

Department of Computing

Realtime Computer Interaction via Eye Tracking

Premnath Dubey

This thesis is presented for the Degree of
Master of Science
of
Curtin University of Technology

June 2004

Abstract

Through eye tracking technology, scientists have explored the eye's diverse aspects and capabilities. There are many potential applications that benefit from eye tracking. Each benefit from advances in computer technology as this results in improved quality and decreased costs for eye-tracking systems.

This thesis presents a computer vision-based eye tracking system for human computer interaction. The eye tracking system allows the user to indicate a region of interest in a large data space and to magnify that area, without using traditional pointer devices. Presented is an iris tracking algorithm adapted from Camshift; an algorithm originally designed for face or hand tracking. Although the iris is much smaller and highly dynamic, the modified Camshift algorithm efficiently tracks the iris in real-time. Also presented is a method to map the iris centroid, in video coordinates to screen coordinates; and two novel calibration techniques, four point and one-point calibration. Results presented show that the accuracy for the proposed one-point calibration technique exceeds the accuracy obtained from calibrating with four points. The innovation behind the one-point calibration comes from using observed eye scanning behaviour to constrain the calibration process. Lastly, the thesis proposes a non-linear visualisation as an eye-tracking application, along with an implementation.

Acknowledgement

As it was always my intention to do a Masters by Thesis, working on it has given me much pleasure. One of the reasons is because I have had the opportunity to work with Prof. Geoff West and Dr. Allan Loh. I have to thank Geoff for his guidance, without which I would have gotten lost on this long journey, and for his patience and understanding because, on a number of occasions he has had to correct my mistakes as well as help me with my understanding. I thank Allan, my co-supervisor who was always actively helping me with problems. He has been giving me a lot of good advice and reminded me of a number of points and concepts that I missed. I have learned a lot about good research from both of them during my working on the thesis.

I have to thank my family for their unconditional loving and support. It has been difficult for them to understand the details of what I have been doing and why I have been away from home for so long. Most of all I would like to thank AusAID, the Australian Government and the people of Australia for the sponsorship of my degree. They have offered one of the greatest opportunities I have ever received. It is not just the opportunity to learn in an academic environment but also the opportunity to understand the differences and the commonalities between our two cultures. Most importantly, it was the opportunity to make friendships that will last forever.

Abstract.....	i
Acknowledgement	ii
Chapter 1 Introduction	1
1.1 Program Features and Implementation Detail.....	3
1.2 Contributions from this Research in Addition to the Thesis.....	5
Chapter 2 Background.....	7
2.1 Introduction	7
2.2 Human Visual Perception.....	8
2.2.1 Eye Anatomy	9
2.2.2 Types of Eye Movements.....	11
2.3 Applications of Eye Tracking	12
2.3.1 Psychology, Psychophysics, and Neuroscience	12
2.3.2 Ergonomics and Human Factors	13
2.3.3 Marketing/Advertising	13
2.3.4 Digital Displays	13
2.3.5 Human-Computer Interaction (HCI) and Collaborative Systems	14
2.4 Current Commercial Eye Tracking Techniques.....	15
2.4.1 Electro-Oculography (EOG)	15
2.4.2 Scleral Contact Lens/Search Coil.....	17
2.4.3 Infrared Oculography (IROG).....	17
2.4.4 Vision Based or Video Oculography Systems (VOG).....	18
2.4.4.1 Vision Based Light Reflection	19
2.4.4.2 Vision Based Non-Reflection.....	21
2.5 Current Research in Video-Based Eye Tracking	22
2.5.1 Face Detection	22
2.5.2 Eye Detection	24
2.5.3 Gaze Estimation.....	25
2.6 Information Visualisation Techniques	29
2.6.1 Polyfocal Display	32
2.6.2 Bifocal Display.....	33
2.6.3 Fisheye View	34
2.6.4 Perspective Wall.....	35
2.6.5 Graphical Fisheye Views.....	36

2.6.6 Hyperbolic Geometry	37
2.7 Summary	37
Chapter 3 Iris Tracking	39
3.1 Introduction	39
3.2 Camshift	40
3.2.1 Creating the Colour Histogram Table	41
3.2.2 Moments	43
3.2.3 The Camshift Process	45
3.3 Iris Tracking by Camshift	48
3.4 Search Window Size	51
3.5 Bin Size	53
3.6 Camera Location	54
3.7 Eye Detection	55
3.8 Updating the Histogram Table	56
3.9 Tracking the Whole Eye.....	58
3.10 Summary	61
Chapter 4 Calibration.....	63
4.1 Introduction	63
4.2 Four-Point Calibration.....	65
4.2.1 Terminology and Notation	69
4.3 One-Point Calibration	69
4.3.1 First Stage – The Offline Information Collection Stage	71
4.3.2 Second Stage - The Online Stage	80
4.4 Performance Comparison of the Two Techniques.....	89
4.5 Discussion	91
4.6 Summary	92
Chapter 5 Integration of Eye Tracking and Non-linear Visualisation	93
5.1 Introduction	93
5.2 Benefit of Eye Tracking and Non-Linear Visualisation.....	93
5.3 Applying Non-Linear Visualisation	95
5.4 Implementation Details	98
5.5 Results and Discussion.....	101
5.6 Summary	101
Chapter 6 Conclusion.....	103

References.....	106
Appendix A.....	112
Paper presented at the International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Perth, Western Australia, December 2004.	112
Eye Tracking as a Computer Interface	112
Appendix B	118
Slides for the paper presented at the International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Perth, Western Australia, December 2004	118

Figure 1.1 Screen shot of the program.....	4
Figure 1.2 Screen shot of the Non-linear visualisation window.....	5
Figure 2.1 Rods and cones cell in retina (Duchowski and Vertegaal, 2000).....	9
Figure 2.2 Eye anatomy (WebVision, 2003).....	9
Figure 2.3 Retina, optic nerve and fovea (WebVision, 2003).....	10
Figure 2.4 The eye's extraocular muscles (Bores, 2002).....	11
Figure 2.5 The Dresden3D Display D4D with two cameras on the top for tracking eye and head (SeeReal Technologies, 2003).....	14
Figure 2.6 Example of the set up of the EOG technique (Duchowski and Vertegaal, 2000).....	16
Figure 2.7 The electric potential differences when an eye moves (Malmivuo and Plonsey, 2002).....	16
Figure 2.8 Example of scleral suction ring insertion for search coil eye movement measurement (Duchowski and Vertegaal, 2000).....	17
Figure 2.9 Limbus tracking from Microguide (Microguide, 2003).....	18
Figure 2.10 Illustrate light setting in limbus tracking technique (Carpenter, 1977)....	18
Figure 2.11 Pupil cornea light reflection a) Bright-eye image b) The first Purkinje (Morimoto, Koons et al., 1998).....	19
Figure 2.12 The Purkinje image (Yang, Dempere-Marco et al., 2002).....	20
Figure 2.13 a) Eye tracking with head frame (Istance and Howarth, 1995), b) remote eye tracker (EyeTech, 2002).....	21
Figure 2.14 Yuille's eye and mouth model.....	25
Figure 2.15 Relative spatial relationship between glint and bright pupil centre used to determine eye-gaze position. (a) bright pupil images, (b) glint images; (c) pupil/glint relationship generated by superimposing glint on to the thresholded bright pupil images (Ji and Zhu, 2002).....	26
Figure 2.16 The head and eye prototypes (Rikert and Jones, 1998).....	28
Figure 2.17 Information visualisation techniques (Leung and Apperley., 1994).....	30
Figure 2.18 Relationship between transformation and magnification functions (Leung and Apperley, 1994).....	32
Figure 2.19 Transform function (a) and magnification function (b) of the polyfocal display with $A = 6$ and $C = 15$	33

Figure 2.20 Transform function (a) and magnification function (b) of the Bifocal display	34
Figure 2.21 Relationship between object image and target image	34
Figure 2.22 Perspective Wall (Robertson, Mackinlay et al., 1991).....	35
Figure 2.23 Graphical fisheye view of graph from Sarkar and Brown (Sarkar and Brown, 1992)	36
Figure 2.24 show transformation and magnification function of graphical fisheye view	37
Figure 3.1 A 3D plot of the 2D histogram of 10×100 bins (H×S).....	42
Figure 3.2 A 3D plot of the 2D histogram of 10×5 bins (H×S).....	43
Figure 3.3 The iterations of Meanshift process to search an object (For the 2 nd to the 5 th diagram, the black cross indicates the mean computed from the window, and the grey cross indicates the position of the mean in the previous image).....	46
Figure 3.4 The iris being tracked by Camshift	48
Figure 3.5 Steps of the iris tracking process	49
Figure 3.6 Back-projection images of hue only, hue and saturation, and original image.....	50
Figure 3.7 Back-projection image when using the 3D histogram of HSV	50
Figure 3.8 Demonstrating the iris tracker	51
Figure 3.9 The iris tracker with unlimited window size	52
Figure 3.10 The iris tracker with 35×35 window size	53
Figure 3.11 Back-projection images of three different bins size, (a) 5×5, (b) 7×7 and (c) 9×9.....	54
Figure 3.12 Iris template.....	56
Figure 3.13 Show a circle within the iris	57
Figure 3.14 Back-projection images of before and after updating the histogram table	58
Figure 3.15 Location of the nine points on the screen used for testing	60
Figure 3.16 Back-projection image of the whole eye	60
Figure 3.17 Graph of iris and the whole eye centroid coordinate when subject look at different location on screen.....	61
Figure 4.1 The four calibration points (red squares halfway along each side) on the screen.	65

Figure 4.2 A picture captured from the camera with the setting used by iris tracking.	66
Figure 4.3 Relationship between the eye plane and the screen plane.....	66
Figure 4.4 Screen capture of calibration point of one-point calibration	70
Figure 4.5 Nine (9) points on screen used for calibration	71
Figure 4.6 Variation in the x video coordinate when subject observed three rows on screen	72
Figure 4.7 Variation in the y video coordinate when subject observed three columns on screen	73
Figure 4.8 Variation in x and y video coordinates from another test	75
Figure 4.9 Fitting straight on graph of y video coordinate from figure 4.7	77
Figure 4.10 Same data as for figure 4.9 except with the axes swapped over	78
Figure 4.11 Shows a graph of a linear function with slope M_{y2}	80
Figure 4.12 Show graph of all of the 3 column of y coordinate	82
Figure 4.13 Drawing a straight line to the last point of each column to find relationship between these columns	83
Figure 4.14 Show the complete information all the three columns achieved from this process	85
Figure 4.15 Resulting shape of the real eye plane	87
Figure 4.16 Comparison of performance between one-point calibration and four-point calibration	89
(a)	90
(b)	91
Figure 5.1 Non-linear visualisation effects on grid lines.....	95
Figure 5.2 Tanh transform and magnification function.....	96
Figure 5.3 comparison of magnification function with 0.015 and 0.05 β value.....	97
Figure 5.4 Comparison magnified grid of 3 different A values 2 3 and 4.....	98
Figure 5.5 Graph of magnification ratio with $A = 2$ and $\beta = 0.015$	100

Chapter 1 Introduction

The aim of this research is to investigate a computer vision based eye tracking system for the application of Human Computer Interaction (HCI). The HCI application of interest in this thesis is non-linear information visualisation i.e. the position of the eye indicates the portion or Region Of Interest (ROI) of an image that should be magnified for analysis. Here the eye would be used instead of a traditional pointer device such as a mouse. The integration between eye tracking and non-linear visualisation is envisaged to yield a novel human computer interaction system that allows users to navigate across large data spaces using their eyes. However the research focuses mainly on developing an efficient eye tracking method to enable interaction in real time. The system uses a simple webcam type camera to track one of the user's eyes. For non-linear visualisation a preliminary study and implementation has been carried out to determine the potential for further development. The eye tracking system will be used to indicate a region of interest in a large data space and to magnify that area. Information within the user's area of interest will be shown in higher resolution and reveal more detail than the surrounding area. With this visualisation technique we can show large amounts of information within the restricted real-estate of a typical computer monitor and still be able to show the detail in the area that the user is interested in.

This thesis is divided into six chapters. In the second chapter, previous research of interest to the current research activity and commercial products for eye tracking will be reviewed. A number of techniques have been commercialised for eye tracking. These techniques can be categorised into four major groups and details of each of these groups will be presented. The latest research in this area introduces a number of new techniques based on advances in modern computer vision technology that offers more user friendly eye tracking systems. The early systems normally require the user to wear special equipment such as a helmet or a head clamp which is uncomfortable and for which the setting up process can be tedious. In this research, we aim to use computer vision techniques to provide a less obtrusive and affordable system. In

order to understand the technology properly, relevant information about the human visual perception system is presented including eye anatomy. The types and characteristics of eye movements are important issues in the eye tracking research. An overview of this topic is also provided.

Chapter three is concerned with the proposed eye tracking system. A colour based object tracking method called Camshift (Intel, 2003) is modified for iris tracking. In this chapter the Camshift algorithm will be explained. Camshift was originally intended for object, face or hand tracking applications. In particular, objects with well defined colours such as the human skin are appropriate for this algorithm. Modifications have been made to enable it to track a much smaller and highly dynamic object such as the human iris. The technique and modifications will be explained in the chapter. Since Camshift is a vision based technique, it is less intrusive and produces a more user friendly tracking system because there is no need for the user to wear any special equipment. Apart from a computer the only equipment required to use the system is a standard webcam. The algorithm has also been modified to determine if the sclera (the white of the eye) can be tracked so that the position of the iris can be measured relative to the sclera or position of the eye. This would enable eye rotation to be separated out from head movement. Other features of the system such as eye detection using template matching, dynamic histogram table update and camera setting are also discussed in the chapter.

In chapter four, calibration and iris to screen coordinate mapping are discussed. To use an eye tracking system for a HCI system requires a function to map gaze direction to the screen location the user is observing. To identify the direction the subject is looking, normally requires not only gaze direction but also head position. Since we do not use a head clamp to position the user's head while using the system, we need a calibration process. For our eye tracking system the calibration involves finding locations of the eye in the camera image that are associated with known locations on the screen. These associated locations will be used to determine where the subject is looking when doing the video to screen mapping process. In this research, experiments have been carried out with two calibration techniques. The first is a four-point calibration process, and the second is a one-point calibration process. In the

chapter each of these techniques will be explained followed by a performance comparison.

HCI is one of the applications of eye tracking. In the field of HCI itself there are a number of possible applications that can be integrated with eye tracking technology. In chapter 5, we investigate the possibility of using eye tracking together with a non-linear visualisation technique to navigate large data spaces. Non-linear visualisation techniques such as modeling a fisheye have been used to overcome the limitation of representing large amounts of data in a small display space such as a computer screen. With the non-linear technique the part of the information being looked at directly will be shown in high detail while the rest of it will be compressed but still be in the display area. Another benefit of using a non-linear visualisation technique is it is possible to deliver both detail and context at the same time. The user will get an overall impression of the data whilst observing the part of interest (the middle of the screen) in great detail. The eye tracking system will be used to indicate the region of interest in a data space and to magnify that area, instead of using a traditional pointer device. Information within the user's interest area will be shown in higher resolution and reveal more detail than the non-interesting areas. The human eye can view a picture in high resolution within 1-2 degrees of arc of focus and the resolution drops sharply outside this angle. If we have a fast and accurate enough system (able to process at 25 frames per second), the user will not be able to see that the visualisation is really being distorted. This means wherever the user is looking will be quickly displayed at a high resolution. This could offer a very beneficial information visualisation system. The chapter will review non-linear visualisation techniques and present methods used with our eye tracking.

1.1 Program Features and Implementation Detail

This program was developed and ran on the Microsoft Windows operating system. It has been developed using Microsoft Visual C++ and the MFC library is used for user interface programming. It is a standalone application with both the eye tracking and non-linear visualisation systems in the one application. Some features of the program apart from iris tracking and non-linear visualisation include:

- Display of video frames from the camera in real time.

- Playing of media files with support for a number of file formats such as avi, mpeg and mov.
- Capture still images from video.
- Open and save in a number of image file formats such as jpg, tiff, bmp, etc.
- Perform some still image processing such as edge detection, template matching, etc.

Figure 1.1 shows the screen shot of the program. The left panel of the program is the part that represents video processing which includes displaying video, playing media files, displaying back-projection video, and iris tracking processing. The right section of the program is for still image processing. For the display of non-linear visualisation, another window will be launched covering the whole screen area (see Figure 1.2).

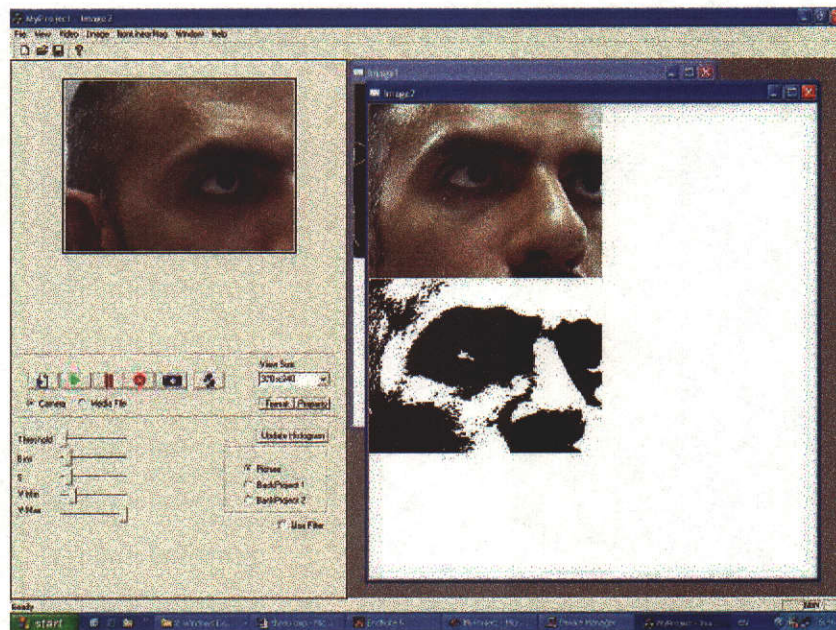


Figure 1.1 Screen shot of the program

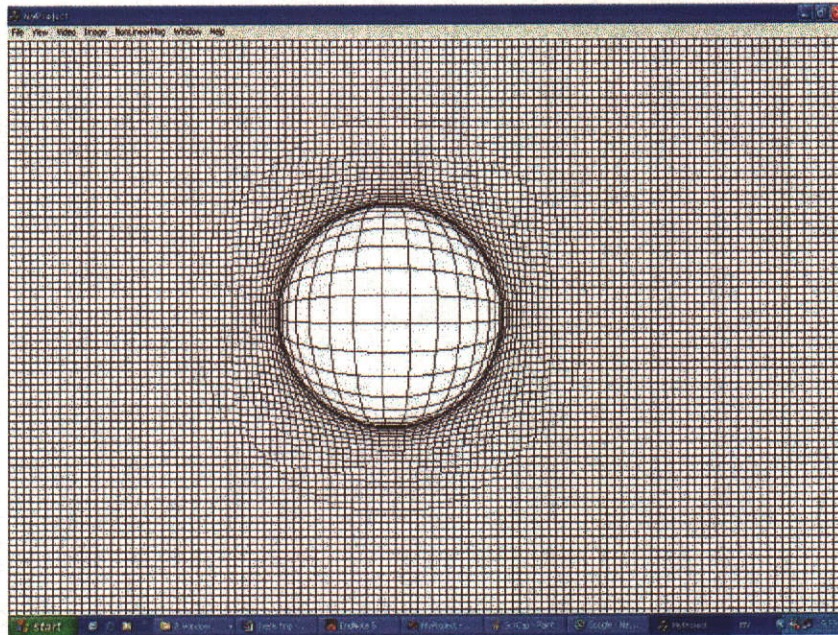


Figure 1.2 Screen shot of the Non-linear visualisation window.

1.2 Contributions from this Research in Addition to the Thesis

Intel's OpenCV library is used for most of the image processing tasks. The original source code for the CamShift algorithm is also taken from the library. Apart from the computer vision API, the OpenCV library also has a library to connect with capture devices called CVCam. We use this library to capture the video stream from webcam. Although the library provides a very easy to use API to control the camera, it lacks a function to adjust video resolution and frame rate. The ability to change video resolution is very important to our research and there is a lot of demand from students and researchers who used OpenCV for this function. Therefore we have implemented this function and submitted our version of CVCam back into the public domain. Our version of CVCam is the version that is currently used in OpenCV. Apart from adding a function for setting video resolution we also fixed a number of bugs in the library such in stereo vision, which was not working previously, that allows the library to synchronise videos stream from two cameras. Moreover we have also fixed bugs in functions related to playing media files such as mpeg and avi video files allowing CVCam to play videos from media files as well from the camera.

Part of this research was presented in the tenth annual IEEE Conference on Mechatronics and Machine Vision in Practice which was held on December 9-11 2003 in Perth, Western Australia (Dubey, West et al., 2003). The copy of the paper can be found in the Appendix A and the conference slides in Appendix B.

Chapter 2 Background

2.1 Introduction

In this chapter, we provide background on eye tracking technologies and review its related topics. There are a number of techniques used in commercial eye tracking products. Early eye tracking technology has its root in the area of psychology and cognitive science where the research was usually called eye movement research. Recently scientists saw the potential of the technology in the area of Human-Computer Interaction (HCI). The new wave of research introduced a number of new techniques based on advances in modern computer vision technology. The trend aims to offer a more user friendly systems, since earlier ones normally required the user to wear special equipment such as a helmet or a head clamp. In our research, we aim to use computer vision techniques to provide a less obtrusive and affordable system.

Eye tracking technology has been applied to many different kinds of application. Although, initially psychology was the most active group, currently, a diverse area of applications have been benefiting from the technology. Applications of eye tracking will be categorised and their examples reviewed.

In order to understand the technology properly, a review of eye related information is included in this chapter. The human visual system and eye anatomy is also briefly reviewed. The type of eye movement is another important issue in eye tracking research and an overview this topic is provided.

Current techniques in current commercial eye tracking system can be divided into four groups: Electro-Oculography (EOG), Scleral Contact Lens/Search Coil, Infrared Oculography (IROG) and Video-Oculography (VOG) or Video based technique. The VOG technique is the most widely used. In section 2.5, current research activity in video-oculography or video-based eye tracking technology is explored in more detail. That section is further divided into three sub-sections describing three sub-technologies: face detection, eye detection and eye-gaze estimation.

Since part of this research is to investigate the possibility of using eye tracking together with a non-linear visualisation technique to navigate a large data space, later in this chapter (section 2.6) various information visualisation techniques are reviewed.

2.2 Human Visual Perception

The human vision system involves the interaction of the eyes and the brain through a network of neurons, receptors, and other specialised cells. When we look at something the light receptors in the eyes are stimulated and electrical signals containing the vision information is transmitted to the brain through the optic nerves.

The basic steps of human vision are as follows:

- First, the cornea receives light and focuses it on the lens, which adjusts for distance. Along the way from the cornea to the lens, the light passes through an aperture called the pupil. This aperture narrows and widens in response to the brightness or dimness of the surrounding light by the action of the iris (the coloured part of the eye).
- The light then passes through a transparent gel called the vitreous humor. The object is viewed on the retina at the back of the eyeball as an inverted image.
- The retina receives the light, converts it into electrical signals, and passes it along through the optic nerve as stimulus to the brain.

The retina is located at the inner most part of the eye. It's where light from an object is focused. All over the surface of the retina there are two types of photoreceptor cells, rods and cones. The cones are located primarily in the central portion of the retina, called the fovea – the place that resolves the fine detail of the image. The cones are highly sensitive to colour, in contrast to rods that are sensitive to illumination. Rods serve to give an overall picture of the field of view. Figure 2.1 shows the density of the different types of sensors in the eye showing variation in density for the rods and cones.

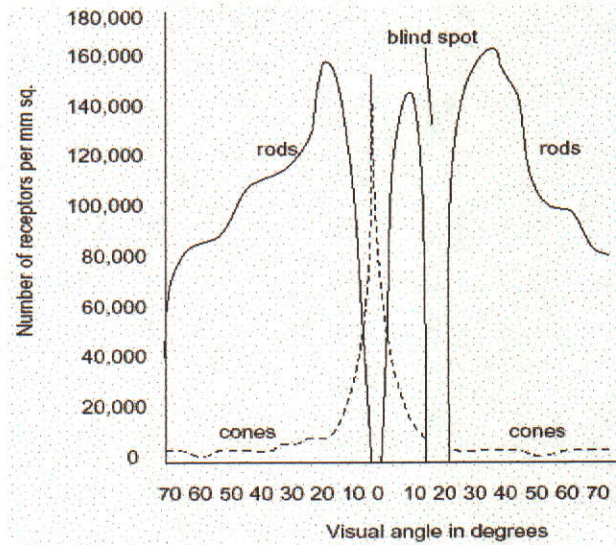


Figure 2.1 Rods and cones cell in retina (Duchowski and Vertegaal, 2000)

2.2.1 Eye Anatomy

The eye is a delicate part of the human body. In order to perceive proper vision, the eyes have to perform various complex tasks before it transfers the visual signal to the brain. These tasks involve different parts of our eye. In this section, we will give a brief review of the human eye.

- Cornea: the transparent area that transmits light into the eye and provides about two-thirds of the eye's focusing power. It is a dome like tissue that covers the front of the eye. The majority (70%) of the bending (refracting) of light rays is accomplished by the cornea.

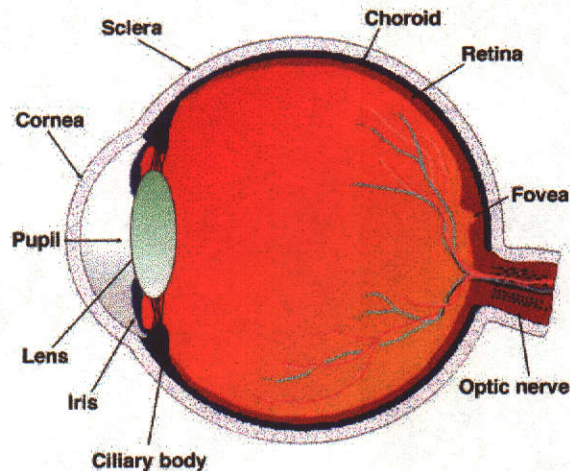


Figure 2.2 Eye anatomy (WebVision, 2003)

- Iris: the coloured part of the eye. The primary function of the iris is to control the light through the size of the pupil opening. The pigmented melanin that gives the iris its colour, aids in absorption of strong or bright light.
- Pupil: the dark centre located in the middle of the iris. It functions like the diaphragm in a camera, controlling the amount of light that enters the eye. When in a bright environment, the pupil becomes smaller to allow less light into the eye. When in a dark environment, the pupil expands to allow more light into the eye.
- Lens: the transparent structure located behind the pupil that serves to focus light passing through the cornea onto the retina. It helps to fine tune vision, and it is able to change shape to allow the eye to focus on near objects.
- Optic Nerve: the carrier of the electrical signals produced in the retina to the brain, where they are then interpreted as images. It has about 1.2 million nerve fibers. The point at which the optic nerve enters the eye is known as the "blind spot" or optic disc. This area contains no photoreceptive cells, and therefore cannot respond to light.

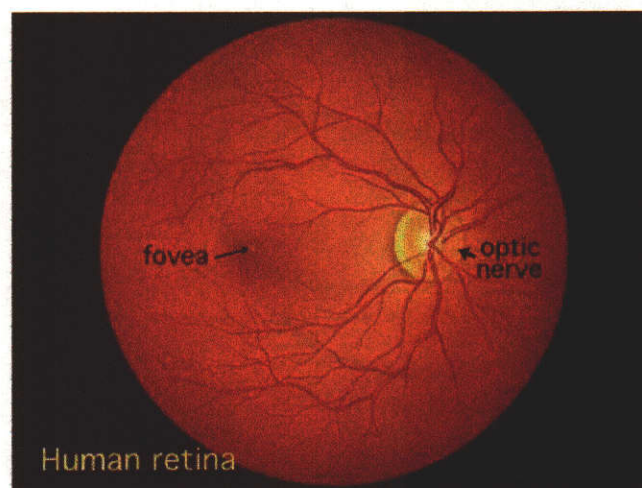


Figure 2.3 Retina, optic nerve and fovea (WebVision, 2003)

- Retina: the fine nerve tissue which lines the inside wall of the eye and acts like the film in a camera. Composed of photosensitive cells (rods and cones), the

retina senses light and creates impulses that are sent through the optic nerve to the brain.

- Fovea: The centre of the retina that contains high concentrations of photosensitive cells (cones). Nerve cells are denser in this area, so images that are focused on the fovea can be seen in greater detail.
- Sclera: the outer white area of the eye.

2.2.2 Types of Eye Movements

In our everyday lives we regularly make various types of eye movements, usually without being aware of them. Such movements occur as a result of the activity of extra-ocular muscles that support each eye. There are six extra-ocular muscles for each eye, four straight or rectus muscles; and two slanted or oblique, ones. Normally, the extra-ocular muscles work together with each other and with the muscles of the other eye to allow the gaze to be focused in any direction and to ensure that both eyes are sending the same image to the brain.

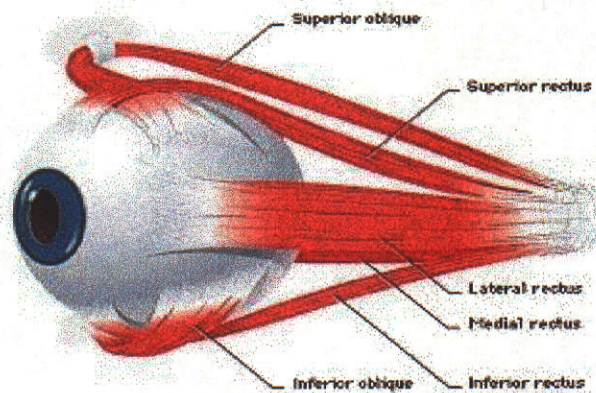


Figure 2.4 The eye's extraocular muscles (Bores, 2002)

Eye movement can be divided into 7 types (Derby, 2002):

- **Saccade:** rapid eye movement or jump that is usually both eyes moving together in the same direction. The purpose is to bring images of particular areas of the visual world to fall onto the fovea.

- **Miniature:** constant and very small movements which are all generally less than 1° in size. The purpose of such small movements is to constantly shift the image minutely over the fovea so that the fovea is constantly being stimulated, otherwise the image of the object would fade (because the eye adapts to constant input and only responds to changes).
- **Pursuit:** smoothly track slow moving objects in the visual field. The purpose is to stabilise moving objects on the retina, enabling us to perceive the object in detail.
- **Smooth:** similar to pursuit movements but can be made in the absence of a moving stimulus.
- **Compensatory:** movements that are related to pursuit movements in order to compensate for movement of the head or body so as partially to stabilise an object on the retina.
- **Vergence:** movements where both eyes move in opposite horizontal directions to permit the acquisition of a near or far object.
- **Nystagmus:** a regular form of eye movement, comprising two alternating components – a slow and fast phase.

