Vision assistant: a human–computer interface based on adaptive eye-tracking

V. Hardzeyeu, F. Klefenz & P. Schikowski
Fraunhofer Institute for Digital Media Technology, Ilmenau, Germany

Abstract

The Vision Assistant is designed as an intelligent tool to assist people with different disabilities. The goal of this project is to replace the mouse and keyboard by an adaptive eye-tracker system (so called mouseless cursor) which will help to establish a universal and easy-to-use human–computer interface. Using a camera, it works with a pattern recognition algorithm based on a Hough–transform core to process the streaming image sequences. This technique is known for its performance in locating given shapes. In particular, it is used to extract the shapes that relate to the human eye and analyze them in real–time with the purpose of getting the position of an eye in an incoming image and interpreting it as the reference position of a mouse cursor on the user's monitor. The possibility of the Hough transform parallelization and its execution on the Hubel–Wiesel Neural Network for ultra fast eye-tracking is also discussed in this paper. The results of several experiments in this paper proved that the system performs quite well with different colours of the subjects' eyes as well as under different lighting conditions. In the conclusion we paid attention to the problems of further improvement of the functional and algorithmic parts of the Vision Assistant.
Keywords: HCI, eye-tracking, gaze estimation, Hough transform, Hubel–Wiesel neural network.

1 Introduction

Real-time systems that use eye-tracking technology have been explored for many years (Duchowski [1]). In our research we wanted to build a non-invasive video-based real-time eye-tracking system to let disabled people take a more active part in every day life, like writing messages, surfing the web, playing games, educating themselves and communicating to each other.

2 Proposed method

The Vision Assistant is based on the architecture shown in figure 1. It starts with an image acquisition step from the streaming video of a camera, which is connected to the computer. After that the image preprocessing takes place. At this stage we prepare the row image for processing in the Hough Core module. The Hough Core makes the transformation from the Cartesian coordinates to the Hough Space in which the shapes that are related to the human eye are located.

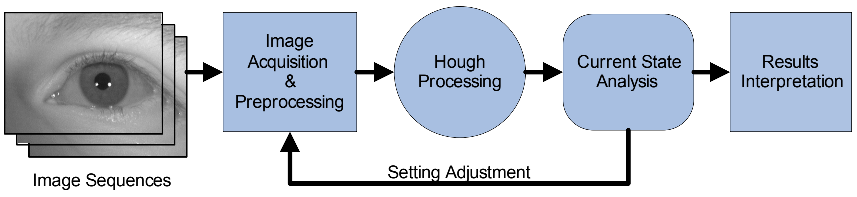

Figure 1: Vision Assistant's architecture.

After the human's eye shapes have been located, the Current State Analysis module inspects the features that were extracted from the image with the help of the Hough Transform and makes a setting adjustment for the next turn. The final block on this diagram is the block of the result interpretation, which interprets the calculated position of a human iris to the reference position of a mouse cursor on the user's monitor.

2.1 Image acquisition and image preprocessing

On the image acquisition stage, Vision Assistant grabs the image from the video stream through Direct X services to internal memory and converts it from the colour RGB-schema into grey scale. After this step we perform the Edge detection [2]. Edges are areas with strong intensity contrasts – a jump in intensity from one pixel to the next. In addition, they characterize object's boundaries on the image and therefore their detection makes further scene decomposition and feature extraction easier. Considering the procedure of edge detection applied to the image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image. In Vision Assistant for each pixel on the frame we apply a convolution algorithm, based on the Sobel 3x3 convolution matrices, which performs four operations:

a) calculation of the gradient information for the vertical direction is presented by eqn. (1):

$$\mathrm{G}_{\mathrm{y}} = \sum_{j}\sum_{i}(P_k \bullet maskY_{ji}) \tag{1}$$

b) calculation of the gradient information for the horizontal direction is presented by eqn. (2):

$$\mathrm{G}_{\mathrm{x}} = \sum_{j}\sum_{i}(P_k \bullet maskX_{ji}) \tag{2}$$

c) gradient calculation for a given pixel is presented by eqn. (3):

$$\left|\mathrm{G}_{\mathrm{k}}\right| = \sqrt{G_y^2 + G_x^2} \tag{3}$$

d) the mechanism of taking the decision upon current pixel is an edge or not is presented by eqn. (4):

$$P_{knew} = \begin{cases} yes, if\ G_k > \tau \\ no, if\ G_k \leq \tau \end{cases} \tag{4}$$

Here P_k is the current pixel of the image; P_{knew} is the pixel of edge; *maskX* and *maskY* are the convolution matrices for the Sobel operator and G_k is a computed value of gradient for the current pixel. The Threshold τ was chosen empirically through performing a number of tests for preprocessing.

