Palpebrae Superioris: Exploring the Design Space of Eyelid Gestures

Ricardo Jota and Daniel Wigdor
Department of Computer Science, University of Toronto.
Toronto, ON, Canada
{jotacosta, daniel}@dgp.toronto.edu

Figure 1 - Example of asymmetrical eyelid gestures. By closing one eye the user is filtering information a 3D edition application.

ABSTRACT

In this paper, we explore the design space of eyelid gestures. We first present a framework for the design space based on the anatomy of the eye, human perception, and complexity of the eyelid gesture. Based on the framework we propose an algorithm to detect eyelid gestures with commodity cameras, already existing in laptops and mobile devices. We then populate the design space by demonstrating prototypes based on 3 form factors: mobile devices, desktop, and horizontal surfaces. These prototypes demonstrate the breadth of eyelid gestures as an input modality. We follow the scenarios with a discussion of how eyelid gestures can contribute to an interactive environment, and conclude with a discussion on insights, design recommendations, and limitations of the technique.

Keywords: Eyelid gestures.

Index Terms: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

1 INTRODUCTION

The human eye is controlled by 9 muscles. Of these, 8 are dedicated to providing precise control over the orientation of the eyeball, which is often measured as gaze direction. The use of gaze direction as a control signal has been the focus of a great deal of research [28],[32],[33],[34]. It has been used as an implicit signal, acting as an indicator of mood [30], as well as an explicit signal, such as in eye tracking [16].

The 9th muscle of the eye is the *palpebrae superioris* (PS), which is dedicated to the control of the eyelids. The use of the PS in control-signal research has largely been limited to acting as a delimiter of input signals from the other 8 muscles of the eye, such as using winks to "click" a gaze-controlled mouse cursor [2]. While the utility of the PS as an explicit signal is clearly useful in combination with the other muscles of the eye, in gaze tracking, its use as an independent input signal has been largely unexplored.

The control of the eyelids is a rich source of expressiveness of the eyes (See Figure 1). Winks, frowns, squints, eye-widening, and conscious blinks each offer different meaning. When examined in isolation from gaze, these expressions provide a significantly different design space which can be explicitly controlled by the person performing them.

In addition to the wide range of available expressions, the PS gestures offer two attributes that make them attractive as an explicit control signal. First, PS movements are largely independent of other body actions. Second, involuntary input is easily heuristically filtered. Further, as we will explain, PS movements can be easily recognized using commodity cameras (such as mobile device front-facing cameras), and thus are near-term candidates for secondary modalities for other forms of input, such as touch input on a mobile phone.

In this paper, we contribute an initial exploration of eyelid gestures made possible mainly by the *palpebrae superioris*. This exploration includes a review of related work and a thorough definition of the design space (including issues of anatomy, perception, and complexity). We then describe our recognition algorithm, and example uses of PS gestures on three form factors: interaction with mobile devices, controlling desktop applications, and surface computing interaction. We conclude with a discussion of our insights, design recommendations, and limitations of the technique.

2 RELATED WORK

There have been several approaches to use facial characteristics as input methods. In particular the vital line of gaze and face muscles have been applied to human-computer interaction.

2.1 Electrooculography

Electrooculography (EOG) is a technique for measuring the resting potential of the retina. With an intrusive apparatus, a pair of electrodes is placed around the eye and allows the eyes' position to be measured based on the recorded potential. LaCourse utilized EOG to create DECS, a communication tool for persons with disabilities [18]. DECS is based on gaze gestures for input and gaze staring for activation. Kaufman [4] presented an EOG interface that supports both eye movement but also left and right winking and blinking gestures. However, authors indicate that EOG provides a noisy signal and attribute a low eye gesture accuracy rate to body actions such as head movements. The Eye Mouse project [26] proposed cursor control by gaze diversion. If the user wants to move the cursor in a certain direction he is only required to divert his gaze 30º in that direction for half a second. Kwon and Kim pushed the idea further and presents a mouse driven EOG system [20]. Using multiple electrodes, they controls a mouse cursor with gaze and utilizes blink gestures to right-click. They report that users can control several window functions after a training session of a few minutes.

