RECOMMENDATION REPORT FOR AN INFRARED EYE-TRACKING SYSTEM FOR P-CUBEE

Ricky Leong
Hei Wang Chan
Project Sponsor:
Sidney Fels

Applied Science 479
Engineering Physics
The University of British Columbia
January 10th, 2010

Project Number 1060

Executive Summary

There are several techniques that can be employed to track real-time eye movements. Currently, pCubee uses magnetic sensor technology and requires the user to wear a head-tracking sensor that is wired to the device itself. The objective of the project was to construct a wireless, non-invasive, low-latency infrared eye tracking system that could replace the existing solution.

The technique that was utilized in this project involved processing of images from a video-based method, which can be performed with inexpensive cameras. A two camera system was setup and calibrated successfully. Triangulation was tested with IR goggles and the accuracy of the system was determined to be less than 1%. Thus, proof of concept was demonstrated and it should not be difficult to extend the system to function with any two camera configuration, provided that the appropriate common field of view is produced.

An algorithm was developed to search for the location of the red-eye effect in a subject"s eyes. However, based on the images gathered from the camera setup, the red-eye effect was not strong enough to be detectable. A difference image technique was also employed to improve the detection of the red-eye effect, but did not yield better results.

There were many limitations or sources of error that prevented the system to be designed optimally. The investigators believe there are several improvements - such as utilizing higher quality cameras and better image processing techniques - that could be made to successfully complete the system.

Table of Contents

List of Figures

List of Tables

1. Background and Motivation

Eye tracking is the process of measuring and tracking real-time eye movements. In general, there are two main techniques to perform eye tracking: the first involves measuring the position and motion of the eye relative to the head, while the second technique involves measuring the absolute location of the eye in space. There are three methods for using the first technique: Electro-OculoGraphy (EOG), scleral contact lens/search coil, and Photo-OculoGraphy (POG) or Video-OculoGraphy (VOG) [1]. About 40 years ago, EOG was the predominant method to track eye movements, and required the user to wear a complex apparatus [1]. With the emergence of faster image processing hardware and software, scleral contact lens and POG were also used. Unfortunately, these methods are all invasive, and usually restrict movement of the head, unless there is separate head tracking performed concurrently.

For applications involving graphical and interactive displays, the second technique is more commonly used. This typically involves processing of images from a video-based combined pupil and corneal reflection method, which can be performed with inexpensive cameras. The advantage of this is that it does not necessarily require the head to be stationary [1]. It is possible to obtain very accurate measurements of eye locations by using devices that involve physical contact with the eyes. However, these techniques are invasive and typically allow much less mobility and flexibility for user interaction with the system.

In our project, the system of interest is the pCubee, an extension of a fish tank virtual reality (FTVR) system, which involves displaying dynamic 3D images based on head positioning of the viewer. pCubee contains five LCD screens arranged into five sides of a cube, so that it appears that there are 3D objects inside the cube. Based on the user‟s physical interactions with the cube, objects “inside” the cube will respond with simulated physics, and can move and bounce around [2]. pCubee currently uses magnetic sensor technology and requires the user to wear a head-tracking sensor that is wired to the device itself.

There is desire to increase the portability of the system and make it easier to use. In designing the system, it is also important to ensure that the tracking is performed at sufficiently high frame rates, and delay is not too high so as to cause jitter. Thus, the objective of this project is to construct a wireless (and non-invasive), low-latency infrared based eye tracking system for the pCubee. Provided that the system functions correctly for a single LCD screen, the system could then be extended to the five screens of the pCubee by introducing additional cameras on each side.

pCubee has many potential applications in the areas of physics-based games, perspective-based 3D entertainment, CAD and architectural design, museum displays, 3D teleconferencing, education, and much more [3]. If the pCubee is improved through increased portability and ease-of-use, it is anticipated that more of these applications may be successfully implemented.

2. Discussion

2.1 Project Objectives

1. Using the method of triangulation, calibrate two pinhole cameras to accurately determine the 3D position (X, Y, Z) of a single IR-emitting source. The calculated position of the IR-source should have errors of less than 10%.

2. Using image processing techniques, implement a software algorithm to accurately locate the eyes of a single person based on a red-eye effect that is produced from the reflection of the IR source. The algorithm should be functional for different orientations of the head.

3. Develop a real-time eye tracking system to determine the eye position at 30 frames per second. The system should predict and/or interpolate the location of the eyes on frames where the eyes cannot be detected (blinking). The lag time of the system should be less than 80ms.

2.2 Theory

2.2.1 Triangulation

Triangulation is the process of reconstructing the coordinates (X, Y, Z) of a point P from two or more image projections of P from different known viewpoints in space. Consider two pinhole cameras placed a distance B apart with parallel optical axis (along Z), as shown in Figure 1. The projection line of point P passes through P and the lens center onto the image plane of the pinhole camera. The image plane is perpendicular to the optical axis and is located at a distance f (focal length) away behind the camera.

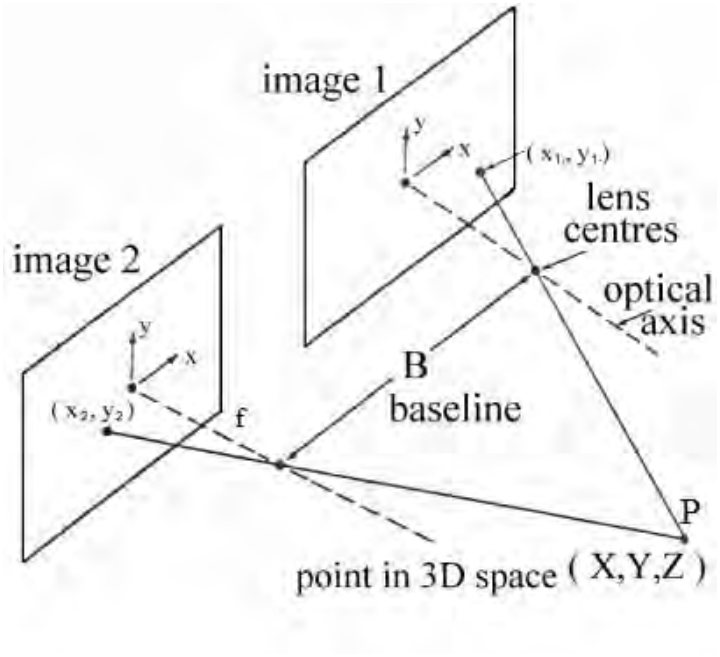


Figure 1. Stereo Camera model.

It should be noted that the resultant image on the image plane is inverted. However, since any practical camera internally reflects the imagine plane across the optical axis, it is simpler to consider the virtual image plane, located at a focal length f in front of the camera. As shown in Figure 2, the image points x1 and x2 form a triangle with P that is similar to the triangle PLR.

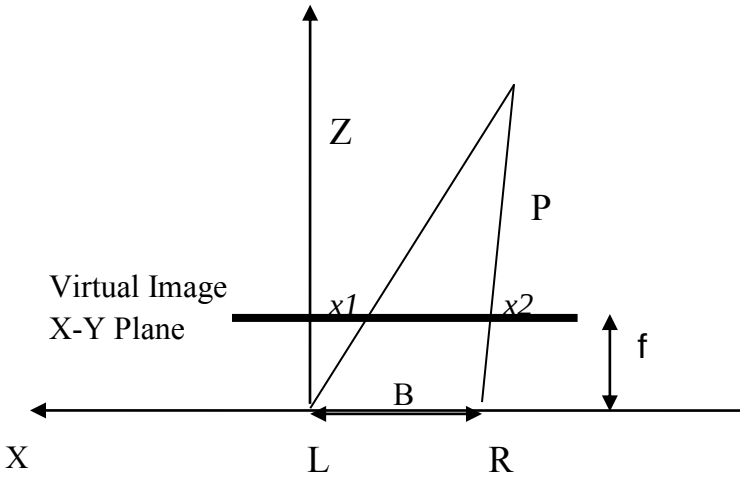


Figure 2. Illustration of Triangulation.

Theoretically, the location of P can easily be determined by equating the ratios of the edges, giving

$$Z = f - \frac{fB}{x2 - x1}$$

$$X = \frac{x_1 Z}{f}$$

$$Y = \frac{y_1 Z}{f}$$

However, in reality, the image points (*x1*,*y1*) and (*x2*,*y2*) cannot be determined precisely due to noise and measurement errors. As a result, the two projection lines do not generally intersect and hence, Z cannot be determined this way. Taubin suggests a general method of determining the location of P by minimizing the distance between the two projection lines [4]. In order to do this, it is appropriate to again consider a point P in space with object coordinates (in meters or inches)

$$P = \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix}.$$

The point P is projected onto the image planes of the camera, resulting in the image coordinates (measured in pixels)

$$P^I = \begin{pmatrix} P_u^I \\ P_v^I \end{pmatrix} .$$

If a pair of stereo camera is used, the left and right image coordinates are be denoted

$$P^L = \begin{pmatrix} P_u^L \\ P_v^L \end{pmatrix} \text{ and } P^R = \begin{pmatrix} P_u^R \\ P_v^R \end{pmatrix}$$

respectively. Note that unlike the method described previously, the image coordinates are measured in pixels and with respect to the camera itself. Hence, the image coordinate $\boldsymbol{P^L = (0, 0)^T}$ represents the top left corner of the left image, and $\boldsymbol{P^R = (0, 0)^T}$ represents the top left corner of the right image. Figure 3 shows the coordinate system for a camera located at **O.**