2.2 Hough processing

After the incoming image has been prepared with the help of Sobelization for feature extraction, the Hough transformation takes place [3]. The Hough transformation is a pattern recognition technique which is known for its performance in locating given shapes on images. In particular, it is used to extract the shapes that relate to the human eye. In addition, the Hough transform is capable of identifying artefacts caused by shadowing, reflections and heavy light conditions, which is very important for eye-tracking. The final results of this transform represent the most reliable centers of both eyes.

Since the human's iris and pupil in most situations looks like a circle, the use of the adapted Circle Hough Transform makes sense for this application. A General circle representation can be described by eqn. (5):

$$(x-a)^2 + (y-b)^2 = r^2 \tag{5}$$

Here, *(a,b)* is the coordinate of the centre of the circle that passes through the set of points *(x,y)*, and *r* is its radius. Since there are three free parameters *(a,b* and *r)* for this equation, it follows that the results of the Hough transform should be represented by three-dimensional array. Considering transformation from Euclid to Hough space can be performed with the eqn. (6):

$$b = y - \sqrt{r^2 - (x-a)^2} \tag{6}$$

This is computationally expensive and circles clearly require more calculations to find than, for instance, lines. To solve this problem and achieve faster processing, we applied two-steps matching process:

In the beginning, the 1st step – 3D Hough Transform for circles takes place in the Hough Space of *a,b* and *r* (assuming that the radius can change only in range of [0 … 100] pixels and *a,b* are the dimensions of the current video frame). During the first iteration the system gathers all necessary information about the iris position *(a,b)* and its optimal radius r_{iris}. This process takes a lot of time, even for a 400 x 500 pixels image and, of course, can't be treated in real-time. However, these first iterations are approximately enough to gather the information about the optimal radius value and the offset position of the eye's centres. Besides, it's feasible that the eye position can't change dramatically from frame to frame if the operation performs at 25 frames per second, even if saccade movement (300 degree/s) takes place. Thus, we introduce the next step of the calculation process, the Reduced Hough Transform.

Since the optimal iris radius r_{iris} has been selected during previous phase, the system doesn't need to perform the Hough for all possible radiuses from 0 to 100 pixels. On this stage we assume that the radius can change from a small number of values and this gives the possibility to decrease the number of radiuses that should be taken into account during the calculation process. This process might be represented within the conditional eqn. (7):

$$r_{iris} - 3 < r_i < r_{iris} + 3 \tag{7}$$

where r_i is a radius on the certain frame.

In addition, a given estimation of the Iris positions on the image brings the possibility to decrease the region of interest for fine search of the Iris centres. These sub regions of interest will be bounded by rectangle with a height in the range of $[b_{i\text{-}1} - 2 \cdot r_i \ldots b_{i\text{-}1} + 2 \cdot r_i]$ and a width of $[a_{i\text{-}1} - 5 \cdot r_i \ldots a_{i\text{-}1} + 5 \cdot r_i]$, where $a_{i\text{-}1}$ and $b_{i\text{-}1}$ are consequently the horizontal and vertical positions of the iris centre on the previous frame. The windowing is shown in figure 2.

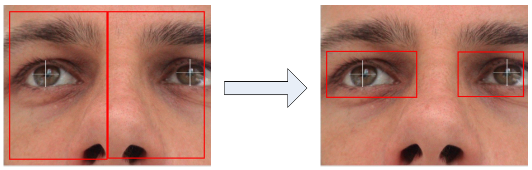

Figure 2: Presentation of the Full (left) and Reduced (right) Hough windows.

