Journal of Eye Movement Research
1(1):1, 1-2

Interacting with Objects in the Environment by Gaze and Hand Gestures

First Author
First affiliation

Second Author
Second Affiliation

Third Author
Third Affiliation

Head-mounted gaze trackers can be used for gaze estimation purposes in mobile scenarios. A wireless gaze tracker in the form of gaze tracking glasses permits the continuous monitoring of a subject's point of regard on the surrounding 3D environment. In this work, we combine gaze tracking and hand gesture recognition to allow a subject to interact with objects in the environment by gazing at them (pointing), and controlling the object using hand gesture commands. A set of gaze tracking glasses is made from low-cost hardware which consists of glasses' frame, eye tracking camera and the scene camera. An open source gaze estimation software is used for eye tracking and obtaining the user's gaze data. A binary beacon recognition library is used to identify objects in the environment. A hand gesture classification algorithm is used to recognize hand-based control commands. When combining all these elements, the emerging system permits a subject to move freely in an environment, select the object he wants to interact with by gaze, and transmit a control command to it by performing a hand gesture. The specific object is identified by the binary beacon visible in its surface. This innovative HCI paradigm opens up new forms of interaction with objects in an smart environment.

Keywords: Keywords: Gaze Tracking, Head-Mounted Gaze Tracker, Mobile Interaction, Gaze Interaction, HCI

Introduction

Body language and gaze are important forms of communication among humans. In this work we present a system that combines gaze pointing and hand gestures to interact with objects in the environment. Our system merges a video-based gaze tracker, a hand gesture classifier and a visual marker recognition module into an innovate HCI device that permits novel forms of interaction with electronic devices in the environment. Gaze is used as a pointing mechanism to select the object with which the subject wants to interact. A visual binary marker in the surface of the object is used for identification of the object by the system. Finally, a hand gesture is mapped to a specific control command that makes the object carry out a particular function.

Eye tracking refers to the monitoring of eye movements. Gaze tracking is the process of measuring the point of regard (where a person is looking). A head mounted eye tracker is a device that permits measuring eye movement while the subject is moving around an environment and estimate the position of gaze in the environment. Eye trackers are used in research on the visual system, in psychology, in cognitive linguistics, product design and in Human Computer Interaction (HCI) research (?, ?). There are a number of methods for measuring eye movement. The most popular variant uses video images from which the eye position is extracted. Other methods use search coils or are based on electrooculography.

Automatic gesture recognition is a topic in computer science and language technology that strives to interpret human gestures via computational algorithms. Gestures can originate from any bodily motion or state but commonly originate from the face or hands. An appealing feature of gestural interfaces is that they make it possible for users to communicate with objects without the need for external control devices. Hand gestures are an obvious choice as a mechanism to interact with objects in the environment (?, ?, ?).

Automated hand gesture recognition is challenging since in order for such an approach to represent a serious alternative to conventional input devices, applications based on computer vision should be able to work successfully under uncontrolled light conditions, backgrounds and perspectives. In addition, deformable and articulated objects like hands represent added difficulty both for segmentation and shape recognition purposes.

Visual marker recognition systems consist of a set of patterns that can be detected by a computer equipped with a camera and an appropriate detection algorithm (?, ?). Markers placed in the environment provide easily detectable visual cues that can be associated to specific objects for identification purposes. In this work, we use the open source library XXXXXXXXXXXXXXXXX FOR DIAKO TO FILL XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX. This library provides our system with the ability to recognize specific objects in the environment.

The hand gesture recognition module we developed here is able to detect a hand in front of the scene camera of the gaze tracking glasses and the number of fingers that the hand is holding up. We define the gesture as holding the hand with a preset number of fingers for a predefined dwell time of 1 second and moving it in a particular direction: up, down, left or right. The hand gesture recognition worked well for natural skin color, but using a latex glove of a color not present in the environment improves performance.