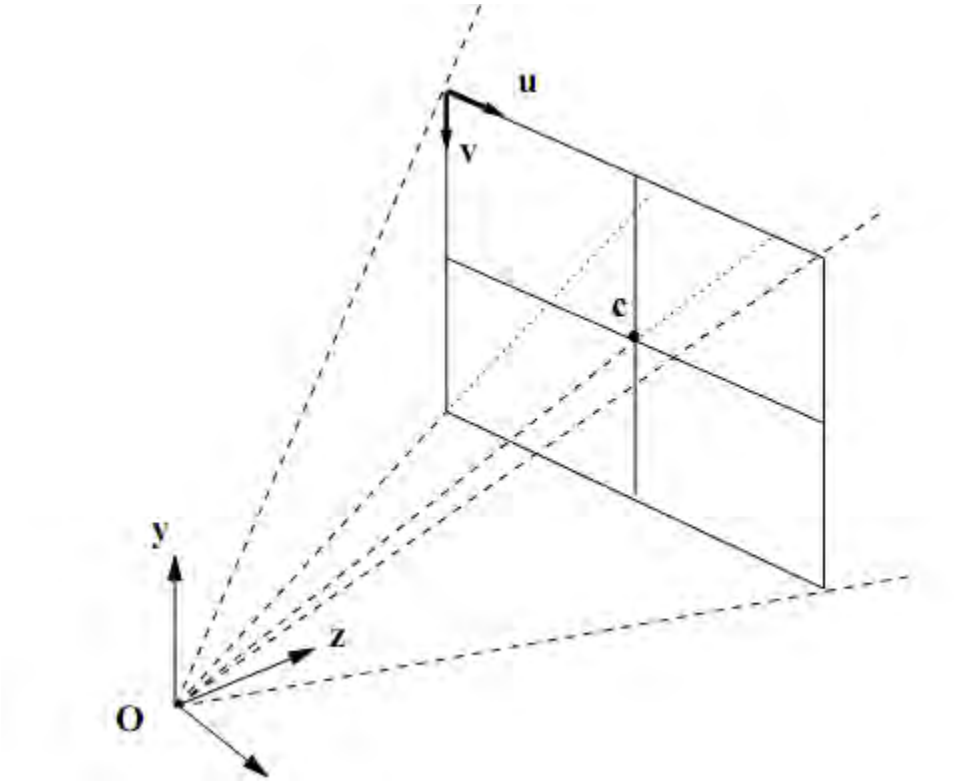


Figure 3. Coordinate system of a camera centered at O

If the radial distortion introduced by the camera lens is ignored, the point P is related to the image coordinates by the equation

$$P = \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} = \lambda \begin{pmatrix} (P_u - c_u)/f_u \\ (P_v - c_v)/f_v \\ 1 \end{pmatrix} = \lambda\omega,$$

where λ is the unknown parameter, f_u and f_v are the focal lengths of the camera (in pixels). Additionally, c_u, c_v measure the intersection of the optical axis with the image plane (in pixels), as shown in Figure 3. The ratio $|f_u / f_v|$ represents the aspect ratio of the camera. Note that for cameras that have square pixels, $|f_u| = |f_v|$.

The parametric equation of the projection line in the object coordinates is then given by

$$l^I = \{O^I + \lambda^I \omega^I : \lambda^I\}$$

Where O^I represents the object coordinates of the indexed camera. As mentioned before, the two lines l^L and l^R do not necessarily intersect due to noise and measurement errors. Thus, we look for $\lambda = (\lambda^L, \lambda^R)^T$ such that the distance between the lines,

$$|(O^L + \lambda^L \omega^L) - (O^R + \lambda^R \omega^R)|^2$$

is minimized. If we denote $\boldsymbol{B} = O^L - O^R$ and $A = [\omega^L | - \omega^R]$, the solution is simply

$$\boldsymbol{\lambda} = \binom{\lambda^L}{\lambda^R} = -(A^T A)^{-1} A^T B.$$

2.2.2 Red-eye effect

In video-based eye tracking, a camera setup is typically used to focus on given features of the eye. However, in order to locate a distinguishing characteristic of the subject"s eye, an external stimulus is often applied. If an infrared (IR) light source is shone onto the subject"s eyes, then a reflection will occur and the direction of the reflection will vary based on the orientation of the light source. When the light source is coaxial with respect to the camera, the light will reflect off the retina (directly back to the camera), and a bright red-eye effect can be observed in the pupil. If the light source is not coaxial with respect to the camera, then the light will reflect off the retina (not back to the camera), but the pupils will instead appear dark [5]. If the red-eye effect is difficult to distinguish from the other features in the image, it is also possible to take a difference image between the bright and dark eye images. In practice, the red-eye effect is often produced in photography when the location of the flash source is close to the camera lens.

The red-eye effect technique that is used in eye tracking can be applied even when there are differences in eye features – such as pupil size, pupil intensity, and presence of eyelashes - between subjects. The technique has been shown to work even with subjects of different ethnic backgrounds, age, and gender [6]. In general, the technique can be applied in most lighting conditions. However, environments with other high sources of IR lighting should be avoided because interference may occur.

2.3 Final Design

2.3.1 Cameras

In the final design, two Logitech QuickCam Messenger cameras are chosen for the application. The Logitech QuickCam Messenger connects directly to the USB port, and it has a resolution of 640x480 pixels and a frame rate of 30 frames per second. The detailed specifications for the Logitech QuickCam Messenger can be found in Appendix A. As shown in Figure 4, the two cameras are attached to the rubber stand using a glue gun, and are pinned to the wooden block using additional screws. Once the positions of the cameras are set, it is imperative that they remain fixed because any translational or rotational movement will change the extrinsic parameters of the camera, which directly affects the accuracy of triangulation.

Figure 4 (a) Two Logitech QuickCam Messengers are positioned side by side for triangulation. (b) The cameras are pinned to the wooden block using additional screws.

As noted previously, the cameras chosen for the application must be very IR-sensitive in order to detect the infrared LED source and the infrared reflection off the eye. Many of the cameras in the market today contain an internal infrared filter that removes most of the incoming infrared light and improves the image‟s colour quality [12]. Often, a special coating is sprayed onto the lens of the camera to act as the infrared filter, making it very difficult to remove from the camera. The main advantage of the Logitech QuickCam Messenger camera is that the infrared filter can be easily removed, as the infrared filter is simply a piece of glass that is placed in front of the lens. With the IR filter removed, the Logitech Messenger QuickCams are exceptionally sensitive to infrared. A 50mA infrared LED at 30cm away would cause the camera to oversaturate, as shown in Figure 5a. The gain and exposure time are adjusted on a trial and error basis to compensate for the high IR sensitivity. The gain and exposure time are chosen to be 0 dB and 1/120s respectively. As seen in Figure 5, the same IR LED with these settings do not cause the camera to saturate, but the location of the IR source can still be easily detected.

Figure 5 An infrared LED is placed 30cm from the cameras. (a) The camera saturates using the automatic exposure and gain settings. (b) The resultant image with 0 db gain and an exposure time of 1/120s.

2.3.2 Camera Calibration

In order to perform Triangulation, the Logitech QuickCam Messenger cameras must first be calibrated to determine the intrinsic and extrinsic parameters. The two cameras are calibrated using the MATLAB Camera Calibration Toolbox [7]. Typically, the intrinsic parameters (focal length, principal point, skew coefficient, etc) and the extrinsic parameters (relative positions of the cameras) can be determined by taking pictures of a checkerboard pattern at various angles and distances. Since stereo calibration requires that the two cameras capture the same scene, the checkerboard must be placed sufficiently far from the cameras such that the entire checkerboard is in the view of both cameras. A large checkerboard pattern is created to improve the accuracy of the calibration results (see Figure 6). The checkerboard pattern consists of 12 rows, with 8 squares on each row. Each square on the pattern is 63.0mm by 63.0mm. After specifying the extreme corners of the checkerboard, as well as the size and number of the squares in the checkerboard pattern, the locations of the squares are interpolated by the calibration tool, as shown in Figure 7. These interpolated locations are compared with the actual locations of the squares to determine intrinsic parameters of the camera. 24 images of the checkerboard pattern are taken from each camera at various distances and angles and used for calibration. The calibration results are shown in Section 2.6.1 below. The meaning of the calibration parameters can be found in Appendix B.

