point. Figure 3.9 illustrates the effect of these lines drawn for an image of a dark circle on a light background. The lines begin on the location of the detected circle edge, then proceed in a direction toward the center of the circle, where they intersect. The next step in the algorithm accumulates the amount of gradient lines passing through a pixel, so that the most votes will be accumulated where many lines intersect (in our example, at the center of the circle). The final step finds the location of the average maximum from the local maxima that are generated in the accumulator array.

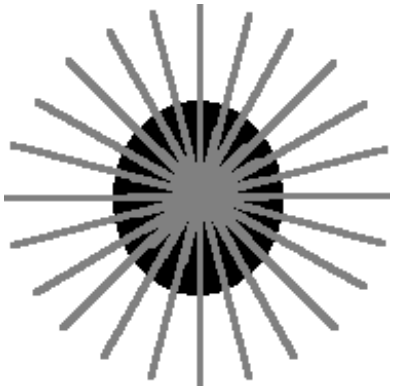

Figure 3.9 Hypothesized result of algorithm

The process above gives an example of a dark circle within a white background—representative of our eye problem (dark iris and pupil on white sclera). What becomes, however, of the other arcs in the image such as the upper and lower eyelids encircling the eye? These edges will also generate lines drawn, so how can our algorithm avoid detecting other maxima? Although we could try to draw our window so that the eyelids are avoided, the special handling of the area is unnecessary. The algorithm is tailored to draw a line from an edge in the direction of light to dark (opposite the gradient direction). So, the lines drawn from an eyelid will diverge outward never traveling into the eye area because the eyelid gray-level value is much darker than the sclera.

We treat the special case of detecting an ellipse because as the eye turns to wide angles, its two dimensional shape becomes more oblong. The ellipse will generate two maxima at its foci in our accumulator array, so statistical averaging must handle finding the true center. The clusters of pixels with the largest values are weighted for a normalized average over the image dimensions. The average location will be the ellipse's true center.

4. INTEGRATED SYSTEM

The integrated system prototype design is pictured below in Figure 4.1. The system is broken up into two major subassemblies: the head gear or eyeglass assembly, and the separate processor and camera assembly. Each module will be described and presented so that its purpose and configuration is linked with the whole system according to its function.