We used an open source gaze tracker, described in (?, ?), to monitor user's gaze. The gaze tracker provides real-time gaze estimation over the user?s field of view, as captured by the scene camera attached to the glasses,

The system recognizes objects in the environment by detecting visual markers associated to them through the scene camera also referred to here as field of view camera. When the subject carrying the glasses gazes at an object, the visual marker placed nearby is recognized by the visual marker recognition library operating over the scene camera video stream. When a visual marker has been detected, a temporal window opens for detection of a hand gesture on the part of the user.

To interact with an object in the environment, the user of the glasses needs to first look at the visual marker associated to the object that identifies the object and then perform the hand gesture. In this way, only that particular object in the environment will react to the hand gesture.

In Summary, this work represents a proof of concept for an innovative form of interacting with objects in the environment by combining gaze and hand gestures. Interaction is achieved by gazing at an object in the environment and carrying out a hand gesture. The hand gesture specifies a certain command and gazing at the object, and the visual marker associated to it, make only that specific object to respond to the subsequent hand gesture. The off-the-shelf components used to build the hardware, the low cost of building the entire system and the open source nature of the algorithms used, make this form of interaction amenable for spreading among academic institutions and research labs to further investigate and stretch the possibilities of this innovative HCI paradigm.

Related Work

Head mounted eye trackers

Gaze for Interaction

Gestures in HCI

Please cite (?, ?), (?, ?) and (?, ?)

Combining gaze and gesture

Gesture for Interaction

Gaze enhanced interaction

Method

Gaze Tracking

Eyes are used by humans to obtain information about the surroundings and to communicate information. When something attracts our attention, we position our gaze on it, thus performing a *fixation*. A fixation usually has a duration of at least 150 milliseconds (ms). The fast eye movements that occur between fixations are known as *saccades*, and they are used to reposition the eye so that the object of interest is projected onto the fovea. The direction of gaze thus reflects the focus of *attention* and also provides an indirect hint for *intention* (?, ?).

A video-based gaze tracking system seeks to find where a person is looking, i.e. the Point of Regard (PoR), using images obtained from the eye by one or more cameras. Most systems employ infrared illumination that is invisible to the human eye and hence it is not distracting for the user. Infrared light improves image contrast and produces a reflection on the cornea, known as corneal reflection or glint. Eye features such as the corneal reflections and the center of the pupil/iris can be used to estimate the PoR. Figure 1 shows a screenshot of an eye being tracked by the open-source Haytham Gaze Tracker (?, ?). In this case, the center of the pupil and two corneal reflections are the features being tracked.

Depending on the hardware configuration of the different components, gaze tracking systems can be classified as either *remote* or *head-mounted*. In remote systems, the camera and the light sources are detached from the user, normally around the device screen, whereas in head-mounted systems the components are placed on the user's head, usually mounted on a helmet or a pair of safety glasses (?, ?).

Head-mounted eye trackers can be used for mobile interaction as well as gaze estimation purposes. In this work, we have build a low-cost headmounted eye tracker using off-the-shelf components. The system consists of safety glasses, batteries, wireless eye camera and the wireless scene or field of view camera. The wireless eye camera is equpped with infrared emitting

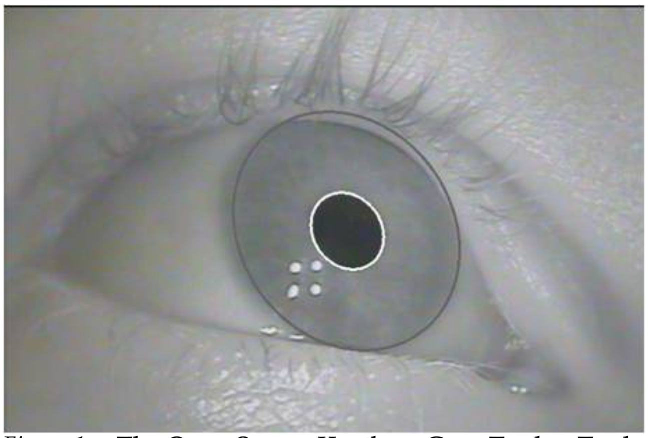

Figure 1. **The Open Source Haytham Gaze Tracker Tracking the Eye.** The features tracked in the image are the pupil center and two corneal reflections. These features are used by the gaze estimation algorithms to determine the PoR of the user on the scene camera.