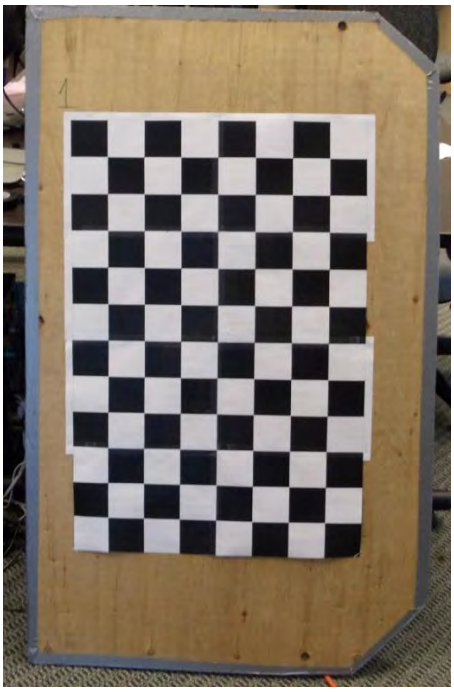


Figure 6. Checkerboard pattern used to calibrate the cameras

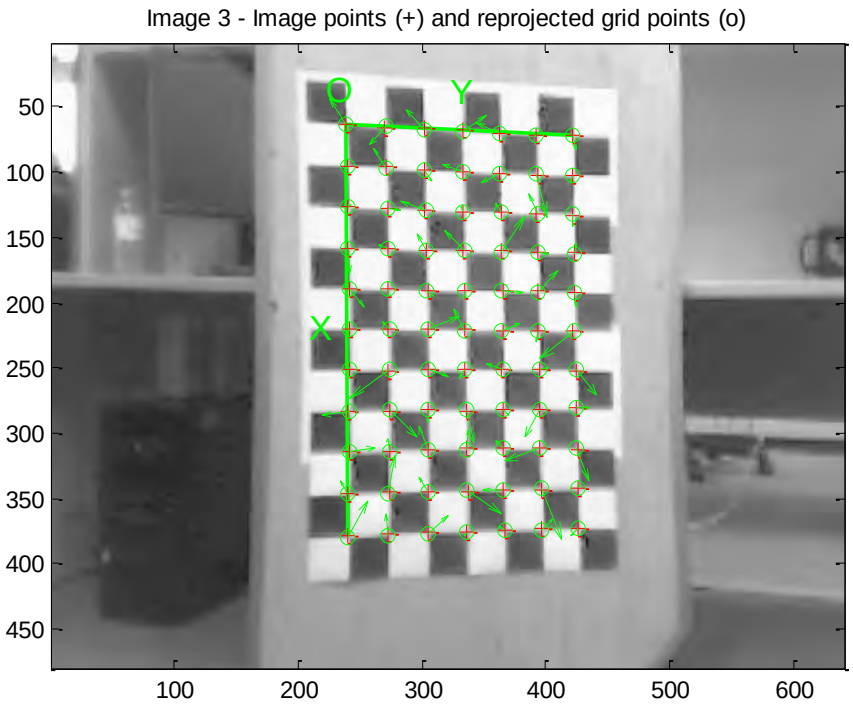


Figure 7. Camera Calibration

2.3.3 Infrared Glasses

In order to test whether the triangulation and real-time tracking system was functioning correctly, IR LED goggles were designed to emulate the IR reflection off the eyes. The objective of the goggles is to create a bright point that can be easily distinguished in an image (in ambient light conditions). Once this point is successfully identified in both camera images, the aforementioned triangulation method can be utilized to determine the accuracy of the system. By moving the goggles in space, it is also possible to estimate the operating range and angle of the tracking system. The goggles were constructed with the following components:

- Laboratory safety goggles
- One 9V battery
- Two IR LEDs
- On/off switch
- Hot glue
- Small PCB

A picture of the IR goggles is shown in Figure 8.

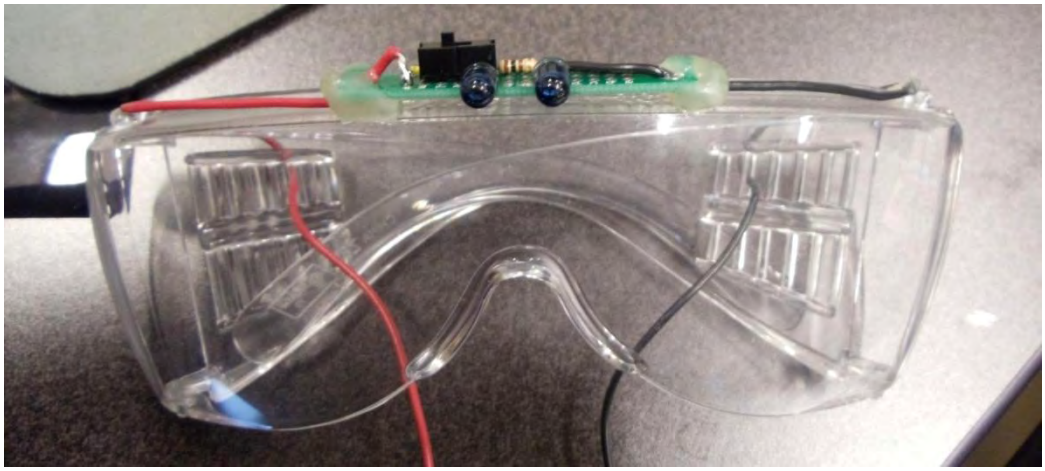


Figure 8. IR goggles

2.3.4 IR illuminator

The primary infrared light source that was utilized in the project was a bright IR illuminator lamp. A picture is shown in Figure 9. Since it is impossible to place the IR illuminator exactly coaxial to the camera, it was instead placed behind the camera. For a given location of the subject‟s eyes, the position and orientation of the illuminator is adjusted slightly, such that the light reflects off the retina and directly back to the camera, thus producing the red-eye effect.

Figure 9. IR lamp

2.3.5 IR Light Emitting Diode (LED) rings

Another source of infrared lighting that was used was the OPA80 Near Infrared Light Ring Series, which contains a set of 18 850nm IR LEDs in a ring configuration. The rings were mounted such that they were coaxial with respect to the camera lens (see Figure 10), which is important because this allows the light to reflect directly off the retina to achieve the red-eye effect. Also, unlike the IR illuminator, the IR rings are fairly secure, so that they do not need to be constantly adjusted if the subject moves closer or further away.

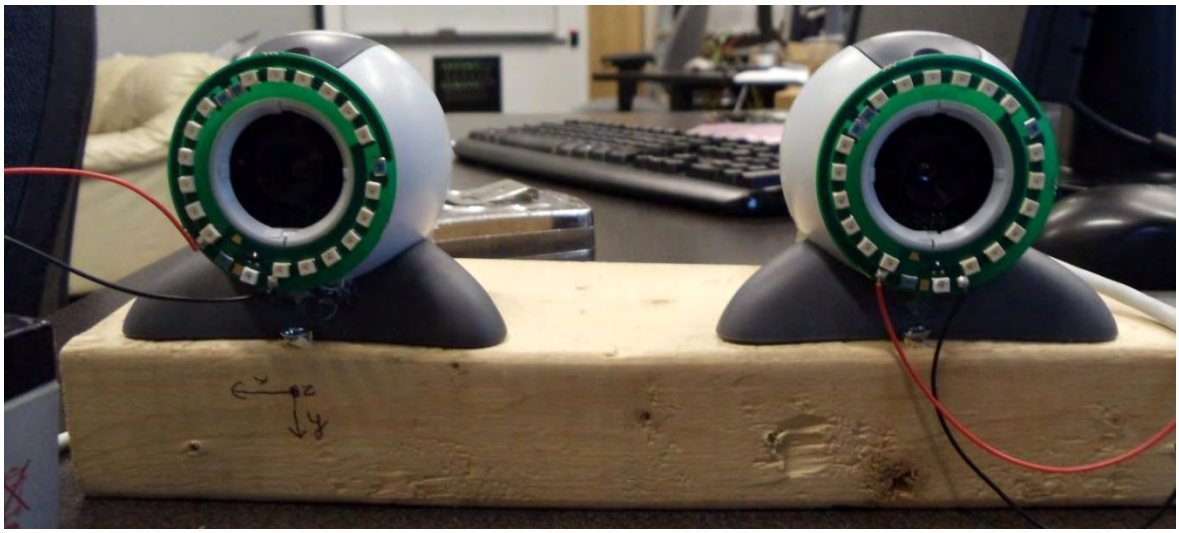


Figure 10. IR rings

2.3.6 Software Design

The software component of this project is responsible for accepting the video input feeds from the two cameras, processing the input images to determine the image coordinates of the IR source, and triangulating based on the image coordinates to determine the 3D coordinates of the IR source. A description of the software code may be found in Appendix C.

Capturing video frames

The software uses a simple video library called *videoInput* to capture from the two Logitech cameras[13]. The *videoInput* library is a simple header library that is capable of capturing from multiple devices simultaneously. It allows the user to set the frame rate and the video settings properties via a graphical user interface, as shown in **Error! Reference source not found.**. Each frame is stored in a 640x480 frame buffer that can easily be converted to openCV or OpenGL format.

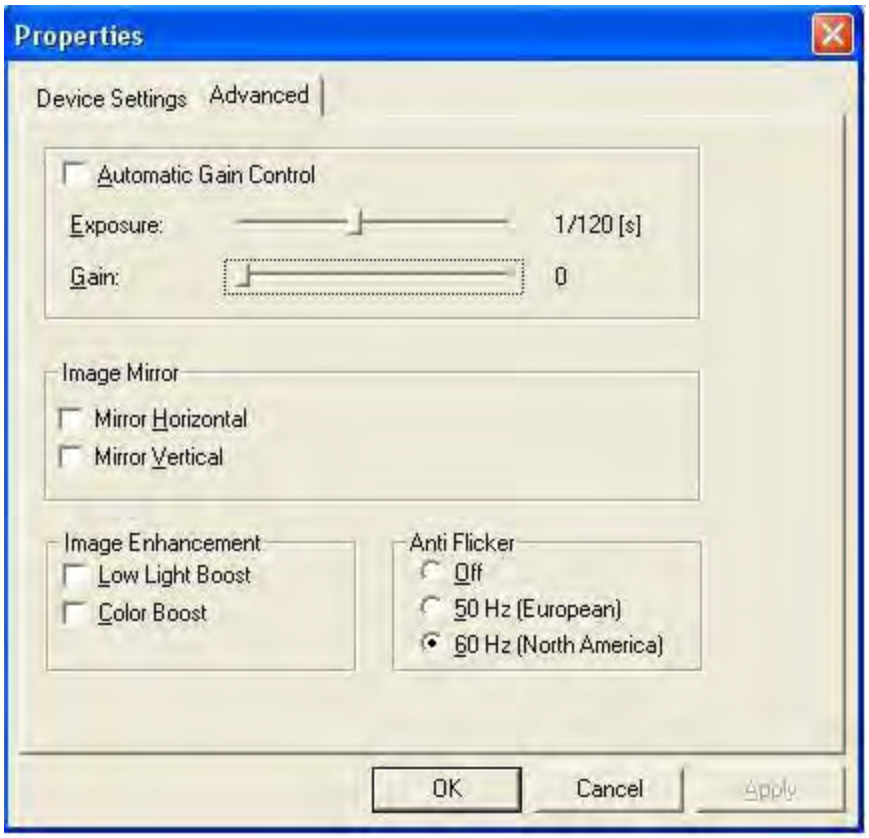


Figure 11. Video Setting Properties Window

Image Processing

Image Processing must be performed on the input images to determine the image coordinates of the IR source. OpenCV is chosen to perform the image processing task, mainly because the video frames from the videoInput library can be easily converted into an OpenCV image format. Initially, the input image is first converted into gray scale, as shown in Figure 12a. Binary thresholding is then used to isolate the white bright spot from the background. Since the cameras are very sensitive to IR, it is possible to use a high threshold value of 245 (8 bit value) to eliminate almost all of the background. An open source library, CvBlob, is used to locate any blobs in a binary image. For each detectable blob, CvBlob calculates the area and the centroid of the blob. In most cases, there is only one detectable blob in the threshold image. However, if the IR LED is placed very close to the camera, there may appear additional blobs due to glare and reflection. In such cases, the largest blob is taken to be the IR source. A red circle is used to indicate the detected location of the IR source, as shown in Figure 12c.

