1. ABSTRACT
This project aims to assist those with limited mobility of their hands by providing a hands-free alternative to using the computer. We plan to achieve this by developing a non-intrusive human-machine interface by means of gaze tracking whereby we will determine where the user is looking and display the results. We will use strategically placed infrared light emitting diodes to produce corneal glints from the user’s eyes. Image processing technology will be utilized to map the location of the glints with respect to a chosen physical landmark of the human eye (e.g. the pupil) onto the corresponding point of gaze on the monitor. The components of this project include: the design of the vision subsystem to capture images of the user’s eye, the image processing algorithm to detect the glint and determine its location, the correlation matrix transformation algorithm, development of a user-friendly GUI for calibration, operation and verification as well as experimental tests for performance analysis.

2. THEORY

Light travels through the human eye through two axes: the optical and the visual axis. The line of sight and the visual axis are functionally the same. The visual axis, however, varies from subject to subject. The optical axis which is a good approximation of the visual axis, is constant among subject. The optical axis passes through the center of the cornea, pupil, lens and retina. Thus, the center of pupil will be used to estimate the optical axis and thus the line of sight and determine where the user is looking.

With the center of pupil, we would still need to know where the user is looking with relation to the computer screen. According to Yoo et Al, an effective way to do this is putting four IR glints at the corner of the computer monitor. These four LEDs would reflect in the user’s cornea. The four LEDs would create a box with the center of pupil somewhere inside the box. Using the box created by the glint reflection and the center of the pupil, one can estimate exactly where the user is looking at the screen proportionally. The location of the pupil center with respect to the four reflected glints represents a scaled version of the user’s point of gaze on the monitor. This is known as the cross ratio approach.

3. HARDWARE

![Example of Output Image Obtained from Set-Up Above](image)

Materials: Logitech QuickCam Communicate MP Webcam, & 6mm IR LEDs (2 mounted on each corner)
Output:

![Hardware Setup to Obtain Images](image)

4. PROCEDURE

4a. Locating Interesting Zone of Eye

Most of the pictures captured contained excessive noise. In order to prevent this noise from interfering with our code, we extract an “interesting” zone from the picture. Our aim was to capture the full pupil and the four glints with as little “extra” as possible. We did this by scanning the entire image and searching for the 50 by 50 pixel rectangle with the most black. This method assumes that the pupil is always the darkest region in the image.

4b. Finding Center of Pupil

To find the center of the pupil, a silver, one pixel wide height of the image, is taken. Initially, the silver is located on the leftmost edge of the image. Note: The "sum value" of the pixel is considered the sum of the three values denoting the amount of red, blue, and green in the pixel respectively. The silver is moved from left to right until it reaches a point such that more than three pixels in the silver have sum values less than or equal to X (details on how we find X to follow). Where the silver stops is considered the left-most edge of the pupil. The same procedure is executed from the right, as well as the top and bottom. After the leftmost, rightmost, top, and bottom locations of the pupil are found, finding the center of the pupil is as simple as taking the average. The left and right averages are taken to find the width-wise location and the top and bottom averages are taken to find the height-wise location. Thus, the center of the pupil is located.

4c. Finding Glints

**GLINT ALGORITHM:** In order to more accurately detect the location of the four glints, we will use edge detection technique provided by Matlab functions. The Prewitt function returns the edges in binary where the gradient of the image is the maximum. Thus for our zone pictures, it returns all of our glints with little to no noise. To locate the glint’s exact location, the program scans the image and puts all pixels with a value of 1 into a glint table. To find G2, we look for the pixel with the least x and y pixel combination (G2 always in the top left). Using the location of G1, G2 table and G3 table are constructed for possible G2 and G3 locations. The brightest location of these two tables, give the location of the G2 and G3 glints. G4 was found similarly using the location of G3.

4d.e. Estimating Point of Gaze

**Estimating Gaze:** Once, the center of pupil is known and the glint locations are known, we can estimate where in the 3x3 grid the user is looking. A box with the four glint locations. The box created is split both in the x and y direction by three, creating a 3x3 grid. The x and y distance between G2 and the center of pupil is calculated and we call this distance-x and distance-y. Thus the program checks the: distance-x and distance-y and checks where in the box it is, thus placing in one of the nine grids. This essentially shows what part of screen the user is looking at.

![Prewitt's Edge Detection of Zone](image)

4f. Feedback – Adding GUI

**GUI:** The last part of the project is to provide feedback to the user. To do this, a GUI was created through MATLAB’s usercontrol functions. The GUI is a 3x3 grid that lights up in green when the user looks at the section. The GUI also sounds off the number that the user is looking at. For future work, this GUI could easily be constructed to create a game such as tic tac toe. It could also be used to help with the calibration process to provide more accurate results.

5. RESULTS

The system was tested with five users 20 times each and it was found that the system has an accuracy of approximately 84%.

6. FUTURE WORK/IMPROVEMENT

- As of now, our system requires the image to be captured manually and the MATLAB code has to be manually run by a 3rd party. Ideally we want a system that is automatic and operated in real time. This would involve adding a microprocessor that would be able to constantly capture images from the camera and automatically run the code.
- Our code is significantly dependent on room lighting. Without adequate infrared illumination, our system would not be able to work reliably. We need to improve our system by providing adequate IR illumination such that the system would not be dependent on sunlight.
- Ideally, this is meant for real time use of a computer as a computer mouse substitute. We could add in a segment that records the user’s previous point of gaze and incorporates this in a logical decision to determine where the user is looking now.

References


Huang, Ping, Tze-Pin, Sun, Juan, Lai, Yante, Wang, Shih-Chung, Chen, Shih-Chi, Huang, Ruey-Sen. “Bin, Qin, Narendra Ahuja, and Technical Guides. Web.”