diodes that permit the associated software to monitor the position of the pupil on the image and the infrared reflection in the iris. The vector difference between the pupil center and the glint reflection in the iris can be tracked during a calibration procedure to build a user specific model of his gaze. The calibration procedure consists on the user looking at a number of points on the environment and marking them on the scene camera video stream while the user fixates on them. Once a calibration procedure is completed, the gaze estimation algorithm is able to determining the point of regard of the user in the environment.

We use the open-source Haytham (?, ?) gaze tracker developed by Diako Mardanbegui at the IT University of Copenhagen for mobile gaze estimation and extended it with a hand gesture recognition module that enables specific interaction with objects in the environment through gaze and hand gestures.

Constructing the eye tracking glasses

Figure 3
Figure **??**

Apparatus

Eye-tracking system.

Results

Materials: Plastic safety glasses Wireless camera with infra-red LEDs (eye camera) Wireless camera (scene camera) Tin strips Araldite Steel wire Tape Double sided tape 9V battery Tin snips

1. An area was traced onto the lens of the glasses where the eyes will be approximately when the user puts on the safety googles. 2. Using the tin snips, the plastic part of the lenses bounded by the previously

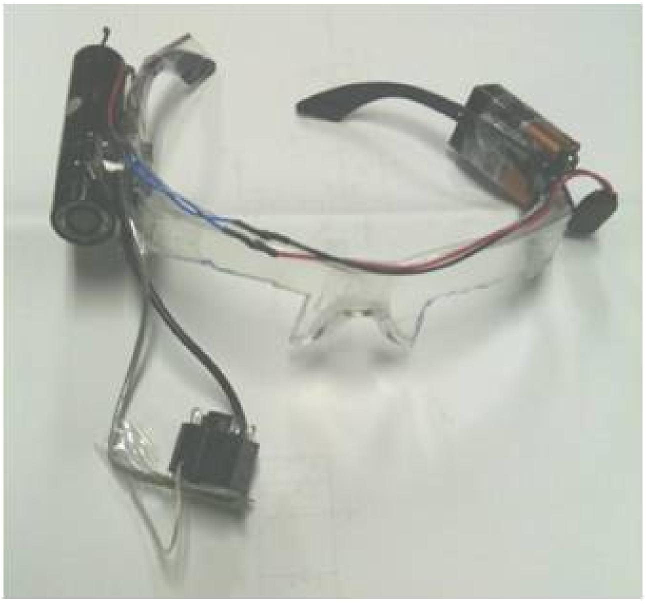

Figure 2. **Low Cost Gaze Tracking Glasses.** Bla bla.