This approach has been organized with the help of a Current State Analysis block, which performs the parameter's recalculation for each step of the Hough Transform. In particular, it decreases the search window on the original image and as a consequence, decreases the complexity of the Hough Space as well. As a result, it brings an increase in the number of images processed in one second and the performance of the whole system grows rapidly to the real-time operation mode.

3 Implementation

We implemented the Vision Assistant with Microsoft Visual Studio.NET 7.1.3088, Framework .NET Version 1.1.4322 SP1. For visualization part, which captures the image from a camera, we used Direct X 9 SDK (Summer 2003 Release) on a computer with Pentium 4 2GHz processor and Direct X 9 compatible graphic card. The SONY DCR-PC110E camera with zoom, nightvision and autofocus functions was attached to the computer through Firewire. This choice was made because this camera has all necessary functions for plug-and-play operation. Its nightvision function replicates the work of InfraRed (IR) filter which is good to use for object tracking applications. Since Vision Assistant is working only with the region of the eyes the camera should have the possibility to zoom directly to the face. If the camera has an Autofocus function, this would be easier to adapt the focus of the camera during on-tracking operation. Using high speed IEEE-1394 (Firewire) camera-to-computer connection instead of USB will improve the performance of the application. All in all, the current system was able to proceed up to 30 frames per second from the camera and show the relevant cursor position on the screen.

Besides, to give more freedom to the user, at his or her choice the system can track left, right or both eyes simultaneously. This approach gives a possibility to choose the best way for tracking when the light conditions are not the same from left and right sides of the face (e.g. when the side sun light beams into the face).

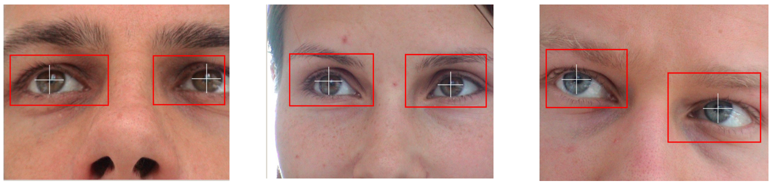

Figure 3: Examples of the subjects' faces.

4 Experiments

To test the Vision Assistant we performed several test sequences on people with different iris colours and different ethnic backgrounds. The subjects were asked to feel free with the computer and non-invasive camera and move their eyes as they do all the time. During the tests we found that ethnic background makes no effect on the software precision and performance. In figure 3 some images that were made during the tests with the Reduced Hough windows are presented.

After this we performed the test for the correct iris position recognition and coordinates interpretation in dependency of the eyes colour. The results of this test are presented in table 1:

Table 1: Correct iris position recognition in dependency of the eye colour.

Brown	Dark Brown	Light Blue	Green Brown	Dark Blue	Blue Grey
96.0%	95.5%	99.1%	97.2%	99.0%	98.6%

Additionally we have tested the algorithms for their performance in dependency of the software operational mode, because this feature is very important for any eye-tracking applications. Here Calibration mode means the work of the 1st processing step - Full 3D Hough Transform. Normal mode means the usage of the Reduced Hough windowing. The tests were performed on the P4-3.2 GHz with a Matrox Millennium G400 graphic card and a firewire camera. The results are presented in the table 2:

Table 2: Dependency between the calculation time and the working mode.

Running mode	Calculation time	Standard deviation	Average framerate
Calibration Mode			
Both Eyes	73ms	± 5ms	13.7 fps
One Eye	44ms	± 3ms	22.7 fps
Normal Mode			
Both Eyes	47ms	± 5ms	21.2 fps
One Eye	35ms	± 3ms	28.5 fps

As the reader can see from the table above the average performance has been reached the point of 22-28 frames per second.

5 Hough parallelization

This part of the development aims to create an Ultrafast Eyetracker which takes high-velocity pictures of the human eye and localizes the iris centres within few milliseconds. As it was mentioned above, the main algorithm for iris localization is based on the Hough transform which uses the “voting” technique for detecting the eyes shapes. However, as it is described in [4], this general method of the shapes localization can be well parallelized using the neural network approach. Being implemented on the FPGA (Field Programmable Gate Array) this will give an outstanding performance improvement of the eye-tracking system.