2.3 Applications of Eye Tracking

There are wide varieties of eye tracking applications. Eye tracking applications can broadly be described by two categories, termed here as diagnostic or interactive (Duchowski and Vertegaal, 2000). In its diagnostic role, the eye tracker provides objective and quantitative evidence of the user's visual and attentional patterns over a given stimulus. An interactive system is expected to respond to, or interact with, the user. Interactive applications are therefore usually expected to respond to the user's gaze in some gaze-contingent manner. Here is a review of applications of eye tracking from both categories.

2.3.1 Psychology, Psychophysics, and Neuroscience

Psychology, psychiatry, and cognitive science are the most active research areas that involve eye tracking. The main objectives of these experiments are to obtain a detailed understanding about the cognitive process of visual search (Yang, Dempere-Marco et al., 2002) and examine the developmental differences of oculomotor

functioning at different stages of our lives. Particular interest is being paid to cognitive processes in infants and animals (Kording, Kayser et al., 2001), so as to improve our understanding of visual recognition under different stimuli. It has been found that eye-tracking techniques are particularly useful in diagnosing reading disabilities and in assessing schizophrenia, and Alzheimer's.

2.3.2 Ergonomics and Human Factors

There have been a number of investigations into using eye trackers in usability studies. Indeed, it is believed that eye movements can significantly enhance the observation of users' strategies while using computer interfaces. A recent study carried out in Stanford University called The Stanford Poynter Project (Poynter, 2000) has started research into the area of 'What the Eye Sees' - how viewers read web sites. The research project focused on a group of subjects who have been reading news web sites over a four-year period minimum. Their study looked at eyeball movement over the pages used and how much of the content was absorbed.

2.3.3 Marketing/Advertising

Eye tracking has been used to analyse human memorisation of brands from advertisements (Wedel and Pieters, 2000). The process by which eye fixations occur for print advertisements and to retained memory for the advertised brands is modeled using a hierarchical Bayesian model. It is useful for assessing advertisement strategies. In particular, they have been considered to assess how people react to 'explicit' advertisements (i.e. depicts the target product together with a related headline in a semantically straightforward way) and 'implicit' advertisements (i.e. the images and text depicted are not directly related to the product).

2.3.4 Digital Displays

There are a number of techniques available to display 3D images, that is, by using polarised glasses, shutter glasses and Virtual Reality headsets, all of which enable a viewer to receive the correct left and right eye images. The general inconvenience of requiring special glasses, or headwear, to view the images for extended periods has led to the development of systems that can be viewed by the naked eye. In order to ensure that the viewer's left eye always sees the left image, and the right eye the right image, regardless of the exact position of the viewer's head and eyes, the system uses

an eye-tracking technique (Harman, 1996). Advanced stereoscopic displays should ensure realistic and comfortable 3-D views.

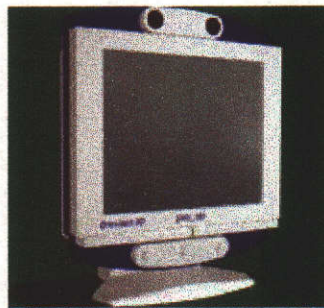


Figure 2.5 The Dresden3D Display D4D with two cameras on the top for tracking eye and head (SeeReal Technologies, 2003)

In typical videoconferencing systems when participants look at each other, they stare into their displays rather than into the camera. Unless people look directly into the camera, people will never perceive them as making eye contact to each other, no matter where they're situated in relation to the display. This loss of gaze awareness has a profound impact on communication, and may in fact be a contributing factor in the failure of videoconferencing to meet with its long-anticipated success. Including gaze tracking into a videoconference system (Gemmell, Toyama et al., 2000) helps the remote communication between participants to occur more naturally. The system analyses the video stream to determine eye contour, head pose and gaze direction then sends this information across the network with the video stream. At the receiving end, the system uses the video frame and vision information to render the head and the corrected gaze in visual 3D space.

2.3.5 Human-Computer Interaction (HCI) and Collaborative Systems

One of the basic eye-based interactive systems uses eye gaze in place of a mouse and keyboard. For assisting disabled people, the gaze may be used to select a given action from a computer screen, enabling them to interact and control their environment. The iEye Project (Interacting with Eyes: Gaze Assisted Access to Information in Multiple Languages) (IST, 2002) developed new forms of hands-free interaction for multimedia applications. In particular the project concentrates on the potentials of

eye-based interaction, where the point of gaze is used as input, either by itself or combined with other forms of input. Beside interactive systems, eye tracking can be utilised to aid multiparty communication in collaborative systems such as GAZE Groupware (Vertegaal, 1999; Vertegaal, 2002).

Another interesting application of eye tracking is the so-called interest and emotion sensitive media that enables audience-directed script navigation by considering the users' interests and emotional states. In the Context Aware Computing Group at the MIT Media Lab, the Eye-aRe system was developed to detect and communicate the information conveyed in eye movement. When the person's eyes are fixated the system infers that they are paying attention to something in their environment and then tries to facilitate an exchange of information in either direction on the user's behalf. Eye-aRe notices patterns of eye motion like gazing (that indicates attention) and blink rate (that can indicate stress level). IBM's BlueEyes research project is chartered to explore and define attentive environments – that is, environments that are user and context aware. The BlueEyes project is looking at ways of extracting information about users from facial expressions.

2.4 Current Commercial Eye Tracking Techniques

There are four broad categories of eye tracking methodologies in use in commercial products: electro-oculography (EOG), scleral contact lens/search coil, Infrared Oculography (IROG) and video-oculography (VOG) or video-based.

2.4.1 Electro-Oculography (EOG)

Electro-oculography, or EOG, was the most widely applied eye movement recording method some twenty years ago (and still used today), and relies on measurement of the skin's electric potential differences, from electrodes placed around the eye, which varies as the eye rotates. Electrode can be placed above, below and to the side of the eyes with potential across the eye indicating the eye position. This method can achieve an accuracy of $\pm 1.5^\circ$. A basic EOG setup consists of an instrumentation amplifier coupled to a chart recorder or to a computer.

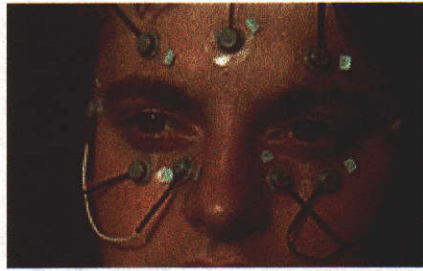


Figure 2.6 Example of the set up of the EOG technique (Duchowski and Vertegaal, 2000)

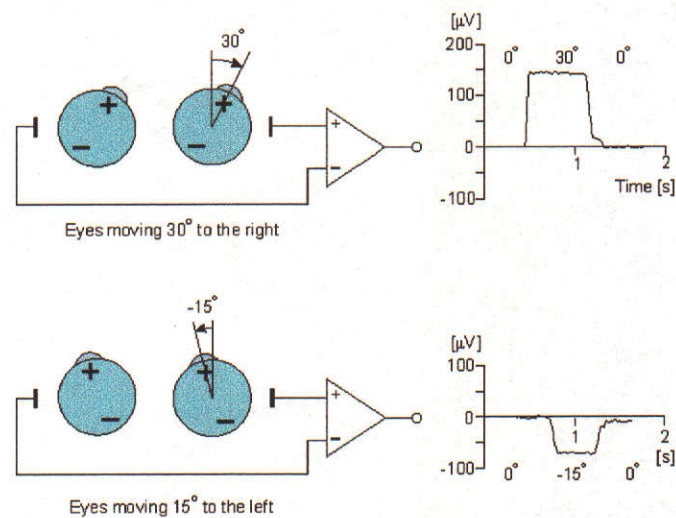


Figure 2.7 The electric potential differences when an eye moves (Malmivuo and Plonsey, 2002)

Figure 2.7 illustrates the measurement of horizontal eye movements by the placement of a pair of electrodes at the outside of the left and right eye.

Electro-oculography has both advantages and disadvantages over other methods for determining eye movement. The most important disadvantages relate to the fact that the corneo-retinal potential is not fixed but has been found to vary diurnally, and to be affected by light, fatigue, and other qualities. Consequently, there is a need for frequent calibration and recalibration. Additional difficulties arise owing to muscle artifacts and the basic non-linearity of the method. The advantages of this technique include recording with minimal interference with subject activities and minimal

discomfort. Furthermore, it is a method where recordings may be made in total darkness and/or with the eyes closed. Today the recording of the EOG is a routinely applied diagnostic method in investigating the human oculomotor system.

2.4.2 Scleral Contact Lens/Search Coil

This is one of the most precise eye movement measurement methods and also the most intrusive. It involves attaching a mechanical or optical reference object mounted on a contact lens, which is then worn directly on the eye. The principle of the method is to employ a wire coil, which is then measured moving through an electromagnetic field. Insertion of the lens requires care and practice. Wearing of the lens causes discomfort. This method also measures eye position relative to the head, and is not generally suitable for point of regard measurement unless head position is also measured (e.g., using a head tracker).

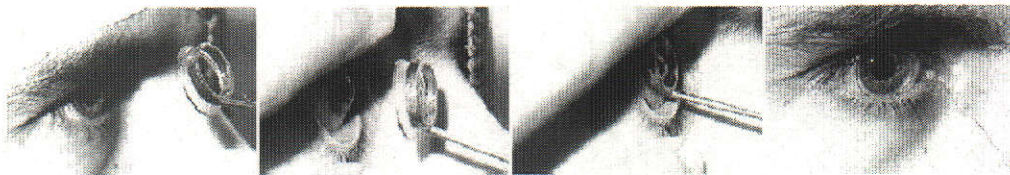


Figure 2.8 Example of scleral suction ring insertion for search coil eye movement measurement (Duchowski and Vertegaal, 2000).

2.4.3 Infrared Oculography (IROG)

With this technique, eye movement is monitored using infrared light-emitting diodes and photo-detectors mounted on goggles. Eye movements cause a different amount of light to be reflected on the sensors. One example of this technique is called limbus tracking. An infra-red beam is bounced off the eyeball and tracks the boundary between the iris and the white of the eye (the limbus). Due to the fact that the sclera is (normally) white and the iris is darker, this boundary can easily be optically detected and tracked. The limbus moves proportionally to the rotation of the eye and its position can be tracked optically with the use of photodiodes. This technique is based on the position and shape of the limbus relative to the head, so either the head must be held quite still or the apparatus must be fixed to the user's head.

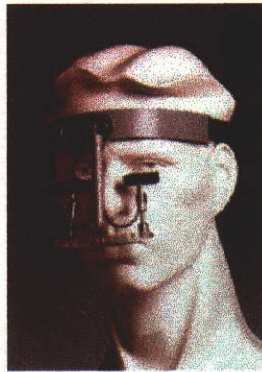


Figure 2.9 Limbus tracking from Microguide (Microguide, 2003)

This technique, however, is only effective for horizontal tracking due to the occasional covering of the top and bottom of the limbus by the eyelids. For vertical tracking the eyelid blocks the limbus and thus prevents its true position from being obtained.

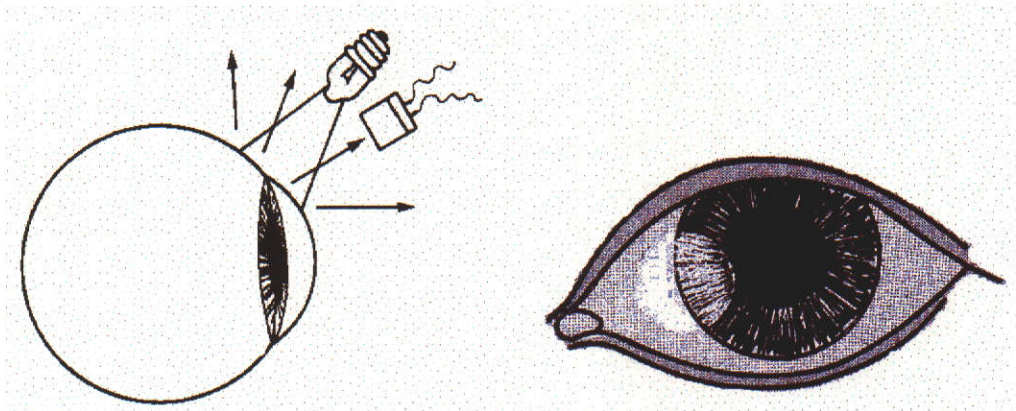


Figure 2.10 Illustrate light setting in limbus tracking technique (Carpenter, 1977)

2.4.4 Vision Based or Video Oculography Systems (VOG)

The vision based system uses a video of the eye taken from a camera typically placed near to the head. Using sophisticated computer vision algorithms it is possible to determine the direction of gaze. It can be divided into two groups, one that uses light reflection and the one that does not.

By measuring the position of the first and fourth Purkinje reflections (Dual-Purkinje Image technique), it is possible to separate translational and rotational eye movements. Although this technique offers more accurate measurement results, the fourth-Purkinje image is normally very weak due to the change of the refraction index at the rear of the lens. Therefore, the surrounding lighting must be strictly controlled in such experiments. Figure 2.12 illustrates how the four Purkinje reflections are formed as well as the relative positions of the pupil and the first Purkinje reflection when the eye fixates to the centre and the four corners of a computer screen. The Dual-Purkinje-Image technique is generally more accurate than the other techniques.

2.4.4.2 Vision Based Non-Reflection

The primary merit of vision based non-reflection techniques is that there is no need for special lighting. The technique is based solely on computer vision and image processing techniques. Although there are a number of different approaches, we can classify them into two groups based on camera placement. The first are systems which point the camera directly into the subject's eye by using a head-mounted apparatus, frame or chin rest to block head movement. The second are systems that place the camera far away from the subject's eye, or the so-called remote system, which captures the whole face of the user including the background. In order to allow the user to move their head naturally, a head tracking function has to be incorporated. The eye gaze direction is usually estimated by using both head orientation and eye. One of the most important problems for the non-reflection techniques is the complex computation required for feature extraction and pose estimation.

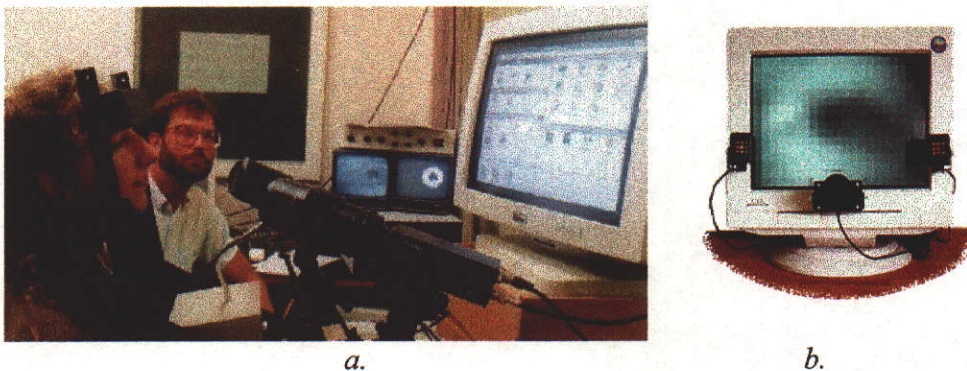


Figure 2.13 a) Eye tracking with head frame (Istance and Howarth, 1995), b) remote eye tracker (EyeTech, 2002)

2.5 Current Research in Video-Based Eye Tracking

In head-mounted systems, the camera points directly at the eyes (or a mirror reflecting the camera directly at the eyes) so the system can deal with the images of the eyes without worrying about other parts of the face. Remote tracking systems, which place the camera away from the subject, require additional processes to locate the face and eyes in the video images. The three basic steps of this kind of system are: to detect the face, to detect the eye and then to track the eye. In this section, techniques applicable to these major steps will be explored. Most of the remote systems do not restrict head movement which means they need to be able to detect the pose and direction of the head as well. However, the head pose detection technique will not be covered.

2.5.1 Face Detection

Several methods for face detection have been published in the literature. Most of the recent methods can be assigned into one of the two broad categories: (1) feature-based and (2) classification-based methods (Wei and Sethi, 1999).

The feature-based methods operate by extracting different facial features and using their geometrical relationship to locate faces. The classification-based methods, on the other hand, try to detect the presence of a face by classifying all possible sub-images of a given image into the categories of face or non-face sub-images.

The feature-based methods can be further subdivided into bottom-up and top-down methods. In the bottom-up approach local facial features, such as the eyes, nose and lips are detected first. These detected features are then combined to establish the presence of a face. On the other hand, the top-down approach consists of locating a face region first and then verifying its presence by detecting the facial features.

There are a number of proposed methods based on the bottom-up approach, such as those by Yow and Cipolla (1997), Jeng et al. (1998), Leung and Burl (1995), Nikolaidis and Pitas (2000) and Tankus (1997). Though this approach can be easily extended to multiple views, it is unable to work well under different imaging conditions because the image structure of the facial features vary too much to be robustly detected by the feature detectors.

Yow and Cipolla (1997) proposed a feature-based-bottom-up algorithm for detecting faces that divide facial features into a number of sub-features to cope with more demanding variations in the imaging conditions. The algorithm detects feature points from the image using spatial filters and groups them into face candidates using geometric and grey level constraints. A probabilistic framework is then used to reinforce probabilities and to evaluate the likelihood of the candidate as a face.

An example of the top-down feature-based method is the scheme used by Yang and Huang (1994). Their face detection system consists of three levels. The first level uses a set of rules to locate candidate face regions in the input image. The next level uses another set of rules to operate on candidate face regions to perform further screening. Finally, the valid face regions are established at the third level by performing facial feature extraction.

Another feature based system is the one designed by Dai and Nakano (1996), in which the extraction of facial regions from a complex background is done based on colour and texture information. Schiele and Waibel (1995) use skin colour to distinguish the face from the background. After adjusting the face colour intensity, the 32×32 block of facial images are fed to a Multi-Layer Perceptron (MLP) neural network. Saber and Tekalp (1996) also use colour to extract regions corresponding to skin-tone as the first step to find facial features. Another example of colour usage for face detection is the approach of Sobottka and Pitas (1998), which segment face-like regions by evaluating colour and shape information. In the first step, skin-coloured regions are detected and then face candidates are selected by verifying the shape information of the skin-coloured regions. The colour space they used is the hue-saturation-value (HSV) colour space.

Examples of classification-based methods for face detection are the systems by Rowley et al. (1998), Schiele and Waibel (1995), Kapoor and Picard (2002), Antoszczyszyn et al. (1998) and Moghaddam and Pentland (1995). The system by Rowley et al. (1998) is a neural network-based system. The most interesting aspect of this system is the use of a bootstrapping technique to collect non-face training examples for neural network learning.

Another classification-based approach is by Principal Component Analysis (PCA) (Kapoor and Picard, 2002), (Antoszczyszyn, Hannah et al., 1998), (Moghaddam and Pentland, 1995). In Moghaddam and Pentland (1995), PCA is used to reduce the dimensionality of the sub-image space. The target probability density of sub-images centred at each spatial position is computed and targets are located by maximum likelihood detection. As mentioned earlier, many of the recent methods work on colour images and often use motion information as well to perform face detection.

2.5.2 Eye Detection

In a top-down face detection by thresholding method (Stiefelhagen, Yang et al., 1996), pupils are searched for by looking for two dark regions within a certain area of the face that satisfy certain geometric constraints. They change the threshold value by using an iterative thresholding algorithm for different people and lighting conditions. Another method that extracts facial features from colour and shape information has been proposed by Sobottka and Pitas (1998). The approach is based on the observation that, in intensity images, eyebrows, eyes, nostrils, mouth and chin differ from the rest of the face because of their low brightness. For example, in the case of the eyes, this is due to the colour of the pupils and the sunken eye-sockets. Even if the eyes are closed, the darkness of the eye sockets is sufficient for eye extraction. The face is characterised by its skin colour and oval shape.

Template matching is another method applied to eye detection. In Li, Qi et al. (2001) a fuzzy template is used and the judgment of eye or non-eye is made according to the similarity between the input image and eye template. Tock and Craw (1996) also used template matching to verify the location of the eye predicted by a Kalman filter while tracking eyes.

Yuille et al. (1989) first proposed the use of a deformable template for locating human eyes. In this method, an eye model is designed and the eye position can be obtained through a recursive energy minimisation process. However, this method is feasible only if the initial position of the eye model is placed near the actual eye position. The template must be started at or below the eye; otherwise, it will locate the eyebrow instead of the eye. Moreover, the deformable template suffers from two limitations.

First, it is computationally expensive. Second, the weighting factors for energy terms have to be determined manually. Improper selection of the weighting factors will often produce unexpected results such as wrong location, missing eyes etc.

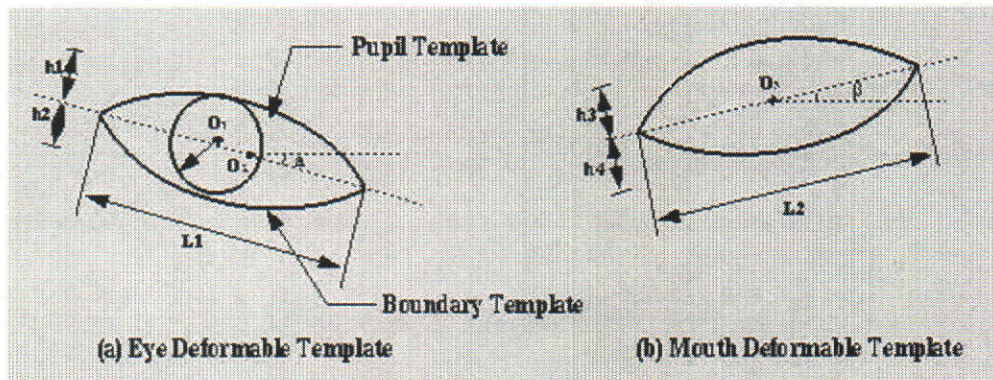


Figure 2.14 Yuille's eye and mouth model

In response to these limitations, Lam and Yan (1996) introduced the concept of eye corners to guide the recursive process and partially solved the problems. In Xie, Sudhakar et al. (1994) the corner detection algorithm is adopted to use for eye detection as well. However, the detection algorithm is based on the edge image – which has the disadvantage of not producing a good edge image when the contrast of the eye image is relatively low. In turn, the performance of the eye detection algorithm will be degraded. Following this, Lam and Yan, Feng and Yuen (1998) developed the Variance Projection Function (VPF) for locating the landmarks (corner points) of an eye. It is observed that some eye landmarks have relatively high contrast, such as the boundary line between eye and eyelid. The located landmarks are then employed to guide the eye detection process. Deng and Lai (1997) proposed a region based deformable template method for locating the eyes and extracting eye model parameters. They used their method to track upper eyelid movement when eye position and eye size are almost fixed in an image sequence.

2.5.3 Gaze Estimation

Gaze, the orientation of the eye when viewing a particular point in the world, is an important visual cue in communication. Where a person looks often indicates their interest and intent, and may provide subtle yet important feedback in social interaction. The goal of gaze tracking is to determine where a subject is looking from

the appearance of the subject's eye. The interest in gaze tracking exists due to the large number of potential applications. The most common uses of a gaze tracker are as an alternative to the mouse as an input modality, as an analysis tool for human-computer interaction studies, and as an aid for the handicapped (Jacob, 1995). Facial animation and video conferencing can also benefit from gaze estimation in generating realistic synthetic faces.

A number of techniques have been used to determine the gaze direction. In this section some of these techniques will be reviewed.

In vision based gaze tracking techniques, a light source shone into the eye is the most common method. Due to the reflection characteristic of the eye, if a light source shone into the eye is parallel to the optical axis, it will create a bright pupil (red-eye) effect, while the off axis light will create a dark pupil or glint. By using this property of the eye, the eye can be tracked easily. Another advantage of the pupil reflection is it can be used to estimate gaze direction by looking at the shape of the pupil. The pupil of the user looking forward is relatively circular. The shape changes to an ellipse when the eye moves to look in other directions.

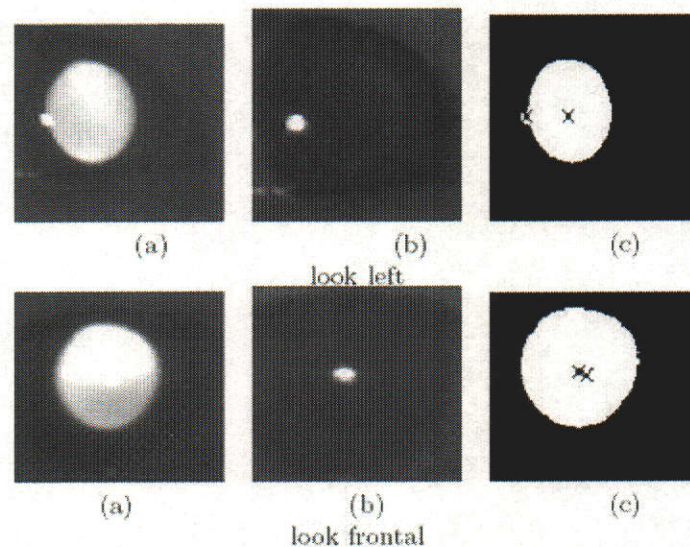


Figure 2.15 Relative spatial relationship between glint and bright pupil centre used to determine eye-gaze position. (a) bright pupil images, (b) glint images; (c) pupil/glint relationship generated by superimposing glint on to the thresholded bright pupil images (Ji and Zhu, 2002).

A number of neural network techniques have been reported to be successfully applied to eye-gaze estimation (Baluja and Pomerleau, 1994; Pomplun, Velichkovsky et al., 1994; Stiefelhagen, Yang et al., 1997; Xu, Machin et al., 1998; Betke and Kawai, 1999; Kim, Park et al., 2000; Ji and Zhu, 2002). Betke and Kawai (1999) presents a gaze estimation system that detects an eye in a face image and estimates the gaze direction by computing the position of the pupil with respect to the centre of the eye. The system is based on unsupervised self-organised learning. They use gray-scale sub-images of eyes as the building blocks of the recognition system. Xu et al. (1998) also use a neural network model based on captured greyscale eye images. The system learns the gaze direction of a user by modeling implicitly corresponding eye appearance – the relative positions of the pupil, cornea, and light reflection inside the eye socket. The 40×15 pixel sub-images of the eye region used are fed into a MLP neural network. In the paper by Baluja and Pomerleau (1994), a three layer feed forward network, trained with standard error back propagation, is used to determine the position of a user's gaze from the appearance of the user's eye. The right eye is located by searching for the specular reflection of a stationary light in the image of the user's face. The reflection's location is used to limit the search for the eye in the next frame. To determine the coordinates of the point the user is looking at, the output units from the neural network are organised with 50 output units for specifying the X coordinate, and 50 output units for the Y coordinate.

In the approach proposed by Bebis and Fujimura (2000), eye-gaze is estimated by using eigenspace. According to the paper, the context of the application is monitoring car drivers; therefore accurate gaze estimates are not really necessary. They quantise eye-gaze into five directions: looking straight, looking up, looking down, looking left, and looking right. In Nikolaidis and Pitas (2000), eye-gaze estimation is based on the estimates of the positions of the eyes and the mouth. Once the correct position of the eyes and mouth are determined, face symmetry properties can be exploited in order to determine the angle between the plane defined by these three facial features (left and right eye and a point in the middle of the mouth) and the image plane. The three facial features form an isosceles triangle. Determining the gaze direction comes from looking at the isosceles triangle as it is projected onto the image plane: the biggest of the eye-to-mouth distances reveals the direction in which the person is staring. The

success of the method depends, as is expected, on the correct detection of the eyes and mouth. This implies that the most accurate results are obtained when it has a precise estimate of the centre of the mouth and of the pupils.

Rikert and Jones (1998) present another technique based on neural networks. Their paper presents a technique for gaze estimation from a single image to provide rough estimates of where a person is looking at a monitor. The approach uses the morphable model framework to model a region around the eyes of a person. The morphable model is matched to the input image. The model is built from face prototypes which span the space of head orientations, iris locations and facial appearance. Figure 2.16 shows the head and eye prototypes. The top left image is the reference image to which all the correspondence fields refer. The next eight prototypes populate the space of different eye positions. The final eight prototypes represent various head orientations.



Figure 2.16 The head and eye prototypes (Rikert and Jones, 1998)

After matching this model to an input image of a person's eye region, the resulting model parameters are sent to a neural network which approximates the screen coordinates being observed.

An application of eye-gaze tracking in interactive stereoscopic displays has been proposed by Talmi and Liu (1999). Their system uses three cameras, with two static video cameras mounted on the left and right side of the display (head cameras). Another camera mounted on a pan-tilt unit is placed under the display for capturing

one of the viewer's eyes (eye camera). In order to measure the gaze direction precisely, the eye is zoomed in on in the image. The restricted viewing angle of the camera means the eye goes quickly out of the image when the head moves. In this case, the eye camera must track the viewer's eye continuously by using the information about the eye position delivered by the head camera. The system uses a LED light source to detect the pupil using the cornea reflection technique. In order to find both eyes in the video image sequences of the two head cameras, principle component analysis (PCA) is applied. In a training phase, the general characteristics of the human eye are learned from various people by using PCA and then stored as reference eye patterns. The current input image of one head camera is then analyzed and compared with the eye patterns in order to find the locations of the eyes. Next, the determined eye regions are searched for in the other camera image. By stereo-matching, the 3-D positions of both eyes can be determined. This information is used to control the eye camera (pan and tilt). To determine gaze direction, the centre of the pupil and of the highlight can be used to measure the vector of eye gaze. A head movement compensation algorithm is also employed.

2.6 Information Visualisation Techniques

Information visualisation strategies are more important nowadays as the amount of information rapidly increases. A number of approaches have been developed to deal with visualizing large amounts of information. The following diagram (Figure 2.17) (Leung and Apperley, 1994) shows the two main types of visualisation techniques.