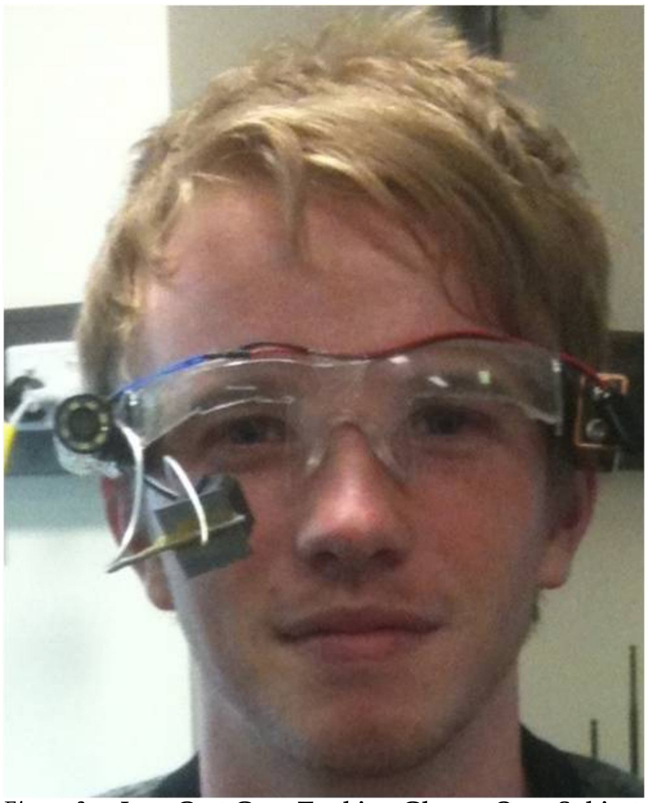

Figure 3. **Low Cost Gaze Tracking Glasses On a Subject.** Bla bla.

traced area was cut away. It is important that the majority of the structure of the glasses is left intact to allow for stability. 3. Tin was cut to the size and shape of the infrared camera using the tin snips. 4. Steel wire was cut to a size of 25cm. 5. The steel wire was then attached to piece of tin using araldite. 6. Using double sided tape, the tin was attached to the back of the camera. 7. The steel wire was then bent into an 'L' shape and attached to the right side of the glasses (frame) using tape. 8. The wires for the battery were extended and the 9V battery was attached to the left side of the glasses to ensure that the glasses are not unevenly balanced. 8. Using the haytham software, the position of the camera was checked to ensure the camera feed is filming the entire eye. It was found that the best position of the eye is below the glasses so it doesn't obstruct the user's vision. 8. The scene camera was mounted to the right of side of the glasses using sticky tape.

Finger detection 1. Scene camera feed is colour thresholded to check for skin colour or latex glove. For checking the skin colour the image is converted to Ycc colour space. When checking for the latex glove the image is converted to HSV space. 2. Blobs are cleaned up using morphological operations. 3. The largest blob in the image is defined as the hand and all other blobs are discarded. If the blob is not above a certain area it is treated as noise and also removed from the image. 4. The convexity hull of the blob is calculated using openCV's library. INSERT IMAGE HERE SHOWING CONVEXITY DEFECTS 5. End and start points of the convexity defects are located. 6. If the location of the end and start points satisfy the custom heuristic equation, the defect is defined as a finger. 7. Each defect is checked and the number of fingers is therefore the total defects that satisfy the equation.

Gaze Tracking Glasses

Visual Markers Recognition

This is a test Figure 4

Hand Gesture Recognition

Figure 5

Putting it all together

Figure 6

Application

We carried out a small pilot study to test the functionality and performance of the system. We decided to test the system in a environment where 3 "smart" objects could be controlled by the system simultaneously: a computer, a set of leds in a bread board and a robot. The hand gesture recognition module could recognize 5 different states of the hands as defined by the number of fingers being held up: 1, 2, 3, 4 and 5. A gesture was

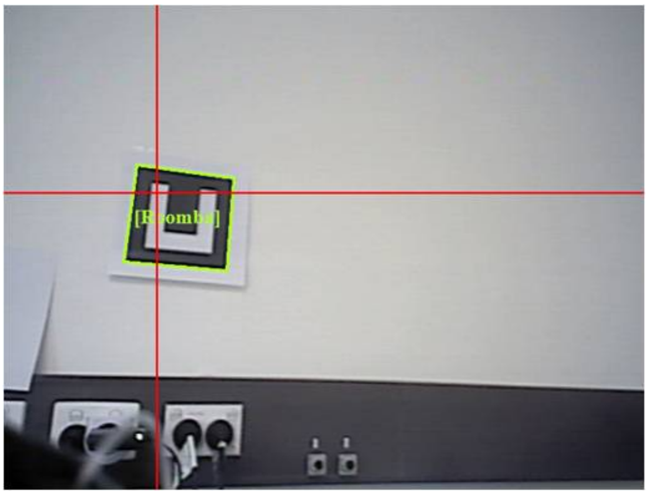

Figure 4. **Visual Marker Recognition.** Bla bla bla.

