Eye Tracking on Unmodified Common Tablets: Challenges and Solutions

Corey Holland, Oleg Komogortsev
Department of Computer Science
Texas State University – San Marcos
ch1570@txstate.edu, ok11@txstate.edu

Abstract

This work describes the design and implementation of an eye tracking system on an unmodified common tablet PC. A neural network eye tracker is employed as a solution to eye tracking in the visible spectrum of light. We discuss the challenges related to image recognition and processing, and provide an objective evaluation of the accuracy and sampling rate of eye-gaze-based interaction with such an eye tracker. The results indicate that it is possible to obtain an average accuracy of 4.42° and a sampling rate of 0.70 Hz with the described system.

CR Categories: B.4.2 [Input/Output and Data Communications] Input/Output Devices; I.4.8 [Image Processing and Computer Vision]: Scene Analysis – Tracking

Keywords: eye tracking, face detection, eye detection, neural networks, eye-gaze-guided interface

1 Introduction

Eye tracking provides an almost seamless form of interaction with the modern graphical user interface, representing the fastest non-invasive method of measuring user interest and attention. While the mouse, keyboard, and other touch-based interfaces have long reigned as the primary input mediums associated with the field of human-computer interaction, as advances continue to improve the cost and accuracy of eye tracking systems they stand poised to contend for this role.

Commercial eye trackers employ a variety of methods to quantify user attention. Popular among these are the scleral search coil, electro-oculography, and pupil tracking techniques [Duchowski 2007; Eggert 2007]. Unfortunately, these techniques are limited by constraints that make them impractical for the average computer user. Commercial eye trackers often require the use of large external apparatus, extensive setup and calibration, and expensive material. The result is a non-portable and difficult to use system that may cost many times the price of a common desktop PC.

Recent work has shown that the techniques employed by infrared pupil tracking are equally applicable to any device with access to sufficient processing power and a camera [Agustin, et al. 2009; Sewell and Komogortsev 2010]. With this in mind, tablet PCs seem to be an ideal candidate for the application of such techniques. The standard mouse and keyboard are often impractical for use with tablets, instead relying largely on touch, a process which is often slow and tedious. Many tablets include a built-in camera and the processing power required for basic image manipulation, the essential components of a video-oculography system [Hansen and Ji 2010] and, in comparison to commercial eye tracking systems, tablets are relatively cheap and portable. In this paper, we consider the design and implementation of an eye tracking system for an unmodified common tablet, along with an objective evaluation of the performance of such a system.

2 Challenges

Though tablets are equipped with the essential elements necessary for the construction of an eye tracking system, they are far from ideal. Limited processing power, a standard/sub-standard camera, and even the portable nature of tablets complicate matters by introducing additional sources of error; and while face detection on an unmodified common tablet [Allan 2012] and eye detection on an unmodified cell phone [Miluzzo, et al. 2010] has been achieved previously, the realm of eye tracking is as of yet completely unexplored.

Tablets are much closer to the common cell phone or PDA than to a desktop computer, maintaining very limited resources for the sake of increased portability, they are incapable of running complex applications with acceptable speed. In addition, the API available to tablets is often only a subset of what is available to their desktop counterparts, placing further constraints on the development process, and making it necessary to re-implement what could be considered common or standard functionality. Similarly, the built-in camera, a common feature of many tablets, often has a lower resolution and frame rate than is available to commercial eye tracking systems. As well, common webcams generally contain filters at the hardware level that remove wavelengths of light outside of the visible spectrum (e.g. infrared).

These issues result in two additional complications: the lack of a consistent external light source causes the eye tracker to be more sensitive to the lighting conditions of the environment; and the free range of motion available to tablets complicates the algorithms responsible for determining relative eye position.

3 Design & Implementation

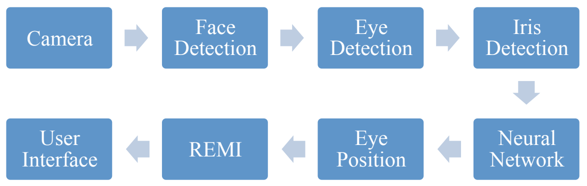

Figure 1. Eye tracking process.

As a starting point for the implementation of an eye tracking system, we consider the steps necessary for determining eye position from an image of the user. A video frame is retrieved from the camera, converted to an appropriate format, and passed to the face detection algorithm. The face detection algorithm identifies the face and passes the coordinates to the eye detection algorithm. The eye detection algorithm uses the bounds of the face to reduce the search area, identifies the eyes within the face, and passes the coordinates of the eyes to the iris detection algorithm. The iris detection algorithm uses the bounds of the eyes to once again reduce the search area, identifies the iris within the eye, and passes the coordinates of the iris to a pre-processing algorithm. The pre-processing algorithm extracts key features of the eye from the image and passes them to the neural network, which in turn identifies the screen position on which the eye was focused.