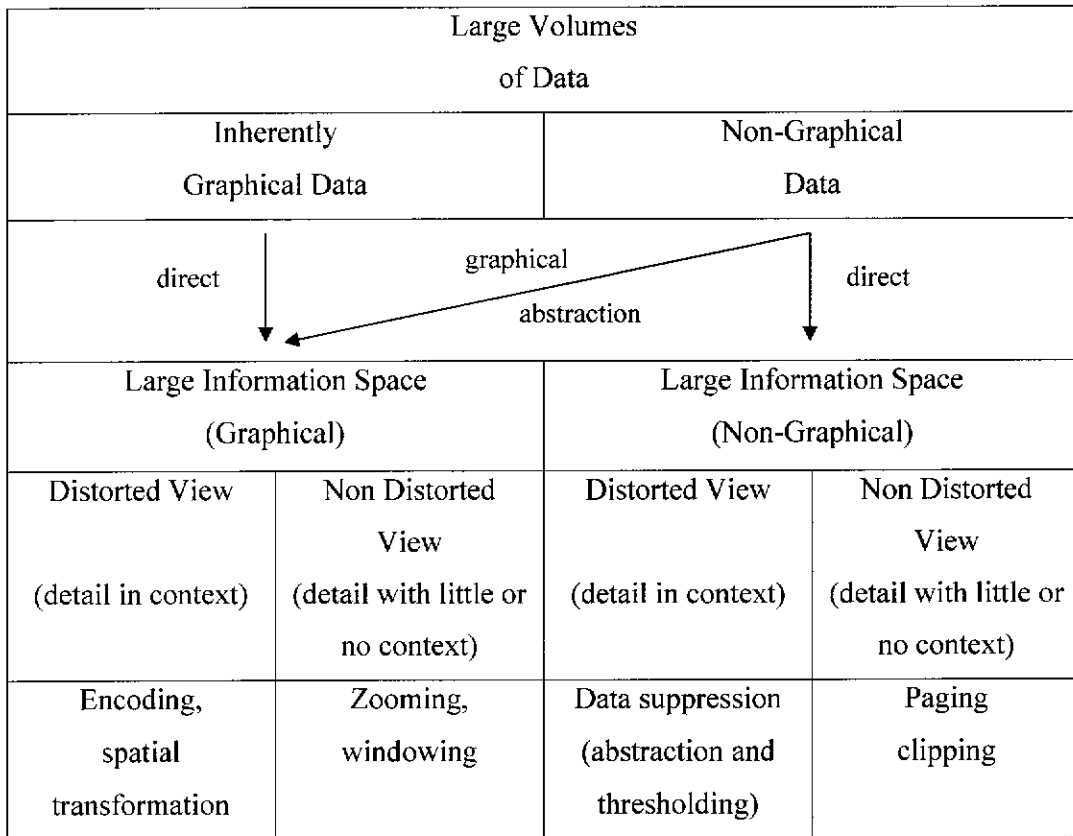


Figure 2.17 Information visualisation techniques (Leung and Apperley., 1994)

In the diagram the data is also divided into two categories, the graphical and non-graphical. The term non-distorted and distorted will be referred to here as linear magnification and non-linear magnification instead. We will omit the word magnification and use the terms linear and non-linear only.

The most common linear technique used to display non-graphical or textual data is paging or scrolling. Most graphical user interfaces use this technique. Only part of the data will be shown at a time and to view the hidden parts the user has to scroll to it. Another technique used for non-graphical data is to divide the data into a hierarchical or tree structure which allows the user to explore in more detail when they move down through a branch. One example of this kind of technique is the World Wide Web. Graphical visualisation of textual data, such as graphs and diagrams, is also very helpful in many situations.

One of the major problems of the linear visualisation technique is the difficulty in viewing an interesting section of the information in detail while keeping the overall context of the information at the same time. The context is the part of the information that helps the user to navigate through the information space. In a large information space, without perception of global context, moving from one part to another part of the information is literally like traveling in an unknown city without a map.

The most common data manipulation in the linear technique for graphical data is known as zooming which will increase or decrease the entire data space with the same degree of magnification (or compression). Scrolling is the method that is usually used to navigate to the parts of data that are outside the visible area.

As mentioned earlier, in order to visualise both detail and context at the same time a number of non-linear visualisation techniques have been introduced. The main feature of these techniques is to allow the user to examine some area in detail on a section of the screen, and at the same time, to present a global view of the space and keep an overall context.

A distorted view is created by applying a mathematical function, which is called a transformation function, to an undistorted image (Leung and Apperley, 1994). The function defines how the original image is mapped to a distorted view. A magnification function, which is the derivative of a transformation function, on the other hand provides a profile of the magnification (or demagnification) factors associated with the entire area of the undistorted image under consideration. The following equation shows the relationship between transform and magnification function:

$$M(x) = \frac{dT(x)}{dx}$$

where $M(x)$ is the magnification function and $T(x)$ is the transformation function. Figure 2.18 shows the relationship between these two functions in graphical manner.

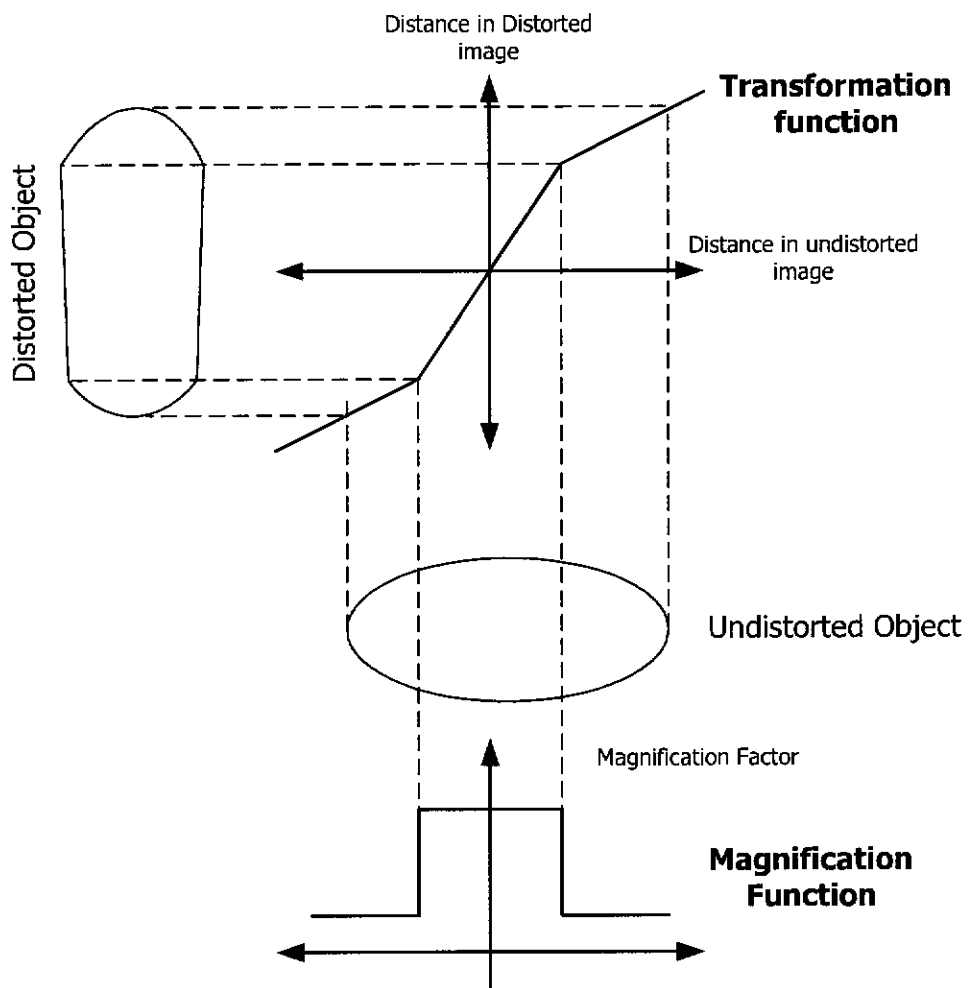


Figure 2.18 Relationship between transformation and magnification functions (Leung and Apperley, 1994)

The following subsection reviews major non-linear techniques.

2.6.1 Polyfocal Display

A technique called polyfocal projection was proposed by Kadmon and Shlomi (Leung and Apperley., 1994) for the presentation of statistical data on cartographic maps. The polyfocal projection distorts the shape of the boundaries of the display. Two sets of parameters control the shape of the magnification function; one controls the magnification at the point of focus and the other the rate of change of magnification with distance from the point of focus. Further, the troughs in the magnification function, which are inherent in polyfocal projections, serve to compensate for the high magnification factors in the area surrounding the point of focus.

The transform function (Figure 2.19 (a)) of the polyfocal display is:

$$T(x) = x + \frac{A * x}{(1 + C * x^2)}$$

where A and C are constants. For the magnification function (Figure 2.19 (b)) the equation is:

$$M(x) = 1 + \frac{A * (1 - C * x^2)}{(1 + C * x^2)^2}$$

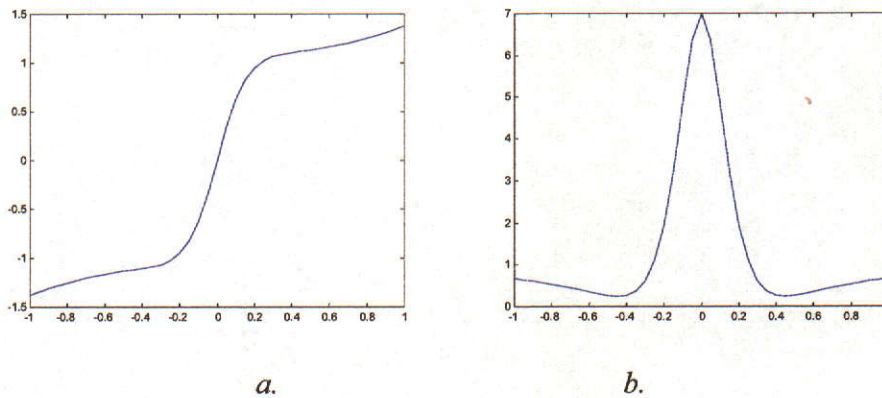


Figure 2.19 Transform function (a) and magnification function (b) of the polyfocal display with $A = 6$ and $C = 15$

2.6.2 Bifocal Display

The Bifocal Display involves a combination of a detailed view and distorted side views, where items around the detailed view are compressed uniformly in the horizontal and vertical direction (Leung and Apperley., 1994). One disadvantage of the bifocal display is discontinuity of magnification at the boundary between the detail view and the distorted view.

The transform function (Figure 2.20 (a)) of the bifocal display is:

$$T(x) = x * \left(\frac{b}{a}\right); \text{ for } x \leq a$$

$$T(x) = b + (x - a) * \left(\frac{1 - b}{1 - a}\right); \text{ for } x > a$$

where a is the boundary point of the two regions of the object image and b the boundary point of the two regions of the target image (Figure 2.21). For the magnification function (Figure 2.20 (b)) the equation is:

$$M(x) = \left(\frac{1-b}{1-a}\right); \text{ for } x > a$$

$$M(x) = \left(\frac{b}{a}\right); \text{ for } x \leq a$$

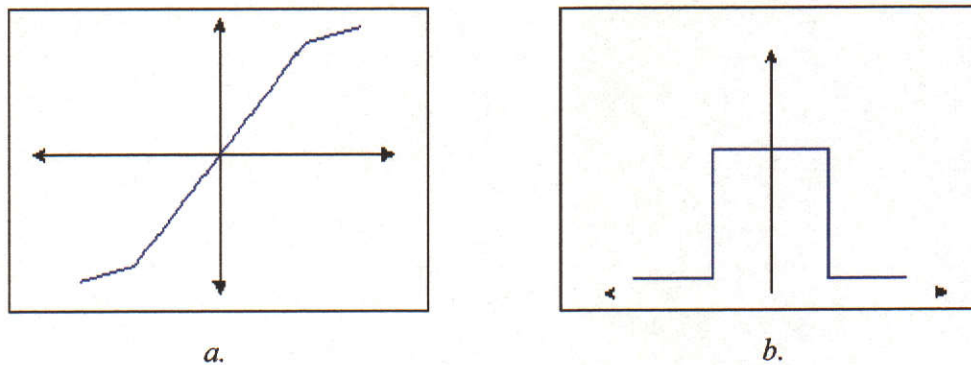


Figure 2.20 Transform function (a) and magnification function (b) of the Bifocal display

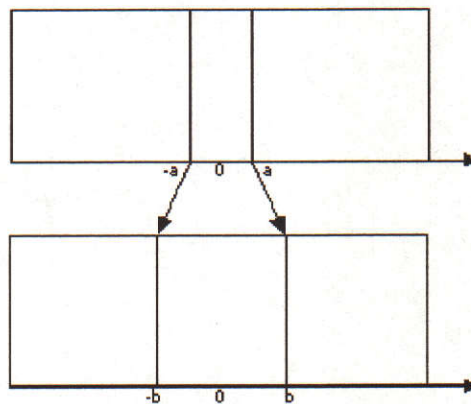


Figure 2.21 Relationship between object image and target image

2.6.3 Fisheye View

The Fisheye View technique was originally proposed by Furnas (1986) for the presentation of hierarchically structured information. The basic idea of this technique is to assign a number to each element in the hierarchical structure based on its relevance and another number based on its distance from the point of focus.

Magnifying or de-magnifying of information will be based on those two values and a threshold value. The concept can be represented using:

$$DOI_{fisheye}(x | . = y) = API(x) - D(x, y)$$

where $DOI_{fisheye}$ is the degree of interest in a point x , given that the current point of focus is y . API stands for *A Priori Importance* or degree of relevance of x and $D(x,y)$ is the distance between x and y .

2.6.4 Perspective Wall

The Perspective Wall (Robertson, Mackinlay et al., 1991) uses a 3D perspective view to represent two side distorted views of the out-of-focus regions which are de-magnified proportionally to their distance from the user. The parameters involved in defining the view of the Perspective Wall are: the length of the wall, the width of the view port, the angle of side panels, and the size of front panel. Figure 2.22 (Robertson, Mackinlay et al., 1991) illustrates the Perspective Wall and some of the parameters.

One drawback of the Perspective Wall is the 3D effect wastes space in the corner areas of the screen.

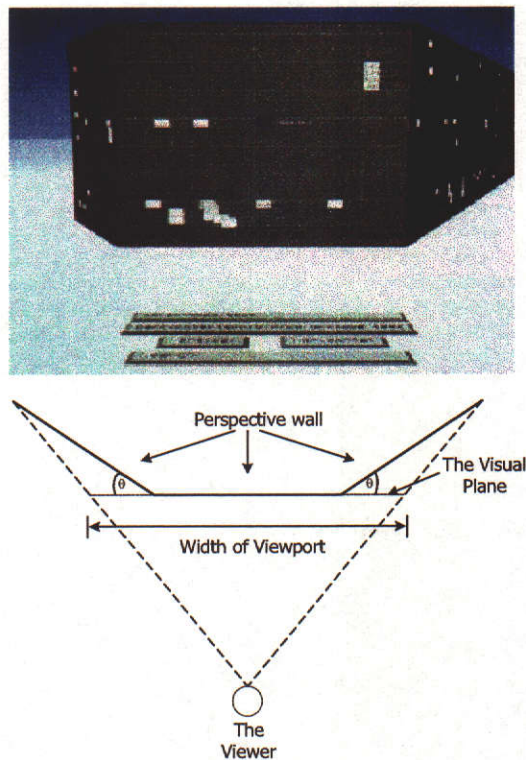


Figure 2.22 Perspective Wall (Robertson, Mackinlay et al., 1991)

2.6.5 Graphical Fisheye Views

Sarkar and Brown (1992) present an extended Furnas' fisheye version for graphical applications. Sarkar and Herman (1992; 2000) use the fisheye lens approach to viewing and browsing graphs. In the fisheye view of a graph (Figure 2.23), the vertex with the thick border is the current point of interest to the viewer or focus. The size and detail of a vertex in the fisheye view depends on the distance of the vertex from the focus, a pre-assigned importance associated with the vertex, and the values of some user-controlled parameters.

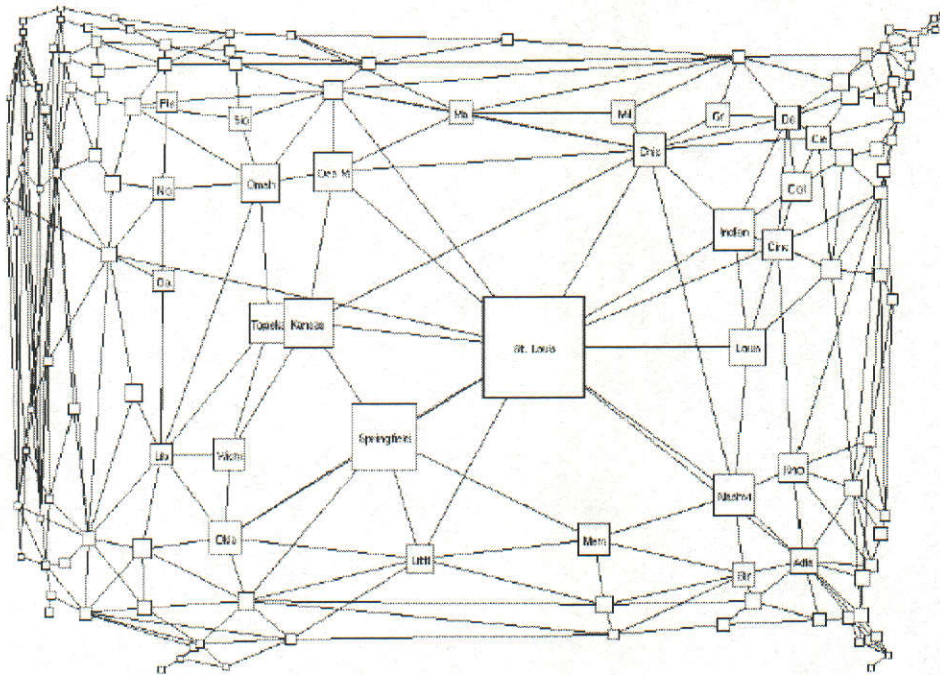


Figure 2.23 Graphical fisheye view of graph from Sarkar and Brown (Sarkar and Brown, 1992)

The transform function of the fisheye display is:

$$T(x) = \frac{(d+1) \cdot x}{(d \cdot x + 1)}$$

where d is called the distortion factor and is used to control magnification dispersion and magnitude. Higher d values will create higher magnification but smaller dispersion.

The x value in the equation has to be normalised into the domain of $[0,1]$. For the magnification function the equation is:

$$M(x) = \frac{(d+1)}{(d*x+1)^2}$$

Figure 2.24 shows the transformation and magnification functions of the graphical fisheye. The graph shows only half of the magnification function since the function is a mirror image about the focus at $x = 0$.

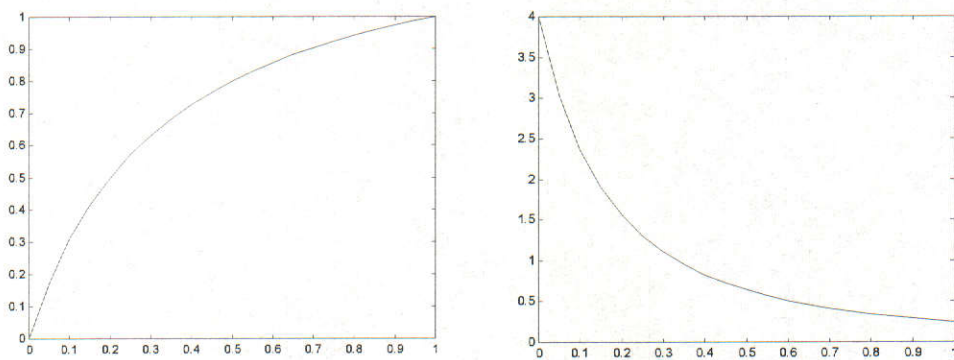


Figure 2.24 show transformation and magnification function of graphical fisheye view

2.6.6 Hyperbolic Geometry

Hyperbolic geometries (Lamping, Rao et al., 1995) provide techniques for visualizing and manipulating large hierarchical information spaces. The essence of this scheme is to lay out the hierarchy in a uniform way on a hyperbolic plane and map this plane onto a circular display region. It is a good place to lay out exponentially expanding graphs, such as trees.

2.7 Summary

In this chapter various aspect of eye tracking technologies as well as non-linear information visualisation techniques have been reviewed. Related topics such as the human visual perception system and eye anatomy have also been introduced. There are a number of applications that use eye tracking technology. These applications have been classified into five groups and some examples of each group have been given as well. Different types of techniques are used in commercial products. These techniques have both advantages and disadvantages. The major problem of the

current commercial systems is obtrusiveness. Vision based techniques that aims to provide a higher level of user friendliness are preferable. Therefore the later parts of the chapter reviewed in greater detail the vision based eye tracking techniques. It was decided, following the survey of the state of the art, to investigate a vision based method. Chapter 3 outlines the methods used for tracking the eyes using a standard web camera and computer. Lastly, since part of this research is to investigate the possibility of using eye tracking together with a non-linear visualisation technique to navigate a large data space therefore various information visualisation techniques are reviewed. Importantly the eye is a non-linear device in terms of resolution, and such non-linear visualisation techniques may be acceptable to a user.

Chapter 3 Iris Tracking

3.1 Introduction

There are a number of proposed techniques for an eye tracking system. A literature review on these techniques has been given in the previous chapter. One of the tracking techniques is colour based tracking. In this chapter a colour based object tracking technique called Camshift will be discussed. The technique has been successfully applied to iris tracking. Camshift was originally intended for object, face or hand tracking applications. Some modifications have been employed to enable it to track much smaller and more dynamic object such as the human iris.

Eye tracking based on colour is a vision based technique. This technique is less intrusive and aims to provide a more user friendly eye tracking system. There will be no need for the user to wear any special equipment. In this research we intent to investigate a technique that is efficient as well as affordable. Apart from a computer the only equipment required is a webcam. A reasonably good quality webcam costs around AUD\$100-200. Continuing developments into low cost cameras will reduce costs still further. Although in the past eye tracking systems were mainly used in medical applications, the trends show that there is increasing use of eye tracking in the area of Human Computer Interaction (HCI). The primary aim of our eye tracking system is in an interactive system for navigating a large data space through eye gaze direction control. However, the system can be applied to other applications as required.

In this research eye tracking will be referred to as iris tracking mainly because the iris is the part of the eye that is actually been tracked. The phrase *eye tracking* will normally be used to refer to a type of technology in general and iris tracking will be used when it is intended to talk about this research in particular.

In the iris tracking section, all detail about modifications to Camshaft and its implementation for use with the iris is discussed. The surface area of an eye can be divided into mainly two parts which are the colour part, or the iris, and the white part, or sclera. An experiment on sclera tracking has also been carried out to determine the possibility of using it as a reference for the position of the whole eye. This means the system has to track two objects at the same time. Other related features of the system such as eye detection using template matching, dynamic histogram table update and camera setting are also discussed in the section.

3.2 Camshift

The Continuously Adaptive Mean Shift or Camshift algorithm is an object tracking algorithm that uses the colour distribution of the object being tracked (Bradski, 1998). The colour model used in Camshift is Hue Saturation Value (HSV) instead of the more common Red Green Blue (RGB). It was stated by Bradski (1998) that RGB colour models are much more sensitive to lighting changes since saturation (which is influenced by lighting) is not separated out of that model. Camshift is derived from the Meanshift Algorithm (Fukunaga, 1990) which is a non-parametric technique for climbing density gradients to find the mode of a probability distribution. The probability distribution of colour in Camshift is represented by a histogram table. The colour sample of the object being tracked will be stored in the histogram and used in the tracking process.

Before the tracking process can be started, a histogram of the pixels making up an object (face, eye) is generated and stored. Each location in the histogram indicates the probability of that particular pixel value occurring in the image. For tracking, all the pixels in the image are replaced by their probability values in the image. As such the image is back projected via the histogram. The result is an image where each pixel whose value matches one of those in the histogram gets a greater than zero probability. Hence, the higher the probability, the brighter the pixel value. Pixels that are not represented in the histogram are given the value of zero. The result is that, without thresholding, the image has been segmented into potential object and non-object pixels. In a demonstration application supplied with the OpenCV library (Intel, 2003), the Camshift algorithm uses the hue channel from the HSV colour system to

create the colour model in a 1D histogram table. However we have found that hue on its own is insufficient for our iris tracking system. Therefore the S or saturation channel is also used. More detail on this issue will be given later in this chapter. The following section will discuss the colour histogram used in Camshift.

3.2.1 Creating the Colour Histogram Table

The colour histogram table represents the probability distribution of the colour of the object to be tracked. The Camshift algorithm uses the HSV colour space. The input video in RGB colour space has to be transformed into HSV before it can be processed. To convert RGB colour to HSV the following algorithm is used (Schaller, 2003):

```
V = max(R,G,B)
If V != 0 then
    S = (V-min(R,G,B))*255/V
Else
    S = 0
Endif
If S = 0 then
    H = 0
Else if V = R then
    H = (G - B) * 60/S
Else If V = G then
    H= 180 + (B - R) * 60/S
Else If V = B then
    H = 240 + (R - G) * 60/S
Endif
Endif
Endif
if H<0 then
    H = H+360
```

Note that intensity (V) and saturation (S) typically have values from 0 to 255 (for the common 8 bits per pixel per colour image representation), whereas hue (H) has values from 0 to 359 (in degrees).

The histogram table used for Camshift can have up to three dimensions for the H, S and V channels. For each dimension the table can be divided into a number of bins for a certain range of colour values. For example in our iris tracking system we divided both H and S channels into 5 equally spaced bins. So for the 360 possible values of H (degrees), the size of each bin will be 72. The appropriate number of bins is determined by application oriented factors. The number could vary from one application to another. In the face tracking demo program supplied with the OpenCV library the number of bins for the H channel was set to 20. It is important to have an appropriate number of bins. If there are too many bins, the distribution of values in the histogram will have a high standard deviation value (Figure 3.1). There will be only a few entries that have distinctive high probability values. Therefore separation of the object being tracked from the background will be poorer. On the other hand, if the bin number was set too low, some ranges of colour will dominate because they have extremely high probability values compare to the rest of the colours. That means some of the relevant colour is missed because their probability values are too low. The number of bins also has an effect on processing time. The smaller the number the faster the processing time will be. The appropriate number of bins is currently found mainly by experiment.

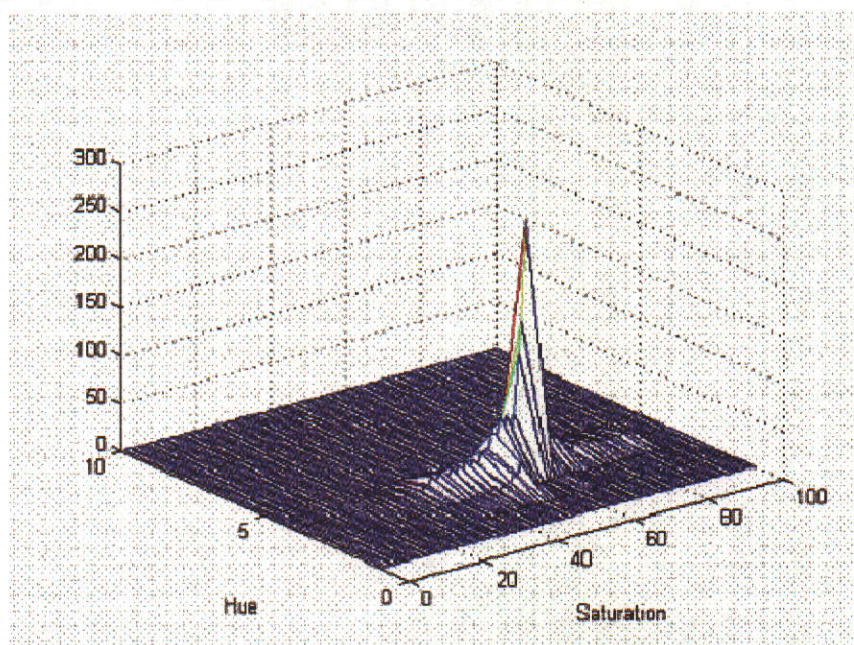


Figure 3.1 A 3D plot of the 2D histogram of 10×100 bins (H×S)

An image or sub-image that contained sample colours of the object being tracked will be supplied to generate information in the histogram table at the pre-processing stage. This can either be by manually segmenting a region of an image known to contain the object of interest or by combining numerous small regions from a number of images of the object. Each pixel from the image will be added to its associated histogram bin. In terms of probability the highest probability is 1 and the lowest is 0. But in image processing, probability in an image can be represented by a 256 greyscale pixel where 0 or black represents the lowest probability value and 255 or white represents the highest probability. For this reason, the histogram is normalised to make the maximum value in the histogram 255.

Figure 3.2 shows a typical histogram for the iris used in iris tracking.

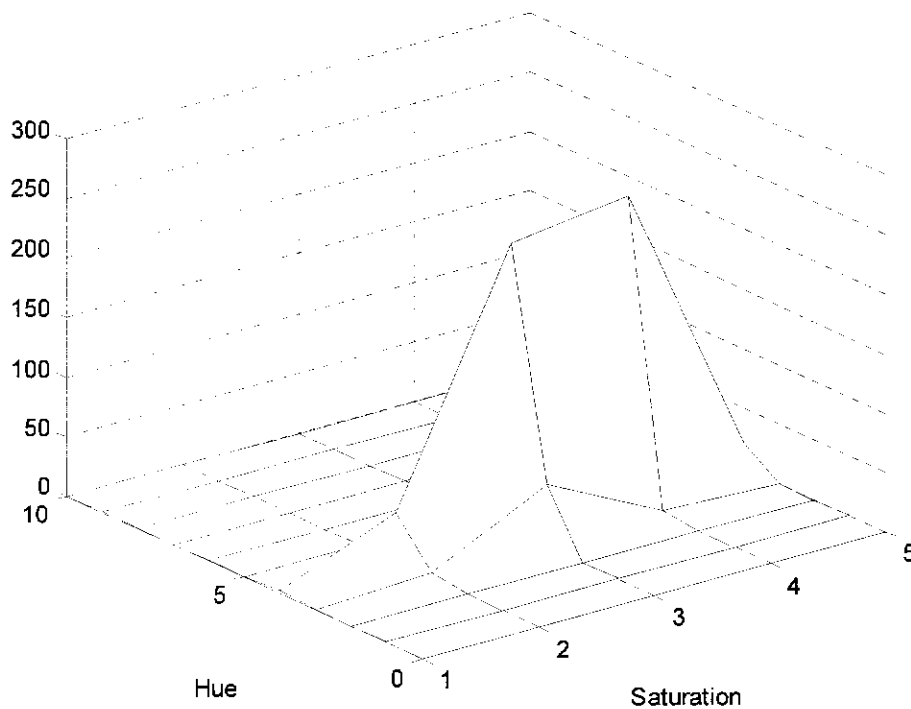


Figure 3.2 A 3D plot of the 2D histogram of 10×5 bins (H×S)

3.2.2 Moments

Moments are introduced here because they are used in the following sections for a number of stages in the iris tracker. Moments are used to describe the probability

density function of a random variable. In image processing, moments can be used for many purposes, including histogram processing, segmentation, and object description. For a grey valued image $f(x, y)$, the general equation for a moment is:

$$m_{p,q} = \iint x^p y^q f(x, y) dx dy$$

In terms of image processing, a discrete function will be used instead:

$$M_{p,q} = \sum \sum x^p y^q f(x, y)$$

A particular moment is classified by its order. The sum of indices p and q of moment $m_{p,q}$ are used to indicate the order of moment. For the zeroth order moment ($p=0, q=0$) the moment is indicated as:

$$M_{0,0} = \sum \sum f(x, y)$$

The zeroth moment describes area or the size of the object. For first order moments where $(p, q) = (1, 0)$ or $(0, 1)$ the equations are:

$$M_{1,0} = \sum \sum x f(x, y)$$

$$M_{0,1} = \sum \sum y f(x, y)$$

We can find the centre of gravity or centroid of the object using first order moments:

$$x_c = \frac{M_{1,0}}{M_{0,0}}$$

$$y_c = \frac{M_{0,1}}{M_{0,0}}$$

where x_c and y_c are the coordinates of the centroid of the object. For second order moments where $(p, q) = (2, 0)$ or $(0, 2)$ or $(1, 1)$ the equations are:

$$M_{2,0} = \sum \sum x^2 f(x, y)$$

$$M_{0,2} = \sum \sum y^2 f(x, y)$$

$$M_{1,1} = \sum \sum xy f(x, y)$$

The third and fourth moments are used to define skewness and kurtosis.