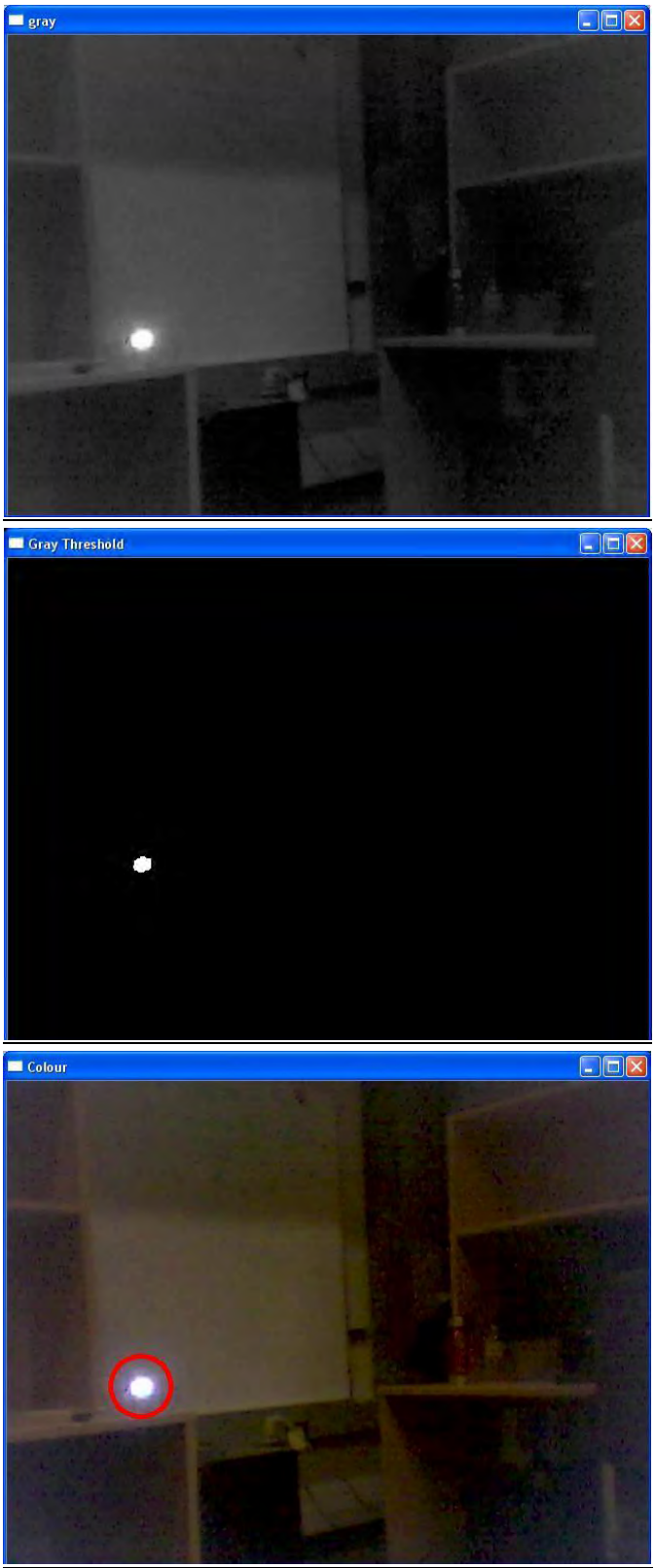


Figure 12 (a) The input image is first converted to a grayscale image. (b) Isolating the bright spot using Binary Thresholding. (c) A red circle is used to indicate the location of the largest bright spot.

Triangulation

The MATLAB Calibration Toolbox contains a triangulation module, stereo_triangulation.m, that computes the 3D location of the left and right image projections. This triangulation module accounts for any radial or tangential distortion of the camera lens, which is significantly more complicated than the method described in the theory section. The triangulation function was originally proposed to be implemented manually in the software package. However, the calibration results shown in 2.6.1 show that the distortion coefficients, **kc**, is in fact significant and should not be ignored. As a result, the triangulation module is converted to a C/C++ shared library using the MATLAB compiler [15]. The output library files are included as part of the project. As the left and right image coordinates of the IR source are generated from the image processing algorithm, they are passed to the external library for triangulation. The resultant coordinates are extracted from the MATLAB interface and displayed to the user in the command prompt. In the case where the IR source is not detected in one or both of the input images, the triangulation process is skipped for the frame.

2.4 Alternative Designs

2.4.1 Purkinje Images

Purkinje image tracking is based on processing images that are produced from reflections on the inner and outer boundaries of the cornea and lens. In general, there are four Purkinje images, but normally only two are of significance. Typically, the first image – which comes from the front of the cornea (and also known as the glint) – can be used because there is a bright-eye effect is produced from the reflected light from the retina, which can be easily distinguished as a very bright spot in image processing [6]. This can then be used to determine the absolute location of the eye in space. Although Purkinje images are relatively stable, the human pupil often rotates, which makes it much more difficult to locate. Fortunately, the fourth image – which comes from the back of the lens – is often used in the dual Purkinje imaging technique. The fourth image is very similar in size to the first image, but it is inverted and its intensity is less than 1% of the first image. By comparing the first and fourth images, it is possible to account for translational and rotational movements of the eye [1]. If the eye undergoes purely translational motion, then both images will move the identical distance. However, if the eye undergoes rotational motion, then the images will move different distances.

Although the dual Purkinje technique is typically much more accurate than other methods of imaging, the drawback is that it depends heavily on the background lighting conditions [6]. Another downside is that the bright spot is significantly smaller than the pupil reflection produced by the red-eye effect, which makes it much more difficult to detect. Figure 13 shows a

bright pupil produced by the red-eye effect, as well as the cornea reflection (seen as a glint) underneath it.

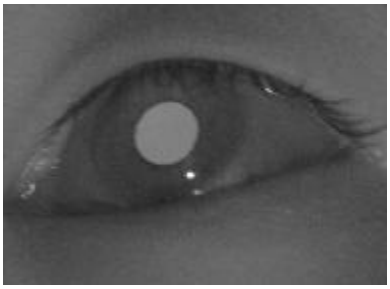


Figure 13. Red eye effect and corneal reflection

2.4.2 Other eye tracking techniques

There are other eye tracking techniques that utilize sensor systems rather than video imaging. For example, Electro-oculography requires the user to wear an apparatus that includes electrodes that are placed around the eye, and contact lens tracking requires the user to wear a magnetic field sensor, mirror, or other mechanical or optical monitoring system [5]. These methods are typically more accurate than video-based techniques. However, both methods are intrusive and cannot be applied without actual physical contact with the subject. The objective of the project was to create a more convenient and easy-to-use system, and to ensure that the subject does not feel uncomfortable.

2.4.3 Image processing

CImg

CImg is an open source image processing toolkit for C++ [9]. It is very lightweight and consists of a single header file that contains all the C++ classes and functions. The CImg library was considered before OpenCV. However, OpenCV was preferred because it was more widely supported and documented, and has integrated camera support in its library functions.

Quartz Composer

Quartz Composer is a graphics development environment with a visual programming language [10]. It directly utilizes hardware acceleration and pixel shaders from the Graphics Processing Unit (GPU), which makes it much more optimized for image processing. However, all software used in this project was in a Windows environment, and Quartz Composer is only available on Mac OS platforms.

2.4.4 Other cameras

Several other camera systems were also tested and explored. Initially, the investigators attempted to use the D-link DCS-950G network cameras for the application. Although the investigators could access the video feed through an internet browser, the network cameras did not have the appropriate APIs to be used locally for the task. Next, the investigators attempted to utilize two Microsoft LifeCam VX-1000 webcams, but found that they were not IR sensitive enough. Lastly, the investigators also attempted to use the Sony DCH-26 Camcorder and the Axis 205 camera for IR detection. The Sony Camcorder has a Nightshot mode, which internally removes the IR filters. While the Sony Camcorder is very IR-sensitive in the Nightshot mode, the camcorder requires a capture card (TV tuner) to be installed in order capture the video feed. This significantly reduced the portability of the system and hence, the Logitech QuickCam Messenger cameras were chosen for the application.

2.5 Testing Method and Protocol

In order to quantify the performance of the system, the investigators examined the speed of the progress, the accuracy of the triangulation module, as well as the working range of the stereo camera setup.

2.5.1 Accuracy of Triangulation

The original proposed method to determine the accuracy was to fix the two cameras in space, and move the IR source to various positions in order to determine the error. The IR source would move in an organized fashion (i.e. along floor tiles) and the distance to the camera would be measured with a measuring tape. However, Dr. Gregor Miller suggested that this method is highly inaccurate, as it is very difficult to align the camera"s optical axis along a particular direction. Any slight rotation would cause the camera"s optical axis to be aligned in a different direction, thus introducing additional error in the measurements. It would be impossible to determine if the error is due to misalignment or triangulation error. Instead, Dr. Miller suggested the appropriate method of determining the triangulation error. The output of the triangulated process, the 3D coordinates (X, Y, Z) of the source point, is re-projected onto the image plane of the camera using a non-linear "Plumb Bob" model , which accounts for radial and tangential distortion. This model uses the normalized coordinates (i.e. X/Z, Y/Z) and the calibration parameters to determine the image coordinates of a point P(X,Y,Z). The detailed re-projection procedure is given in Appendix B. The re-projected image coordinates (x_p, y_p) is compared with the original image coordinates (x_i,y_i), and the percent error is computed as

$$\%\ Error\ in\ x-direction = \frac{|x_p - x_i|}{Image\ Width} \text{ and}$$

$$\%\ Error\ in\ y-direction = \frac{|y_p - y_i|}{Image\ Height}.$$

The investigators examined the triangulation error by choosing a common point (i.e. a specific corner of the checkerboard pattern) of the two input images, and computed the % Error in x and y directions as described above. The procedure is repeated at various distances. Although the absolute error scales with distance, it is hoped that the percent error remains the same at all distances.