3.1 Image Recognition

Image detection algorithms are computationally expensive, a limited resource on most portable devices. To alleviate this issue, the size and resolution of the original image is reduced by ½ (from 640 × 480 pixels to 320 × 240 pixels) and converted to gray-scale before being passed to the face detection algorithm.

Haar classifiers, a method of object detection originally developed for facial recognition [Paul 2001], are employed for face and eye detection. The classifier is trained with a set of features and is able to quickly reject false regions of the image, reducing the amount of time spent searching. Iris detection makes use of visual template matching to search for the iris by sequentially comparing overlapping regions of the image [Cole, et al. 2004].

At each sequential stage of image recognition (face, eye, and iris), the size and coordinates of the identified image segment are passed to the subsequent stage, reducing the search area. To improve the speed of face recognition, the most computationally expensive step, the previously identified window is used as a starting point when searching each frame.

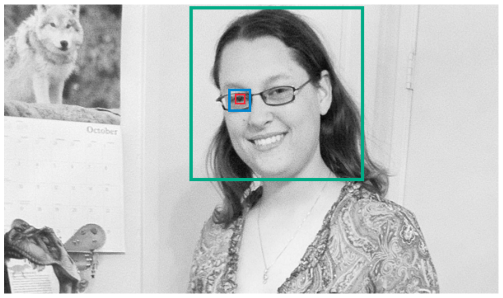

Figure 2. Image recognition.

By steadily narrowing the search field from the easiest to identify to the most difficult, the ability to accurately identify the eye is improved. Unfortunately, a failure to identify the appropriate features at any point in the process results in the loss of the frame and any information it may have contained.

With the limited processing power available to tablet PCs, each stage of image recognition has a substantial impact on the overall performance. We hypothesize that in a mobile environment the accuracy gains of face detection may be outweighed by the reduction in sampling rate.

3.2 Neural Network

The common analogy is that artificial neural networks are designed to mimic the ability of biological neural networks to recognize and identify patterns [Nissen 2005]. Neural networks are primarily used for function approximation; the neural network learns to map a set of inputs to the appropriate outputs.

We wish to map the image of the eye to a position on the screen. To do so, however, we must first select a quantifiable aspect of the image that is capable of indicating the position of the iris. This requires that our input values be derived from the pixels that compose the image, rather than more general qualities of the image itself (i.e. size, resolution, etc.). This also requires that the number of input values remains constant; hence the size of the image must be normalized before input values are derived.

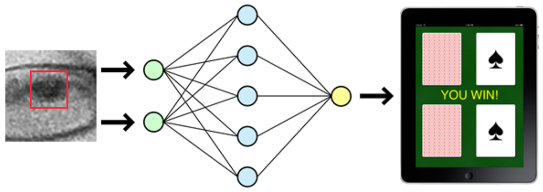

Figure 3. Neural network.

The size and position of the iris is mapped to the gray-scale image, and a segment of fixed size is selected from the image around the iris. The image size is selected to be large enough to encompass the eye itself, but small enough to exclude extraneous portions of the face. The brightness of each pixel within the segment is used as an input value to the neural network, giving a number of input neurons equal to the number of pixels in the image segment, and two output neurons for the (x, y) screen coordinates.

A basic two-layer neural network with a symmetric sigmoid activation function was employed in the current context. Additional layers may improve the accuracy of a neural network, but increase the size of the required training set considerably. Conversely, increasing the number of nodes in each layer does not necessarily increase the accuracy of the network.

3.3 REMI

The real-time eye movement identification protocol (REMI) was designed to standardize the processing and interpretation of eye movement data at the software level [Koh, et al. 2010]. The REMI protocol consists of three primary modules: processing, classification, and action. These modules describe the steps common to many eye-gaze-guided applications developed for desktop environments, and as such require some amount of modification.

To this end, we have reduced the implementation of the REMI protocol to the bare minimum necessary to convert the raw eye movement data into an appropriate interface event. The solution chosen for our purposes is the removal of the processing module, and implementation of an area-of-interest identification algorithm (I-AOI) for classification [Salvucci and Goldberg 2000]. When a new set of coordinates is provided by the neural network, the classification module determines whether the point falls within one of the primary interface elements; if so, the point is classified as part of a fixation.

The action module monitors the duration of fixations within each interface element. If the duration of a fixation exceeds a given threshold (which is largely dependent on the temporal resolution of the eye tracker), a touch event is simulated at the positional centroid of the fixation points within the region. To avoid the Midas Touch problem, it is necessary to define a sufficient duration for gaze point collection in each region, as well as a set radius around interface elements within which interface events are registered, so that the region assigned to each interface element is greater than the accuracy of the eye tracking system.