Another important type of moment called the central moment or moment about the mean is a moment that is computed with reference to the centroid of the object. All the moments explained above were not computed with reference to the centroid and are called moments about the origin. The second order moments about the centroids are also known as covariances, which are denoted as functions of the variance.

The 2D orientation of the object can be obtained from the second order moments using:

$$\theta = \frac{\arctan\left(\frac{2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right)}{\left(\frac{M_{20}}{M_{00}} - x_c^2\right) - \left(\frac{M_{02}}{M_{00}} - y_c^2\right)}\right)}{2}$$

To find the length and width, the first two Eigenvalues of the object probability distribution can be used. Let

$$a = \frac{M_{20}}{M_{00}} - x_c^2,$$

$$b = 2\left(\frac{M_{11}}{M_{00}} - x_c y_c\right),$$

$$c = \frac{M_{02}}{M_{00}} - y_c^2$$

Then the length and width of the object are:

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}} \quad \text{and} \quad w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}}$$

3.2.3 The Camshift Process

When the Camshift operation begins, the video stream will be processed frame by frame. A new video frame will, first, be used to create a back-projection image. The back-projection is a 256 gray scale image with the same size as the input video frame. Each pixel in the input video frame will be indexed onto the histogram table. The probability value of the bin then will be assigned to the same location as the input pixel on the back-projection image. The finished product of this process, the back-projection, is an image where each pixel has a probability value representing a colour of the object of interest.

The search window size can be initialised to any reasonable value. The minimum size of the window is 3×3. The window size together with its initial location affects the number of iterations required for Camshift to converge to cover the object being tracked. The difference between a good and a bad initial window setting (estimate of object position and window size) will affect only the first few video frames. After the

whole object has been covered (located) less iterations will be need to track the object.

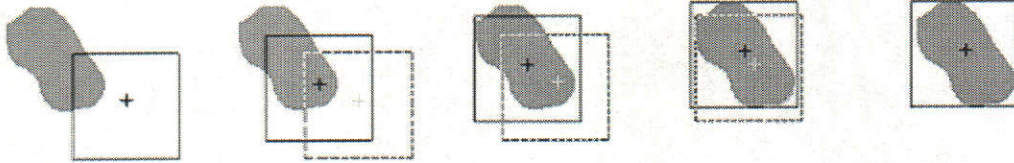


Figure 3.3 The iterations of MeanShift process to search an object (For the 2nd to the 5th diagram, the black cross indicates the mean computed from the window, and the grey cross indicates the position of the mean in the previous image).

Camshift, in essence, is an extension of the MeanShift algorithm. Camshift uses MeanShift to search for the centroid of the object. The probability of the input video pixels in the back-projection image will be used to search for the mean or centroid location of the object. The search window will first calculate the centroid within the window area. Then the window will be re-centred to the new centroid. The process repeats until each window move is less than a threshold or the number of iterations exceeds the preset loop count. See Figure 3.3 for a MeanShift algorithm demonstration. In this case the window size is kept constant and the way the mask moves to get the centroid in the middle can be seen. The centroid of the object inside the search window can be obtained using:

$$x_c = \frac{M_{10}}{M_{00}}$$

$$y_c = \frac{M_{01}}{M_{00}}$$

where M_{00} is the zeroth moment defined as:

$$M_{00} = \sum_x \sum_y I(x, y)$$

and the first moments M_{10} and M_{01} are defined as:

$$M_{10} = \sum_x \sum_y xI(x, y)$$

$$M_{01} = \sum_x \sum_y yI(x, y)$$

Whereas the MeanShift algorithm is designed for static distributions (or histograms), Camshift is designed for dynamically changing distributions i.e. video streams.

Camshift relies on the zeroth moment to determine the width and height of the new window size. For every incoming video frame Camshift starts by expanding the initial window size by some constant value. Currently this value is 20 pixels, therefore the window size will be 20 pixels bigger in both width and height while keeping the centre of the window at the same location. Then Meanshift uses the larger search window and back-projection image to compute a new centroid for the object. The information from Meanshift about the centroid and the size (area) of the object is then used to re-size the search window using moments.

The following is a summary of the steps in the Camshift algorithm (Bradski, 1998):

1. First, set the calculation region of the probability distribution to the whole image.
2. Choose the initial location of the 2D Meanshift search window.
3. Calculate the colour probability distribution in the 2D region centred at the search window location in an area slightly larger than the Meanshift window size.
4. Meanshift to convergence or for a set number of iterations. Store the zeroth moment (area or size) and mean location.
5. For the next video frame, centre the search window at the mean location stored in Step 4 and set the window size to a function of the zeroth moment found. Go to Step 3.

The 2D orientation of the object also can be found using second moment information received during the Camshift operation (see the section 3.2.2 on Moments).

Figure 3.4 shows the cross lines drawn to show the current area that are covered by Camshift. The centre of the cross is the location of the centroid of the iris region. Orientation and length of the cross lines indicate the size and orientation of the ellipse representing the iris. The cross lines can be computed using the centroid, width, height and orientation information for the object. Note that the values of the width and height of the object are not needed to be the same as for the search window. The search window is a rectangular area (with zero or no orientation) that covers the object.



Figure 3.4 The iris being tracked by Camshift

3.3 Iris Tracking by Camshift

There are some major differences between face tracking and iris tracking when using Camshift. Some of these differences create problems for iris tracking that do not occur for face tracking. First of all the iris is much smaller than the face in typical images. Moreover when the subject looks down, the eyelid obscures most of the eye area, which makes it more difficult to find and track. Next the reflectivity of the eye's surface is much higher than the human face in normal lighting conditions (i.e. the eye is a glossy object because of moisture, whereas the face is usually dry and matt). Normal light sources and such light emitting objects as computer monitors can affect the colour of the iris by superimposing the reflections off the surface of the eye onto the reflections of the actual iris that is below the surface of the eye. Another important difference is the nature of eye movement, which is much more dynamic than that of the head or face. Taking all of those issues into consideration, a number of modifications to Camshift from the original face tracking application are proposed to handle these specific requirements.

One important modification of Camshift for iris tracking is to use more than just the hue channel for the histogram table. For face tracking only hue is needed for Camshift to track effectively. Hue is excellent for face tracking because most faces in the world have similar hue, and the hue doesn't change much across a face (Bradski, 1998). By comparison, there is much variability in the hue or colour of the iris. Different ethnic groups generally have different iris colours and it also varies within groups. In this research we have concentrated mainly on irises with dark colours. However the technique used here can be applied to any iris colour.

A major problem when tracking an almost black iris is that the range of hue values from the iris is mostly within the blue colour band, and has a wide overlap with the hue values of the white eye area (sclera) as well as some of the lighter areas of the face. To distinguish the iris from those areas of the face, more information is needed, which is obtained from saturation. The saturation channel with hue is used to form a 2D histogram instead of the 1D one used for face tracking. Figure 3.5 shows the stages in the process of iris tracking.

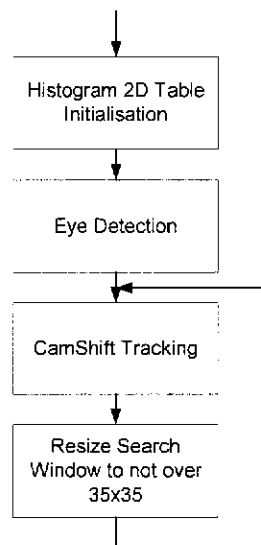


Figure 3.5 Steps of the iris tracking process

By including saturation, Camshift performance on iris tracking is improved significantly. Figure 3.6 shows the back-projection pictures of the iris using hue only, and using hue and saturation, for the face image. It is obvious in the latter image that the iris is clearly differentiated from the background. However it can be seen that some parts of the eyelash and eyebrow are still recognised as part of the iris. These errors are significant because they can cause Camshift to lose track of the iris. To deal with this problem we introduce a fixed size search window size instead of a variable sized window as used by Camshift for face tracking. Details on the window size will be given in next section.

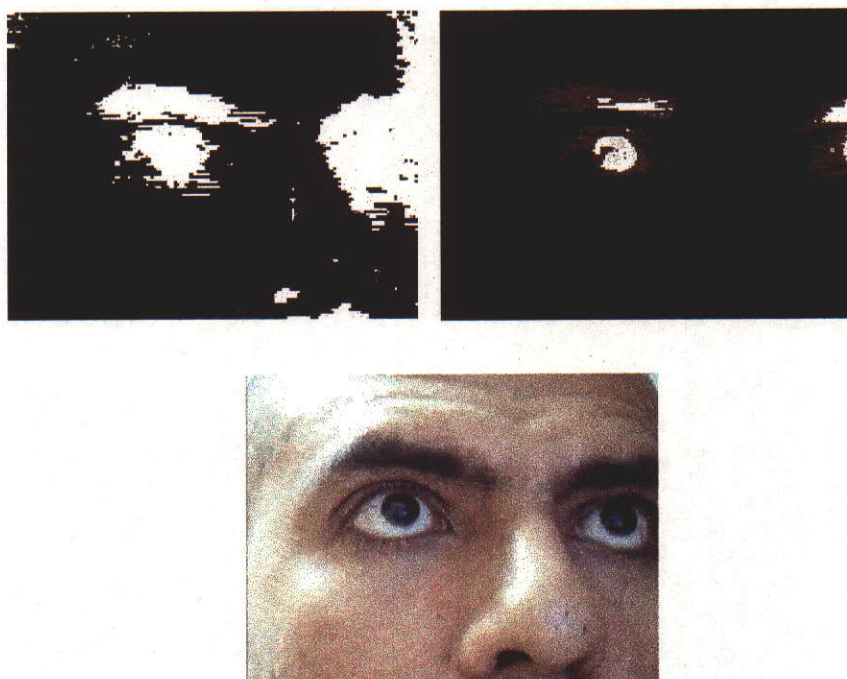


Figure 3.6 Back-projection images of hue only, hue and saturation, and original image.

In order to achieve the best performance, we also experimented with the value channel V (essentially the grey scale intensity). This was added to the other two channels to create a 3D histogram table. After trying a number of different settings and bin sizes, we have found that adding the V channel does not help to increase the performance of Camshift. Figure 3.7 shows a back-projected image when using the 3D HSV histogram. The result is somewhat similar to the back-projection image for the 2D HS histogram. In terms of processing time, adding another dimension to the histogram table reduces the frame rate from 25 fps to 23 fps.



Figure 3.7 Back-projection image when using the 3D histogram of HSV

Figure 3.8 shows the demonstration of the Camshift iris tracker operation when the subject is moving his eyes.



Figure 3.8 Demonstrating the iris tracker

3.4 Search Window Size

A major difference between Camshift and Meanshift, is that Camshift dynamically changes the size of the search window. Unlike Meanshift which is limited to still images, the changing window size helps Camshift to track an object in real time from video frames. The changing window size occurs twice during processing a video frame in the Camshift algorithm. The first time is when a new video frame is acquired. The window size will be increased to ensure that the object is still within the window given that it would move between adjacent frames captured. Currently the window is increased by 20 pixels in both width and height. This helps Camshift to keep tracking the object. After a number of iterations or once the search is finished, the search window will move to cover the object as explained for the Camshift algorithm (see section 3.2.2). The second change in size will happen when

the size (or area) of the object is calculated and the window is resized to cover the entire object. This means that the window can be of any arbitrary size. For an object that has distinct colour compared with the background this will not be a problem. However for iris tracking, increasing the window size will encompass unwanted areas close to the iris such as the eyelash and eyebrow. This could cause shifting of the centroid of the iris which will affect the accuracy of tracking (see Figure 3.9). The mapping process requires the centroid of the iris to be accurately computed to signify where the subject is looking. This is a very serious issue if the iris tracking system will be used in an HCI application. For example, although the tracker may still be able to track the iris, computing the wrong centroid coordinate will cause the wrong screen coordinate to be determined.

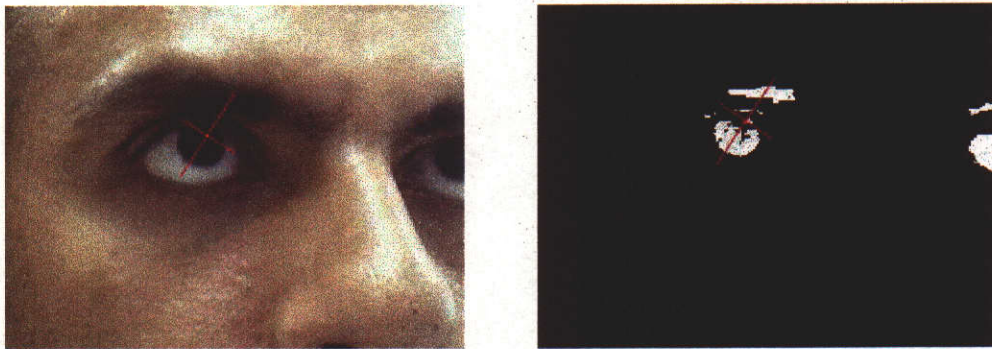


Figure 3.9 The iris tracker with unlimited window size

To solve the problem we increase the window size at the beginning of the process as normal. However when the searching process is finished, instead of allowing the window to be any size, we set a maximum window size so the window will not cover any unwanted area. Therefore we can guarantee that the window will always cover the main part of the object and not include other unwanted regions. The current maximum window size that we use is 35×35 pixels, which is a few pixels bigger than the approximate size of the iris image using typical camera geometry. Note that this is aided by the fact that the iris is a constant size and people do not generally vary the distance of their heads from the computer screen. Figure 3.10 shows the result when using our controlled window size.



Figure 3.10 The iris tracker with 35×35 window size

3.5 Bin Size

As explained in the section on Camshift (section 3.2.2) the number of bins in the histogram table is an important factor. A number of different combinations of bin size for the H and S channels have been tried and 5×5 (H×S) is the optimum choice. The iris is clearly segmented from the background. Figure 3.11 shows the back-projection image resulting from three different bins size, 5×5, 7×7 and 9×9. The picture of the iris with 7×7 and 9×9 bins comprise more low probability pixels than the picture for 5×5 bins. Higher probabilities in the back projection image will have a higher attraction to the tracker. As can be seen from Figure 3.11 the centroid of the iris is not at the centre of the iris when the bin size is increased. Although the lower probability is still good enough for Camshift to track, we need sharper segmentation to indicate the real iris's centroid location. There is also a drawback of small bin size in that unwanted areas such as eyelash and eyebrow also have their probabilities increased. Note that a small change of bin size will not have much effect on the performance.

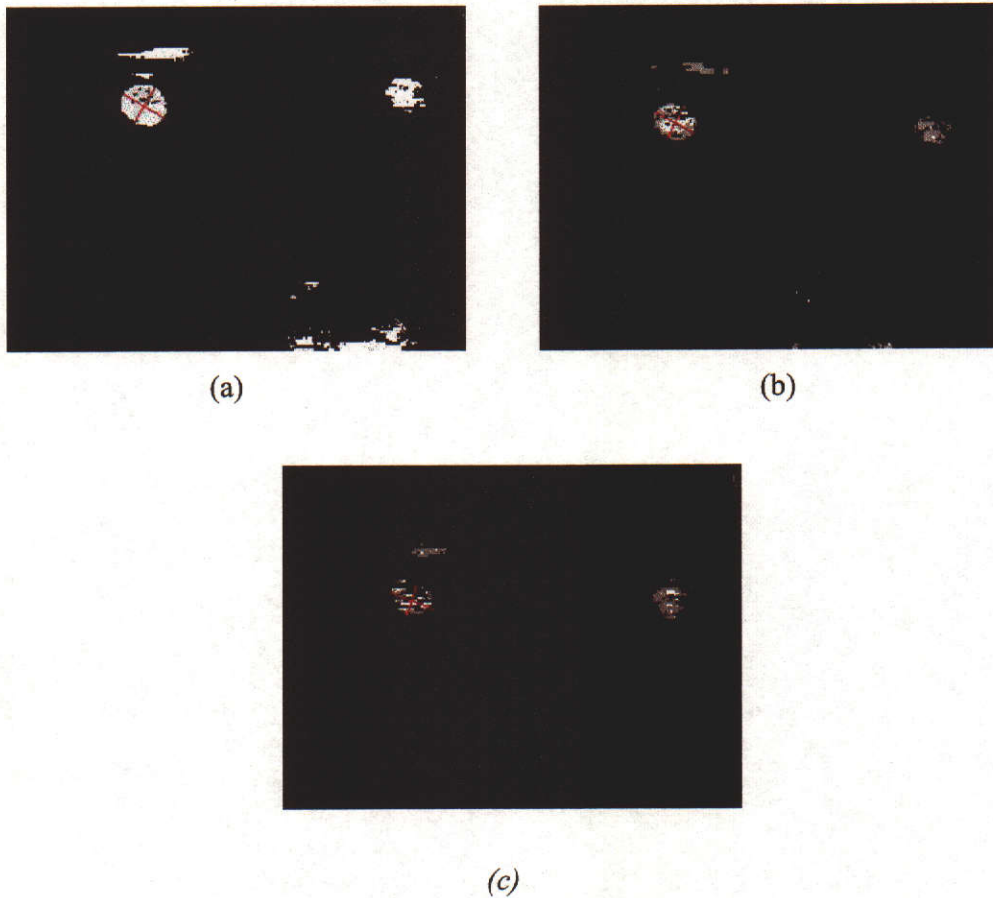


Figure 3.11 Back-projection images of three different bins size, (a) 5×5 , (b) 7×7 and (c) 9×9

3.6 Camera Location

Camera set up is another important issue for the eye tracking system. Most of the eye tracking products on the market use cameras that are equipped with head mounted devices to always keep them focused directly on to the eye. This technique helps reduce the number of problems involved in eye tracking, such as head movement, clarity of the eye picture and occlusion from the eyelid. However the head mounted technique is regarded as intrusive and not flexible. Although this research does not focus on the problem of head movement in the vision based technique, a properly set up camera is still an important issue.

Some different camera settings have been tested. The camera has been placed at different locations (e.g. on top of the monitor, at the bottom of the monitor, and on a stand at one side of the monitor). We have found that, with the camera on top of the

monitor, the subject's eyes are covered by the eyelids when he/she looks down. Almost 70-80% of the iris is occluded and causes Camshift to lose tracking. When the camera is placed at the bottom of the monitor, it is able to capture the whole iris area whether the subject looks up or down. But the problem is, by pointing the camera upwards, it is highly likely for the camera to be looking directly at a light source which can overload the camera and produce high contrast images in which the face detail and hence the iris are not easily seen. Adjusting the angle or camera parameters to solve the problem has been found to be difficult and different environments would require different adjustments. The solution for the camera location has been found by placing it at one side of the monitor on a stand (or tripod) at a height slightly lower than eye level. By placing it lower than eye level, we can be assured that the iris will not be covered by the eye lid when the subject looks down and also the angle is not too high to possibly be looking directly at a light source.

To get a picture of the iris that is big enough for the Camshift tracker, the camera was set to zoom in on the eye area. The average size of the iris image with the current setting is 30×30 pixels using an image of 320×240 pixels in size. We have found that the current size is good enough for Camshift to track effectively.

3.7 Eye Detection

An important pre-processing stage for eye tracking is to detect and find the location of the eye. There are a number of different methods proposed in the literature for eye detection (Yuille, Cohen et al., 1989; Xie, Sudhakar et al., 1994; Lam and Yan, 1996; Tock and Craw, 1996; Deng and Lai, 1997; Feng and Yuen, 1998; Li, Qi et al., 2001). The simplest way is to ask the user to guide the system using a user interface. However it is much more preferable if the system is able to perform this task automatically. But that is on condition that it would not cause too much delay to the overall processing time. We applied a simple template matching technique to search for the location of the maximum matched location in the first video frame. An iris sub-image acquired from the user is used as a template image (Figure 3.12). This process is done only once at the start of tracking. The searching method is less intrusive and does not take much time.



Figure 3.12 Iris template

The cross-correlation coefficient template matching method (Gonzalez and E.Woods, 2001) is used. This is defined as:

$$R(m,n) = \frac{\sum_x \sum_y (w(x,y) - \overline{w(x,y)})(t(x+m,y+n) - \overline{t})}{\sqrt{\sum_x \sum_y (w(x,y) - \overline{w(x,y)})^2 \sum_x \sum_y (t(x+m,y+n) - \overline{t})^2}}$$

where $R(m,n)$ is the result image, w is the area in the image that corresponds with the current location of the template, \overline{w} is the average value of $w(x,y)$, t is the template and \overline{t} is the template's average value. Note \overline{t} only has to be computed once as it is a constant value – the template doesn't change during processing.

3.8 Updating the Histogram Table

As stated previously, the histogram table is initialised by a pre-defined (or pre-acquired) iris colour model. This colour model was created off-line by selecting sample colours of the iris from a few images of the eye. Experiments show that this model represents pixels of the iris very well in most circumstances. However when the lighting conditions change drastically, a very small number of pixels from the iris get a high probability from the histogram table. It is impossible to create a static colour model that will be able to cover all the possible lighting conditions. It would be better if the system could update its histogram table when needed. To do this, the system allows the user to update the histogram table when the performance is found lower than normal.

Updating the histogram table can be done only while the iris is still being tracked. As mentioned in section 3.2.2 about Camshift, the algorithm provides information about the centroid, width, height and orientation of the object. The width and centroid value

Every pixel within the search window will be examined if it is inside the circle. If the result from the above equation is true, it will be used to create a new histogram table. There are two ways to do this: (1) update the histogram table with the new data, or (2) create a new one. To update the old table we have to remove the old values and add in the new values to the table then normalise to the range of 0 to 255. We have found that there is not much difference in using both techniques. As updating the old histogram table is more complicated than creating a brand new one (using the same procedure that we use when we first initialise the histogram), we decided to use the second method.

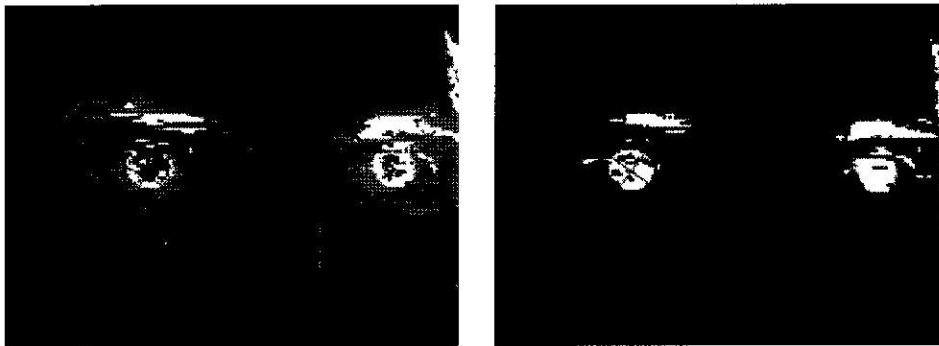


Figure 3.14 Back-projection images of before and after updating the histogram table

3.9 Tracking the Whole Eye

Since both head and eye are moveable, the measured displacement of the eye can not be guaranteed to have occurred solely because of the movement of the eye. It is possible that the movement is also caused by the head or a combination of the head and eye. A solution for dealing with the head movement is to use a head clamp to make sure that only the eye can cause the movement. However this means we have to use additional hardware resulting in an invasive method that we do not want. An alternative to the head clamp is to assume the user is keeping his/her head relatively still all the time. The best method is to determine head pose or the position of the iris relative to the rest of the eye. There are many proposed methods in the literature (Baluja and Pomerleau, 1994; Heinzmann and Zelinsky, 1998; Talmi and Liu, 1999). One of the types of method is to determine the eye direction relative to some reference points on the face, such as the corners of the eye (Xu, Machin et al., 1998).

However the corner detection procedure is very time consuming. To find a reference point by using a similar algorithm to that for tracking the iris should be less expensive than applying a totally different approach. In other words, could Camshift be used to obtain information about head position? An object that may be suitable to use as a reference is the eye itself, which is used in Camshift to determine the position of the whole eye – including both the iris and the sclera (white part of the eye). Although the shape of the sclera changes while the eye is moving, intuitively, the sum of the area of the sclera and iris should stay the same. That also means that the centroid should stay at the same location whatever the iris movement. Therefore we could use the centroid of the eye as a reference point on the face. To track the whole eye we have to be able to track the sclera together with the iris. To do that we used another colour model for the sclera to create a back-projected image of the sclera which we merged into the back-projected image of the iris by taking the XOR of the two images pixel by pixel.

The result showed that the sclera has less uniqueness than the iris. The white colour of the sclera has a lot of similarity to the colour of light reflected from the face. The reflection on the forehead skin and nose caused the tracker to occasionally not track the eye. However we can easily remove this unwanted effect by defining a region of interest that covers only the eye area. The area outside the ROI will then be ignored and hence won't cause any problem. Moreover we have also found that the back-projected picture of the sclera showed that not all the sclera area had been mapped to the picture. Most of the missing parts are at the corners of the eye (see Figure 3.16). Another unexpected result is the centroid of the whole eye seems to follow the iris when the iris moves. The centroid did not stay as still as we thought it would. Figure 3.17 shows graphs of the coordinate of the iris and the whole eye when looking at different locations on the screen. The y coordinates of the iris and sclera are recorded when the subject observes nine points on screen (see Figure 3.15). The sequence is top-down and left-to-right. At each point 10 samples are recorded (see details on this in chapter 4). To be able to use the sclera as a reference the graph has to be straight line independent from the iris position. We have found that the area of the iris has high influence over the centroid of the whole eye.

We have also experimented with the camera positioned at the centre of the screen which should allow more of the sclera to be observed because a more front on view is

obtained. Although some improvement was obtained, the result was still inaccurate. One of the reasons for the unreliability of the sclera tracker is because the sclera colour is mostly white, and hence the HS histogram has low values (lack of colour) leading to poor segmentation of the sclera. Secondly the size of the eye changes when we look up or down. When we look down the eyelid will cover the eye therefore the area of the eye will be smaller. When we look up the eye will open wider. Since the shape of the eye changes the centroid changes. Finally since the iris doesn't change its shape dramatically, is one region, and is always visible, it will always be captured better than the sclera and yield a good estimate of its position.

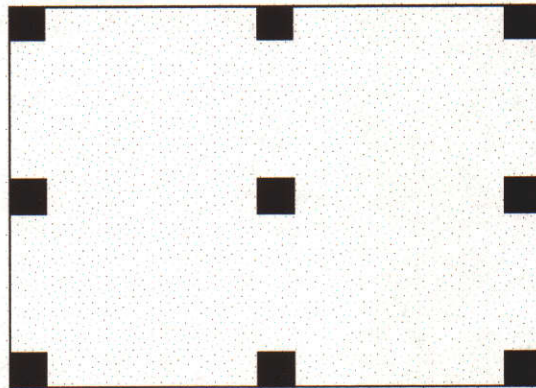


Figure 3.15 Location of the nine points on the screen used for testing



Figure 3.16 Back-projection image of the whole eye

Although the Camshift task is doubled by tracking both the iris and the whole eye together, the processing time increases only marginally. This is because much of the time is spent acquiring the image and back projecting (which has to be done for both

iris and sclera but can be done at the same time). Tracking the iris is the only different operation to tracking the whole eye.

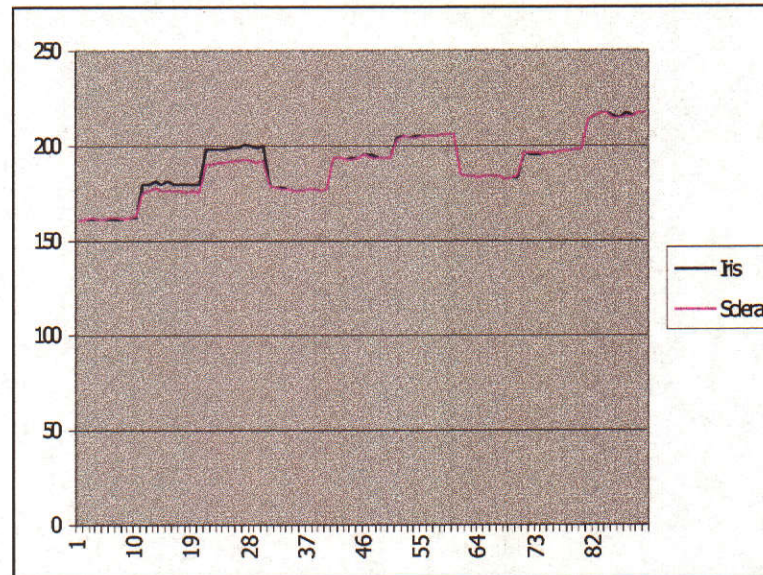


Figure 3.17 Graph of iris and the whole eye centroid coordinate when subject look at different location on screen

3.10 Summary

In this chapter we introduce the Camshift object tracker technique, to which we have made significant changes for iris tracking. Camshift is a colour based tracker designed to be used mainly for face tracking applications. A number of changes have been made to make it suitable to tracking the iris. There are a number of differences between face and iris that have to be taken into account such as its size, colour, reflective properties and speed of movement. In addition to these differences, the application that we intend to use it for is HCI, which requires high accuracy for the iris location to produce correct mapping results. A significant modification is to use a 2D colour histogram instead of a 1D colour histogram by adding saturation to the hue used in Camshift. The histogram will be created at the beginning of the process with an image of sample iris colours. The histogram table can also be updated dynamically during runtime to adjust it to suite the environment. Another modification is the way the algorithm resizes the search window. We decided to define the maximum window size to avoid the window from including unwanted areas such as eyelashes and

eyebrows. This we could do because we have a good idea of the size of the iris in the image. To detect the iris before the start of tracking we use a simple template matching method using an iris image as a template.

Chapter 4 Calibration

4.1 Introduction

Using an eye tracking system for a human computer interaction (HCI) system requires a function to find the gaze direction that will be used to determine the screen location that the user is looking at. To identify where the subject is looking normally requires not only gaze direction but also head position. The information about head pose can be found by using a 3D vision technique. The problem with the 3D head pose modeling technique is it requires complex processing and hence needs much system resources. In many HCI eye tracking solutions the issue about head pose is overcome by fixing the head position. This can be achieved by using a head clamp for which the user has to rest his/her head in a clamp that will help to keep the head position stationary while they use the system. With both head and camera position fixed, the only parameter left to deal with is eye location. One of the benefits of using a head clamp is every time the user starts using the system it can be assured that it is at the same place every time, which is not true for the method we use. With six degrees-of-freedom the initial head position will never be the same each time the subject uses the system. The solution to this problem is to enforce calibration of the eye location each time a new user wants to use the system.

Although in this research there is no special equipment used to guarantee a stationary head position, we assume that while using the system the user's head stays still. The subject has to keep his/her head still as much as possible. Although it is almost impossible for any one to keep their head perfectly still, the system should be able to handle small head movements and still give reasonably accurate results. A preliminary study addressing how much people move their head when they use computer in the normal condition reveals there will be only small amount of head movement when they concentrate on the content on screen. The head movement is more apparent when they move from content to menu, button or controls that lay on the border of the screen. Observing a few people working with computers reveals

they normally move only their eye to observe content within around one-third of the screen radius from the fixation point. For distance more than that there will be the combination of both head and eye movement. For example when people want to look at something at the opposite side of the screen they will move their head along with the eye. This quick study tells us that our assumption about restricted head movement is reasonable.