2.5.2 Working Range of System

The operating range of the system is also measured to quantify the performance of the system. The minimum distance to common field of view and the common viewing angle is determined. The minimum distance is measured by initially placing the checkerboard pattern close to the cameras and moving it back slowly until a common point appears in the image of both cameras. The minimum distance can also be computed using the separation distance of the cameras and the field of view of the Logitech QuickCam Messenger, as seen in Figure 14 below. The two values are compared to examine the minimum distance to common field of view. The common viewing angle is measured by placing the checkerboard pattern at a known distance away and identifying the left-most and right-most pixels that appear on the image of both cameras. Since the size of each checkerboard squares is known, the horizontal distance, and hence the common viewing angle is determined.

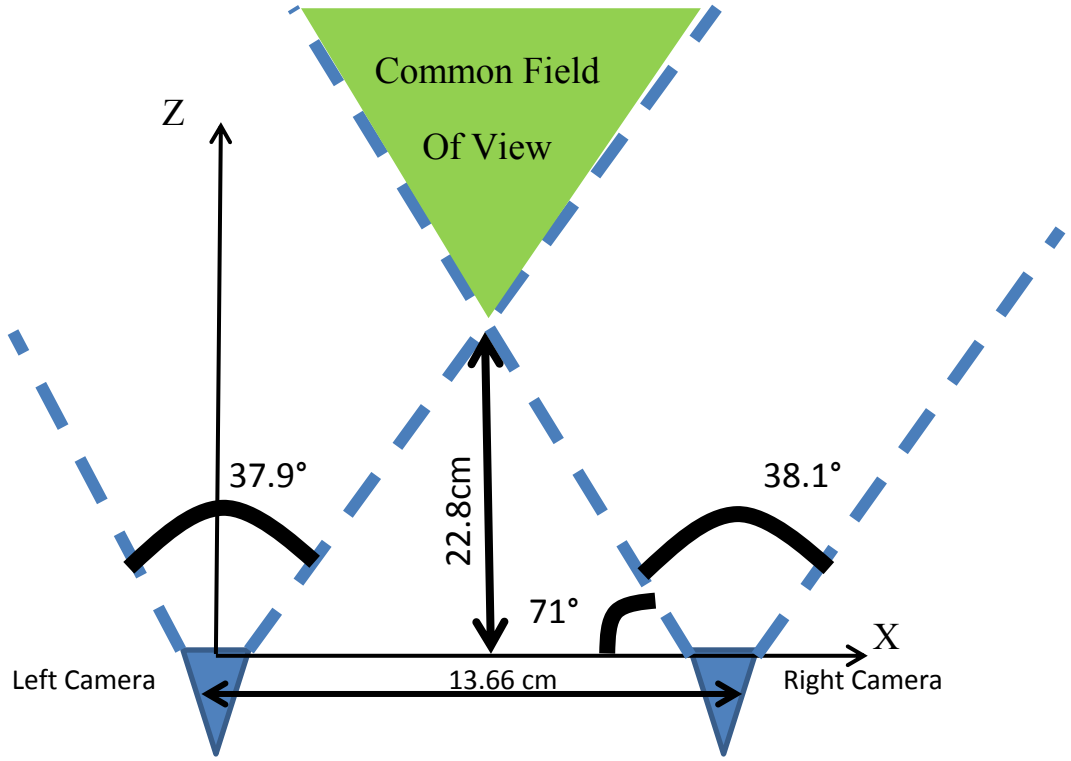


Figure 14. Common Field of View of Left and Right Camera

In addition to determining the minimum distance and the common field of view, the investigators also examined the maximum head movement and head rotation that causes the IR source to be undetected by the image processing algorithm. All these measurements are taken at 1.5m away, which is a typical operating distance. Simple tape measurements are used to evaluate the allowable head movement in the horizontal and vertical direction. The allowable rotation is measured by placing the IR goggle on a flat surface and rotating it until the light reaching the stereo cameras is insufficient to be processed.

2.6 Results

2.6.1 Calibration

After the two cameras were calibrated with MATLAB, the intrinsic and extrinsic parameters were determined and recorded so that they could be used in the triangulation measurements. The description of these parameters may be found in Appendix B. Field of view was also measured (see Table 1).

Intrinsic parameters of left camera

Focal Length (mm): fc_left = [901.88519 901.49456] ± [10.00869 10.43267]
Principal point: cc_left = [371.50513 237.31749] ± [8.85621 4.97275]
Skew: alpha_c_left = [0.00000] ± [0.00000]
Distortion: kc_left = [0.75565 -3.16262 -0.00299 0.00095 0.00000] ± [0.05779
0.44345 0.00374 0.00930 0.00000]

Intrinsic parameters of right camera

Focal Length (mm): fc_right = [893.88032 894.30765] ± [10.62383 10.89368]
Principal point: cc_right = [308.27280 302.79057] ± [7.75198 7.21819]
Skew: alpha_c_right = [0.00000] ± [0.00000]
Distortion: kc_right = [0.80027 -3.24039 0.00743 -0.00024 0.00000] ± [0.06126
0.55570 0.00668 0.00730 0.00000]

Extrinsic parameters (position of right camera with respect to left camera)

Rotation vector: om = [0.05058 0.14347 -0.04562]
Translation vector (mm) : T = [136.64322 -12.07266 -20.13129]

Table 1. Field of view measurements

	Left Camera	Right Camera
Field of view (angle, degrees)	37.9 ± 0.7	38.1 ± 0.7
Field of view, horizontal span, 20 inches away	13.2 ± 0.2	13.1 ± 0.3
Common field of view, minimum distance (inches)	9.00 ± 0.25	
Common field of view (angle, degrees)	38.0 ± 0.5	

The parameters of each camera were kept very similar to each other to make the system as symmetrical as possible. Skew was determined to be zero, so that the x pixel and y pixel were 90 degrees with respect to each other. However, other parameters were not exact, because the focal length could only be manually adjusted by hand. Furthermore, the cameras were mounted on top of a wooden block, which was inclined at a very small angle. The cameras were also not perfectly level with respect to the block. This accounts for the discrepancy between the field of view of both cameras. See Figure 14 for details on the field of view and common field of view of the two camera setup.

The absolute minimum distance in order to see an object at the edge of both cameras was determined to be 9.00 ± 0.25 inches. However, at this distance only a single point can be detected, as the horizontal span is effectively zero.

At a more nominal distance of 23.25 ± 0.25 inches, the field of view spans a horizontal distance of 8.25 ± 0.25 inches. The common field of view angle was determined to be 38 degrees, which makes sense because it should be very similar to the field of view of one camera (since both are almost identical). For several different subjects, the minimum distance where both eyes are visible on both cameras was determined to be approximately 16 ± 1 inches. At a more typical operating distance of about 6 feet, the maximum vertical and horizontal distance – to stay within the common field of view - that the subject is able to move is about 10 ± 1 inches.

2.6.2 Triangulation

In order to evaluate the accuracy of the triangulation, the triangulated point was reprojected onto the two image planes. The error is then defined the difference between the reprojected points on the image planes and the actual points. This process takes into account the distortion of both cameras, and is described in greater detail in 2.5.1. The errors, expressed in terms of pixels, were measured for various distances of triangulation (see Table 2).

Table 2. Triangulation errors

Triangulation distance (inches)	Pixel error in x	% error	Pixel error in y	% error
13.5	0.520	0.08118	1.061	0.22099
22	0.544	0.08502	1.570	0.32718
36	0.555	0.08677	1.999	0.41649
56.5	0.575	0.08983	2.095	0.43652
76	0.594	0.09276	2.189	0.45600
110	0.610	0.09535	2.254	0.46965

When selecting the actual point, an inherent half pixel error was applied because corners of the checkerboard squares sometimes appeared to be between the edges of different pixels. At larger triangulation distances, errors in the points in the image plane will always translate to larger errors in the triangulated point. It can also be observed that there were larger errors in the y direction (480 pixels) compared to the x direction (640 pixels). This could be due factors such as distortion.

2.6.3 Tracking with IR goggles

Actual tracking of a subject‟s eyes was not achieved, so several measurements of performance were tested using the IR goggles as a substitute. The maximum range of tracking was found to be greater than 6 meters, which was the maximum dimension of the room that the test was conducted in. The maximum angle of rotation for tracking to work was determined to be approximately 60 degrees (horizontally and vertically). At distances greater than 15 cm, the system was able to detect and track the location of the IR bright spot at roughly 25-30 frames per second. At distances less than 15 cm, the frame rate would sometimes drop below 15. This would occur when the bright spot appeared to be too large or anomalous; in this case, typically the incorrect center of the spot would be detected.

2.6.4 Red-eye effect

A visible red-eye effect was observed in the subject when the IR illuminator was placed in the correct location. The IR illuminator was primarily used, because the intensity of the IR rings was often found to be insufficient to achieve the red-eye effect. The effect could be observed when the subject was located between 49.5 and 81 inches away from the camera, and the maximum angle of rotation of the head was found to be approximately 16.8 ± 2.7 degrees. At distances beyond this range, the illuminator source could not be placed on axis.

An example of an image of the red-eye effect is shown in Figure 15. A picture was also taken with the light source off-axis, so that the dark eye effect would be produced (see Figure 16). Both were taken with the subject in approximately the same location.