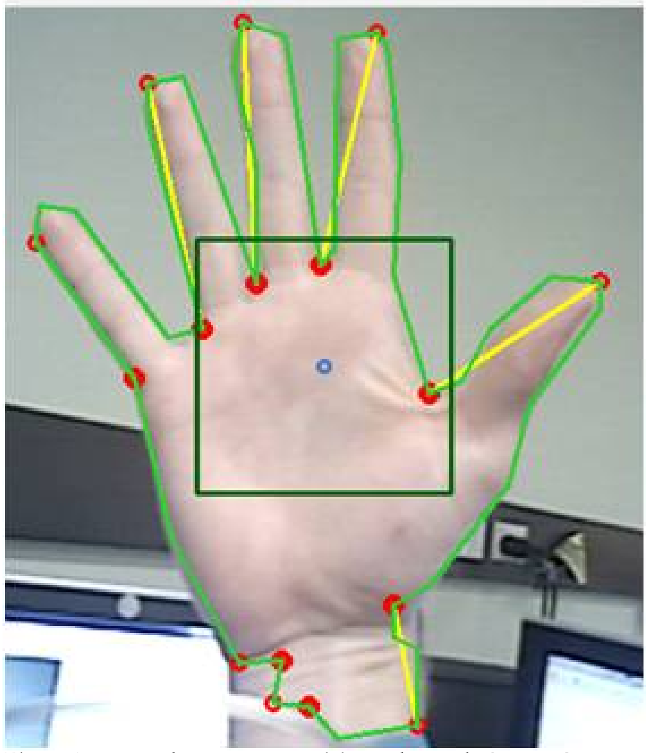

Figure 5. **Hand Pose Recognition Through Scene Camera.** Bla bla.

defined as one of these 5 states plus one of four spatial directions: up, down, left and right.

The infrared leds could just be turned on and off. Two fingers being held up and an upward movement would turn on the leds. Four fingers being hold up and a movement to the right would turn them off.

The same hand gestures were used to control the computer. The gestures were mapped to minimizing the current window on display in the computer GUI and to bring them back up. This was done just as a

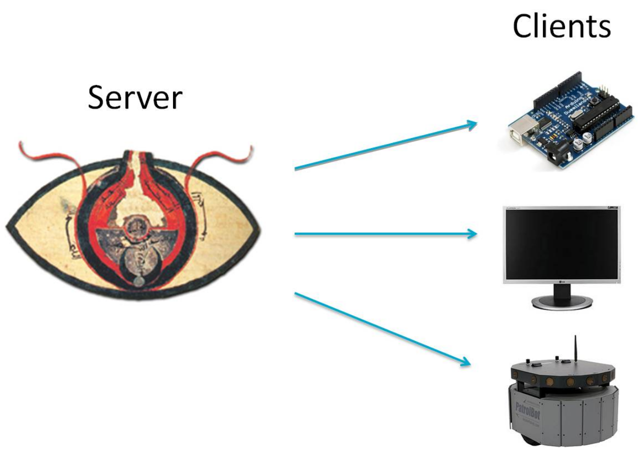

Figure 6. **System Diagram.** Bla bla.

proof of concept, any other type of gestures associated to different control commands could be envisioned and implemented.

The robotic control was the most elaborated one. The robot could be made to move forward, backward, right and left. The numbers of fingers being held up indicated, either the speed for forward and backward movements or the amount of turn to be made for right and left movements.