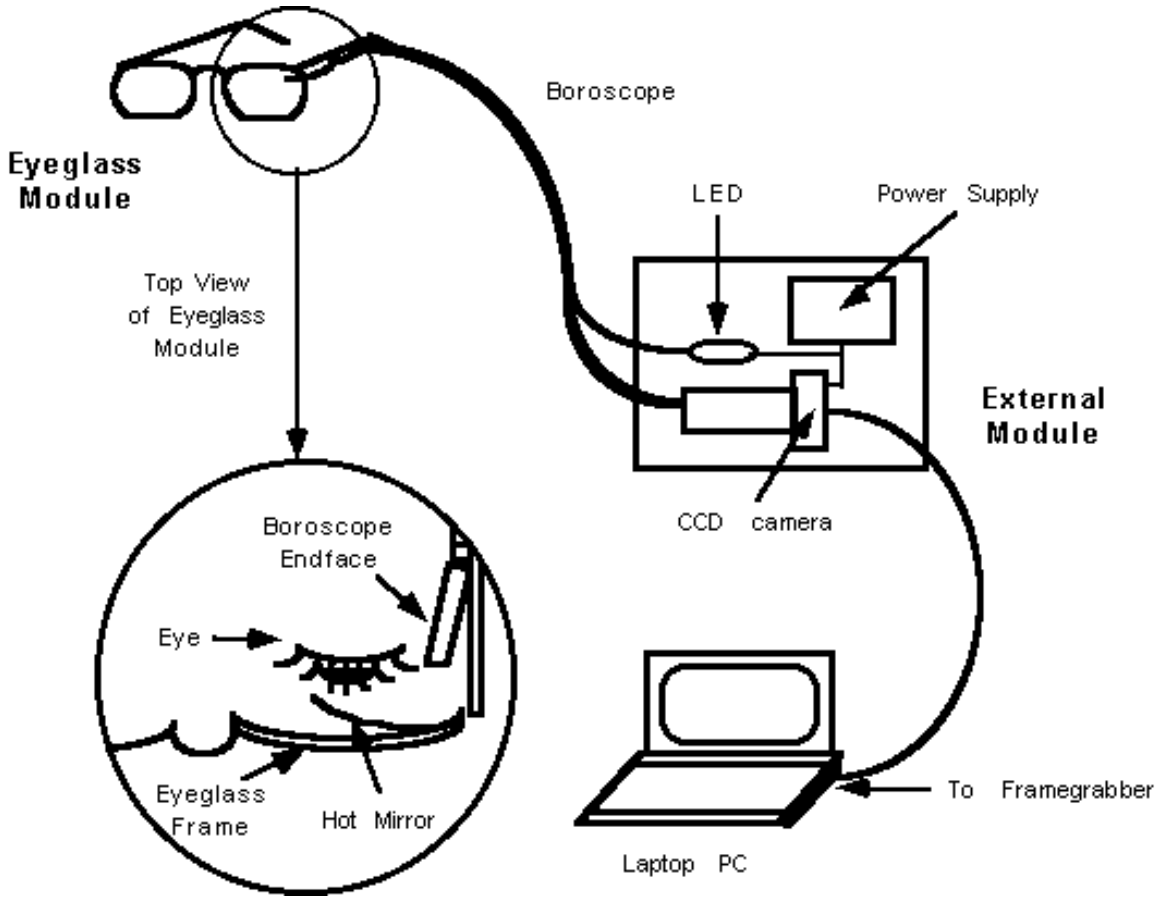

Figure 4.1 Integrated System Design

4.1 THE EYEGLASS MODULE

In Figure 4.1, the parts of the eyeglass module are illustrated as they would be assembled in their respective positions on the head gear. Each of these parts will be described in turn. Information on part mounting and assembly, as well as how each part contributes to the eyeglass assembly, is given below.

4.1.1 Hot mirror

The hot mirror used for experiments in this thesis, depicted in Figure 4.2, is merely a prototype for the ideal hot mirror we define for our final design. However, it serves to show the function of the hot mirror in our eyeglass assembly—as a method of directing the IR illuminated eye image back into the boroscope. Ideally, the hot mirror would be deposited directly on the eyeglass lenses, or on a small extended piece that attached to the eyeglass frame as was seen in Figure 4.1. This piece could easily slide into place in plastic slots cut in the frames, or glued or fastened if a permanent fixture was desired.

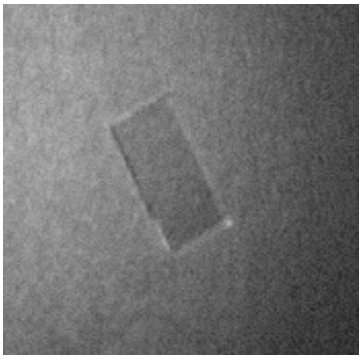

Figure 4.2 Hot Mirror

The prototype hot mirror was used in experiments to demonstrate the effectiveness of its filtering capabilities. We observed a white light transmission spectrum with and without the reflector in front of the source to note that the IR portion of the spectrum changed in the second case—indicating that most of the IR light had been reflected. Figure 4.3 indicates the image of the LED in the hot mirror as seen by our CCD camera, demonstrating that the camera and hot mirror produce an excellent image of the 880 nm LED. Objects illuminated by the LED only, like our practice eye shown later in Figure 4.6, also image well even in when ambient light is not present (in total darkness).

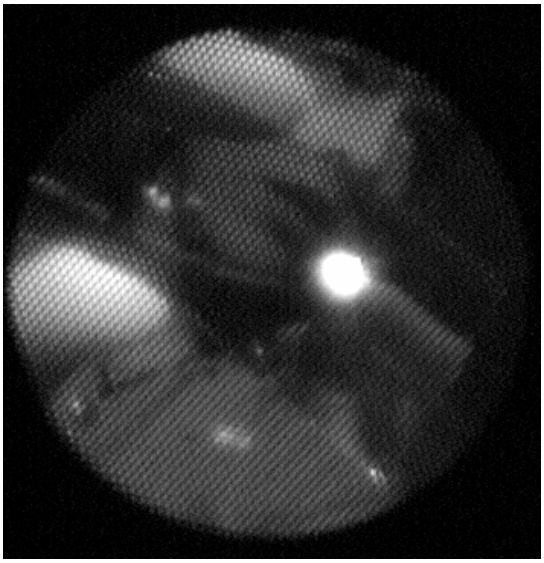

Figure 4.3 Image of LED in hot mirror

4.1.2 Fiber optic bundle

The boroscope purchased from Edmund Scientific is shown in Figure 4.4. The device has three ports: one for image viewing (or eyepiece), one for light injection, and one for placement near the object to be viewed. The last end mentioned would be fastened to our eyeglass assembly as shown previously in Figure 4.1, so that the hot mirror image would be incident on that imaging bundle endface.