Figure 15. Red-eye effect

Figure 16. Dark-eye effect

Based on the red-eye effect image alone, it is difficult to distinguish the location of the eyes in image, since the subject‟s face appears to be quite illuminated. A difference image between the red-eye and dark-eye images was also taken (see Figure 17). However, it still appears difficult to isolate the location of the eyes, because there is a lot of background noise. Small bright specks can be observed, but it cannot be known for certain which ones correspond to the eyes.

Figure 17. Difference image

2.7 Discussion of Results

Based on the calibration results of the stereo camera configuration, and for the purposes of Cubee, the viewing angle (common field of view) is relatively small. However, the results from triangulation demonstrate that the system can locate a point correctly. It should be realized that the calibration parameters are system specific. The investigators attempted to keep the parameters of each camera as similar to each other as possible, so that the system would be symmetrical. However, the calibration technique should still function correctly for two entirely different cameras, provided that their configuration yields an appropriate common field of view.

In the range of distances that were tested, the maximum pixel error was always found to be less than 2, and the percentage error always less than 1%. Sources of error in triangulation testing could have occurred when selecting the pixel location for a checkerboard square. Although this was inherited in the addition of the half pixel error, there could have also been errors while designing the checkerboard. e.g. Each square is not exactly the same so there may be slight discrepancies in the number of pixels per square.

The red-eye effect could clearly be identified when the IR illuminator was used. Although the triangulation module is able to determine the 3D coordinates of the IR source (IR goggle) quite accurately, the proposed method of determining the head position based on the red-eye effect did not work well. As seen in Figure 15 previously, the bright-eye effect is not strong enough to be used for image processing. This can be explained by several reasons. IR tracking based on the bright-eye effect typically uses one camera, with the user being sufficiently close to the camera (~0.5m). However, since the user‟s eyes must be in the common field of view of the cameras for triangulation, the user must be sufficiently far away (~1.5m). However, at such distances, the image quality becomes very poor, and the bright eye effect is much less pronounced than the one seen in Figure 18. Furthermore, the cameras that were available for the project were relatively low quality. With higher quality cameras, it is expected that image processing will be easier to perform, and the location of the eyes will be easier to locate.

Figure 18. Ideal bright-eye effect.
Source: http://collab.eyewriter.org/wp-content/uploads/2010/04/brightEye.jpg

The investigators also attempted to use a difference image to separate the bright-eye effect from the background. However, as shown in Figure 15 and Figure 16, the IR lamp did not illuminate the background evenly when it is moved off the optical axis, resulting in shadows and dark spots in the dark-field image. As the images are subtracted from each other, the resultant image contains multiple bright spots that might be misinterpreted as the IR reflection. Even so, the difference image requires the user to remain stationary for both the bright and dark field images. At 30 frames per second, a small movement may make it impossible to detect the location of the

eyes in the different image. This is a particularly challenge because the even and odd frames of the camera are not synchronized with the LED lamp, meaning that the LED lamp must be turned on and off manually to capture the bright/dark field image. For future testing purposes, it may be beneficial to utilize a camera with a higher frame rate, and integrated coaxial IR light source, which should be synchronized so that better red-eye and dark-eye images can be produced.

The investigators were unable to track the eyes because of difficulties in pinpointing the location in the red-eye effect and difference images, but the IR goggles as a substitute demonstrated that if the bright spot were the actual location of one of the eyes, then the system would be able to track it successfully.

3. Conclusion

The objective of the project was to construct a wireless, non-invasive, low-latency infrared eye tracking system that could replace the existing solution. While the system was not completed, it is a work-in-progress. In using a simple two webcam system mounted on a wooden block, the proof of concept was clearly demonstrated, and it should not be difficult to extend the system to function with any two camera configuration, provided that the appropriate common field of view is produced.

The two cameras were successfully calibrated with MATLAB; the calibration parameters corresponding to the two camera setup are system dependent. Using the triangulation method, the cameras were utilized to accurately determine the 3D location of a single IR-emitting source. Several methods were utilized to characterize the performance of the triangulation system. For all distances that were tested, the triangulation error was found to be less than 3 pixels, or less than 1% with respect to the image resolution.

Image processing techniques were developed to locate the eyes of a subject. A software algorithm - based on searching for the location of a bright-eye effect created by the reflection off the retina - was developed. The algorithm worked with the IR goggles, but based on the images that were taken with this particular camera setup, it was unable to locate the eyes of the subject. The investigators also attempted to utilize a difference image to improve the detection of the red-eye effect. However, results did not improve.

The investigators were unable to satisfy the final objective, which involves tracking of the eyes in real-time. This is because accurately detecting the eyes is a crucial pre-requisite.

There were many limitations or sources of error that prevented the system to be designed optimally. The investigators believe that with more development and refinement, there are many improvements that can be made so that the system would satisfy all the requirements. Several of these improvements are discussed in the following Recommendations section.

4. Recommendations

To improve the accuracy and usability of the system, the investigators propose the following future course of actions:

1. **Use camera with higher resolution and field of view**
   It would be beneficial to use cameras with higher resolution and wider field of view for our application. Currently, one of the main problems is that the common field of view is very narrow, and thus the user must be positioned far away (> 1.5m) from the cameras to allow for any reasonable head movement. However, the IR reflection at such distances is not strong enough to be detected by the Image Processing Algorithm. If cameras with wider field of view were used, they would decrease the minimum distance at which the two cameras share the common field of view. As a result, this allows the user to move closer to the cameras and provides additional room for head movements. Furthermore, the IR reflection from the user would be more pronounced, making it much easier to detect with the image processing algorithm.

   It should also be noted that the associated radial distortion of a wide field of view camera (fish-eye camera) should not affect the accuracy. The MATLAB Calibration Toolbox accounts for any radial distortion in the calibration process, and the distortion vector, **kc**, is adjusted accordingly for triangulation. On the other hand, the improved resolution of the camera would allow the system to determine the image coordinates of the IR source more accurately, and thus reduce any triangulation error.

2. **Use IR Ring with brighter LEDs**
   The IR LED Ring chosen for the application is not bright enough for IR reflection. The investigators did not see the bright-eye effect when using the IR ring, even when the IR ring is placed on the optical axis and the user is positioned close to the camera. Instead, the IR lamp was used for the bright-field image. However, the IR lamp must be placed slightly off the optical axis due to size constraints. As a result, the user must move further back to compensate for this adjustment. This was a design flaw that was overlooked, as the required brightness for IR reflection was initially unknown. After additional research, the investigators feel that Agilent HSDL-4220 880nm diodes may be used for the application. It is a high brightness LED that is used by others with successful results [14]. The datasheet for the LED may be found in Appendix D. Ideally, the Agilent LEDs should be placed in inner and outer rings for on and off axis illumination, as shown in Figure 19.

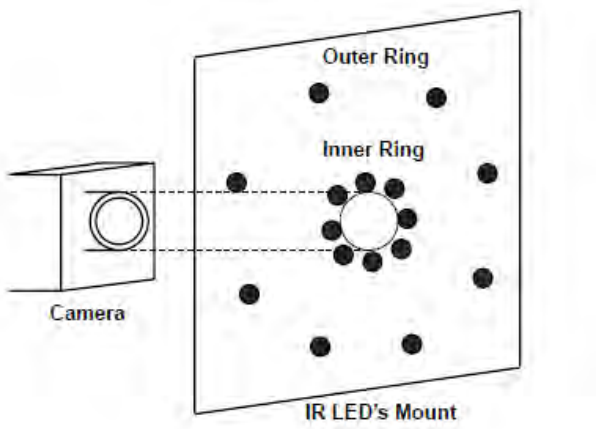


Figure 19. Inner and Outer Ring Illumination.

3. **Synchronize bright/dark field images with odd and even frames**
   The bright and dark field images should be synchronized with the odd and even frames of the camera. Since the difference image is used to isolate the bright-eye effect, the inner and outer LED rings must be activated automatically for each alternating frame. Bradshaw et al. suggests a simple circuit and Arduino code that can be used synchronize the inner and outer LED ring [14]. Ideally, the camera‟s shutter should also be synchronized with the LED rings. However, this may be more difficult implement as it requires one to have access to the internal API of the camera.

4. **Add additional cameras when required**
   Even if the IR reflection worked, the system may not function correctly for the purposes of Cubee. Since Cubee resembles the shape of a cube, and there are LED screens on five sides, there is a strong possibility that the camera could be blocked on one or more of the sides. In order to have line of sight with the user‟s eyes, it may be necessary to add additional cameras on different sides.

5. **Use more advanced image processing techniques**
   It may be insufficient to only utilize binary thresholding. Dr. Sidney Fels mentioned that it may be necessary to apply other steps of processing before attempting to locate the bright red-eye effect. For example, the eyes may not appear as uniform bright spots on the image, and edge detection might be required. Also, in order to eliminate background noise it may be necessary to perform histogram adjustment.