4 Methodology

To evaluate the proposed eye tracking system, a proof-of-concept application was developed, using the Open Source Computer Vision library (OpenCV) for image recognition and the Fast Artificial Neural Network library (FANN) for the neural network [Intel 1999; Nissen 2002].

4.1 Apparatus

The experiment was conducted on the Apple iPad 2, with a 1 GHz dual-core processor, 512 MB memory, and a screen resolution of 1024 × 768. The built-in front camera has a resolution of 0.3 megapixels and is capable of video capture at 30 frames per second. A portable easel was used to secure the iPad in a fixed, upright position to establish a reliable performance baseline.

4.2 Participants

A total of 13 subjects volunteered for the first experiment phase and provided informed consent, of which there were 11 male and 2 female. Among these, 8 had normal vision and 5 required corrective lenses. Of the participating volunteers, 11 provided usable data and 2 were unable to complete the experiment. Participants unable to complete the experiment suffered from astigmatism, due largely to failure of the face detection algorithm.

A total of 9 subjects volunteered for the second experiment phase and provided informed consent, of which there were 5 male and 4 female. Among these, 3 had normal vision and 6 required corrective lenses. All participants completed the experiment.

4.3 Procedure

The tablet was secured on an easel with the front-facing camera at approximately eye-level. Participants were seated 23 inches (± 2.75 inches) from the camera and were allowed a free range of head motion. Each participant was given 5 minutes to complete calibration and 5 minutes to complete verification.

During calibration, a series of dot stimuli were presented on the screen in an evenly spaced 4 × 4 grid (16 stimuli total) covering the entire screen. Each stimulus point remained visible until 2 valid image samples were collected for the point, the stimulus point was then hidden, and the next stimulus point in the sequence was presented.

During verification, the calibration procedure was repeated, presenting the same series of dot stimuli on the screen in an evenly spaced 4 × 4 grid (16 stimuli total). Again, each stimulus point remained visible until 2 valid image samples were collected for the point, the stimulus point was then hidden, and the next stimulus point in the sequence was presented.

Training of the neural network occurred incrementally for each stimulus sample collected during both calibration and verification. Accuracy calculations were performed during verification, while the sampling rate was measured during both calibration and verification.

The first experiment phase tested the eye tracking system in its described state. The second experiment phase tested the eye tracking system with the removal of face detection to determine if the overall usability of the system could be enhanced.

5 Results

5.1 Accuracy

The accuracy is calculated as the error in degrees of the visual angle between the stimulus position and the position calculated by the neural network. The average accuracy of the eye tracking system was 3.55° (±0.42) or 186.38 pixels (±21.13) for the first phase, and 4.42° (±0.55) or 203.85 pixels (±17.02) for the second phase, roughly a fifth of the available screen size. These results were statistically significant and indicate an accuracy reduction of 24.5% with the removal of face detection from the eye tracking system ($F(1, 18) = 15.61$, $p < 0.001$). In comparison, the neural network eye tracker described by [Sewell and Komogortsev 2010] and run on a standard desktop PC provided an accuracy of 3.68° (±4.24) and made use of a greater training set for each subject.

5.2 Sampling Rate

The sampling rate is calculated as the usable frames processed by the neural network per second. The average sampling rate of the eye tracking system was 0.23 Hz (±0.08) or roughly 1 usable frame every 5 seconds for the first phase, and 0.70 Hz (±0.08) or roughly 1 usable frame every 1.5 seconds for the second phase. The average task completion time was 5.35 minutes (±2.19) for the first phase, and 1.55 minutes (±0.18) for the second phase. These results were statistically significant and indicate a sampling rate increase of 304.3% with the removal of face detection from the eye tracking system ($F(1, 18) = 156.96$, $p < 0.001$).

6 Discussion

The iPad 2 is capable of processing roughly 8 medium-resolution frames per second from image recognition to the neural network. The low sampling rate obtained during experimentation is due in large part to the failure of image recognition algorithms to identify the area of interest within each frame. By comparing the results of the first and second experiment phases, the effects of the iPad's limited processing power are obvious. Face detection may provide a marginal improvement in eye tracking accuracy; however, the sampling rate is substantially improved with its removal. As well, a portion of the accuracy difference between the two systems may be explained by the skewed proportion of subjects with normal vision (8 in the first phase, 3 in the second phase).

7 Limitations

The primary limitation of the described system is its reliance on a successive series of image recognition techniques. A failure at any point to identify the required feature (face, eye, or iris), due either to a varied spatial configuration or subject features dissimilar to the recognition template, results in the loss of any information that may have been gained from the frame. With the available frames per second already limited by camera and processing constraints, the loss of any frame has a noticeable effect on the overall accuracy and consistency of the eye tracker.