EOG have two main disadvantages: the intrusive apparatus [13], that requires specific hardware; and the baseline drift that obscures eye-movement signal [9].

2.2 Vision-Based Approaches

Gaze-tracking interfaces are generally controlled using an optical sensor focused on one or both eyes. Most modern eye-trackers use contrast to locate the center of the pupil and use infrared and near-infrared light to create a corneal reflection, the vector between these two features can be used to compute gaze intersection [28],[32],[33],[34]. This has mostly been explored by Attentive User Interfaces (AUIs) [6] and has been applied to mouse control [6],[17],[28]. Its use has been demonstrated as a secondary control signal in EyeWindows; proposed as gaze as input method for focus window selection and finds eye tracking to be faster than mouse and hot keys [8]. Vertegaal presents a Fitt's Law evaluation that show that the eye tracking techniques outperformed the mouse from 16% up to 46% faster, however they present a high error rate [29].

Face recognition has been thoroughly researched and is often used to control mouse cursors[24],[35]. Ashdown further explores the concept by combining head tracking with mouse input to facilitate mouse movement over multiple monitors [25] and Kaiser applies the same principle to disambiguate targeting in virtual environments [10]. Finally, Nguyen presents an algorithm for eye gaze tracking that allows free head movements with a single USB camera [5]. Similar to Nguyen, our tracking algorithm builds on top of Saragih's face tracker [15] by adding eyelid state detection.

2.3 Blink gestures

Blink gestures have been proposed with both electrooculography and video based approaches. Kaufman [4] supports detection of the state of the eyelids, and suggests using a sequence of left/right blinks as an input signal. Kwon [20] utilizes conscious blinking (two eyes) as an activation mechanism to accompany eye tracking. The Eye Wink Control Interface (EWCI) relies on eye winks of varying length to generate input for "severely disabled" users [31]. To this we add an additional state for each eye (squinting), as well as the use of eyelid gestures as a secondary control signal for users who give simultaneous input with other devices. Further, EWCI is based on an IR emitter/detector clamped on the earpiece of an eyeglass frame. We demonstrate a less intrusive method of eyelid gesture detection. Recently a comparative study on blink detection and gaze estimation methods has been published [19]. As with previous work in this space, this study focused on the use of blink detection as a secondary signal for eye gaze detection. From the revision of previous work, it is clear that we are now able to easily track eyelid gestures, without resorting to exotic hardware solutions.

It is also clear, however, that these efforts have mostly focus on the detection of winks as a means to either detect the face position, or facilitate gaze tracking. Moreover, although there has been a strong focus on the accessibility community, the HCI community has preferred gaze to eyelid gestures, leaving eyelid gestures underexplored from an HCI application perspective. To the best of the authors' knowledge there is no exploration of the design space provided by eyelid available. Moreover, other than the cursor position and AUI based interfaces, there is little focus on the application of eyelid gestures on non-accessibility scenarios such as mobile devices or surface computing. Thus, a goal of our work has been to define and explore that design space, which we now present.

3 Design Space of Eyelid Gestures

We will divide our exploration of the design space into three areas: eye anatomy, perception, and complexity & sustainability.

3.1 Eye Anatomy

The human eye is a two-piece spherical unit (eyeball) that is controlled by six extra-ocular muscles for eyeball movement: the lateral rectus, the medial rectus, the inferior rectus, the superior rectus, the inferior oblique, and the superior oblique, shown in Figure 2. Each of the rectus muscles rotates the eyeball in one direction; while the oblique muscles stabilize the eye to prevent double vision (diplopia). The eye also includes three intra-ocular muscles: the ciliary regulates the shape of the lens and the circular and radial muscles of the iris contract and dilate the pupil respectively. Throughout this section, we refer to anatomy features explained in The Anatomy of Human body, to which we refer the reader for a thorough review [12].