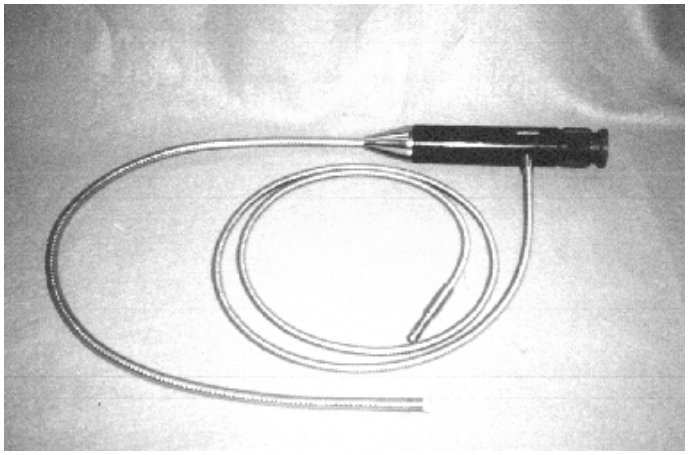

Figure 4.4 Boroscope

This approximately 3400 fiber imaging bundle endface is encircled by a ring of light injection fibers in a circular configuration. Our LED was coupled into the other end of the injection bundle so that the light illuminated the object that the imaging fiber bundle targeted. The design of the boroscope was ideal for our application in that it required no other LED or camera attachments to reside on the eyeglass assembly.

4.2 THE EXTERNAL MODULE

In Figure 4.1, we presented the external gear that accompanies the eyeglass unit. We begin our description of this gear with the design of a small pocket-sized unit to house the CCD camera and the LED. The remaining parts to the system, the PC and the framegrabber, are then discussed.

4.2.1 CCD camera

The C-mount CCD camera purchased from Edmund Scientific, shown in Figure 4.5, provided excellent resolution for display while requiring very little power and taking up little physical space. The CCD images presented earlier in this thesis show evidence of the camera's superb imaging abilities when the camera was lensed. The low power requirements were proven by our finding that a small 100 mA, 9.0 V power supply powered the unit adequately. Also, the 1 ¼" square unit was compact enough for us to design a pocket-sized enclosure to contain both the camera and the LED.

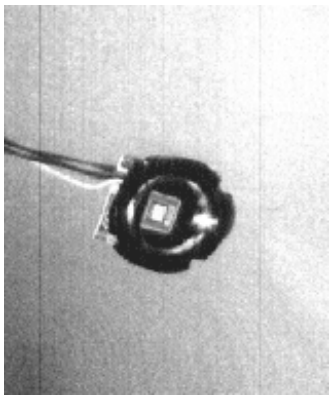

Figure 4.5 Edmund Scientific CCD camera

In order to mate our boroscope directly to our lensless CCD camera, we purchased a C-mount coupler to clamp the boroscope directly onto the camera's C-mount. The lens elements in the boroscope were then used to focus the boroscope's images onto the CCD. An example image captured through the boroscope of our practice eye as it was illuminated by the LED is shown in Figure 4.6. Clearly, the

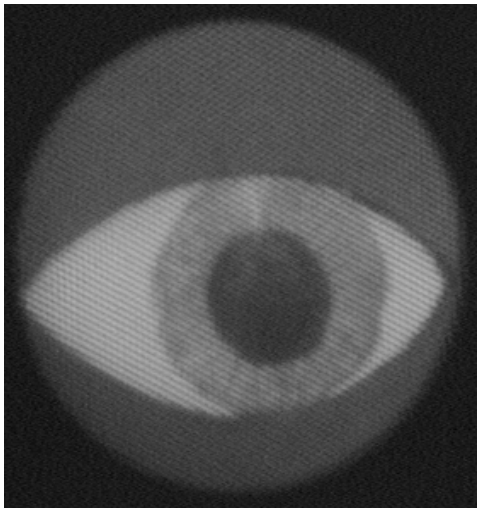

Figure 4.6 Practice Eye

image of our practice eye produced by our proposed setup was adequate for vision purposes.

4.2.2 LED

The other element to be housed in the pocket-sized unit was the 880 nm LED. The Gilway Technical Lamp LED we chose for this thesis is shown below in Figure 4.7.

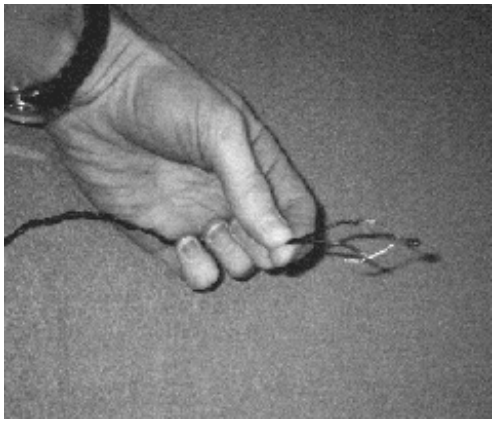

Figure 4.7 Gilway Technical Lamp LED

The LED is 4.0 mm in diameter and extremely lightweight. We measured its power output—given in the table below for three different configurations.