The parallelization approach uses the Hubel–Wiesel neural network. Neurobiologists David H. Hubel and Torsten N. Wiesel [5], who shared a Nobel Prize in medicine for discovering the V1 region of the brain, proposed a wiring diagram of the visual cortex. Based on this knowledge the considered neural network has been investigated its digitalized model has been created. The natural shape of the Hubel–Wiesel neural network and its digital representation is shown in figure 4:

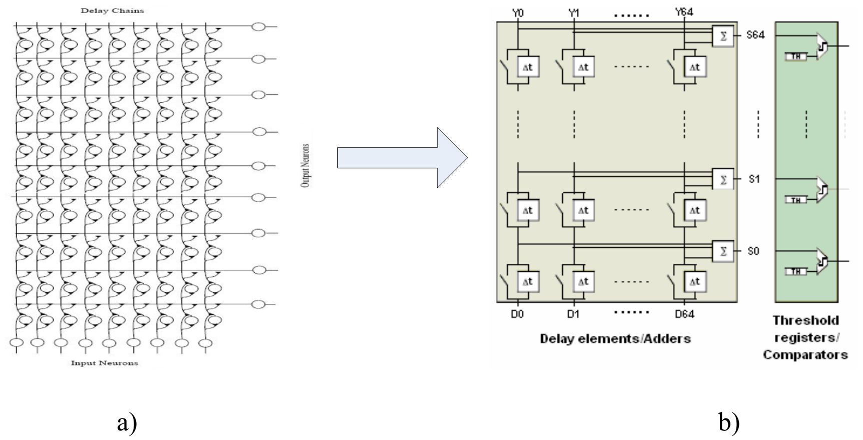

Figure 4: The natural (a) and digital (b) representation of the Hubel-Wiesel neural network.

This structure has proved to be a good basis for a highly dynamical computer vision system that is based on the cognitive principles like self-organization and self-modification. Furthermore, it is well suited for extracting the 2-D slopes and sinusoidal-like shapes from the image as well [6].

Using this technology, the Parallel FPGA Core is now under development and by the end it will be able to simulate the work of the Hubel–Wiesel neural network in order to execute the Hough eye center's localization routine on it.

Current implementation uses Xilinx Virtex II 6000 platform. An estimated frame rate of the considered system varies from 300 fps for the images of 512 x 512 pixels to 1500 fps for the pictures of the lower resolution.

6 Conclusion

In this paper we present a robust eye-tracking system that was adapted for the needs of the handicapped people. The system uses the Hough transform for circles. It produces information about the centers and radius of the eyes in real time. The possibility of the system design for high speed eye-tracking based on the bio-inspired approach was also discussed here.

Future work will concentrate on improving the robustness of the centre localization as an accuracy of one – two pixels is not sufficient enough. In addition, we will investigate and implement a method of dynamical thresholding for the preprocessing steps. This will give more tolerance to the different light conditions, e.g. darkness or intensive artificial light. We will also work on the prediction of the eyes region position during the tracking process. This will be used to create smooth cursor movement on the screen and, in consequence, will give the user more convenience during controlling the computer with his/her eyes.

References

[1] Duchowski, Andrew T. Eye Tracking Methodology: Theory and Practice. Springer-Verlag, p. 251, 2003.
[2] Nixon, Mark S., Aguado, Alberto S. Feature Extraction and Image Processing. Newnes, p. 350, 2002.
[3] Leavers, V.F. Shape Detection in Computer Vision Using the Hough Transform. Springer-Verlag, p. 201, 1992.
[4] Epstein, A., Paul, G.U., Vettermann, B., Boulin, C., Klefenz., F. A parallel systolic array ASIC for real time execution of the Hough-transform. E. S. Peris, A. F. Soria, V. G. Millan (Editors). Proceedings of the 12th IEEE International Congress on Real Time for Nuclear and Plasma Sciences, Valencia, pp. 68–72, 2001.
[5] Hubel, D.H., Wiesel, T.N., Stryker, M.P. Anatomical demonstration of orientation columns in macaque monkey. Journal of Comparative Neurology 177, pp 361–380, 1978.
[6] Brueckmann, A., Klefenz, F., Wuensche, A. A Neural Net for 2D-Slope and Sinusoidal Shape Detection, International Scientific Journal of Computing. Vol. 3, Issue 1, pp. 21 – 26, 2004.