As well, due to the lack of adequate correction for the spatial configuration of the head in relation to the tablet, it is necessary to maintain relatively fixed positions for both the tablet and head when utilizing the described system. This is less than desirable, as the primary benefit of tablets is their portable nature; however, this issue is not insurmountable.

In addition, the low spatial resolution, temporal resolution, and spatial accuracy of the proposed system make it unsuitable for use with data collection or complex interfaces. This imposes a severe limitation on the applications of such a system, but as mobile devices become ever more powerful, the range of applicable eye tracking techniques will improve in parallel.

8 Future Research

Future research in this area will likely focus on: optimization of image recognition algorithms, improvements in the heuristics used to estimate eye position, and more advanced methods of image manipulation and machine learning.

Optimization of image recognition algorithms is necessary to improve the achievable temporal resolution of the eye tracker. With limited processing power available, and the large overhead of image recognition and manipulation algorithms, every clock cycle saved is a clock cycle earned.

Improvements in the heuristics used to estimate eye position may improve both the spatial resolution and accuracy of the eye tracker. For example, estimating the user's distance from the screen based on the distance between pupils, or more precise pre-trained Haar classifiers for face and eye detection. As well, the accuracy of the neural network can easily be enhanced with the implementation of additional layers and an extended training routine.

More advanced methods of image manipulation and machine learning are necessary to increase the overall portability of the described system. Specifically, head modeling is necessary to measure the relative angles between the tablet and the eye, allowing automated correction for the variable orientation of user and tablet. Further, image manipulation techniques may be applied to normalize images distorted by the position or environment of the user, and more advanced machine learning techniques may be applied to improve image recognition and eye movement prediction, directly affecting the core functionality of the eye tracker.

As mobile devices continue to evolve, increased computational power will have the greatest effect on the efficacy of these techniques, allowing greater complexity and accuracy in image recognition and processing. At present, the iPad is unable to take full advantage of the front camera due to processing restrictions.

9 Conclusion

In this paper, we have described the design and implementation of an eye tracking system with the use of an unmodified common tablet. Through an objective evaluation, the described system obtained an average accuracy of 4.42° and a sampling rate of 0.70 Hz. It was also found that the removal of face detection from the system provided a substantial improvement in the sampling rate with minimal loss of accuracy. While the accuracy of the described system has much room for improvement, it represents a portable and inexpensive alternative to commercial eye tracking systems, and presents a viable means of interaction for disabled users. Eye tracking promises to fill the gap left by the keyboard/mouse as portable devices continue to evolve.

This work was partially funded by a grant from the Na-tional Institute of Standards #60NANB10D213 and grants from Texas State University. An open-source proof-of-concept application, NNET, will be made available for download through the App Store for instructive purposes, and source code will be made available from http://cs.txstate.edu/~ok11/.

References

AGUSTIN, J. S., SKOVSGAARD, H., HANSEN, J. P. and HANSEN, D. W. 2009. Low-cost gaze interaction: ready to deliver the promises. ACM, Boston, MA, USA.

ALLAN, A. 2012. Face Detection. *iOS Sensor Programming*, O'Reilly. 1-320.

COLE, L., AUSTIN, D. and COLE, L. 2004. Visual Object Recognition using Template Matching. In *Proceedings of Australian Conference on Robotics and Automation*, vol.

DUCHOWSKI, A. T. 2007. Eye Tracking Methodology: Theory and Practice. Springer-Verlag, London.

EGGERT, T. 2007. Eye Movement Recordings: Methods. *Neuro-Ophthalmology*, Karger. 15-34.

HANSEN, D. W. and JI, Q. 2010. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 3, 478-500.

INTEL. 1999. Open Source Computer Vision (OpenCV). Available: http://sourceforge.net/projects/opencvlibrary/.

KOH, D. H., GOWDA, S. M. and KOMOGORTSEV, O. V. 2010. Real time eye movement identification protocol. ACM, Atlanta, Georgia, USA.

MILUZZO, E., WANG, T. and CAMPBELL, A. T. 2010. EyePhone: activating mobile phones with your eyes. ACM, New Delhi, India.

NISSEN, S. 2002. Fast Artificial Neural Network (FANN). Available: http://sourceforge.net/projects/fann/.

NISSEN, S. 2005. Neural Networks Made Simple. *Software 2.0 2*, 14-19.

PAUL, V. 2001. Rapid Object Detection using a Boosted Cascade of Simple Features. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 511-511.

SALVUCCI, D. D. and GOLDBERG, J. H. 2000. Identifying fixations and saccades in eye-tracking protocols. ACM, Palm Beach Gardens, Florida, United States.

SEWELL, W. and KOMOGORTSEV, O. 2010. Real-time eye gaze tracking with an unmodified commodity webcam employing a neural network. ACM, Atlanta, Georgia, USA.