2 mm away from LED …….……. 10.23 mW
reflected from hot mirror……..…..……..627.2 μW
2 mm away from fiber bundle port (LED light injected into other port)….1.370 mW

In Figure 4.7, the LED was under full power when the photograph was taken. Observe that the picture does not indicate any light emitting from the diode. This image duplicates what the human eye would see upon viewing the LED under full power—no light. This effect was what we desired in an IR LED so that the illumination of the eye was unobservable by the user. An image from the CCD, however (shown in Figure 4.8), proves that the IR LED is quite visible from the camera's standpoint.

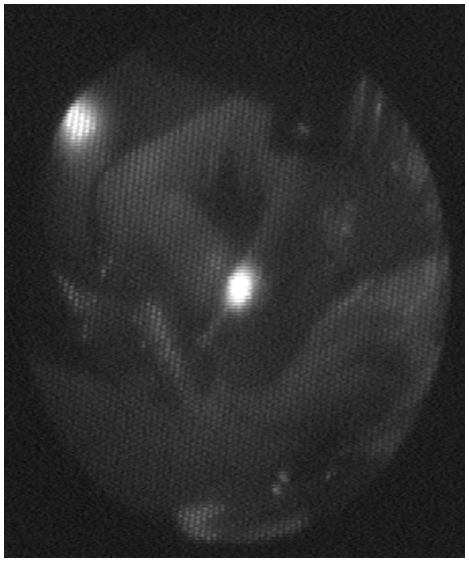

Figure 4.8 LED in CCD image

Although the LED could have been mounted on the eyeglass assembly without interfering with a user's sight or movement because of its small size, we opted to use the fiber ring encircling our boroscope fibers for light injection in our design of the final system. The fiber ring design would require more power than our LED could give, however, since light was lost in the fiber length and good coupling from the LED to the fiber was difficult. In the majority of our actual tests with the LED, the lamp was mounted directly onto the boroscope endface in order to provide enough light to the object being monitored.

We avoid problematic injection light loss by coupling the LED properly to the light injection portion of our boroscope. This coupling is provided by a conforming housing to sheath the LED and the endface of the boroscope piece in alignment. A picture of this assembly is given below in Figure 4.9.

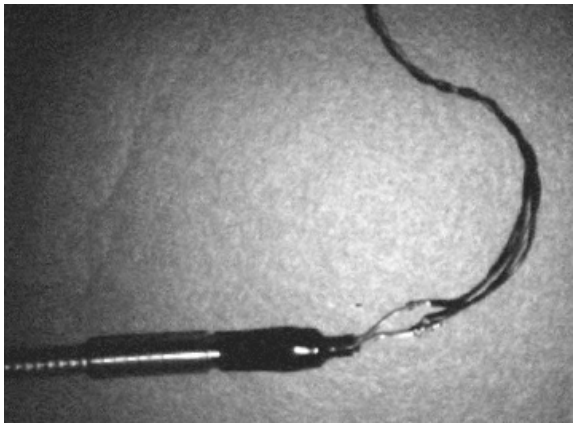

Figure 4.9 Coupling of LED to boroscope

4.2.3 The Framegrabber

The framegrabber card purchased from Imaging Technology captured the camera images presented earlier is this thesis using the driver software provided by Imaging Technology and original software generated by the thesis author. The programs, written in Microsoft Visual C++, use functions to capture the image data and fill an array with the frame information.

Frame information provided by the card is provided in 32-bit words—each of which represents four pixels of 8-bit grey-level data. These words had to be separated into individual pixel data so that image manipulation could be conducted more easily. The data was simply written to a raw data file whenever a saved picture of the image was desired. The raw images were altered in Photoshop to .BMP format or .TIF format for display in other files such as this thesis.

4.2.4 The Program

The software written by the author utilized a windows-based framework in order to facilitate a non-programmer's use of the program. A window with a drop-down menu contains a File Save button and a Pupil Find button—the most used features. Additional

features are available from the drop-down menus to set up the camera and to snap (single frame acquire) and grab (video acquire) images. A picture of the window with a sample snapped image is shown in Figure 4.10.

Figure 4.10 Sample window from program

Although in an actual eye tracking system program, pupil tracking would be automatic and continuous, our program was written to search for the pupil center when the user designated it by the push of the Pupil Find button. We designed our prototype to accept user input because of constant changes that would most likely occur in the program while experiments were being conducted.