The calibration process prepares the system for the new environment or acquisition of a new set of information. The calibration process involves some adjustment to set up the system for proper operation. For our eye tracking system the calibration involves finding locations of the eye in the video stream that are associated with locations on the screen. These associated locations will be used to determine where the subject is looking when doing the video to screen mapping process. After the calibration the system can tell where on screen that user is looking, if we make assumptions about the linearity of the eye tracking process, or have some model to fit the pattern of eye coordinates while it moves across the screen.

To calibrate the system, points are displayed on the computer screen that the subject looks at. While the eye tracking software is running, it records the coordinates of the centroid of the iris for each observed point. The system will record the eye coordinates of each calibration point on the screen. To improve the accuracy of the process, a number of coordinate samples are taken. The centroid coordinates later will be used as references in the mapping process. Any eye coordinates that go outside the boundary of the screen will be regarded as if the user is looking at the boundary of the screen.

In this research, experiments have been carried out with two calibration techniques. The first is four-point calibration, and the second is one-point calibration. The four-point method uses four points on the screen to determine the mapping of camera pixel coordinates to screen coordinates. This is a process required to be performed for every session to obtain reasonable accuracy. The one-point method uses two phases. An off-line phase is performed using nine points which gives a better mapping than the four-point method. Then, for each session, one point is used to determine the right calibration. The on-point method is more convenient for a user as it is quick to

perform for each session. In the following sections each of these techniques will be explained, followed by a performance comparison later in the chapter.

4.2 Four-Point Calibration

The main purpose of the calibration is to define a boundary of the eye movement that happens when the user scans the computer screen. This boundary will define left, right, top and bottom limits of the iris centroid coordinates that are associated with the left, right, top and bottom of the screen coordinates. When the calibration process starts, the system will show a window with four-points at the top, bottom, left and right. Figure 4.1 displays a screen capture of the calibration screen. Note that the points chosen could not have been the corners of the screen because we need to know the accurate position for lines through the center of the screen.

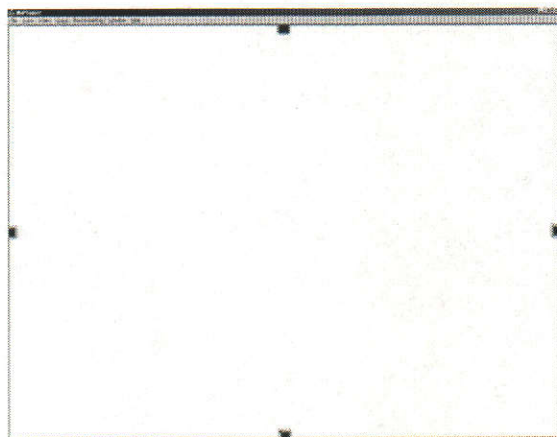


Figure 4.1 The four calibration points (red squares halfway along each side) on the screen.

When the user looks at each point, a number of iris coordinates will be recorded. The mean value of the iris coordinates will be used instead of a single coordinate because noise in the measurements can be reduced by taking the mean.

Figure 4.2 shows a picture from the camera (using the camera settings previously described in Chapter 3) of the subject when he looks at a particular calibration point. This is a normal face pose and iris image captured during the iris tracking process.

For this research only the left iris is used for tracking. Only one eye is needed and the camera is on the left side of the monitor making the left eye more visible.



Figure 4.2 A picture captured from the camera with the setting used by iris tracking.

After the recording of the calibration points is complete, the iris coordinates will be used to find the width and height of the range of the eye movement across the screen. The range of the eye coordinates can be considered as a virtual plane of the eye. Figure 4.3 shows this virtual plane of the eye and the computer screen plane.

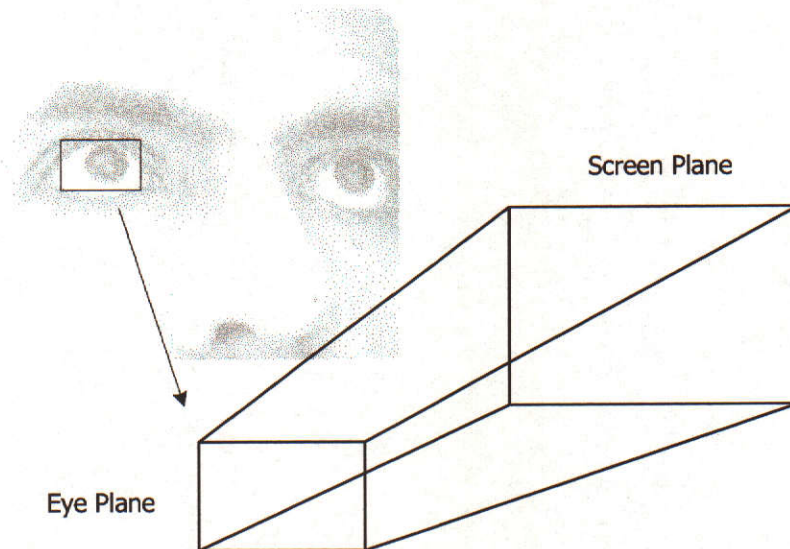


Figure 4.3 Relationship between the eye plane and the screen plane.

Figure 4.3 also represents how the eye coordinates are translated into screen coordinates. By using iris or eye plane terminology, as shown in Figure 4.3, the explanation about calibration and eye to screen coordinate mapping will be much

easier. The range of the iris coordinates that cover the screen area will be referred to as an eye plane.

We can find the width and height of the eye plane by:

$$EyePlane_width = Calibration_right_x - Calibration_left_x;$$

$$EyePlane_height = Calibration_bottom_y - Calibration_top_y;$$

where $Calibration_right_x$, $Calibration_left_x$, $Calibration_bottom_y$ and $Calibration_top_y$ are the mean values of the iris coordinates from the calibration process at right, left, bottom and top of the screen respectively.

For our current camera setting the width and height of the eye plane will be about 23 and 20 pixels respectively. It is quite obvious that the mapping from eye plane to screen is not a 1:1 mapping. With our default screen resolution, 1280×960, the ratio is around 1:55 for the x coordinate and 1:48 for the y coordinate.

In the mapping process, a centroid of the iris from each video frame will be processed to find the screen coordinate using a linear scaling process. The following formula explains the process:

$$Screen_x = \frac{Iris_x - Calibration_left_x}{EyePlane_width} \times Screen_width;$$

$$Screen_y = \frac{Iris_y - Calibration_top_y}{EyePlane_width} \times Screen_height;$$

with the results clamped to lie between 0 and the width (or height) respectively, and $Iris_x$ is the current iris x centroid coordinate, $Iris_y$ is the current iris y centroid coordinate, $Screen_x$ is the result x coordinate on screen, $Screen_y$ is the result y coordinate on screen, $Screen_width$ is the width of the screen in pixels and $Screen_height$ is the height of the screen in pixels.

The above analysis assumes that there is a linear relationship between iris position and screen location. This is not true given the position of the camera observing the eye. In fact the image plane of the camera is tilted with respect to the screen and hence linearity is only an approximation.

For the formula above, any coordinates that fall outside the screen area will be set to left, right, top or bottom of the screen depending on which side of the screen that it went outside. Given perfect calibration, mapping of the eye coordinate to points outside of the screen area is due mostly to head movement. This method of mapping is very sensitive to the head movement and noise. As stated above, the mapping is one-to-many, one pixel of the eye coordinates represents 55×48 pixels on the screen. Hence, a one pixel error, whether caused by head movement or measurement noise, could create up to 55 pixels error on the screen. Unfortunately the error from both head movement and video input are not controllable and are likely to happen all the time. As already mentioned, because we do not use a head clamp it is impossible for any human being to keep their head perfectly still. The error from the video signal is mostly dependent on the quality of the camera. A better camera is likely to produce less noise. In this research a simple webcam (Intel PC Camera model cs430) camera, with the capability of 320×240 at 30 fps, is used for the video source. The reason for choosing this type of camera is to develop an application that is as affordable as possible. Although the webcam we chose can be considered as one of good quality in its category, imperfections from the video signal are still inevitable. Therefore it is important to investigate methods to deal with this error. Until now there has not been much research on this issue - although some filtering techniques have been used to reduce the error. In this work, three successively acquired eye coordinates are averaged and used instead of a single coordinate value. The result shows some improvement of the output screen coordinates accuracy (see results in Table 4.1).

	Average error in screen pixels	Standard deviation of error
Mapping without filter	135.80	65.35
Mapping with filter	115.95	60.70

Table 4.1 Result of mapping with and without using average value of coordinate

4.2.1 Terminology and Notation

The mapping of a point (x, y) from a video frame or image point to a point (x, y) on the screen is done separately for the x and y values. The x video coordinate will be mapped to an x screen coordinate using one function while mapping of the y coordinate will use another function. Therefore all the graphs that will be shown represent functions of either y -to- y or x -to- x coordinates between screen and video. The axis of a graph, instead of being X and Y , will be labelled as S for screen and V for video. In any equation related to y coordinate mapping S_y and V_y will be used while S_x and V_x will be used for x coordinate mapping. A set of coordinates at a particular x value on the screen will be referred to as a column or a column at $x = X_n$. A set of coordinates at a particular y value on screen will be referred to as a row or a row at $y = Y_n$. The axis arrangement can be S on the horizontal and V on the vertical or vice versa. To refer to any point on the graph, $p(v,s)$ or $p(s,v)$ notation is used depending on the axis arrangement. The constants `SCREEN_HEIGHT` and `SCREEN_WIDTH` will be used to refer to the height and width of the screen respectively. This terminology and notation will be used extensively in the following explanation.

4.3 One-Point Calibration

As already mention above, the calibration process is very important for the system so that it can interact with the user correctly. This section describes how to create a new eye plane with the information provided by the one-point calibration process. As stated previously, every time a user starts using the system the location of the eye plane cannot be assumed to be in the same place as for the last execution. That is why calibration to initialise a new eye plane has to be performed. With one-point calibration the process is more complicated to implement than the four-point method because it uses a training phase in which nine points are used, but it is easier for the user for each session because only one point is needed for calibration. From the user's point of view the less overhead or time devoted to this task the better. Therefore reducing the number of calibration points from four to one for each session will make the system more user-friendly. For four-point calibration, the equations used to determine the parameters assume a linear mapping. For one-point calibration the idea is to determine if this is a true or useful assumption, and to allow some

general parameters to be acquired. With one-point calibration, the user has to look only at one-point on the computer screen located at the centre of the screen as in Figure 4.4. The recording process is the same as for the four-point method. The difference is in the eye-to-screen mapping process.

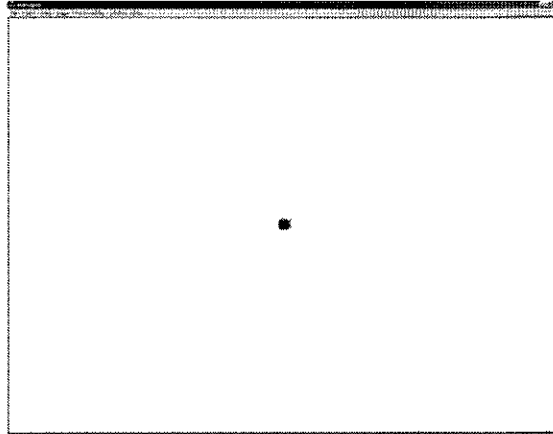


Figure 4.4 Screen capture of calibration point of one-point calibration

The one point calibration is an enhancement over the four-point counterpart. By using knowledge gathered from the four-point experiment we have found that if the camera position has not changed, we can substitute a number of parameters by constant values and make the calibration process easier and more efficient. Therefore the one point calibration process is divided into two stages. First, the offline stage, in which we gather information about the pattern of the eye coordinates when the subject scans the computer screen. We display nine points on the screen for the subject to observe which is the four points used in four point calibration and five additional points at the four corners and one in the middle of the screen. This information then will be processed to acquire parameters for the next stage. In the second stage, the online stage, the user will have to observe a point in the middle of the screen every time he/she uses the system. Then system will record the eye coordinates and fill in the rest of the information needed to initialise the mapping function. Given the camera settings remain the same, the user only has to perform the second stage each time he/she uses the system.

4.3.1 First Stage – The Offline Information Collection Stage

At this stage we gather information about the x and y coordinates of the iris for the nine points on the screen.

In order to find out how the x and y coordinate of the iris changes, the 9 points as in Figure 4.5 is displayed. For the x coordinate, the subject has to observe each of these points row by row. At each point 10 samples of the x coordinate will be recorded. The rows have values $y=0$, $y=SCREEN_HEIGHT/2$ and $y=SCREEN_HEIGHT$ for the first, second and the third row respectively. Figure 4.6 shows the results.

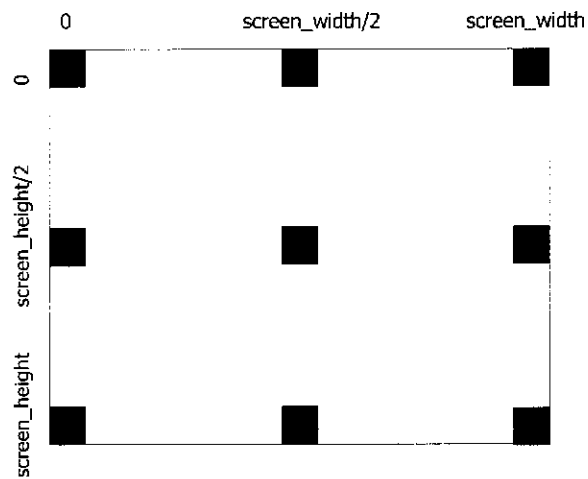
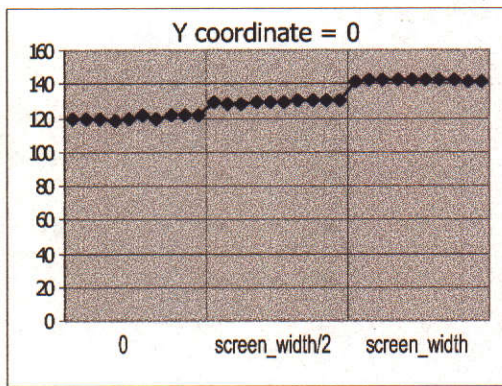
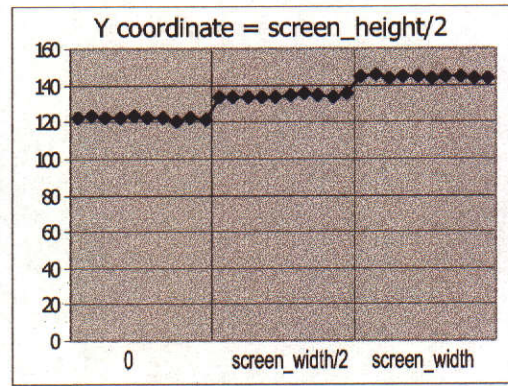


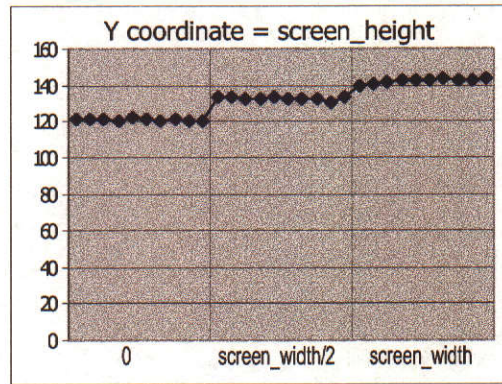
Figure 4.5 Nine (9) points on screen used for calibration



(a)



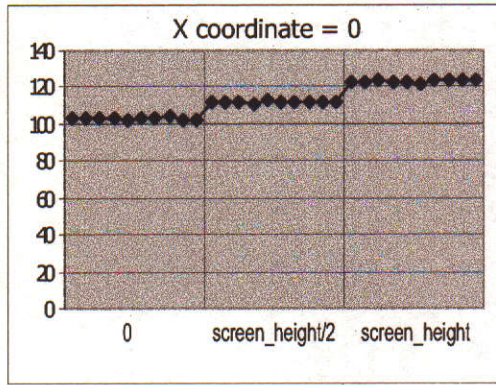
(b)



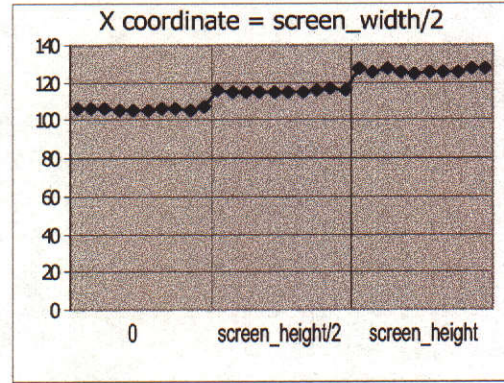
(c)

Figure 4.6 Variation in the x video coordinate when subject observed three rows on screen

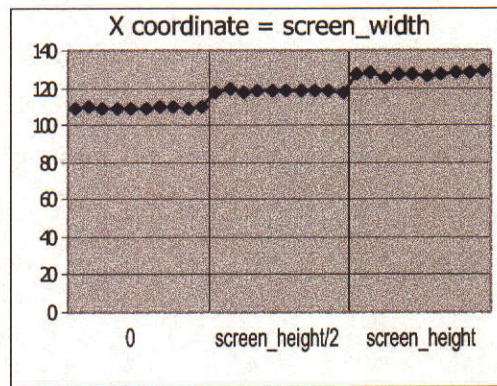
To record the y coordinate of the iris, the subject has to observe each of three points column by column. As before, we capture 10 samples of each y coordinate. The columns are at $x = 0$, $x = \text{SCREEN_WIDTH}/2$ and $x = \text{SCREEN_WIDTH}$ for the first, second and the third columns respectively. Figure 4.7 shows the results.



(a)



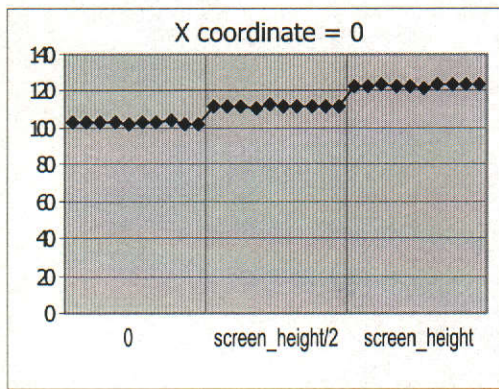
(b)



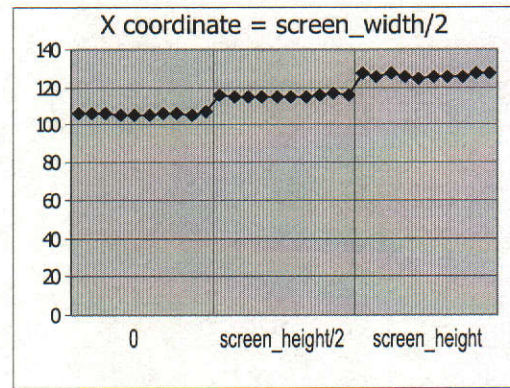
(c)

Figure 4.7 Variation in the y video coordinate when subject observed three columns on screen

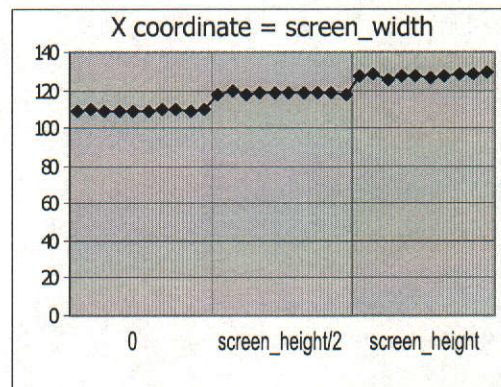
Figure 4.6a shows the x coordinates when the subject looks at points of the first row on the screen. Figures 4.6b and 4.6c show the second and third rows respectively. With the screen resolution used the first row is where the y coordinate is 0. For the second and third row the y coordinates are 479 and 959 respectively. The horizontal axis of the graphs is the x coordinate of the screen and the vertical axis represents the x coordinate in the video frame. For the first row the x coordinate of the iris starts (on average) from 120 at the top left of the screen and ends at 142 at the top right of the screen. Hence the total number of pixels for the iris to scan from one end to the other end of the screen is 22 i.e. for screen coordinates from 0 to 1279, which is the horizontal resolution the screen. The start and end values of the iris coordinate are not the same for each row. They are in the range of 122 to 145 and 121 to 143 for the second and third row respectively. The reason to this drift is, firstly, because the



(a)



(b)



(c)

Figure 4.7 Variation in the y video coordinate when subject observed three columns on screen

Figure 4.6a shows the x coordinates when the subject looks at points of the first row on the screen. Figures 4.6b and 4.6c show the second and third rows respectively. With the screen resolution used the first row is where the y coordinate is 0. For the second and third row the y coordinates are 479 and 959 respectively. The horizontal axis of the graphs is the x coordinate of the screen and the vertical axis represents the x coordinate in the video frame. For the first row the x coordinate of the iris starts (on average) from 120 at the top left of the screen and ends at 142 at the top right of the screen. Hence the total number of pixels for the iris to scan from one end to the other end of the screen is 22 i.e. for screen coordinates from 0 to 1279, which is the horizontal resolution the screen. The start and end values of the iris coordinate are not the same for each row. They are in the range of 122 to 145 and 121 to 143 for the second and third row respectively. The reason to this drift is, firstly, because the

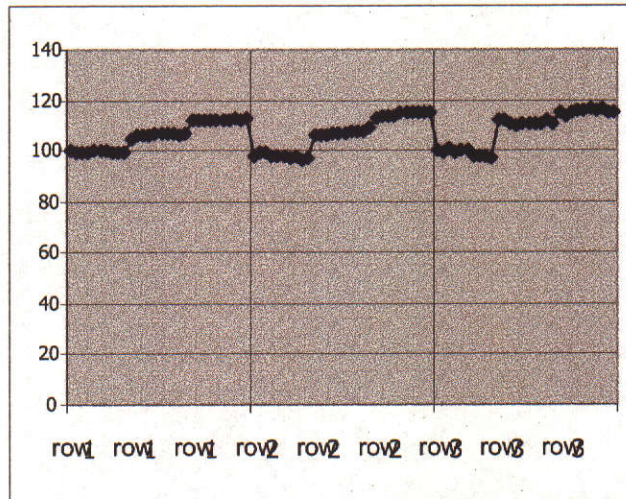
location of the camera located at the left side of the monitor. Secondly the way the eye moves is not in a 2D manner but really in 3D. In fact the eye rotates in 3D, which is modeled here as a 2D translation.

For the y coordinate, Figure 4.7a, b and c - shows the coordinates when the subject looks at points of the first, the second and the third columns on the screen with screen coordinates 0, 639 and 1279 for the first, the second and third column respectively. While coordinate in the video frame of the iris are in the range of 102-124, 106-126 and 109-129 for the first, second and third columns respectively. The horizontal axis of the graphs is the y coordinate of the screen and the vertical axis represents the y coordinate in the video frame. As the x coordinate, the start and end values of the iris coordinate are not the same for each column.

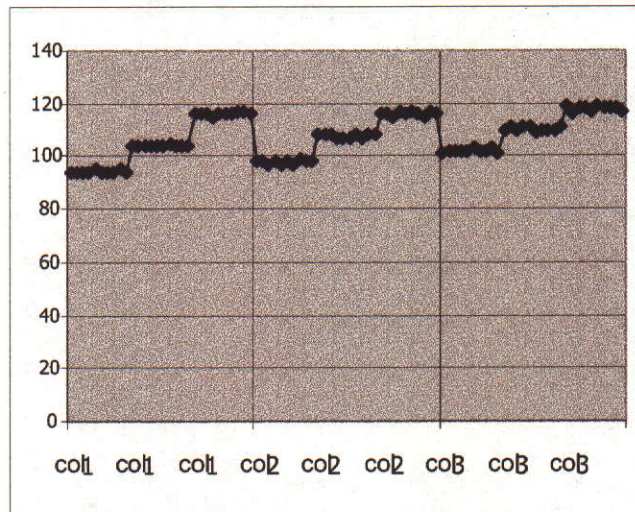
Note, these values are taken from an experiment for the sake of demonstration of how the one-point calibration technique is processed. The numbers could change from one experiment to another depending on the initial head position (which is the reason for doing the calibration).

The graphs show almost linear changes of the values of both x and y coordinates. To make sure that the system performance is consistent, another test of the same procedure has been carried out. Figure 4.8 shows the result from the second test (the figure includes all three rows of x in a single graph and all three columns in a second single graph for y). The graphs reveal important information about the eye coordinates in that the system produces the same pattern of graphs for both directions and appears to be approximately linear across and up the screen. The only difference between the first and second tests is where the eye plane starts and ends. As mentioned previously, the eye plane is an area within the video frame that defines the boundary of the eye while it scans the computer screen area. If the camera and the monitor are fixed, the initial location of the head when the subject starts using the system each time will be the only reason for the plane to shift from one place to another place in the video frame. Another factor that is affected by the initial head location is the number of pixels used for the eye to scan in the screen area or the width and height of the eye plane. The number of pixels will increase when the subject

moves closer to the camera. But for the relatively small difference of head location, there will be little effect on the size of the eye plane.



(a)



(b)

Figure 4.8 Variation in x and y video coordinates from another test

Another important characteristic of the way the x and y coordinates of the iris increase is the slope of their graphs. From the two tests, the slopes of the same row or same column have similar values. All this information shows that the system is quite consistent allowing for the possibility of reducing the number of calibration points from four to one.

Note: from information collected from a number of experiments, the slope value will be as claimed previously only if all system settings stay the same and the initial head position does not change too much. If all of these conditions are true, we can assume the slope will be approximately the same every time the user scans the screen. However, the results could change drastically if some of these conditions are false.

Since the relationship between the screen coordinates and the iris coordinates found from the graphs is almost linear, a linear function is used for the eye plane to screen plane mapping function in the one-point calibration technique. The linear function used here is derived from the equation of a straight equation, expressed in our application as:

$$\text{ScreenCoordinate} = M \times \text{Video(iris)Coordinate} + C \quad \text{e.q. 4.1}$$

The screen coordinate and iris coordinate in the equation can be either the x or y coordinate. Before we go into detail about how the one-point calibration method works we need to explain the M parameter in the equation which is the slope or gradient of a line. As stated previously, the relationship between the position of the iris scanning and the position on the screen is quite consistent. The slope of the graphs from the two tests are almost the same, therefore it is possible to use the slope value from the experiment which is a constant.

In the linear equation the value of M can be found by:

$$M = \frac{Y}{X} \quad \text{e.q. 4.2}$$

In terms of our application, the X value in the equation above is the numbers of pixels in the video from the iris tracking that correspond to the Y pixels on the screen which is better to represent as:

$$M = \frac{S}{V} \quad \text{e.q. 4.3}$$

It is the slope M of the x and y that we obtain from this stage of calibration. This is the parameter that helps us to achieve the one point calibration. Knowing the value of

slope in the linear equation in advance the rest of the parameters can be easily extracted using a single calibration point. In our case, the slope of x and y iris coordinate can be calculated from Figure 4.6 and 4.7.

To find M value of the y coordinate we use information from Figure 4.7. To make the picture clearer in Figure 4.9 we shows the data of graphs 4.7a, 4.7b and 4.7c modelled as single straight lines.

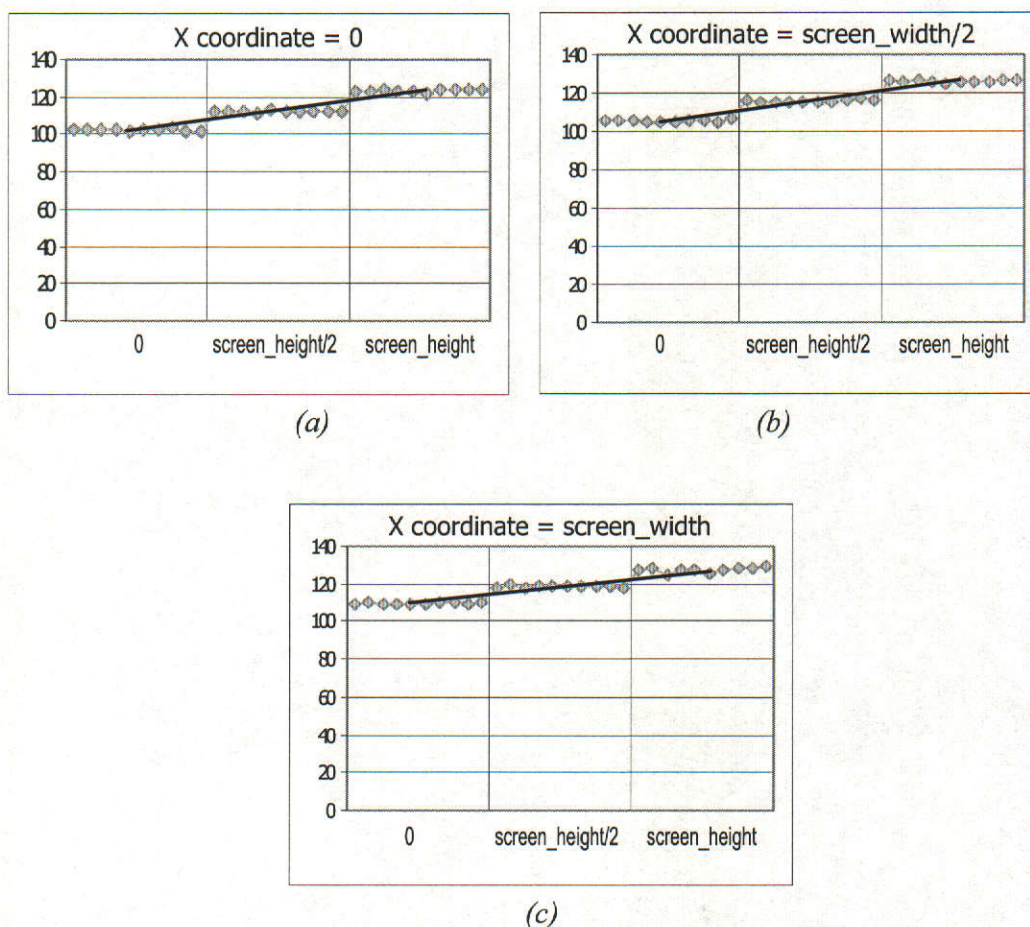


Figure 4.9 Fitting straight on graph of y video coordinate from figure 4.7

One thing that needs to be changed before we go further is the axis that represents screen coordinates and video coordinates. All the graphs that have been shown up until here have screen coordinates on the vertical axis and video coordinates on the horizontal axis. The conventional representation of the two variables is that the values on the vertical axis are a function of, or derived from, the value of the

horizontal axis. The mapping function we are using is from iris coordinates in the video frame to screen coordinates, therefore Figure 4.10 shows graphs that rearrange the axis of Figure 4.9. The horizontal axis will be referred to as V (Video) and the vertical axis will be referred to as S (Screen).