APPENDIX A. Logitech QuickCam Specification

Logitech QuickCam Messenger Specifications	
General	
Device Type	Web camera
Localization	English / Canada , French
Optical sensor type: CMOS	CMOS
Software Included	Yahoo! Messenger , ImageStudio , MGI PhotoSuite , Drivers & Utilities , Logitech IM Video Companion
Camera	
Type	Color
Still Image	640 x 480
Image Sensor Type	CMOS
Lens Construction	
Focus Adjustment	Manual
Interfaces	
Computer Interface	USB
Expansion / Connectivity	
Interfaces	1.0 x USB - 4 pin USB Type A
Compatible Slots	None
Miscellaneous	
Cables Included	1.0 x USB cable - Integrated - 6.0 ft
Mounting Kit	Included
Power	
Power Device	None
Software / System Requirements	
OS Required	Microsoft Windows XP , Microsoft Windows 98/ME/2000
Min Processor Type	Pentium II 400.0 MHz
Min RAM Size	64.0 MB
Min Hard Drive Space	200.0 MB
Peripheral / Interface Devices	Sound card , USB port , CD-ROM , Speaker(s)
Manufacturer Warranty	
Service & Support Details	Limited warranty - 2 years
Override	
Service & Support	2 years warranty

Source: http://reviews.cnet.com/webcams/logitech-quickcam-messenger/1707-6502_7-20439317.html

APPENDIX B. Description Intrinsic Parameters and Re-projection method

Source: http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/parameters.html

Intrinsic parameters (camera model):

The list of internal parameters:
Focal length: The focal length in pixels is stored in the 2x1 vector fc.
Principal point: The principal point coordinates are stored in the 2x1 vector cc.
Skew coefficient: The skew coefficient defining the angle between the x and y pixel axes is stored in the scalar alpha_c.
Distortions: The image distortion coefficients (radial and tangential distortions) are stored in the 5x1 vector kc.

Definition of the intrinsic parameters:

Let P be a point in space of coordinate vector $XX_c = [X_c;Y_c;Z_c]$ in the camera reference frame.
Let us project now that point on the image plane according to the intrinsic parameters (fc, cc, alpha_c, kc).
Let x_n be the normalized (pinhole) image projection:

$$x_n = \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix},$$

Let

$$r^2 = x^2 + y^2.$$

After including lens distortion, the new normalized point coordinate x_d is defined as follows:

$$x_d = \begin{bmatrix} x_d(1) \\ x_d(2) \end{bmatrix} = \left(1 + kc(1)\,r^2 + kc(2)\,r^4 + kc(5)\,r^6\right) x_n + dx$$

where dx is the tangential distortion vector:

$$dx = \begin{bmatrix} 2\,kc(3)\,x\,y + kc(4)\left(r^2 + 2x^2\right) \\ kc(3)\left(r^2 + 2y^2\right) + 2\,kc(4)\,x\,y \end{bmatrix}$$

Therefore, the 5-vector kc contains both radial and tangential distortion coefficients (observe that the coefficient of 6^{th} order radial distortion term is the fifth entry of the vector kc).
It is worth noticing that this distortion model was first introduced by Brown in 1966 and called "Plumb Bob" model (radial polynomial + "thin prism"). The tangential distortion is due to "decentering", or imperfect centering of the lens components and other manufacturing defects in a compound lens. For more details, refer to Brown's original publications [7].

Once distortion is applied, the final pixel coordinates x_pixel = $[x_p;y_p]$ of the projection of P on the image plane is:

$$\begin{cases} x_p = fc(1)\left(x_d(1) + alpha\_c * x_d(2)\right) + cc(1) \\ y_p = fc(2)\, x_d(2) + cc(2) \end{cases}$$

Therefore, the pixel coordinate vector **x_pixel** and the normalized (distorted) coordinate vector $\mathbf{x_d}$ are related to each other through the linear equation:

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = KK \begin{bmatrix} x_d(1) \\ x_d(2) \\ 1 \end{bmatrix},$$

where **KK** is known as the camera matrix, and defined as follows:

$$KK = \begin{bmatrix} fc(1) & alpha\_c * fc(1) & cc(1) \\ 0 & fc(2) & cc(2) \\ 0 & 0 & 1 \end{bmatrix}.$$

In MATLAB, this matrix is stored in the variable **KK** after calibration. Observe that **fc(1)** and **fc(2)** are the focal distance (a unique value in mm) expressed in units of horizontal and vertical pixels. Both components of the vector **fc** are usually very similar. The ratio **fc(2)/fc(1)**, often called "aspect ratio", is different from 1 if the pixel in the CCD array are not square. Therefore, the camera model naturally handles non-square pixels. In addition, the coefficient **alpha_c** encodes the angle between the x and y sensor axes. Consequently, pixels are even allowed to be non-rectangular. Some authors refer to that type of model as "affine distortion" model.

In addition to computing estimates for the intrinsic parameters **fc**, **cc**, **kc** and **alpha_c**, the toolbox also returns estimates of the uncertainties on those parameters. The MATLAB variables containing those uncertainties are **fc_error**, **cc_error**, **kc_error**, **alpha_c_error**. For information, those vectors are approximately three times the standard deviations of the errors of estimation.

APPENDIX C. Description of Software Code

The software for this project is all written in C++ using Visual Studio Express 2010. All required files are packaged in 1060_Cubee_IR_Tracking_Software.rar. The project is named „Cubee IR Tracking System". The software package contains the library files for videoInput, openCV, CvBlob, and MATLAB Triangulation. All library header files are stored in the folder „LibHeaders". The openCV and the cvBlob library headers are grouped together inside the folder „LibHeaders/openCV", while the required MATLAB headers files and the triangulation module is stored inside the folder „Triangulation". Correspondingly, all source and object library files are stored in the folder „Lib", with the same file hierarchy described above. The software package also contains two main files: Camera.cpp and IR_Tracking.cpp. These are described in details below.

Camera.cpp

Camera.cpp is the associated file with the Camera class. The Camera class is used to represent and model the cameras that are used for the application, which contains the intrinsic parameters of the camera. The image size, camera position, focal length, principle point, distortion coefficients are all stored in the Camera class. The code snippet below shows the associated functions with the Camera class.

```
class Camera
{
public:
	Camera(void);
	Camera(double fccX, double fccY, double focal_X, double focal_Y, int imgWidth, int
	imgHeight, double CamLocX, double CamLocY, double CamLocZ);

	// Camera function
	void CaptureFrame();
	// Getters
	double* getFcc();
	double* getFocalLength();
	double* getCameraLocation();
	double* getKc();
	unsigned int* getImageSize();

	// Setters
	void setFcc(double fccX, double fccY);
	void setFocalLengthX(double length);
	void setFocalLengthY(double length);
	void setCameraLocation(double X, double Y, double Z);
	void setKc(double fourthOrd, double thirdOrd, double secondOrd, double firstOrd,
	double zerothOrd);
	~Camera(void);

private:
	double fcc[2];
	double focalLength[2];
	double CamLocation[3];
	double kc[5];
```

```
	unsigned int imageSize[2];
};
```

IR_Tracking.cpp

The project solution‟s „main‟ function is located inside IR_Tracking.cpp. IR_Tracking.cpp is procedural based and is responsible for capturing input images from the cameras, identifying the image coordinates of the IR source, and triangulating to find the coordinates of the IR source in real space. After initializing the cameras and the selecting the appropriate gain and exposure levels, it begins to capture new frames in a continuous loop. For each frame, the colour image is converted into grayscale and segmented using a binary threshold of 245 (Max = 255). CvBlob is then used to locate the centroid of the largest bright spot of the image. Lastly, the left and right image coordinates of the IR source is passed into the triangulation module to determine the 3D coordinates of the IR source. The results are then displayed in the command prompt. The key code fragment for each process is shown below.

Capturing new frames:

The code runs as a continuous loop. If a new frame exists, the new pixels are copied into the frame buffer. Note that the variable size1 is the image size, 640x480. This process is repeated for the left camera as well.

```
while(true)
	{
		// If there is a new frame
		if(VI.isFrameNew(rightDevice))
		{
			// Get the Pixels from camera and copy it to the frame buffer
			VI.getPixels(rightDevice, rightBuffer, false, true);	//fills
pixels as a BGR (for openCV) unsigned char array - no flipping
			for (int i = 0; i < size1; i++)
			{
				*((tempPt)+i) = *(rightBuffer+i);
			}

		}
```

Image Processing

The images are converted to grayscale images and segmented using a binary thresholding technique. CvBlob is then used to search for the white blobs in the image. All the blobs are iterated through to find the largest one, and the coordinates are stored accordingly.

```
int rightThreshold = 245;
int leftThreshold = 245;
int maxValue = 255;
int thresholdType = CV_THRESH_BINARY;
```

```
// Convert the colour image to grayscale
cvCvtColor( imgRight, grayRight, CV_BGR2GRAY);
cvCvtColor( imgLeft, grayLeft, CV_BGR2GRAY);


// Threshold the greyscale image to isolate the bright spot
cvThreshold(grayRight, grayThreshRight, rightThreshold, maxValue, thresholdType);
cvThreshold(grayLeft, grayThreshLeft, leftThreshold, maxValue, thresholdType);


// Declare blob variables for the cvBlob Library
cvb::CvBlobs leftBlobs, rightBlobs;

// Label the threshold image to search for any white blobs
unsigned int rightResult = cvLabel(grayThreshRight, labelImgRight, rightBlobs);
unsigned int leftResult = cvLabel(grayThreshLeft, labelImgLeft, leftBlobs);


// Iterate the blobs to find the largest one.
for (cvb::CvBlobs::const_iterator it=rightBlobs.begin(); it!=rightBlobs.end(); ++it)
              {

                     int area = (int)(*it).second->area;
                     int x = (int)(*it).second->centroid.x;
                     int y = (int)(*it).second->centroid.y;

                     if ( area > maxArea_right)
                     {
                            maxArea_right = area;
                            largestx_right = x;
                            largesty_right = y;
                     }
              }
```