When the Pupil Find button is pushed, the program was designed to snap the current image and begin targeting the pupil through a series of initial image processing modules. After the image is snapped, the 32-bit word (4 pixels of information) data is collected across each row in succession until the entire image fills a 32-bit word array.

The next process reorders the array into a new separated 8-bit pixel array. The pixel array is then diminished according to the smaller window chosen inside the whole image.

After the initial image processing has prepared our windowed image array, several computer vision techniques are applied to the array data to find edge points, thresholds, and gradient directions. The image array is an input parameter for a modified Sobel operator function that finds edge points through two masks:

$$Sx = \begin{matrix} -1 & 0 & 1 \\ -3 & 0 & 3 \\ -1 & 0 & 1 \end{matrix}$$

$$Sy = \begin{matrix} 1 & 3 & 1 \\ 0 & 0 & 0 \\ 1 & 3 & 1 \end{matrix}$$

The magnitude of the convolution of each mask with the image (computation of the gradients G_x and G_y) gives floating point numbers for edge points in the image. These edge points are normalized to values between 0 and 255 so they can replace the values in the input image in a new, edge-detected output image. The gradient components, G_x and G_y, at every edge point are saved in an array for later processing. Thresholding is conducted in the next function; the minimum and maximum of the input edge-detected image's values are used to create the threshold value. If a pixel's value is below the threshold value, that pixel is not considered an edge point and is assigned the value 0 in the output array. If the pixel value exceeds the threshold, it is output as a value 255 in the output array. Thus, when both the edge-detection and the thresholding functions have processed our windowed image, we are left with pixels that are either on an edge or not on an edge.

With our binarized image we need to conduct the last step of our algorithm—the drawing of the inverse gradient lines to statistically mark our pupil center. An accumulator array the size of the windowed input image is established and initialized so that all of its values are zero. The input image to our CenterDetect function is inspected for a percentage of its edge points (to conserve processing time not all edge points are used). If an edge point is detected in the window, a line is drawn in light grey (value 100) in the opposite direction to that point's gradient (saved in an array during the Sobel convolution). The lines drawn are saved in the accumulator array; every point that one of the lines crosses in the window has its value incremented. Once the whole window has been investigated, the accumulator array is checked for local maxima that could be an ellipse focus or a circle center. If the maxima are within limits of the pupil's or iris's span, then the weighted distance between them is calculated as the location of the center of the pupil.

The calculation of the center location from the weighted maxima is conducted through an equation resembling a relation for center of mass. A maxima is defined as a location where more than two hits have occured. The number of hits for a maxima is multiplied by the location component (x or y) and added to the total number of hits. The next maxima is multiplied by its respective hit number and added to the first multiple. Its number of hits is also added to the total number of hits. The sums of number of $hits_n * x_n$ and number of $hits_n * y_n$ are accumulated as well as the sum of number of $hits_n$. The center location is then determined by the relation

$$x_{center} = \Sigma \text{ number of } hits_n * x_n / \Sigma \text{ number of } hits_n$$

$$y_{center} = \Sigma \text{ number of } hits_n * y_n / \Sigma \text{ number of } hits_n$$

The calculated center is displayed, along with the lines drawn by the CenterDetect algorithm, in a window in our user-friendly image capture environment. An example of

the output given for our practice eye is shown in Figure 4.11 A white square, indicating the approximate center of the pupil, is shown in the figure as a result of statistical calculation. Notice that the window drawn around the region of interest of the eye contains all image manipulation. The remainder of the image is left unchanged.

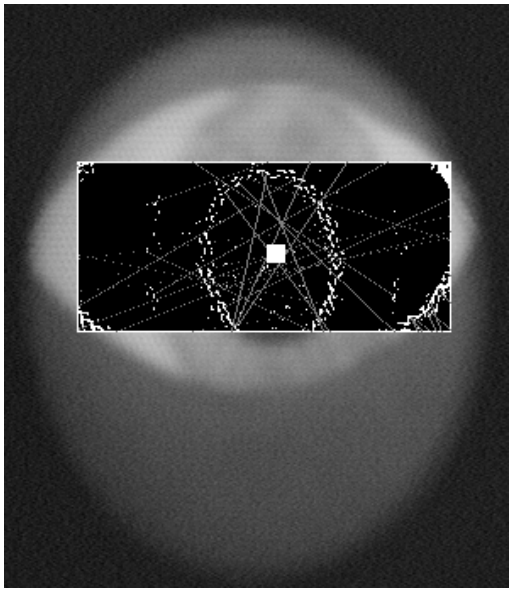

Figure 4.11 Sample program output

5. EXPERIMENTAL PROCEDURES AND RESULTS

The experiments run using our system were conducted using a practice eye for the majority of the tests, although some critical findings could only have been made with our brief studies with an actual human eye. We captured 20 frames of the practice eye and counted the correct targeted centers during the test. Possible errors are categorized and explained. The test with a human eye was also conducted, with different setup conditions as explained below.