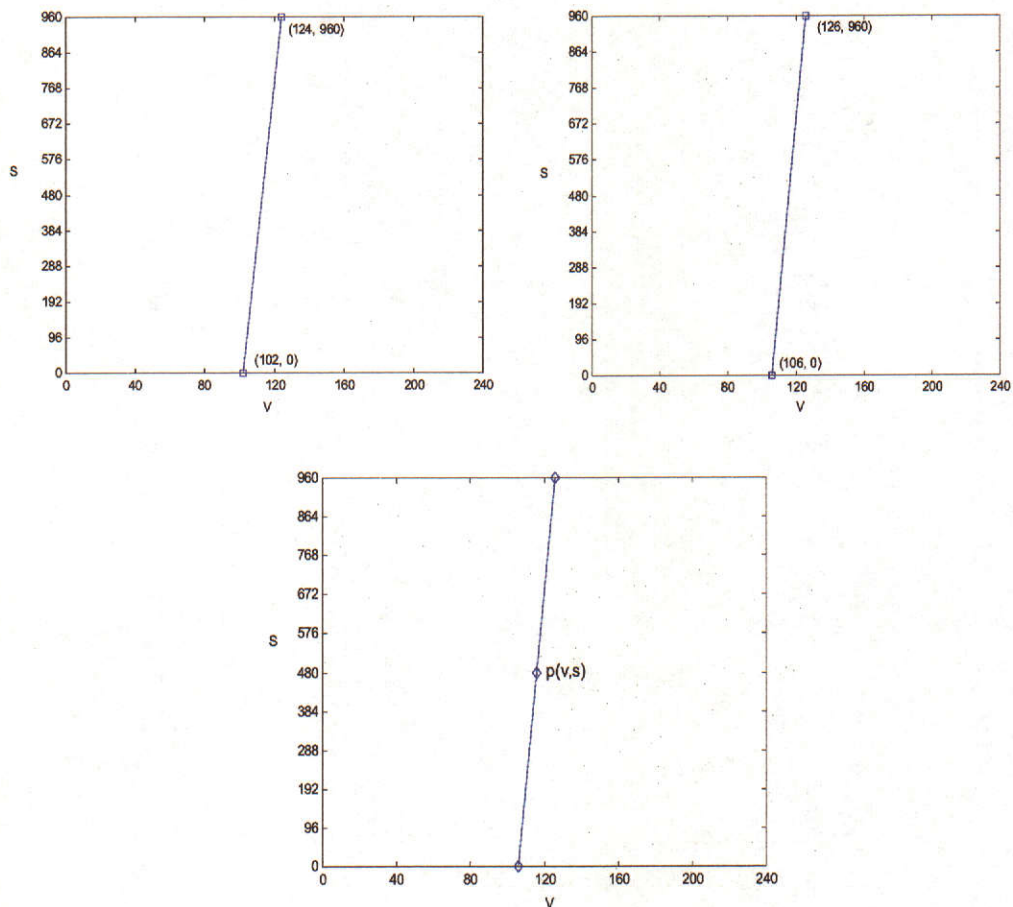


Figure 4.10 Same data as for figure 4.9 except with the axes swapped over

Therefore our linear equation will become:

$$S_y = M_y \times V_y + C \quad \text{e.q. 4.4}$$

First of all we need to know the slope value of these graphs. To find the value of M , we have to know the number of pixels of iris coordinate change when the subject scans from the top to the bottom of the screen. From experiments, the first column, the y coordinate will change about 22 pixels on average. The average number of pixels for the second and third columns is about 20 and 18 respectively. With

1280×960 screen resolution the parameter S in equation 4.4 is 960. Therefore M will be:

$$M_{y1} = \frac{960}{22} = 43.636$$

The same way we can find the M value second and third column.

$$M_{y2} = \frac{960}{20} = 48$$

$$M_{y3} = \frac{960}{18} = 53.33$$

In the case of the slope of the x iris coordinate, we have found that most of the time the slopes of the graphs of each row have similar values. This is also true for the end points that have similar coordinates. Figure 4.6 shows three rows of x coordinates from the video while the subject observes three rows on the screen: top, middle and bottom. The top row is where the y coordinate on the screen is equal to zero. For the middle row, the y coordinate is SCREEN_HEIGHT/2 and for the bottom row, is SCREEN_HEIGHT. By using this characteristic, the mapping function for the x coordinate from video to screen, will be performed by applying only one linear function for the whole screen. This assumption might not create the perfect result but it is the easiest way to deal with the unreliability of the x coordinate. The mapping function for the x coordinate is:

$$S_x = M_x \times V_x + C \quad \text{e.q. 4.9}$$

As with the y coordinate, the slope M will be obtained by experiment and will be used as a constant value in the mapping process. From the slope equation 4.3 the value of S here is SCREEN_WIDTH or 1280 and the V value is number of pixels used to scan the screen horizontally which is estimated to be 22 pixels. Therefore the slope value is:

$$M_x = \frac{1280}{22} = 58.18$$

What we have found from this stage are three slope values (M_{y1} , M_{y2} and M_{y3}) of three column of y coordinate and a slope value (M_x) of x coordinate.

4.3.2 Second Stage - The Online Stage

This stage can be considered as the online stage. It will be carried out every time a user uses the system. The user will be asked to observe a calibration point at the middle of the screen after which the rest of the parameters needed for eye to screen coordinate mapping will be computed.

Specifically, the X and Y coordinate calibration approach of this stage will be explained separately. The process for the Y coordinate is more complicated and has some common calculation to the X coordinate calculation. Therefore the treatment of the Y coordinate will be described first.

4.3.2.1 Y Coordinate Calibration and Mapping

To demonstrate how the method works, we assume the subject observed the calibration point and the iris coordinate recorded is $(107,118)$.

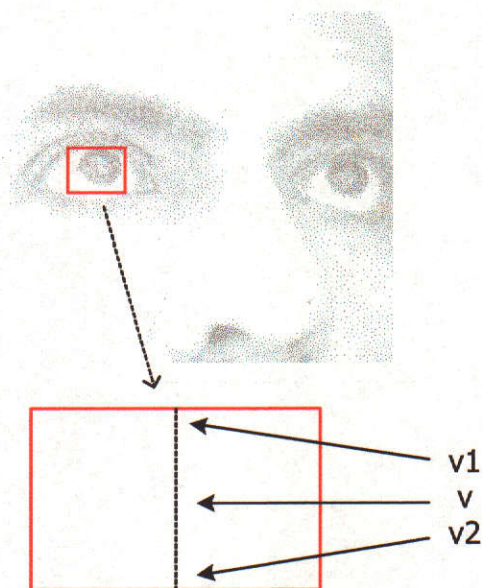


Figure 4.11 Shows a graph of a linear function with slope M_{y2}

Figure 4.11 shows a graph of a linear function with slope M_{y2} . In the middle of the line there is a point $p(v,s)$, plotted using information gathered from the calibration process. That is when the subject is asked to observe a point located at the middle of the screen, x and y coordinates of the iris in the video frame will be recorded. Apart from that we also know the x and y coordinates of the point on the screen that the

subject observed. Because the subject looked at the middle of the screen therefore s is equal to SCREEN_HEIGHT/2 or 480. The value v is the y coordinate of the iris in the video frame when the subject looked at the calibration point which is 118 in this case. The point $p(v,s)$ is a point where a line with slope M_{y2} intersects. This value is also the y coordinate of a point at the centre of the eye plane (see Figure 4.11). What we finally need to do is find what the start and end coordinate of this particular column of the eye plane are (v_1 and v_2 in Figure 4.11) and what is the value of C in equation 4.4.

The parameter C , usually referred to as the y -Intercept, has a direct connection with the initial head location. As explained above, comparing the graphs from the two tests, reveals the only major difference between them is where the eye plane starts and ends. The shifting of the eye plane is caused by a different initial head location. In terms of the linear equation it means that the function will cross the y axis at a different point.

To find the value of parameter C , we simply rearrange equation 4.4 and substitute S with 480 and M with M_{y2} (48) and V with 118 to get:

$$C = S_y - M \times V_y \quad \text{e.q. 4.5}$$

$$C = 480 - 48 \times 118 = -5184$$

After the value of C has been determined, we can find the start and end coordinates of the iris in the video frame. In Figure 4.11 we referred to the start coordinate as v_1 and the end coordinate as v_2 . We know that v_1 screen coordinate at this point is zero. To find v_1 we substitute S in the equation 4.6 with 0, C with the value we just found, -5184, and M with 48. Therefore the value of v_1 is computed using:

$$V_y = \frac{(S_y - C)}{M} \quad \text{e.q. 4.6}$$

$$v_1 = \frac{(0 - (-5184))}{48} = 108$$

Next to find v_2 , we will do almost the same as for v_1 except at v_2 the screen coordinate will be SCREEN_HEIGHT or 960.

$$v_2 = \frac{(960 - (-5184))}{48} = 128$$

We now have complete information for the second column. What we need now is the information of the first and the third columns so we can get a complete description of the eye plane. However we do not have any information about these two columns apart from their slope value. To solve the problem, first of all the original graph of the three columns has to be reconsidered. In Figure 4.12 all the graphs have been put together in new graph and the axis system also changed back to the original one which is the vertical axis for the video coordinate and the horizontal axis for the screen coordinate.

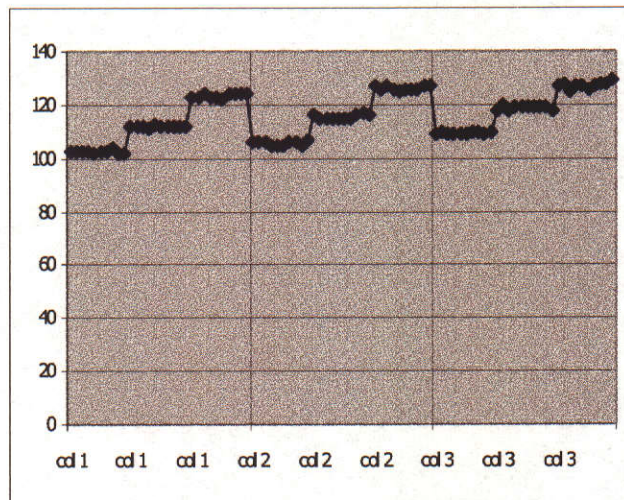


Figure 4.12 Show graph of all of the 3 column of y coordinate

One important thing that the graph depicted here is that the iris coordinate has different start and end points on each column. In addition, the number of pixels for the iris to scan each column are also different (it has to be different because each column has a different slope value). As we have stated before the shape of the eye plane is not rectangular but it can still be described mathematically. If we look at the end point of each column we will see that it increases slightly from the first column to the last. This increasing rate also appears to be consistent and linear. If we draw a line joining these three points, the output will be a straight line as shown in Figure 4.13. Again we will exploit another linearity property of the tracker for our benefit.

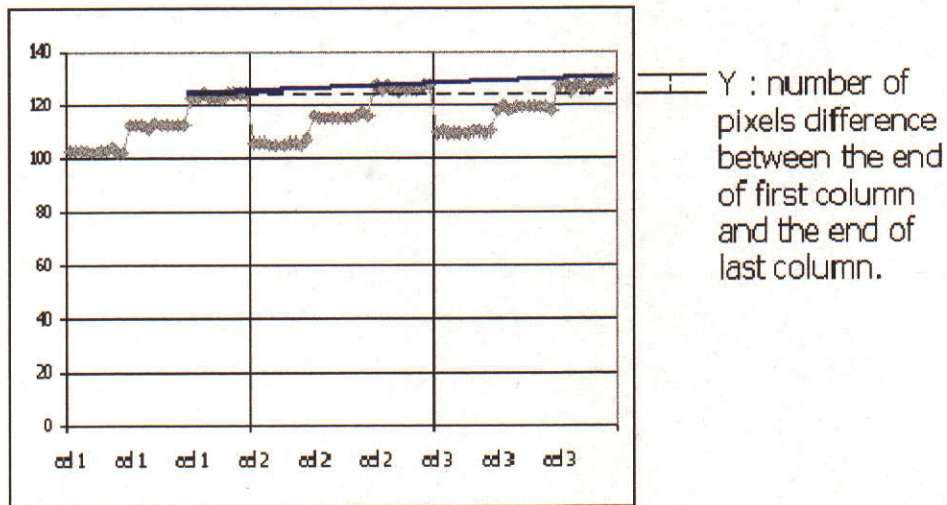


Figure 4.13 Drawing a straight line to the last point of each column to find relationship between these columns

The three columns the subject observed on the screen have x coordinates 0, SCREEN_WIDTH/2 and SCREEN_WIDTH for the first, second and third columns respectively. The line connecting the ends of these columns will be used to find the relationship between them. First of all we need to know the slope of the line. The same method used above will be applied here.

From the experiment the average value of Y is estimated to be approximately 5.0 and the value of X is the width of the screen or 1280. Therefore the value of M will be:

$$M = 5/1280 = 0.0039$$

The slope M above will be referred to as M_e and its line equation will be represented as:

$$V_e = M_e \times S_e + C_e \quad \text{e.q. 4.7}$$

where the video coordinate V_e is a function of the screen coordinate S_e .

At this stage what is known is the end coordinate of the second column at the middle of the screen. At the end coordinate all of the three columns have the same screen coordinate, which is SCREEN_HEIGHT. We also know the slope of the line that

connects them. What we need to know now is the video coordinate at the end point of the first and third columns and this is where the line equation we just mentioned will be used.

The value of C_e has to be known first. Since the value of V_e , M_e and S_e of the second column are known which is 128 (v_1), 0.0039 and 640 (SCREEN_WIDTH/2). C_e can be found by substituting those values into the equation:

$$C_e = V_e - M_e \times S_e \quad \text{e.q. 4.8}$$

$$C_e = 128 - 0.0039 \times (640) = 125.5$$

When the value C_e is known, the video coordinate at the end point of the first and third columns can be determined. For the first column the $S_e = 0$ so the V_e will be:

$$V_e = M_e \times 0 + 125.5 = 125.5$$

For the third column S_e is 1280 (SCREEN_WIDTH) so V_e will be:

$$V_e = 0.0039 \times 1280 + 125.5 = 130.5$$

Since the coordinate of the end point of the first and third columns are now known, they can be used to complete the linear equation. For both columns first find the C value of the equation. From eq. 4.6, $S_y = 960$, $M_{y,l} = 43.636$ so the C value for the equation of the first column is:

$$C = 960 - 43.636 \times 125.5 = -4516.3$$

The C value for the equation of the third column is:

$$C = 960 - 53.3 \times 130.5 = -5995.65$$

What is left to be done is to find the coordinate of the start point of these two columns. By using equation 4.6 the start point of the first column ($M = M_{y,l}$) is:

$$V = \frac{(S - C)}{M_1}$$

$$V = \frac{(0 + 4516.3)}{43.636} = 103.5$$

For the third column ($M = M_{y3}$) it is.

$$V = \frac{(0 + 5995.65)}{53.33} = 112.5$$

Now all the required information for the y coordinate for video to screen mapping has been found and is depicted in Figure 4.14. From the calibration we know only one point at the centre of the eye plane and with the process described in this section, all the information of the y coordinate of the eye plane has been constructed. The start and end points of the three columns are [103.5, 125.5], [108, 128] and [112.5, 130.5] respectively.

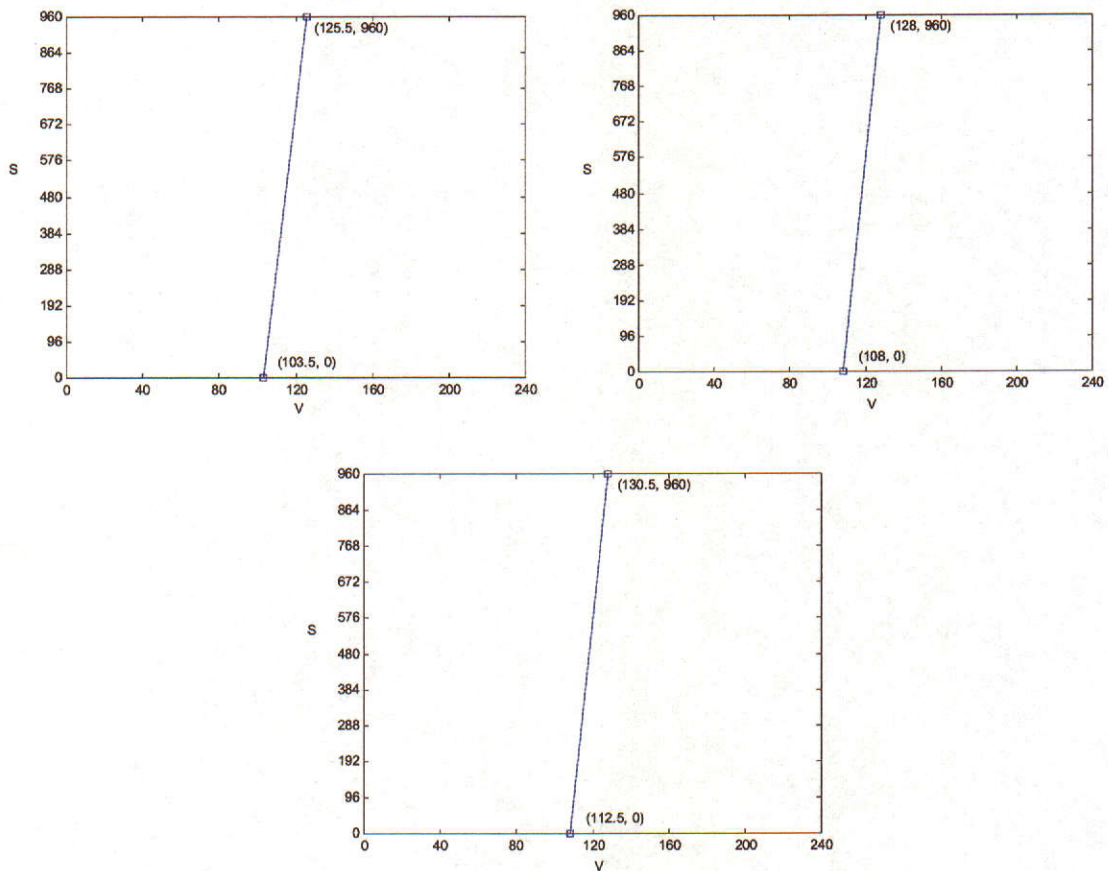


Figure 4.14 Show the complete information all the three columns achieved from this process

4.3.2.2 X Coordinate Calibration and Mapping

Computing the values for the x coordinate is different to those for the y coordinate. From a number of test results the value of the x coordinate has a less reliable pattern than for y . This may be because the head is more likely to move in the left or right directions than up and down.

The value of M_x , 58.18, will be used for the slope value of the x coordinate mapping function. Similar to calculating the y mapping function, another important parameter in the linear equation that we have to know is the intercept or C value. Since we assume that the x coordinate has the same start and end points, therefore what we need to know is the C value of only one row and that is the middle row. The process is the same as for the y coordinate to find the C value of the middle row.

The information about the iris centroid and screen coordinates that are available after the subject has observed the calibration point will be used here again. For demonstration purposes, as mentioned in the coordinate calibration section, the value of the iris x coordinate is 107. The S is SCREEN_WIDTH/2 or 640 and the M is the value that just been found, 58.18. Therefore from:

$$C = S_x - M \times V_x \quad \text{e.q. 4.10}$$

C can be derived:

$$C = 640 - 58.18 \times 107 = -5585.26$$

The -5585.26 is the s -intercept which will be used for every row of x coordinate mapping. The start point of x , which is the same for every row, can be determined as:

$$V_x = \frac{(S_x - C)}{M} \quad \text{e.q. 4.11}$$
$$V_x = \frac{(0 + 5585.26)}{58.18} = 96$$

The end point is:

$$V_x = \frac{(1280 + 5585.26)}{58.18} = 118$$

With this information we can draw a new eye plane within a 320×240 video frame (see Figure 4.15). The resulting plane shape is not a symmetric rectangular shape. This is because the camera is located at the left side of the eye which causes the left side of the plane to be higher than the right.

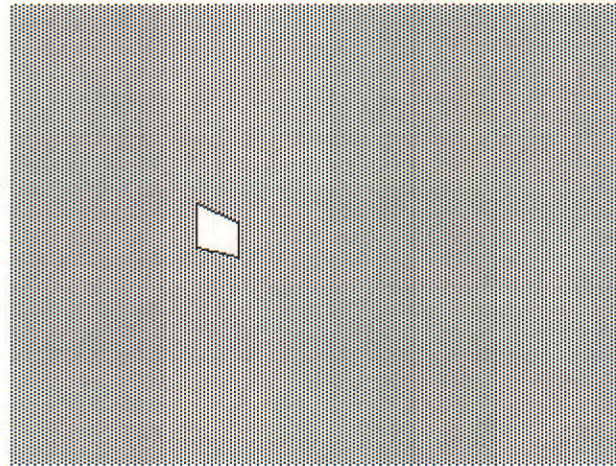


Figure 4.15 Resulting shape of the real eye plane

4.3.2.3 Video to Screen Coordinate Mapping

This process can start only after the subject has carried out the screen calibration. All the information obtained from the calibration process will be used to find a point on the screen that is associated with the centroid of the iris. When the process starts, the iris tracking sub-system will send the coordinates of the iris centroid for the current video frame to the mapping process. The x coordinate will be processed first. It will be converted to screen coordinates using the x coordinate mapping equation:

$$S_x = 58.18 \times X - 5585.26$$

It has to be verified that the output S value is located inside the screen area. If not then it is just clamped to the end of the screen. After the x screen coordinate is known, the y screen coordinate can be found. However, as explained in the y coordinate mapping section, the y coordinate mapping is not the same as the x because the slope and the start and end points of the y coordinate are not the same for each column. What we know so far is how to map the y coordinate if the x screen coordinate is at

one of the three columns ($S_x \in \{0, \text{SCREEN_WIDTH}/2, \text{SCREEN_WIDTH}\}$). If the x screen coordinates that is found from the equation above falls in between those columns, we have to find the slope and the intercept value for those columns before mapping can be done. What is known at this stage are the slopes and end points of the three columns. To find the slope value if the x screen coordinate is lying between two of these columns, the known slope values will be used to find the proportion of the new slope. For example if the x screen coordinate is lying between the first and second columns, the slope of the first and the second column will be used in linear interpolation:

$$M = \left(\frac{M_2 - M_1}{\frac{\text{SCREEN_WIDTH}}{2}} \right) \times S_x \quad \text{e.q. 4.12}$$

Another parameter is needed to complete the y mapping function which is the intercept value. The same method used to find the end points of the first and the third columns will be applied here. By using the equation 4.7 for the end point of the y coordinate, we can find the C value. The end point of y coordinate at x screen coordinate is found using:

$$V_e = M_e \times S_x + C_e \quad \text{e.q. 4.13}$$

The M_e and C_e values are from eq. 4.7 and 4.8 which are 0.0039 and 125.5. S_x is the x screen coordinate we have just found. Substituting these values into the equation, we will get the end point of the screen coordinate with $x = S_x$:

$$V_e = 0.0039 \times S_x + 125.5$$

To find the value of C for the equation of this column:

$$C = \text{SCREEN_HEIGHT} - M \times V_e \quad \text{e.q. 4.14}$$

Now the equation for the y coordinate mapping is ready. With a Y coordinate from the eye tracker, a Y (S_y) coordinate for the screen can be obtained from:

$$S_y = M \times Y + C \quad \text{e.q. 4.15}$$

4.4 Performance Comparison of the Two Techniques

In this section the accuracy of four-point and one-point calibration methods are analysed. The testing is done by measuring the distance between points that are mapped to the screen by each technique and the point that the user is actually looking at. Random points on the screen are chosen for the user to look at. Figure 4.16 shows the result of the tests and compares performance of the two techniques. The vertical axis of the graph represents the distance between the points shown on the screen and the point mapped to the screen by the two calibration techniques. The distance, in other words, is the error in calibration. The horizontal axis represents a number of random points generated by the system and observed by the eye. The graph reveals the one-point mapping technique is more consistent with 60.26 pixels average error. The average error of four-point mapping is 116.57 that is twice as much as the one-point counterpart. A similar trend is also apparent for the standard deviation, which is 28.18 and 61.10 for one and four point mapping respectively.

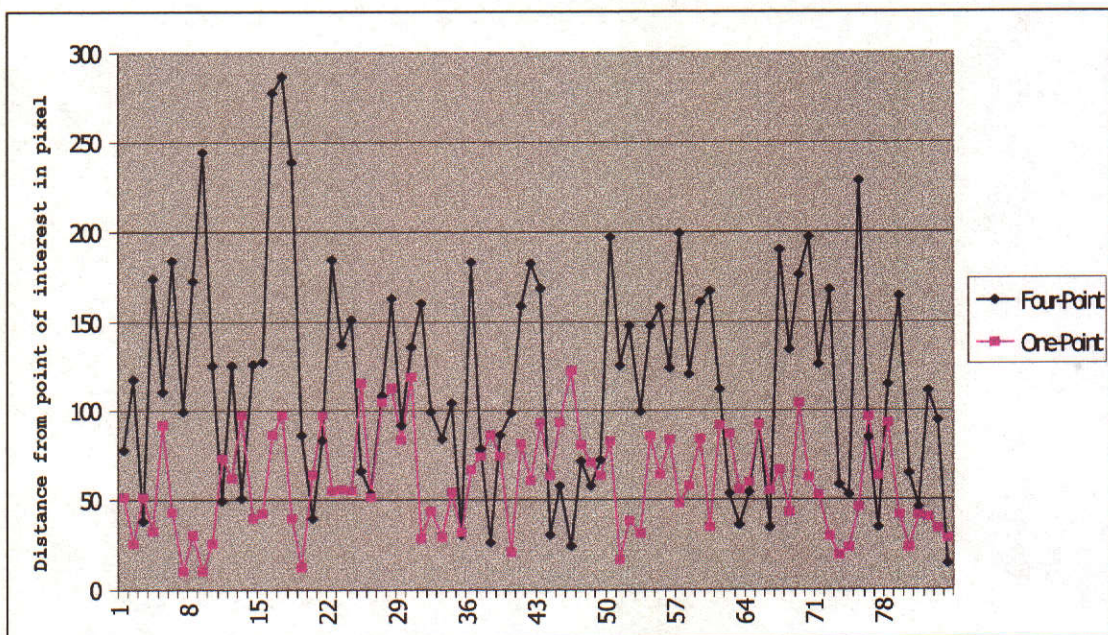
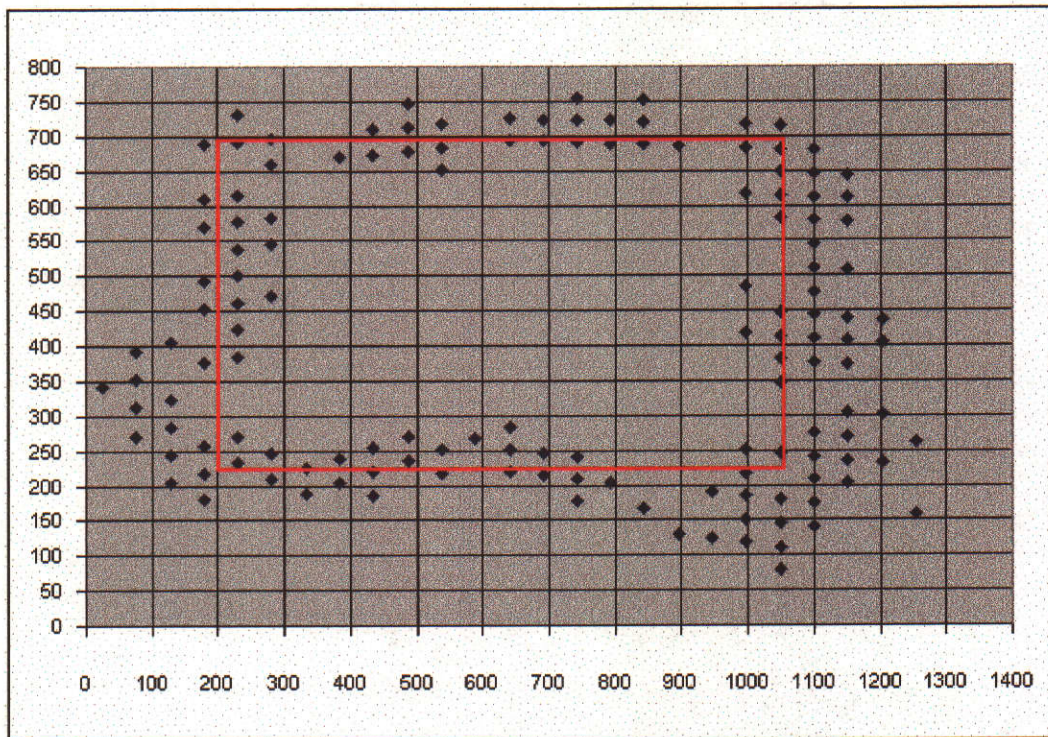


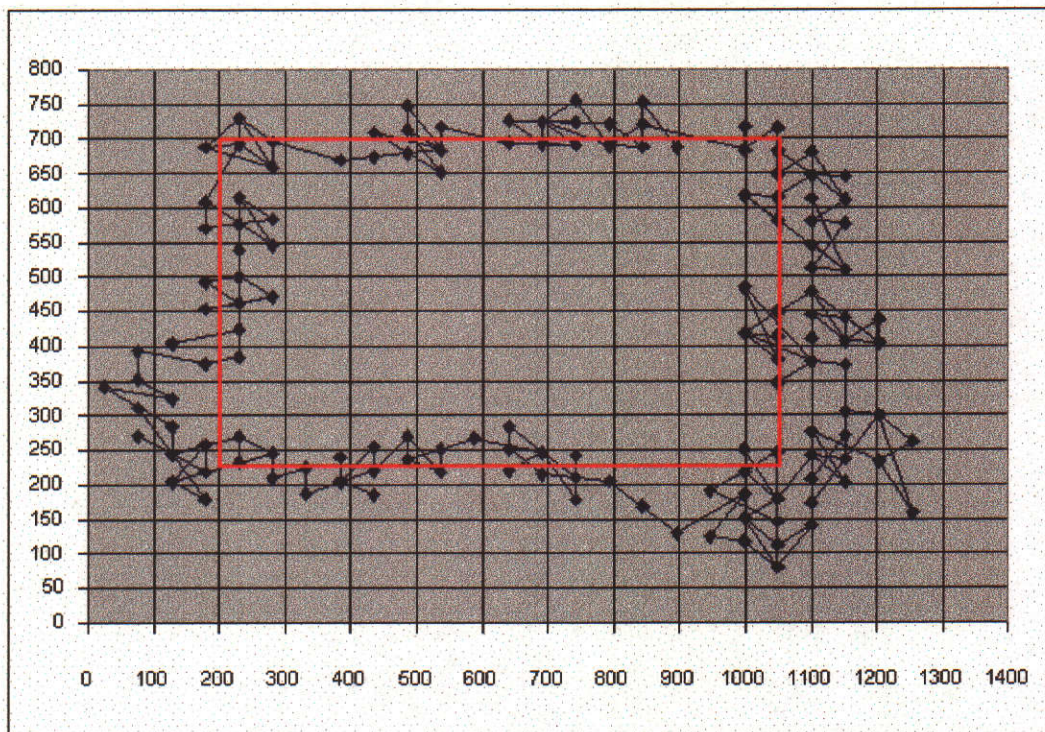
Figure 4.16 Comparison of performance between one-point calibration and four-point calibration

The test result has shown that the one-point mapping technique is not only more convenient but also more accurate than the four-point technique.