The hand gestures could be done with bare hands, but we notice that in environments where the color of the walls could resemble the skin hue, hand gesture recognition performance would suffer. Using a glove with a distinctive color, not present in the rest of the environment enhanced hand recognition performance.

This manuscript's associated video [1] provides a good visual overview of the system at work and how it is being used by two different users to interact with a computer, a breadboard with a set of light emitting diodes and with a robot.

Discussion and Conclusion

In this work we have shown how to interact with objects in the environment through an innovative combination of gaze and hand gestures using a set of gaze tracking glasses and a hand gesture recognition module. The method is easily extensible to multiple objects in the environment and a wide array of hand gestures.

The low-cost headmounted eye tracker used and the gaze estimation algorithms employed do not compensate for parallax error, i.e. the inability to differentiate between the working plane and the calibration plane (?, ?). This limits the ability to alternating interaction with objects at a distance and objects up close. Nonetheless, since the scene camera used in the glasses is relatively close to the eye being tracked, see Figure 3, the parallax error was minimized. Furthermore, we notice that during the calibration, using calibration points situated at different distances (from 1 to 10 meters) would find a compromise between objects far away and objects up close and would generate good gaze estimation for all sort of distances. We noticed that gaze estimation accuracy was never an issue for our system. Only over time, if the glasses would move slightly from their position during calibration, due to sweat on the skin and the glasses sliding slightly, would gaze estimation degrade marginally.

We did notice problems with the skin detection algorithms when the hand was position within the field of view of the scene camera. This was markedly noticeable, when the colors of the background were similar to the skin color. Usage of more sophisticated skin detection algorithms could help to solve this issue.

An important issue of the system was the fact that the user wearing the glasses did not have any sort of feedback in terms of where within the field of view of the scene camera the hand was placed when it was about to initiate a hand gesture. This was due to the lack of a display on the glasses to provided visual feedback in terms of how the hand is positioned within the field of view of the scene camera. We implemented an auditory feedback signal to indicate that the system had found the hand within the field of view of the scene camera and it was therefore ready to receive a gesture. We found it that this helped the user but still did not provide real time feedback to carry out small corrections of hand positioning for proper positioning within the field of view of the scene camera. This issue was due to the usage of an scene camera with a relatively narrow field of view. Using an scene camera with a wider field of view should prevent the need of feedback for proper hand positioning since the hand would always fall within the field of view of the scene camera as long as the arm was stretched in front of the user.

Further work should strive to carry out an extensive quantitative analysis of the performance of the system within a large user study. More shopisticated hand gestures can also be envisioned. However, complex gaze gestures generate a cognitive and physiological load on the user. Cognitively it is difficult for users to remember a large set of complex gestures, and physiologically it is tiring and challenging to complete them. Finding the right trade-off between simple and complex hand gestures is therefore paramount to successfully use hand gestures as a control input device.

Discussion

[1] `http://www.youtube.com/watch?v=tBOfvZboGfk`

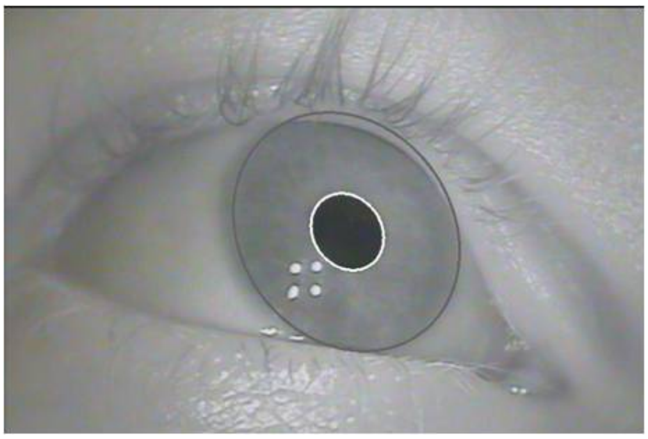

Figure 7. caption...