5.1 FEASIBILITY

The setup of the system entailed using the prototype design as planned to demonstrate feasibility, but to use a modified setup for program tests. We set up the hot mirror in a lensless glasses frame with the boroscope affixed to the side of the frame as depicted in Figure 5.1. The only difference in this setup from the proposed design was that the LED was mounted on the boroscope endface near the hot mirror. This modification resulted from repeated unsuccessful attempts to couple enough light from our LED into the injection portion of our boroscope.

Figure 5.1 Eyeglass assembly

The author wore the device while images were captured and trial runs of the pupil detection program were run. Figure 5.2 shows an image captured of the author's eye in the hot mirror. From this image, we can see that the light level is too low and that the LED produces a detrimental glint on the eye surface.

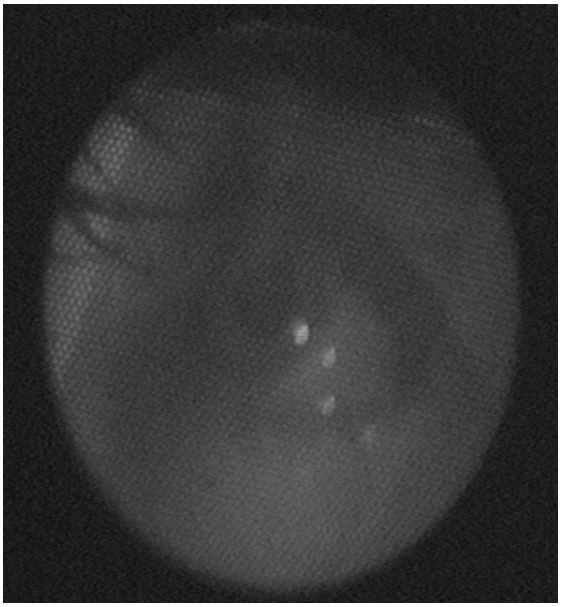

Figure 5.2 Reflected eye in hot mirror

Nevertheless, a sample run of the program was conducted with the system set up as indicated. A sample output of the program from one of these trial runs is shown in Figure 5.3. Instead of detecting the pupil center, the program located the center close to the pupil edge. Notice the great amount of noise introduced into the image from the low light level.

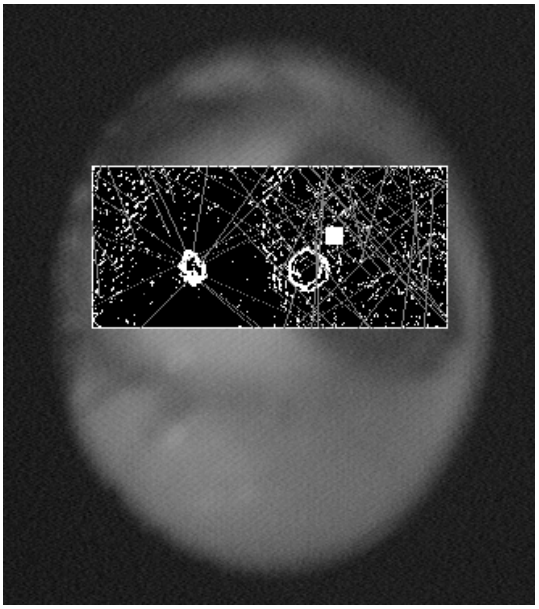

Figure 5.3 Human eye program output

5.2 PRACTICE EYE

Most of the Pupil Find program tests were conducted using a practice eye image drawn in pen ink on white paper. The practice eye had a movable iris and pupil made of a cut out slot of paper that could move over a white oval to resemble eye movement. This object was placed in the line of sight of the boroscope and illuminated with the LED through direct mounting on the boroscope for 20 frames of program testing, and then through injection through the fiber bundle for the next 20 frames.