We also carried out another test for the one-point calibration technique by having a subject to observe a rectangular line on screen while the system recording the mapped coordinates from the system. The rectangular coordinates are 216, 210, 700 and 1050 for top, left, bottom and right respectively. Figure 4.17 shows the result of the test in two ways. In both figures, dots represent the coordinates calculated by the system while the subject followed the red line. In the second figure, the dots are joined together to show the sequence of points calculated by the system. The second figure shows there is error along the line as well as away from it (e.g. horizontal errors when tracking a horizontal line). Table 4.2 shows the result of the test. From the graph the large fluctuations occur mostly at the left and the right side of the screen where the iris is either too close and too far from the camera.



(a)



(b)

Figure 4.17 Testing the one-point calibration technique by having a subject observe a rectangular shape on the screen. (a) shows just the points, and (b) shows the path given by the sequence of points.

<i>Mean of error</i>	<i>Max error</i>	<i>Min error</i>	<i>Stdev of error</i>
45.41455	204	1	40.77

Table 4.2 Result of the one-point calibration test

The errors produced can be from two sources. The first is sub-conscious eye movements (saccade, miniature – see Section 2.2.2)), and errors in segmentation due to lighting etc. Figure 4.17(b) implies that averaging a number of time adjacent points would give more accurate results as eye coordinates computed at similar times have similar values i.e. there are no large errors between time adjacent coordinates.

However this was not explored in this thesis.

4.5 Discussion

It is obvious that more research is needed for the calibration and mapping techniques. One of the major causes of the limitation is because we do not have a good reference

point to provide information about the head position. It is important to the mapping process to know the relative position of the eye to the head in order to map the coordinates correctly. A frequent problem during use of the system is that the mapped point falls out the focus area because of accidental head movement. If we try to follow the point it will move away even further. However, with some experience, we can correct it by moving our head back slightly and looking to the opposite direction until we can move the point into our focus again. On the issue of how high the precision of the mapping should be, we believe that for our application as long as the mapped point is still in the focus area (or we can still see it under area of interest) it should be acceptable as the magnified area will have radius wide enough to cover the area of interest.

4.6 Summary

In this chapter, calibration techniques have been discussed. The calibration process is an important part of the eye tracking system for HCI applications. In HCI, calibration will be used to help in the coordinate mapping process. Two calibration techniques have been developed in this research namely four-point and one-point calibration. The four-point technique uses four marked points on the screen for the user to observe while the one-point uses only one marked point. The one-point technique presents a novel calibration technique that requires only one point, by using *a priori* eye model (the assumptions of linearity and results from the four-point mapping process). Moreover, the results show that the one-point technique is also more accurate than the standard multi-point counterpart.

Chapter 5 Integration of Eye Tracking and Non-linear Visualisation

5.1 Introduction

Human computer interaction (HCI) is one of the applications of eye tracking. Within the field of HCI itself there are several possible applications that can incorporate eye tracking technology. In this research we investigate the possibility of using eye tracking together with a non-linear visualisation technique to navigate a large data space. Non-linear visualisation techniques such as fisheye projection have been used to overcome limitations in representing large amounts of data in a small display area such as a computer screen. With non-linear techniques, only the interesting part of the information will be shown at high detail while the rest of the information will be compressed at the fringes of the display area. Another benefit of using non-linear visualisation techniques is the ability to deliver both detail and context at the same time. This contrasts against linear visualisation techniques that allow observation of the overall context of the information at the expense of detail (and vice versa). The eye tracking system in this thesis will be used as a means for the user to indicate a region of interest in the data space. Information within the user's interested area will then be shown at a higher resolution, revealing more detail while representation of the other areas will be compressed (Leung and Apperley., 1994).

5.2 Benefit of Eye Tracking and Non-Linear Visualisation

In the multi-modal HCI research area, eye tracking has been recognised as an important part of the future user interface. The cooperation between eye tracking and a non-linear visualisation technique will deliver a friendlier user interface. The message conveyed by the eye has immense usefulness as stated in chapter 3, especially in the field of Psychology and Cognitive science that has long been using eye tracking for abnormality diagnosis. For HCI, apart from using the eye for direct interaction, the eye can also be used to extract some natural but unconscious messages

that are very useful, such as intention. The system can use staring or looking at a certain point for a certain period of time as an indicator of user interest. By reacting to that indicator without any extra effort, the user will get an impression that the system can read their mind. The system will deliver an interaction method, which requires less effort but with more sophistication. That is, by looking, the system will respond to the user's intention by displaying information the user is interested in at a higher level of detail (using zooming).

For a non-linear visualisation system, real time response to the eye movement might not be beneficial because fixation rather than movement may be used to represent user intention. While the user is moving their eye, very small details can be perceived. It is unnecessary to interact during eye movement because it usually means the user just wants to search for something. Only when he/she find what he/she wants, the eye will stop moving and rest there to observe more detail. This assumption is also true when using a mouse as a pointer. The system should use the fact that the mouse has stopped as an indicator that the user really intends to find more detail about that area.

Although it has been shown that eye tracking is not yet accurate enough to replace pointer devices such as a mouse (Jacob, 1993; Hyrskykari, 1997), it may be used if the object of interest to be interacted with is large enough to accommodate the error in position (e.g. using a resolution of 640×480 instead of 1280×1024 would make icons and many items much larger).

Two important elements, the context and detail, have to be considered when discussing the benefits of non-linear visualisation. The context of the data provides an overall picture and relationship between each item in the data space, while the detail gives specific information of each item. In linear visualisation techniques when information is shown with high compression it will reveal more context but very little or no detail. However when it is shown with high magnification it will give high detail but little or no context.

For a non-linear visualisation system the magnified area normally will have a large radius to trade off the need for detail and context. In Figure 5.1, although the eye gaze is interpreted to be somewhat different to what the user intends, the result still

covers the area the user needs i.e. the centre region is magnified, the whole image is visible, but there is distortion.

In this thesis, non-linear visualisation concentrates on images and maps i.e. areas of interest are zoomed-in parts of the image or map to reveal the detail. This task requires high processing power and data manipulation. In this research the non-linear visualised effect is represented by grid lines to reduce processing time. The point of interest will be shown like a swelling surface (see Figure 5.1).

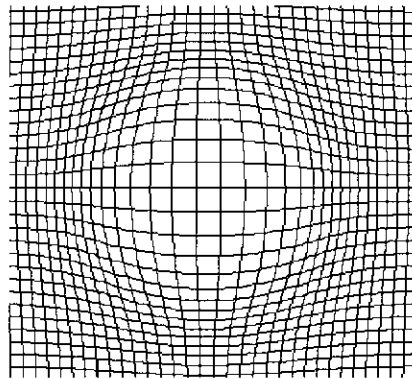


Figure 5.1 Non-linear visualisation effects on grid lines

5.3 Applying Non-Linear Visualisation

As mentioned above, there are a number of methods to achieve non-linear visualisation. In our work we will use the fisheye technique for preliminary evaluation of the integration of the eye tracking and non-linear visualisation systems. Selection of the most appropriate technique requires further study which includes study about performance and user satisfaction. While the performance can be easily determined by the complexity and processing time of the algorithm, user satisfaction of any particular technique is a rather subjective test. We will leave the comparative study of these techniques for future research.

One important property of a fisheye function for non-linear transformations is a smoothly varying slope over the area of magnification to demagnification as Sarkar and Brown(1992) proposed:

$$T(x) = \frac{(d+1) \times x}{(d \times x + 1)}$$

to perform this transformation. The function produced a desirable non-linear characteristics only over the [0,1] domain. So the coordinates that will be transformed have to be normalised into the [0,1] domain first. However there are many possible functions that can produce the same result and do not require domain normalisation. One such function is the hyperbolic tangent function (Keahey and Robertson, 1996):

$$T(x) = \tanh(x)$$

The tanh function performs in the desired way over the infinite domain. Another parameter β added to the function will be used to control the degree of magnification.

$$T(x, \beta) = \tanh(x \times \beta)$$

The derivative or magnification of the tanh function is:

$$M(x) = \text{sech} = 1 - \tanh^2(x)$$

Figure 5.2 shows the tanh transform and magnification function for $\beta=0.015$.

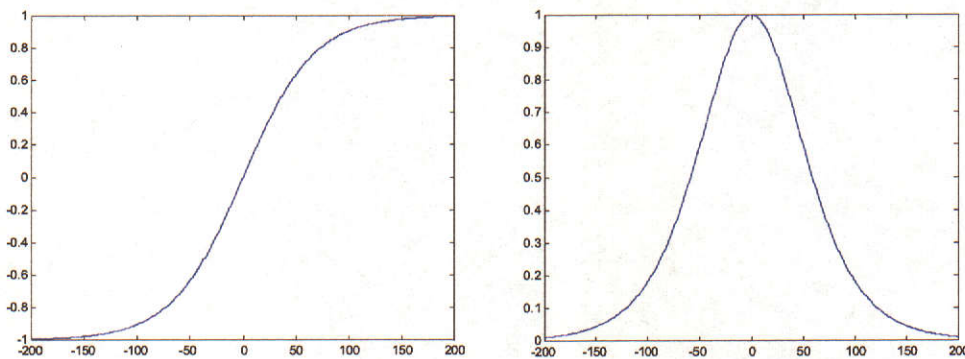


Figure 5.2 Tanh transform and magnification function

In this research we apply the tanh function to produce the non-linear visualisation effect. Figure 5.4 shows screen shots of the tanh function applied to the grid lines. The parameter β used in the equation is 0.015. A smaller value of β will create a wider radius of magnification. Figure 5.3 shows a comparison of magnification function with 0.015 and 0.05 values for β .

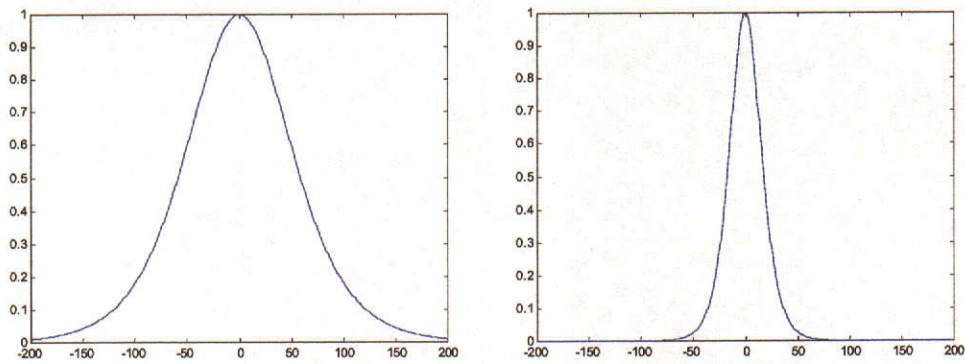
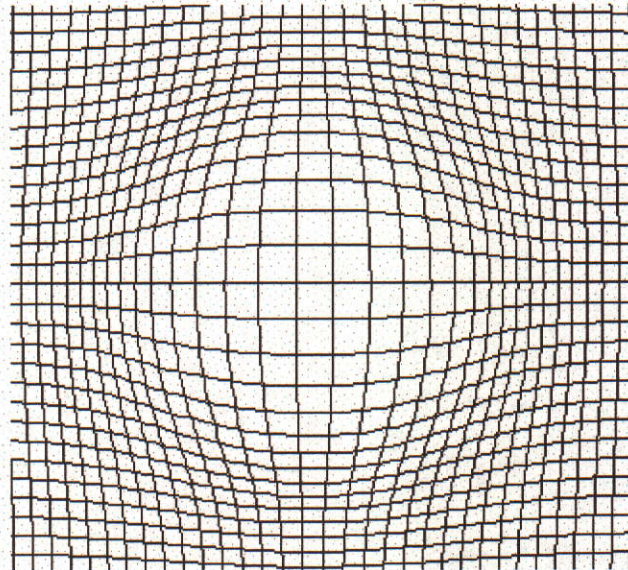


Figure 5.3 comparison of magnification function with 0.015 and 0.05 β value

While β can be used to control the dispersion of the magnified area, the magnitude of the fisheye can be as high as one with the equation above. In order to be able to create a more extreme curve and hence mapping, some modification to the equation is needed:

$$M(x) = A - (A-1) \times \tanh^2(\beta \times x); A > 1$$

The parameter A will be used to control the magnitude of the fisheye curve. The \tanh^2 function has domain $[0,1]$, and to display a magnified effect A has to be more than or equal to two. In Figure 5.4 show the comparison of three different A values, 2, 3 and 4.



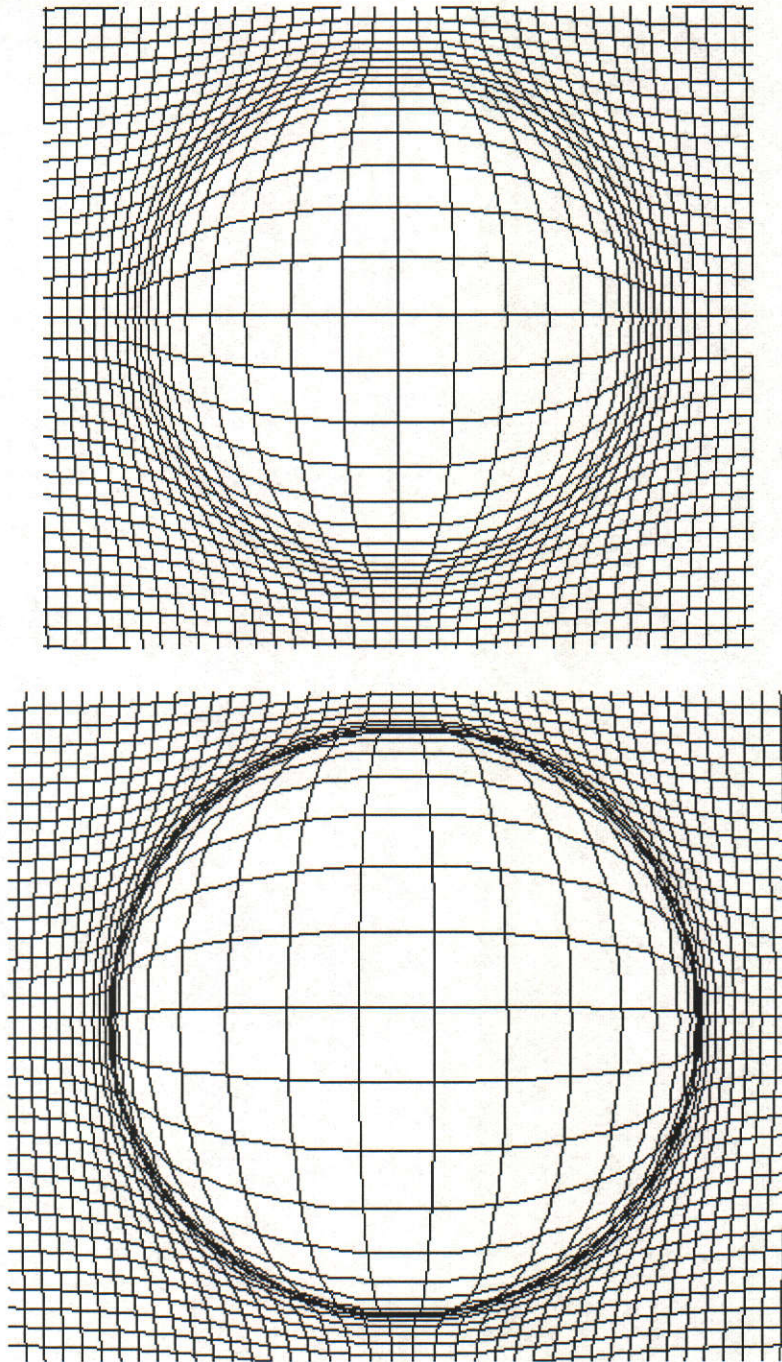


Figure 5.4 Comparison magnified grid of 3 different A values 2 3 and 4

5.4 Implementation Details

All the experiments about non-linear visualisation that have been carried out so far used an image of grid lines. In a real application, the system has to be able to handle the transformation of every pixel, or at least those that reside within the range of magnification. This may require some more study in terms of implementation.

However the basic idea will still be the same to the way things are operated on the grid line.

The grid is drawn as a series of horizontal and vertical lines. The magnification process can be performed on horizontal and vertical lines separately. The operation is almost identical for both of them so we will demonstrate the operation on a general line which can be applied to any of them.

Before the process starts a magnification ratio look-up table has to be initialised. Figure 5.5 shows the graph used to form the table. The size of this table is two times that of the size of the screen (currently twice the screen width). At the centre of the graph (or focus point) is the point of interest that has the highest magnification ratio. In this graph the magnitude control value A , is 2 and the dispersion control value, β , is 0.015. So the equation used here is:

$$M(x) = 2 - \tanh^2(0.015 \times x)$$

For any particular line, every pixel on the line will be examined. To find the magnification ratio for a pixel on the line, first a distance between that pixel to the focus point has to be calculated. The distance will be used as an index into the magnification ratio look-up table. The distance value can be positive or negative. Positive distances occur when the pixel is to the right of the focus point and negative distances occur when the pixel is to the left of the focus point. At the focus point the distance is zero.

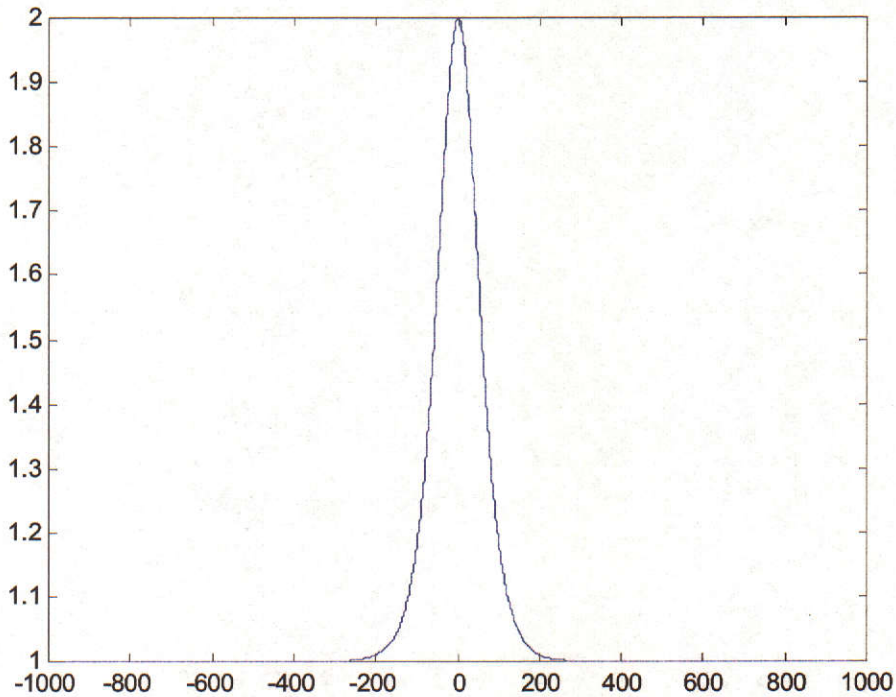


Figure 5.5 Graph of magnification ratio with $A = 2$ and $\beta = 0.015$

For a point (x,y) , distance from focus point (x_c, y_c) and be found using :

$$D = \sqrt{(x_c - x)^2 + (y_c - y)^2}$$

where (x_c, y_c) is the point of focus and (x,y) is the pixel being mapped. The magnification ratio R can be found as:

$$R = \text{MagRatio}[D]$$

where MagRatio is the magnification ratio look-up table.

By applying a magnification ratio to a point (x,y) we will get a magnified *point* (X', Y') :

$$X' = x_c + R \times (x - x_c)$$

$$Y' = y_c + R \times (y - y_c)$$

5.5 Results and Discussion

The testing of non-linear visualisation was carried out with traditional means using a mouse as well as iris tracking. The major difference between the two is the ability to resolve screen coordinates. While the mouse can resolve every pixel of any screen resolution, the iris has a very limited resolution of about 25×25 pixels. That means, with an 800×600 screen, the mapping ratio is 1:32 horizontally and 1:24 vertically. A small change of iris centroid coordinate causes a large shift on the screen. This issue is apparent in real time experimentation. The non-linear effect appears jerky even when the subject is keeping his eye still. The jerking can be easily caused by noise in the video or a little unintentional moving of the head and/or eye and may be removed via smoothing (not investigated here). If the non-linear visualised system responds to the iris coordinate change in real time for every video frame, it is impossible to distinguish between the effects caused by noise or that caused by intentional movement (see Figure 4.17 in Chapter 4). To solve the problem we have to define what is intentional and what is not. If fixation is used instead of producing the non-linear effect in real time while the eye is moving this problem can be minimised if not totally eliminated. Some certain amount of fixation time can be used to indicate user intention. Therefore the system has to wait for a certain amount of time (or a number of video frames) to respond to the point the user is looking at. The iris coordinates have to stay at approximately the same location for some time to be able to indicate the location on screen the user is really interested in.

Considering processing time, there is a reduction in frame rate during the processing of non-linear visualisation from 25 fps to about 20 fps. This is for processing the grid lines and not a real image. The processing required will be much higher for images and maps because every pixel in the image has to be processed but the processing for these image manipulations could be offloaded onto another CPU.

5.6 Summary

In this chapter, work on non-linear visualisation has been discussed. There are many non-linear visualisation techniques with different visualisation effects and goals. We have carried out some preliminary testing on integration of the fisheye technique with

our iris tracking system. The visualised effect was performed on grid lines. The experiments showed that there are some problems that need to be resolved in future research such as jittering of the eye that causes the focus of attention to change and the non-linear effect to move around. However the technique appears viable but needs higher speed processing to deal with real images and maps at high frame rates.

Chapter 6 Conclusion

We have investigated a computer vision based eye tracking system for the application of Human Computer Interaction (HCI). The research focuses mainly on developing an efficient iris tracking method to enable interaction in real time. The system uses a simple webcam camera to track one of the user's eyes. A colour based object tracking technique called Camshift has been successfully applied to iris tracking. Although Camshift was originally intended for object, face or hand tracking applications with some modification it is now able to track a much smaller and more dynamic object such as the human iris. The major modifications that contribute to the success of the iris tracker are the use of two channels, Hue and Saturation, instead of only the Hue channel, and the changing of the search window resizing technique. Adding the Saturation channel to the colour distribution model instead of using only the Hue channel, enabled the performance of the Camshift algorithm for iris tracking to improve significantly. By defining a fixed search window we solved the problem that the window ends up covering some unwanted areas such as the eyebrow. This would affect the accuracy of the mapping process because it requires the centroid of the iris to be accurately computed.

As we aimed to use our tracker for the application of HCI, it required a process for coordinate mapping and calibration. One of the major issues we faced is how to deal with the instability of the head as well as of the eye itself. Since we do not intend to use any extra devices to control the head, we need to have some reference points on the face to detect the head displacement. We have proposed a technique to use the whole eye (sclera plus iris) as a reference point. We use the same technique we used for iris to find the centroid of the whole eye. In fact much of the code is reused. We just created another histogram table for Camshift for tracking the sclera. The summation of the area of sclera and the iris should stay the same no matter which direction the subject is looking at. The result showed that the centroid of the whole eye also moved in the same direction as the iris. We have found that the area of the iris has great influence on the accurate detection of the centroid of the whole eye. The reason is because the sclera is mostly white, that can be easily influence by light and

shadow, and hence is difficult to track. In addition, when we look down the eyelid will cover the eye therefore the area of the eye will be smaller, and when we look up the eye will open wider. Since the size of the eye changes, the centroid of the whole eye also changes,. Finally since the iris doesn't change its shape and visibility much, it will always be captured better than the sclera and influence the outcome no matter where the camera is located.

To determine the screen location that the user is looking at, we investigated the function that maps the coordinates of the iris in the video to coordinates of the computer screen. To be able to do the coordinate mapping, first we needed to perform calibration. The calibration process prepares the system for a new environment (e.g. change of camera position). The calibration process involves some adjustment to set up the system for proper operation. For our eye tracking system the calibration involves finding locations of the eye in the video stream that are associated with locations on the screen. These associated locations will be used to determine where the subject is looking when doing the video to screen mapping process. After calibration the system can tell where on the screen the user is looking, if we make assumptions about the linearity of the eye tracking process or have some model to fit the pattern of eye coordinates while it moves across the screen. In the calibration process, points are displayed on the computer screen that the subject looks at. While the eye tracking software is running, it records the coordinates of the centroid of the iris for each observed point. In this research, experiments have been carried out with two calibration techniques. The first is four-point calibration, and the second is one-point calibration. The four-point technique uses four marked points on the screen for the user to observe. In the mapping process, a centroid of the iris from each video frame will be processed to find the screen coordinate using a linear scaling process. The one-point technique presents a novel calibration technique that requires only one point, by using *a priori* eye model. From the user's point of view the less overhead or time devoted to this task the better. Therefore reducing the number of calibration points from 4 to 1 will make the system more user-friendly. Moreover, the results show that the one-point technique is also more accurate than the standard multi-point counterpart.

In the last part of the research, we investigate the possibility of using eye tracking together with a non-linear visualisation technique to navigate a large data space. Non-linear visualisation techniques such as fisheye projection have been used to overcome limitations in representing large amounts of data in a small display area such as computer screen. The eye tracking system in this thesis is used as a means for the user to indicate a region of interest in the data space. Information within the user's area of interest will then be shown at a higher resolution, revealing more detail while representation of the other areas will be compressed. The experiments show that the non-linear effect appears jerky even when the subject is keeping his eye still. We have found that the jerking can be easily caused by noise in the video or a little unintentional moving of the head and/or eye. Considering processing time, more computation is needed for images and maps because every pixel in the image or map has to be examined and many of them have to be processed.

References

- Antoszczyszyn, P. M., Hannah, J. M. and Grant, P. M., (1998), Tracking of the motion of important facial features in model-based coding, *Signal Processing*, vol. 66, no. 2, pp. 249–260.
- Baluja, S. and Pomerleau, D., (1994), *Non-intrusive gaze tracking using artificial neural networks*, Technical Report CMU-CS-94-102, Carnegie Mellon University.
- Bebis, G. and Fujimura, K., (2000), An eigenspace approach to eye-gaze estimation, *ISCA 13th International Conference on Parallel and Distributed Computing Systems*, Las Vegas, Nevada USA, August 8-10.
- Betke, M. and Kawai, J., (1999), Gaze detection via self-organizing gray-scale units, *International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, Corfu, Greece, pp. 70-76.
- Bores, L. D., (2002), *Ocular anatomy - extraocular*, *The Nexus*: http://www.e-sunbear.com/anatomy_01.html [last accessed: 8-Feb-2004].
- Bradski, G. R., (1998), *Computer vision face tracking for use in a perceptual user interface*, Microcomputer Research Lab, Intel Corporation, Santa Clara, CA, USA.
- Carpenter, R. H. S., (1977), *Movements of the eye*, 2nd. ed. Pion, London.
- Dai, Y. and Nakano, Y., (1996), Face-texture model based on SGLD and its application in face detection in a colour scene, *Pattern Recognition*, vol. 29, no. 6, pp. 1007-1017.
- Deng, J. Y. and Lai, F. P., (1997), Region-based template deformation and masking for eye-feature extraction and description, *Pattern Recognition*, vol. 30, no. 3, pp. 403-419.
- Derby, U., (2002), *Types of eye movements*, *Institute of Behavioural Sciences*, University of Derby, <http://ibs.derby.ac.uk/gallery/types.shtml> [last accessed: 25-June-2003].
- Dubey, P., West, G. and Loh, A., (2003), Eye tracking as a computer interface, *Mechatronics and Machine Vision in Practice*, Perth, Western Australia, December 9-11, pp. 221-228.
- Duchowski, A. and Vertegaal, R., (2000), Eye-based interaction in graphical systems: theory & practice, Course 5, *SIGGRAPH 2000*, www.vr.clemson.edu/eyetracking/sigcourse/notes/c05-TOC.pdf [last accessed: 20-June-2004].
- EyeTech, (2002), *Quick glance*, EyeTech Digital Systems, Inc., <http://www.eyetechds.com/> [last accessed: 30-June-2003].

- Feng, G. C. and Yuen, P. C., (1998), Variance projection function and its application to eye detection for human face recognition, *Pattern Recognition Letters*, vol.19, no. 9, pp. 899–906.
- Fukunaga, K., (1990), *Introduction to statistical pattern recognition*, Academic Press.
- Furnas, G. W., (1986), Generalised fisheye views, *ACM SIGCHI Conference on Human Factors in Computing Systems*, Boston, Massachusetts, USA, April, 13-17, pp. 16-23.
- Gemmell, J., Toyama, K., Zitnick, C. L., Kang, T. and Seitz, S., (2000), Gaze awareness for video-conferencing: A software approach. *IEEE MultiMedia*. vol. 7, pp. 26-35.
- Gonzalez, R. C. and E. Woods, R., (2001), *Digital image processing*, Prentice Hall, 793 pp.
- Harman, P. V., (1996), Autostereoscopic display system, *SPIE Proceedings*, vol.2653, pp. 56-64.
- Heinzmann, K. and Zelinsky, A., (1998), 3-d facial pose and gaze point estimation using a robust real-time tracking paradigm, *International Conference on Automatic Face and Gesture Recognition*, April, pp. 142-147.
- Herman, I., Melanon, G. and Marshall, M. S., (2000), Graph visualisation and navigation in information visualisation: A survey, *IEEE Transactions on Visualisation and Computer Graphics*, vol.6, no. 1, pp. 24-43.
- Hyrskykari, A., (1997), Gaze Control as an Input Device, *Proceedings of ACHI'97*, Tampere, Finland, pp. 22-27.
- Intel, (2003), *Intel's OpenCV*, Intel Corp., www.intel.com/research/mrl/research/opencv/ [last accessed: 31-Jan-2004].
- IST, (2002), *iEye*, *Information Society Technologies (IST)*, <http://www.cs.uta.fi/hci/ieye/> [last accessed: 29-June-2002].
- Istance, H. and Howarth, P., (1995), *IDRG eye-tracking project*, De Montfort, <http://www.cms.dmu.ac.uk/Research/IDRG/ET/> [last accessed: 30-June-2002].
- Jacob, R. J. K., (1993), Eye movement-based human-computer interaction techniques: Toward non-command interfaces, In: *Advances in Human-Computer Interaction*, Ed. J. Nielsen, H. R. Hartson & D. Hix, Ablex Publishing Co., Norwood, N.J., pp. 151-190.
- Jacob, R. J. K., (1995), Eye tracking in advanced interface design, In: *Virtual Environments and Advanced Interface Design*, Ed. T. A. Furness, Oxford University Press, New York, pp. 258-288.