Triangulation

The triangulation module is shown below. After initialize the variables for the MATLAB Interface, the external library for triangulation is called to perform stereo triangulation using the image coordinates and the calibration parameters. The results can easily be extract using a mxPtr.

```
bool Triangulate(const double* leftImgCoord, const double* rightImgCoord, double*
leftObjCoord, double* rightObjCoord, Camera LeftCam, Camera RightCam, double*
rotationalVector, double* translationalVector)
{
       printf("leftImg: %f, %f ; right Img: %f, %f\n", leftImgCoord[0], leftImgCoord[1],
       rightImgCoord[0], rightImgCoord[1]);

       // Removed: initializing and setting variables for Matlab interface

       double *retValues;
       mxArray *y_ptr=NULL;

       // Library Call to do the stereo triangulation
       mlfStereo_triangulation(2, &RET_XL, &RET_XR, m_xL, m_xR, m_om, m_T, m_fc_left,
       m_cc_left, m_kc_left, m_alpha_c_left, m_fc_right, m_cc_right, m_kc_right,
       m_alpha_c_right);
```

```
        // Grab the pointer to access the return values
        retValues = mxGetPr(RET_XL);
        // Display the results
        printf("3D Coordinates of IR Source wrt to Left Camera:\n");
        printf("X: %lf\n", retValues[0]);
        printf("Y: %lf\n", retValues[1]);
        printf("Z: %lf\n", retValues[2]);

        printf("\n");

        // Grab the pointer to access the return value
        retValues = mxGetPr(RET_XR);
        // Display the results
        printf("3D Coordinates of IR Source wrt to Right Camera:\n");
        printf("X: %lf\n", retValues[0]);
        printf("Y: %lf\n", retValues[1]);
        printf("Z: %lf\n", retValues[2]);

        return true;
}
```

HEWLETT PACKARD

High-Performance T-1¾ (5 mm) TS AlGaAs Infrared (875 nm) Lamp

Technical Data

HSDL-4200 Series
HSDL-4220 30°
HSDL-4230 17°

Features

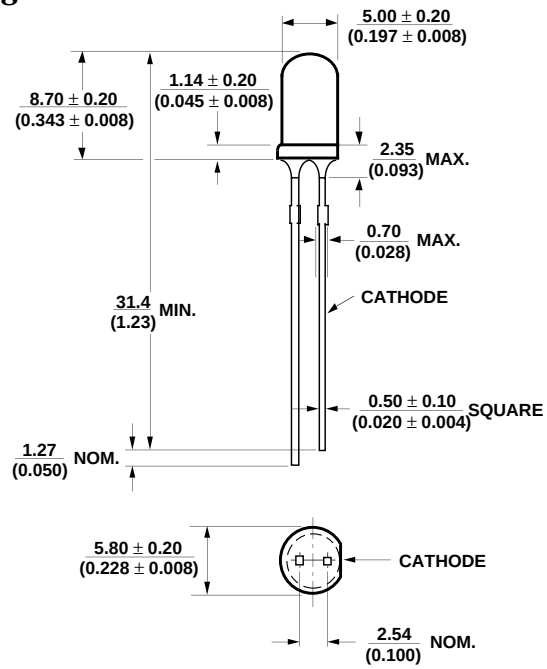
- **Very High Power TS AlGaAs Technology**
- **875 nm Wavelength**
- **T-1¾ Package**
- **Low Cost**
- **Very High Intensity:**
  HSDL-4220 - 38 mW/sr
  HSDL-4230 - 75 mW/sr
- **Choice of Viewing Angle:**
  HSDL-4220 - 30°
  HSDL-4230 - 17°
- **Low Forward Voltage for Series Operation**
- **High Speed: 40 ns Rise Times**
- **Copper Leadframe for Improved Thermal and Optical Characteristics**

Applications

- **IR Audio**
- **IR Telephones**
- **High Speed IR Communications**
  IR LANs
  IR Modems
  IR Dongles
- **Industrial IR Equipment**
- **IR Portable Instruments**

- **Interfaces with Crystal Semiconductor CS8130 Infrared Transceiver**

Package Dimensions



Description

The HSDL-4200 series of emitters are the first in a sequence of emitters that are aimed at high power, low forward voltage, and high speed. These emitters utilize the *T*ransparent *S*ubstrate, double heterojunction, *Al*uminum *Ga*l-lium *As*enide (TS AlGaAs) LED technology. These devices are optimized for speed and efficiency at emission wavelengths of 875 nm. This material produces high radiant efficiency over a wide range of currents up to 500 mA peak current. The HSDL-4200 series of emitters are available in a choice of viewing angles, the HSDL-4230 at 17° and the HSDL-4220 at 30°. Both lamps are packaged in clear T-1¾ (5 mm) packages.

The package design of these emitters is optimized for efficient power dissipation. Copper leadframes are used to obtain better thermal performance than the traditional steel leadframes.

The wide angle emitter, HSDL-4220, is compatible with the IrDA SIR standard and can be used with the HSDL-1000 integrated SIR transceiver.

Absolute Maximum Ratings

Parameter	Symbol	Min.	Max.	Unit	Reference
Peak Forward Current	I_{FPK}		500	mA	[2], Fig. 2b Duty Factor = 20% Pulse Width = 100 µs
Average Forward Current	I_{FAVG}		100	mA	[2]
DC Forward Current	I_{FDC}		100	mA	[1], Fig. 2a
Power Dissipation	P_{DISS}		260	mW	
Reverse Voltage (I_R = 100 µA)	V_R	5		V	
Transient Forward Current (10 µs Pulse)	I_{FTR}		1.0	A	[3]
Operating Temperature	T_O	0	70	°C	
Storage Temperature	T_S	-20	85	°C	
LED Junction Temperature	T_J		110	°C	
Lead Soldering Temperature [1.6 mm (0.063 in.) from body]			260 for 5 seconds	°C	

Notes:
1. Derate linearly as shown in Figure 4.
2. Any pulsed operation cannot exceed the Absolute Max Peak Forward Current as specified in Figure 5.
3. The transient peak current is the maximum non-recurring peak current the device can withstand without damaging the LED die and the wire bonds.

Electrical Characteristics at 25°C

Parameter	Symbol	Min.	Typ.	Max.	Unit	Condition	Reference
Forward Voltage	V_F	1.30	1.50 2.15	1.70	V	I_{FDC} = 50 mA I_{FPK} = 250 mA	Fig. 2a Fig. 2b
Forward Voltage Temperature Coefficient	$\Delta V/\Delta T$		-2.1 -2.1		mV/°C	I_{FDC} = 50 mA I_{FDC} = 100 mA	Fig. 2c
Series Resistance	R_S		2.8		ohms	I_{FDC} = 100 mA	
Diode Capacitance	C_O		40		pF	0 V, 1 MHz	
Reverse Voltage	V_R	2	20		V	I_R = 100 µA	
Thermal Resistance, Junction to Pin	$R\theta_{jp}$		110		°C/W		

Optical Characteristics at 25°C

Parameter	Symbol	Min.	Typ.	Max.	Unit	Condition	Reference
Radiant Optical Power							
HSDL-4220	P_O		19 38		mW	I_{FDC} = 50 mA I_{FDC} = 100 mA	
HSDL-4230	P_O		16 32		mW	I_{FDC} = 50 mA I_{FDC} = 100 mA	
Radiant On-Axis Intensity							
HSDL-4220	I_E	22	38 76 190	60	mW/sr	I_{FDC} = 50 mA I_{FDC} = 100 mA I_{FPK} = 250 mA	Fig. 3a Fig. 3b
HSDL-4230	I_E	39	75 150 375	131	mW/sr	I_{FDC} = 50 mA I_{FDC} = 100 mA I_{FPK} = 250 mA	Fig. 3a Fig. 3b
Radiant On-Axis Intensity Temperature Coefficient	$\Delta I_E/\Delta T$		-0.35 -0.35		%/°C	I_{FDC} = 50 mA I_{FDC} = 100 mA	
Viewing Angle							
HSDL-4220	$2\theta_{1/2}$		30		deg	I_{FDC} = 50 mA	Fig. 6
HSDL-4230	$2\theta_{1/2}$		17		deg	I_{FDC} = 50 mA	Fig. 7
Peak Wavelength	λ_{PK}	860	875	895	nm	I_{FDC} = 50 mA	Fig. 1
Peak Wavelength Temperature Coefficient	$\Delta\lambda/\Delta T$		0.25		nm/°C	I_{FDC} = 50 mA	
Spectral Width–at FWHM	$\Delta\lambda$		37		nm	I_{FDC} = 50 mA	Fig. 1
Optical Rise and Fall Times, 10%-90%	t_r/t_f		40		ns	I_{FDC} = 50 mA	
Bandwidth	f_c		9		MHz	I_F = 50 mA ± 10 mA	Fig. 8

Ordering Information

Part Number	Lead Form	Shipping Option
HSDL-4220	Straight	Bulk
HSDL-4230	Straight	Bulk

References

[1] Duchowki, A.. Eye Tracking Methodology: Theory and Practice (2007)

[2] Stavness, I., Lam, B., and Fels, S. pCubee: A Perspective-Corrective Handheld Cubic Display. *CHI 2010: Pointing and Selecting* (2010), 1381-1390

[3] Cubee. http://www.cubee.ca/

[4] Taubin, Gabriel. Camera Model and Triangulation. http://mesh.caltech.edu/ee148/notes/projections.pdf

[5] Morimoto, C.H., Koons, D.,Amir, A., and Flickner, M. Pupil detection and tracking using multiple light sources. Image and Vision Computing 18 (2000), 331–335

[6] Nguyen, K., Wagner, C., Koons, D., and Flicker, M. Differences in the Infrared Bright Pupil Response of Human Eyes

[7] MATLAB Camera Calibration Toolbox. http://www.vision.caltech.edu/bouguetj/calib_doc/

[8] Duchowki, A.. Eye Tracking Methodology: Theory and Practice (2007)

[9] CImg. http://cimg.sourceforge.net/

[10] Quartz Composer. http://developer.apple.com/graphicsimaging/quartz/quartzcomposer.html

[11] Xbox Kinect. http://www.xbox.com/en-ca/kinect

[12] Twede, David. Introduction to Full-Spectrum and Infrared Photography. http://surrealcolor.110mb.com/IR_explained_web/IR_explained.htm#CamColor

[13] videoInput software. http://www.muonics.net/school/spring05/videoInput/software

[14] Bradshaw et al. Bright-Eye Effect. http://collab.eyewriter.org/?p=102

[15] http://www.mathworks.com/support/solutions/en/data/12QTWCE/index.html?product=CO&solution=1-2QTWCE