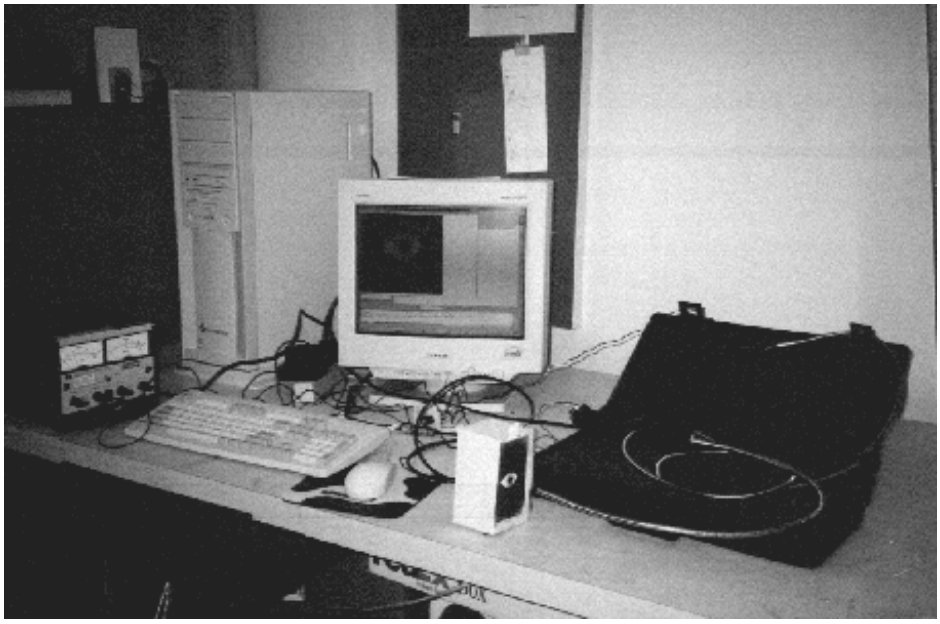

Figure 5.4 Test setup

The physical layout of the tests conducted is pictured in Figure 5.4. Notice the placement of our practice eye in front of the boroscope. The eye was moved to a random location, then the Pupil Find menu option was chosen in the program. This process was repeated 20 times using direct LED illumination of the eye, and 20 more times using the boroscope as the light injection means. An example of a trial run of the Pupil Find program with direct light illumination is given in Figure 5.5, while a trial run using the boroscope for illumination is given in Figure 5.6. Notice the additional noise due to the low light level.

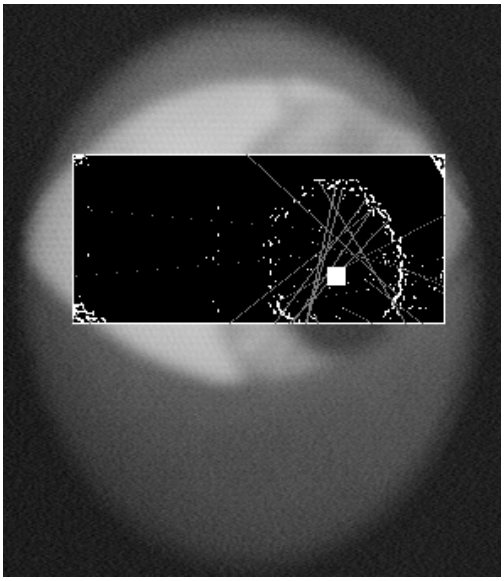

Figure 5.5 Practice eye illuminated directly by LED

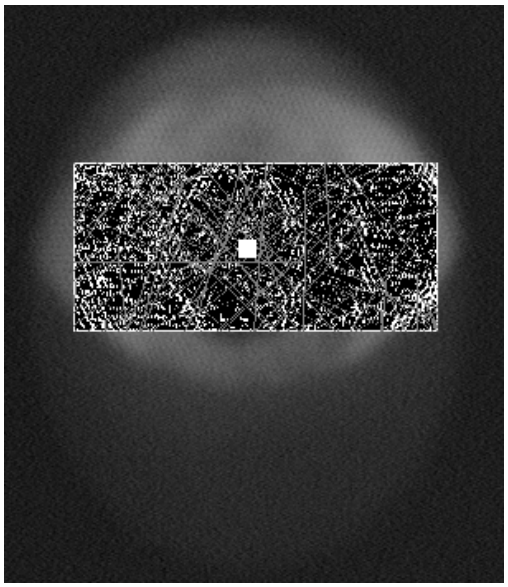

Figure 5.6 Practice eye illuminated through boroscope

The results of the direct illumination tests show that on average the x coordinate of the center was accurate to within ±6.95 pixels The y coordinate was accurate to within ±7.7 pixels. Since the pupil height was 102 pixels and the pupil width was 79 pixels, our center was accurate within a 1/6.15 pupil radius about the center. The boroscope

illumination results did not fare as well, with a ±11.5 x coordinate and a ±11.5 y coordinate shift. The new accuracy radius for the boroscope illumination test was 1/3.69 of a pupil radius.