- Jeng, S. H., Yao, H. Y. M., Han, C. C., Chern, M. Y. and Liu, Y. T., (1998), Facial feature detection using geometrical face model: An efficient approach, *Pattern Recognition*, vol.31, no. 3, pp. 273-282.
- Ji, Q. and Zhu, Z., (2002), Eye and gaze tracking for interactive graphic display, *Symposium on Smart Graphics*, Hawthorne, NY, USA., June 11-13.
- Kapoor, A. and Picard, R. W., (2002), Real-time, fully automatic upper facial feature tracking, *5th International Conference on Automatic Face and Gesture Recognition*, May, Washington DC, pp. 10-15.
- Keahey, T. A. and Robertson, E. L., (1996), Techniques for non-linear magnification transformations, *IEEE Conference on Information Visualisation*, San Francisco, CA, 28-29 October, pp. 38-45.
- Kim, J., Park, K. R., Lee, J. J. and LeClair, S. R., (2000), Intelligent process control via gaze detection technology, *Engineering Applications of Artificial Intelligence*, vol.13, no., pp. 577-587.
- Kording, K. P., Kayser, C., Betsch, B. Y. and Konig, P., (2001), Non-contact eye-tracking on cats, *Journal of Neuroscience Methods*, vol. 110, no. 1-2, pp. 103-111.
- Lam, K. M. and Yan, H., (1996), An improved method for locating and extracting the eye in human face images, *International Conference on Pattern Recognition*, Vienna, Austria, pp. 411-415.
- Lamping, J., Rao, R. and Pirolli, P., (1995), A focus+context technique based on hyperbolic geometry for visualizing large hierarchies, *CHI '95: ACM Conference on Human Factors in Computing Systems*, New York, USA, pp. 401-408.
- Leung, T. K., Burl, M. C. and Perona, P., (1995), Finding faces in cluttered scenes using random labeled graph matching, *International Conference on Computer Vision*, Cambridge, Massachusetts, June, pp. 637-644.
- Leung, Y. K. and Apperley., M., (1994), A review and taxonomy of distortion-oriented presentation techniques., *ACM Transactions on Computer-Human Interaction*, vol.1, no. 2, pp. 126-160.
- Li, Y., Qi, X. L. and Wang, Y. J., (2001), Eye detection by using fuzzy template matching and feature-parameter-based judgement, *Pattern Recognition Letters*, vol.22, no. 10, pp. 1111-1124.
- Malmivuo, J. and Plonsey, R., (2002), *The electric signals originating in the eye, bioelectromagnetism*, <http://butler.cc.tut.fi/~malmivuo/bem/bembook/28/28.htm> [last accessed: 22-June-2003].
- Microguide, (2003), *Microguide BIRO system*, <http://www.eyemove.com/Biro.htm> [last accessed: 8-Feb-2004].

- Moghaddam, B. and Pentland, A., (1995), A subspace method for maximum-likelihood target detection, *International Conference on Image Processing*, Washington DC, vol. 3, pp. 3512-3515.
- Morimoto, C., Koons, D., Amir, A. and Flickner, M., (1998), Real-time detection of eyes and faces, *Workshop on Perceptual User Interfaces*, San Francisco, November, pp. 117-120.
- Nikolaidis, A. and Pitas, I., (2000), Facial feature extraction and pose determination, *Pattern Recognition*, vol. 33, no. 11, pp. 1783-1791.
- Pomplun, M., Velichkovsky, B. and Ritter, H., (1994), An artificial neural network for high precision eye movement tracking, *Advances in Artificial Intelligence. 18th German Annual Conference on Artificial Intelligence*, Berlin, Germany, pp. 63-69.
- Poynter, I., (2000), *Eye tracking online news*, Poynter Institute, <http://www.poynter.org/eyetrack2000/> [last accessed: 17-June-2003].
- Rikert, T. D. and Jones, M. J., (1998), Gaze estimation using morphable models, *International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, April, pp. 436-441.
- Robertson, G. G., Mackinlay, J. D. and Card, S. K., (1991), The perspective wall: Detail and context smoothly integrated, *ACM SIGCHI Conference on Human Factors in Computing Systems*, New Orleans, USA, April, pp. 174-179.
- Rowley, H. A., Baluja, S. and Kanade, T., (1998), Neural Network-Based Face Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.20, no. 1, pp. 23-38.
- Saber, E. and Tekalp, A. M., (1996), Face detection and facial feature extraction using colour, shape and symmetry-based cost functions, *International Conference on Pattern Recognition*, Vienna, Austria, pp. 654-658.
- Sarkar, M. and Brown, M. H., (1992), Graphical fisheye views of graphs, *ACM Conference on Human Factors in Computing Systems*, Monterey, CA, USA, May, pp. 83-91.
- Schaller, N. C., (2003), *Colour conversion algorithms*, http://www.cs.rit.edu/~ncs/colour/t_convert.html [last accessed: 20-Feb-2004].
- Schiele, B. and Waibel, A., (1995), Gaze tracking based on face-colour, *International Workshop on Automatic Face and Gesture Recognition*, Zurich, pp. 344-349.
- SeeReal Technologies, (2003), *SeeReal Technologies*, www.seereal.com [last accessed: 1-Jan-2003].

- Sobottka, K. and Pitas, I., (1998), A novel method for automatic face segmentation, facial feature extraction and tracking, *Signal Processing: Image Communication*, vol.12, no. 3, pp. 263-281.
- Stiefelhagen, R., Yang, J. and Waibel, A., (1996), A model-based gaze tracking system, *International Joint Symposia on Intelligence and Systems*, Rockville, MD, November, pp. 304-310.
- Stiefelhagen, R., Yang, J. and Waibel, A., (1997), Tracking eyes and monitoring eye gaze, *Proceedings of the Workshop on Perceptual User Interfaces*, Alberta, Canada, pp. 98-100.
- Talmi, K. and Liu, J., (1999), Eye and gaze tracking for visually controlled interactive stereoscopic displays, *Signal Processing: Image Communication*, vol.14, no. 10, pp. 799-810.
- Tankus, A., Yeshurun, Y. and Intrator, N., (1997), Face detection by convexity estimation, *Pattern Recognition Letters*, vol.18, no. 9, pp. 913-922.
- Took, D. and Craw, I., (1996), Tracking and measuring drivers' eyes, *Image and Vision Computing*, vol.14, no. 8, pp. 541-547.
- Vertegaal, R., (1999), The GAZE groupware system: Mediating joint attention in multi-party communication and collaboration, *Proceedings of CHI'99*, Pittsburgh, pp. 294-301.
- Vertegaal, R., (2002), *GAZE groupware system*, Queen's University, <http://www.cs.queensu.ca/~roel/gaze/home.html> [last accessed: 30-June-2003].
- WebVision, (2003), *A simple anatomy of the retina*, <http://webvision.med.utah.edu/sretina.html> [last accessed: 8-Feb-2004].
- Wedel, M. and Pieters, R., (2000), *Eye fixations on advertisements and memory for brands: A model and findings, marketing science*, The University of Florida, <http://bear.cba.ufl.edu/centers/mks/abstracts/vol19/no4/wedelpieters.html> [last accessed: 22-June-2003].
- Wei, G. and Sethi, I. K., (1999), Face detection for image annotation, *Pattern Recognition Letters*, vol.20, no. 11-13, pp. 1313-1321.
- Xie, X., Sudhakar, R. and Zhuang, H., (1994), On improving eye feature-extraction using deformable templates, *Pattern Recognition*, vol.27, no. 6, pp. 791-799.
- Xu, L. Q., Machin, D. and Sheppard, P., (1998), A novel approach to real-time non-intrusive gaze finding, *British Machine Vision Conference*, Southampton, UK, September.
- Yang, G. Z., Dempere-Marco, L., Hu, X. P. and Rowe, A., (2002), Visual search: psychophysical models and practical applications, *Image and Vision Computing*, vol.20, no. 4, pp. 291-305.

- Yang, G. Z. and Huang, T. S., (1994), Human face detection in a complex background, *Pattern Recognition*, vol.27, no. 1, pp. 53-63.
- Yow, K. C. and Cipolla, R., (1997), Feature-based human face detection, *Image and Vision Computing*, vol.15, no. 9, pp. 713-735.
- Yuille, A. L., Cohen, D. S. and Hallinan, P. W., (1989), Feature extraction from faces using deformable templates, *Computer Vision and Pattern Recognition*, pp. 104-109.

Appendix A

Paper presented at the International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Perth, Western Australia, December 2004.

Published in "Mechatronics and Machine Vision 2003: Future Trends", ed. John Billingsley, pages 221-228.

Eye Tracking as a Computer Interface

Premnath Dubey, Allan Loh & Geoff West
Department of Computing
Curtin University of Technology
Perth, Western Australia
{dubey, lohawk, geoff}@cs.curtin.edu.au

Abstract

There are a number of proposed techniques in the literature for eye tracking. One of the most important disadvantages of the current commercially available eye tracking systems is their intrusive and, in some cases, active nature. There is a need for non-intrusive and passive methods that do not require physical contact with the eye or the use of scanning systems such as lasers. In this paper we present an eye tracking system based on the object tracking technique called CAMSHIFT. This general technique can track a variety of regions using colour. It has been designed mainly for tracking hands and faces i.e. regions of similar colour different from the surroundings. The modified algorithm presented in this paper for eye tracking is shown to be very robust and effective for moving a pointer around a computer screen in real time.

Keywords:

Eye tracking, human computer interface, vision.

1. INTRODUCTION

Eye tracking technology has been applied to many different applications. Initially the disciplines of psychology and cognitive science were the groups most interested in using eye tracking technology, mainly to determine the characteristics of the human visual system and where people look for different tasks. Today, a diverse range of applications are benefiting from the technology including human computer interfaces. A number of different techniques have been proposed for eye tracking that can be broadly divided into the following categories [1]:

- Electro Oculography (EOG)
- Scleral contact lens and search coils
- Infrared Oculography (IROG)
- Vision Based or Video oculography systems (VOG)

All of the above methods are designed for medical uses and require equipment to be attached to the person or for the person to position themselves accurately relative to the apparatus. This research focuses on a vision based eye tracking technique. Vision based techniques can be divided into two broad groups: Vision Based Light Reflection and Vision Based Non Light Reflection. The former technique requires a special light source, usually infrared to illuminate the eye. The advantage of this technique is that it is much easier to locate and track the eyes. The non-reflection technique, on the other hand, requires more complex processing to achieve the same task but it is more flexible and less

intrusive. Our primary intention is to explore the non-reflection technique to determine the potential of the technique.

The vision based light reflection technique usually requires precise eye locations in order to process corneal reflection images. Many of the non-light reflection techniques estimate the gaze direction by using sub-images cropped from the area around the eyes. This means they need only the relative location of the eyes. These non-light estimation methods include: neural networks [2-4], eigen analysis based techniques (PCA, eigenspace) [5, 6] and morphable models [7]. Among these techniques, the eigen analysis based technique has been successfully applied to many areas of computer vision; especially face recognition [8]. One of its renowned qualities is robustness in the presence of variable input.

In this research, the CAMSHIFT technique has been investigated for tracking eyes. The CAMSHIFT algorithm is a colour-based tracker designed initially for face or hand tracking. Faces and hands are successfully tracked because of the mainly similar colour of the pixels in the region and the contrast with the background. CAMSHIFT is used to track either the iris or both the iris and the surrounding sclera depending on the application. Tracking both is needed to determine the orientation of the eye relative to the head and just the iris to determine the position of the iris relative to the camera. Results show that both of these applications have good enough performance under typical indoor lighting conditions and can operate in real time, typically 24 to 25 frames per second.

2. THE CAMSHIFT ALGORITHM

The Continuously Adaptive MeanShift or CAMSHIFT algorithm is an object tracking algorithm that uses the colour distribution of the object being tracked [9]. Initially a histogram of the pixels making up an object (face, eye) is generated and stored. Each location in the histogram indicated the probability of that particular pixel value occurring in the image. For tracking, all the pixels in the image are replaced by their probability values in the image. As such the image is back projected via the histogram. The result is an image where each pixel whose value matches one of those in the histogram gets a greater than zero probability. Pixels that are not represented in the histogram are given the value of zero. In a demonstration application supplied with the OpenCV library [10], the CAMSHIFT algorithm uses the hue channel, from the HSV colour system, to create the colour model in a 1D histogram table. This technique means that the usual method of amplitude thresholding is not needed to segment relevant areas in the image. For example, in Figure 1(b) a histogram of the eye has been back projected from Figure 1(a). The histogram should be as small as possible to obtain high speed because each pixel has to index into the histogram. We use hue and saturation in the histogram (see below for explanation) and a 5x5 histogram has been found to give good results. Once back projection has been carried out, then tracking can commence using the mean shift algorithm [11] which is basically a non-parametric technique for climbing a probability density gradient to find the mode of the distribution. Given an estimate of the position of the object to be tracked, pixels inside a search window are used to compute the centroid (x_c, y_c) of the pixels using the moments:

$$x_c = \frac{M_{10}}{M_{00}}$$

$$y_c = \frac{M_{01}}{M_{00}}$$

where M_{00} is the zeroth moment defined as:

$$M_{00} = \sum_x \sum_y I(x, y)$$

and the first moments M_{10} and M_{01} are defined as:

$$M_{10} = \sum_x \sum_y xI(x, y)$$

$$M_{01} = \sum_x \sum_y yI(x, y)$$

The following are steps involved in the mean shift process [9]:

1. User specifies search window size and its initial location.
2. Compute the mean location (x_c, y_c) in the search window.
3. Center the search window at the mean location computed in step 2.
4. Repeat steps 2 and 3 until convergence (or until the mean location moves less than some preset threshold).

For tracking in successive frames, a similar set of steps are performed:

1. Position the window at the last location and make it large enough to accommodate the object after it has moved (typically double the size).
2. Invoke the mean shift process as above.
3. Set the window size to be a function of the zeroth moment.
4. Repeat steps 2 and 3 until convergence.

This process will track the object of interest over a video sequence adapting the window size to keep the object centred in the window.

3. IRIS TRACKING USING CAMSHIFT

There are some major differences between face tracking and eye tracking when using CAMSHIFT. Some of these differences create problems for eye tracking that do not occur in face tracking. First of all the iris is much smaller than the face in typical images. Moreover when the subject looks down, the eyelid obscures most of the eye area that makes it more difficult to find and track. Next the reflection property of the eye's surface that has a much higher reflection than the human face in normal lighting conditions i.e. the eye is a glossy object because of the moisture whereas the face is usually dry and matt. Normal light sources and such light emitting or reflecting objects as computer monitors can affect the colour of the iris by superimposing the reflections off the surface of the eye onto the reflections off the actual iris that is below the surface of the eye. Another important difference is the nature of eye movement which is much more dynamic than that of the head or face. Taking all of those issues into consideration, a number of modifications to CAMSHIFT for the original face tracking application are proposed to handle these specific requirements.

In iris tracking we use a 2D histogram. The saturation channel has been included in addition to hue because hue on its own does not work well. Hue is excellent for face tracking because most faces in the world have similar hue, and the hue doesn't change much across a face [9]. For instance to track a subject's iris that is dark in colour, the range of hue values from the dark iris, are mostly within the blue colour band, and has a wide overlap with the hue values of the white eye area (sclera) as well as some of the lighter areas of the face. By including saturation, the performance on iris tracking has improved significantly. Figure 1 shows the back-projection pictures of the iris using hue only and using hue and saturation for the face image.

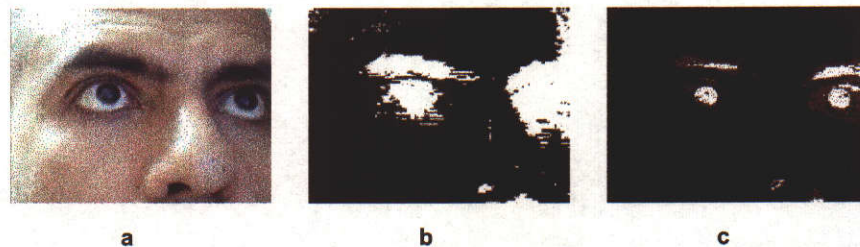


Figure 1: a) original video frame from camera, b) hue only back-projected picture and c) hue and saturation back-projected picture.

To get the results of figure 1, the histogram of an iris is generated and used for subsequent detection of the irises in subsequent images. Back projecting highlights the pixels whose values are present in the histogram with intensity values representing the probability of that pixel value occurring. This is easily seen in Figure 1(c) where the backprojected pixels have different shades with the highest probability ones associated with the irises, and to some extent with the eyebrows. This good segmentation is because of the use of hue and saturation. Hue on its own results in the poor segmentation in Figure 1(b). Although currently our system works only with dark irises, it is possible for the system to have a set of pre-defined eye colour models for people with differing eye colour. Users will be allowed to choose the model that suits them. These models can be created from colour sample taken from pictures of irises representing each group off line.

To find the initial location to start tracking, a simple template matching method has been used. An iris image is used as a template for searching for an iris in the first video frame. This process is done only once at the start of tracking. An alternative method is to get the user to position their eye at a particular location in front of the camera guided by the camera image on a computer monitor. This is

more difficult than it seems because you have to look at the monitor and move your head around to line up the eye. The searching method is less intrusive and does not take much time.

Instead of allowing the search window to change to any size as in CAMSHIFT, it is better to define a maximum window size to suit our application. With 320×240 video resolution and our specific camera setting, the size of the iris in the video frame will have a diameter of around 30 pixels. Hence the maximum search window size is set to be 35×35 pixels in size. This helps the system ignore unwanted areas in the window.

We have also tried tracking the whole eye, that is the iris and sclera (white of the eye) together. The reason for this is we would like to discriminate between movement of the iris because of head movement and because of eye rotation. By tracking the whole of the eye using the sclera we could use the centroid of the eye as a reference point on the face. To track the whole eye we have to be able to track the sclera together with the iris. To do that we used another colour model for the sclera to create a back-projected image of the sclera that we merge into the back-projected image of the iris by taking the xor of the two images pixel by pixel. The result showed that the sclera has less uniqueness than the iris. The white colour of the iris has a lot of similarity to the colour of light reflected from the face. The reflection on the forehead skin and nose can cause the tracker to lose the eye. Moreover the back-projected picture of the sclera showed that not all the sclera area had been mapped to the picture. Most of the missing parts are at the corners of the eyes (see Figure 2(b)). Another unexpected result is the centroid of the whole eye seems to follow the iris when the iris moves. The centroid did not stay as still as we thought it would and the results are not presented here.



Figure 2: a) image from video, b) back-projection of the whole eye.

4. RESULTS

The computer used for this system is a 1Ghz PC computer with the Microsoft Windows™ operating system. A simple webcam is used for the video source. The webcam supplies 320×240 pixel images at 30 frames per second for input to the system. There is no special lighting setup required. A single 40 watt fluorescent light from above is a typical lighting scenario. The webcam should be placed directly in the middle of the screen but this is impractical. We have chosen to place it below the eye line on the left of the monitor looking upwards (see Figure 1 – note the image is a mirror image to help user interpretation). This gives the best view of the eyes because the eyebrows aren't in the way of the eyes. Putting the webcam directly under the monitor is more ideal but this produces an extreme view of the eyes from below.

The system was tested with a subject with a dark iris colour. It has proved to be highly accurate when tracking the iris in real time. It failed to track only when the subject moves their head rapidly. This result is based on normal office lighting conditions. The test on environment lighting has shown that the system is robust enough to endure a certain amount of lighting difference. In the environment that too dim or too bright, the tracker will be more likely to lose the iris. It has been found that most of the time the system loses track because of the influence of the eye brow. The eye brow appears to have very similar hue and saturation value to the iris. Figure 1(c) of the back-projection provides clear evidence of this. The center of mass (centroid) of the iris can be used to tell the exact location. We have found that while the subject keeps their eye and head still (there is no head clamp used so we can not say that it is perfectly still), the centroid moves +/- 1 to 2 pixels between frames.

POSITION	X		Y	
	MEAN	STDEV	MEAN	STDEV
Top-Left	161.1	0.56	94.6	0.69
Centre Middle	194.1	0.56	105.4	0.69
Bottom-Right	216.2	1.03	116.2	0.63

Table 1: Statistics for viewing three points on the screen.

Table 1 shows results for the x and y coordinates of the eye's centroid for the user looking at three points on the screen; the top-left, the center middle and the bottom right. Each test was performed ten times for each point – and the statistical results show very little variation in the x, y coordinates given there would be some eye movement and error due to noise in the camera, lighting and other parts of the system.

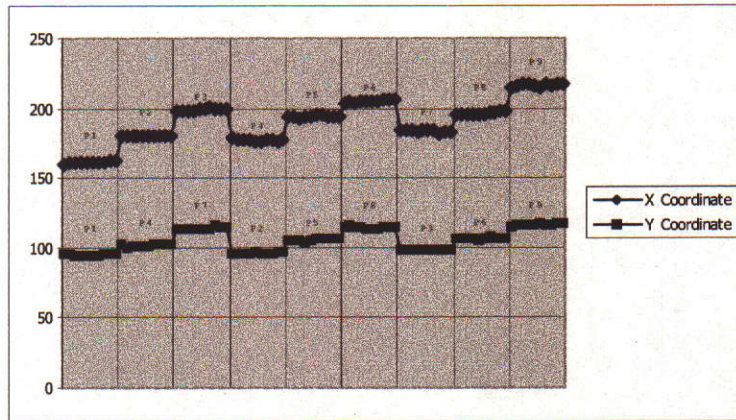


Figure 3: Show the value of x (a) and y (b) coordinate when subject look at nine points on the screen.

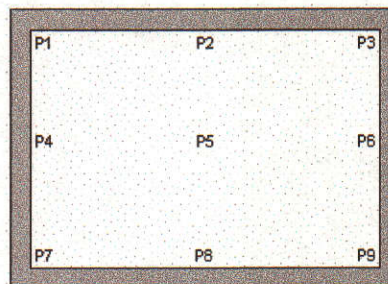


Figure 4: The nine points used for Figure 3.

Figure 3 displays the value of the x (a) and y (b) coordinates when the subject looks at a calibration screen of nine points arranged in a grid on the screen as shown in Figure 4. At each point ten observations were recorded. The graph shows that as the subject scans the screen from left-to-right or top-down, the eye coordinate values increase in a consistent way. Note that groups of 30 measurements represent the changes in measurement along a row or down a column. Although the eye's movement within the eye socket occurs in three dimensions, over the short range of movement within the monitor area, in our experiment, the graphs appear to be almost linear for both of the x and y values. Assuming the eye movement follows a linear pattern across the screen, it is possible to apply a direct mapping from the 2D coordinate of eye centroid location from the video to 2D screen coordinates. This assumption means that determining where the user is looking on the screen can occur using a simpler 2D transformation, and not the more complicated 3D eye coordinate space to 2D screen space transform.

5. CONCLUSION

This paper has presented a real time eye tracking algorithm for manipulation of a pointing device on a computer monitor. It uses a modified version of the CAMSHIFT algorithm for the eye tracking which works at between 24 and 25 frames per second. Numerous modifications have been investigated and presented along with results for tracking the iris. Tracking the iris and sclera is possible but the results






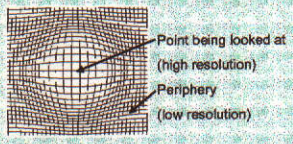






are not as good as for tracking the iris on its own. Tracking both would allow the orientation of the eye to be determined (comparing the position of the iris and the iris and sclera combined) so that head movement could be ignored and just eye orientation used. This is the subject of further investigation. The technique has proved to be very robust for tracking the iris and produces repeatable measurements at low error. The technique is being used to allow the interactive examination of maps and images in which the map is enlarged in the area being looked at to enhance the interaction. It is also being used to replace a mouse and in such cases, the absolute position of the iris can be replaced by changes in position to drive a mouse pointer. For both of these applications, experiments with users will show the most useful methods to use.

References

1. Young, L.R. and D. Sheena, *Survey of Eye Movement Recording Methods*. Behavior Research Methods & Instrumentation, 1975. 7(5): p. 397-439.
2. Baluja, S. and D. Pomerleau, *Non-Intrusive Gaze Tracking Using Artificial Neural Networks*. 1994, Carnegie Mellon University.
3. Betke, M. and J. Kawai. *Gaze Detection Via Self-Organizing Gray-Scale Units*. in *International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*. 1999.
4. Pomplun, M., B. Velichkovsky, and H. Ritter. *An artificial neural network for high precision eye movement tracking*. in *Advances in Artificial Intelligence. 18th German Annual Conference on Artificial Intelligence*. 1994. Berlin, Germany: Springer-Verlag.
5. Bebis, G. and K. Fujimura. *An Eigenspace Approach to Eye-Gaze Estimation*. in *ISCA 13th International Conference On Parallel And Distributed Computing Systems*. 2000. Las Vegas, Nevada USA.
6. Talmi, K. and J. Liu, *Eye and gaze tracking for visually controlled interactive stereoscopic displays*. Signal Processing: Image Communication, 1999. 14(10): p. 799-810.
7. Rikert, T.D. and M.J. Jones. *Gaze Estimation Using Morphable Models*. in *International Conference on Automatic Face and Gesture Recognition*. 1998.
8. Turk, M.A. and A.P. Pentland. *Face recognition using eigenfaces*. in *IEEE Conference on Computer Vision and Pattern Recognition*. 1991. Hawaii.
9. Bradski, G.R., *Computer Vision Face Tracking For Use in a Perceptual User Interface*. 1998, Microcomputer Research Lab, Intel Corporation: Santa Clara, CA, USA.
10. Intel, *Intel's OpenCV*. 2003, Intel(www.intel.com/research/mrl/research/opencv/).
11. Fukunaga, K., *Introduction to Statistical Pattern Recognition*, ed. W. Rheinboldt. 1990: Academic Press. 591.

Appendix B

Slides for the paper presented at the International Conference on Mechatronics and Machine Vision in Practice (M2VIP), Perth, Western Australia, December 2004

 <h2 style="text-align: center;">Eye Tracking as a Computer Interface</h2> <p style="text-align: center;">Premnath Dubey, Allan Loh & Geoff West Department of Computing Curtin University Perth, WA {dubey,lohaw,geoff}@cs.curtin.edu.au</p> <p style="font-size: small;">6/9/2004 1</p>	 <h2 style="text-align: center;">Motivation</h2> <ul style="list-style-type: none"> • Need for different interfaces for different people: <ul style="list-style-type: none"> - Handicapped and disabled • Need to be non-intrusive and passive • Needs to be fast and accurate • Easily set up and cheap • Easily adapted for different people (training) • Student's disability • Foveation on areas of interest <p style="font-size: small;">6/9/2004 2</p>
  <p style="font-size: small;">6/9/2004 3</p>	 <h2 style="text-align: center;">Non-linear Magnification and Foveation</h2>  <p style="font-size: small;">6/9/2004 4</p>
 <h2 style="text-align: center;">Background</h2> <ul style="list-style-type: none"> • Eye trackers: <ul style="list-style-type: none"> - Electro oculography - Scleral contact lens and search coils - Infrared oculography - Vision based or video oculography <ul style="list-style-type: none"> • Light reflection using special illumination • Passive light reflection • Mainly for medical uses • Intrusive and restrictive  <p style="font-size: small;">6/9/2004 5</p>	 <h2 style="text-align: center;">Solution</h2> <ul style="list-style-type: none"> • Use of the CAMSHIFT algorithm - Continuously Adaptive Meanshift algorithm <ul style="list-style-type: none"> - Colour based region tracker: <ul style="list-style-type: none"> • Originally for face or hand tracking • Reasonably stable colours different to the background • Part of the OpenCV software library • Tracking eyes: <ul style="list-style-type: none"> - Iris (absolute position) - Iris and sclera (rotation relative to head) - Passive: using normal illumination <p style="font-size: small;">6/9/2004 6</p>
 <h2 style="text-align: center;">Moments</h2> <p>Zeroth order moment: $M_{00} = \sum_x \sum_y I(x, y)$</p> <p>First order moments: $M_{10} = \sum_x \sum_y x I(x, y)$ $M_{01} = \sum_x \sum_y y I(x, y)$</p> <p>Centroid: $x_c = \frac{M_{10}}{M_{00}}$ $y_c = \frac{M_{01}}{M_{00}}$</p>  <p style="font-size: small;">6/9/2004 7</p>	 <h2 style="text-align: center;">CAMSHIFT - Tracking</h2> <ul style="list-style-type: none"> • Initialisation (mean shift process): <ol style="list-style-type: none"> 1/ User specifies search window size and location 2/ Compute mean (x_c, y_c) in the search window 3/ Centre window at mean (x_c, y_c) 4/ Repeat steps 2 and 3 until mean stabilises • Tracking: <ol style="list-style-type: none"> 1/ Position window at last location and make double the size 2/ Perform the mean shift process 3/ Set window size as a function of zeroth moment 4/ Repeat steps 2 and 3 until convergence <p style="font-size: small;">6/9/2004 8</p>

Mean Shift in Two Dimensions

Window

6/9/2004 9

Tracking in Two Dimensions

Position (old)

Position (new)

Window

6/9/2004 10

Histogram Projection

- For a given region, build histogram of pixels - store
- For a new image, project pixels through histogram to form a new image

1	2	4
3	6	5
4	7	5

Count

Pixel value

10	20	10
25	0	0
10	0	0

6/9/2004 11

Parameters

- Hue and saturation histogram: 5x5 bins
- Image Size: 320x240 pixels
- Maximum search window: 35x35 pixels
- Iris diameter: 30 pixels

6/9/2004 12

Iris or Iris and Sclera?

Iris only:

Absolute coordinates: (x_i, y_i)

Iris and Sclera:

Relative coordinates: $(x_s - x_i, y_s - y_i)$

Note: (x_s, y_s) computed from XOR of iris and sclera regions

6/9/2004 13

Histogram Projection - Example

Original image

Backprojected (hue only)

Backprojected (hue and saturation)

6/9/2004 14

Iris and Sclera Backprojection

Original image

Backprojection of iris and sclera

6/9/2004 15

Implementation

- Windows 2000
- Visual C++
- OpenCV library (Intel)
- Intel webcam (320x240 pixels at 30 fps)
- 40 watt fluorescent lighting

6/9/2004 16

Iris Tracking with CAMSHIFT

6/9/2004 17

As Seen by the Camera

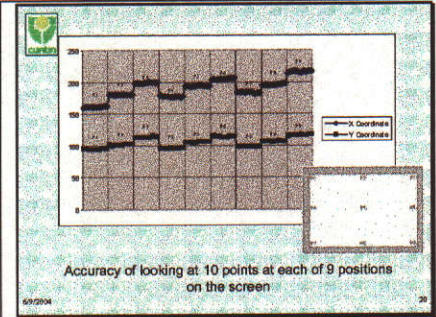
6/9/2004 18

Fixation Accuracy

POSITION	X		Y	
	MEAN	STDEV	MEAN	STDEV
Top-Left	161.1	0.56	94.6	0.69
Centre Middle	194.1	0.56	105.4	0.69
Bottom Right	216.2	1.03	116.2	0.63

Statistics for looking at each point 10 times

6/9/2004 19



Summary

- Use of CAMSHIFT for tracking the iris
- Reasonable accuracy across the screen
- Real time operation
- Applicable to viewing images/maps at variable resolution
- Need to consider iris and sclera tracking to ignore head movement

6/9/2004 21