One last item to note about the differences between these two tests was their process times. The first test condition had process speeds of 3 seconds whereas the speed of the second test was about 13 seconds. The addition of noise and therefore addition extraneous edge points added significantly to the effort required to calculate the pupil center.

6. CONCLUSIONS

We have seen through the results presented here that our eye tracking prototype is feasible as an application, yet needs many modifications to be a working device. The program is accurate to within less than a 1/5 pupil radius about the pupil center, yet is quite slow due to calculation quantity. In addition, the system speed and accuracy is hindered by improper illumination The reflection in the hot mirror should also be investigated because of other detrimental images that appear in transmission through the hot mirror glass.

The processing problems could be lessened if more attention were paid to the processor itself. To increase speed, parallel processing techniques have been developed for many imaging applications. We should consider this kind of processing—perhaps on a different operating system like UNIX. In addition, we could use more advanced windowing to minimize calculation time.

The illumination problem could be eliminated through a redesign of the illumination of the eye. A more powerful IR LED source or a filtered white light source injected through the boroscope fibers could provide the added light. LEDs could be placed on the glasses for direct illumination with little added conflict for the user. For all of these proposed redesigns, a method of diffusing the light (for example, frosted glass) before it strikes the eye and produces a glint is crucial.

The problem of the camera detecting not only the LED illuminated image on the hot mirror, but also any object through the glass could be handled by simple filtering. An IR sensitive camera would also solve the problem, but would increase the cost of the system substantially. We propose instead that a visible light filter (high bandpass) be used in front of our inexpensive camera to block out images of objects appearing through the hot mirror.

We believe that through the addition of these optimizing elements, we will have introduced a novel and practical solution to the eye-tracking problem for the case where

driver drowsiness monitoring is key. We also hope that the simple algorithm for detecting the ellipse center can aid in detecting ellipses in various other images—not specifically the eye-tracking case. With the work accomplished in this thesis, a progression towards eye tracking development has been made, as well as a merger of computer vision and optical engineering solutions.

REFERENCES

1. http://www.adm.duke.edu/alumni/dm3/sleepc.txt.html, March 1997.

2. Scott, D. and Findlay, J.M., Visual Search, eye movements and display units, Human factors report, University of Durham, South Road, Durham DH1 3LE, UK, before 1993.

3. http://www.diku.dk/~panic/eyegaze/node9.html, March 1997.

4. Tock, D. and Craw, I., “Tracking and Measuring Drivers’ Eyes,” Image and Vision Computing, Vol. 14, 1996, pp. 541-547.

5. Mooij, H.A. and Howieson, J., "A GazeTracker for Real-Time Digital Point-of-Gaze Determination" , ESA Preparing for the Future, Vol. 6, No. 2, June 1996, pp. 12-13.

6. http://hermes.cns.uiuc.edu/research/EyeMovement.html, March 1997.

7. Gilway Technical Lamp “T-1 and T-1 ¾ Infrared LEDs: 940 nm, 880 nm” Data Sheet, October 1997.

8. Electro-Optics Handbook, Burle Industries Inc. Inc., 1974, p.174.

9. Machine Vision: Theory, Algorithms, Practicalities, Davies, E.R., Academic Press Inc., SanDiego, CA, 1990, pp. 1, 102, 266.

10. Machine Vision, Jain, R., Kasturi, R., and Schunk, B., McGraw-Hill Inc., New York, NY, 1995, pp. 26, 84, 87, 135, 136, 150, 219.

11. Tsuji, S. and Matsumoto, F., “Detection of Ellipses by a Modified Hough Transform,” IEEE Trans. Comput., Vol. 27, 1978, pp. 777-781.

12. Wanatabe, T.and Shibata, T.; “Detection of Broken Ellipses by the Hough Transforms and Multiresolutional Images”, Systems and Computers in Japan, Vol.22, No. 8, February 1990, pp. 159-166.

VITA

Noel Annette Zabaronick was born in Columbus, Ohio on November 24, 1970. She received her Bachelor of Science degree in Physics in May of 1992 from the Virginia Polytechnic Institute and State University. She worked as an Associate Member of the Technical Staff at Burle Industries, Inc. from July of 1992 to July of 1995. She then worked as a Graduate Research Assistant at the Fiber and Electro-Optics Research Center while pursuing her Masters of Science degree in Electrical Engineering. She received her MS degree from the Bradley Department of Electrical Engineering at the Virginia Polytechnic Institute and State University in May of 1997. Her work interests include computer vision, automation, and data acquisition.