Humans cannot control these muscles directly, however their actions can be controlled in aggregate for conscious actions [12]. The extra-ocular muscles react to instructions of gaze direction, and the ciliary can be relaxed by the attempt of defocusing a viewed object. Absent of invasive tools, such as electromyography [13], gaze can only represent the aggregate of the extra-ocular muscles' movement. As we have described, gaze has been extensively explored on previous work, but forces the user maintain target on

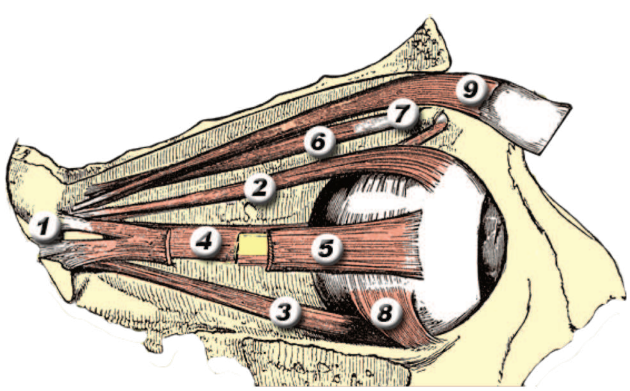

Figure 2 - 2,3,4, and 5: Rectus muscles. 6 and 8: Oblique muscles. 9 - *levator palpebrae superioris*. Other structures: 1 = Annulus of Zinn, 7 = Trochlea [extracted from Gray's Anatomy, 1918.]

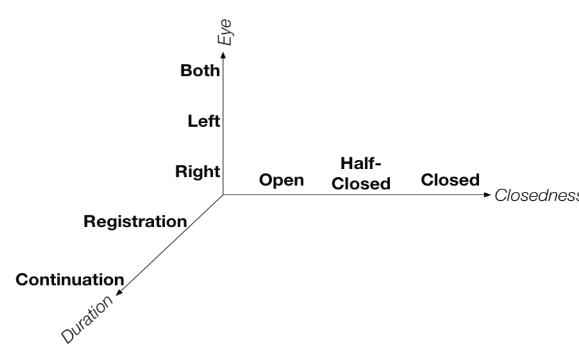

Figure 3 - The anatomical characteristics which form parameters of eyelid gestures.

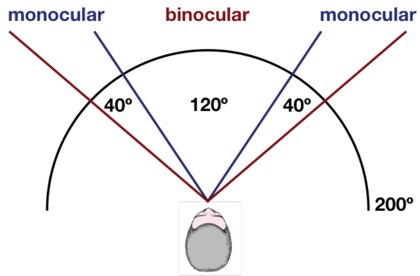

Figure 4 - Human field of view is composed by a binocular region of 120 degrees, flanked by two monocular fields of 40 degrees.

sight. Further, eye saccades, or small and rapid unconscious changes in gaze, make eye tracking a noisy channel for input [21].

Human eyes are equipped with eyelids, which serve the function of maintenance, and protection, of the eye. They are a thin layer of skin that can cover the eye. Eyelids are controlled by the *palpebrae superioris* muscle (Figure 2, structure 9), whose function is to elevate the upper eyelid. The lower eyelid is not controlled by muscles and is static during this process [11].

3.1.1 Temporal Parameters

Humans unconsciously blink their eyelids to help spread tears or remove impurities across the surface of the cornea. Blinks typically last 300-400 milliseconds. Longer lengths are used to safeguard the eyes against threats (such as when falling, sneezing, when an object rapidly approaches the eye), and when sleeping. Blinks can be easily heuristically filtered to build eyelid gestures with few false positives. We avoid unconscious blinks by ignoring eyelid gestures occurring in both eyes simultaneously that are below a threshold of 300-400 milliseconds.

Generally, eyelid gestures are less prone to false positives than are gestures based on modulation of eye gaze, which is known to be noisy due to unconscious movements (such as eye saccades, or responding to movement or other pre-attentive cues), as well as conscious actions (such as glancing, or responding to post-attentive cues, such as the user's name being called). For eyelid gestures, we focus on the conscious actions of the *palpebrae* muscle.

3.1.2 Symmetry and Closedness

A large quantity of the population is able to selectively control each eyelid, which allows humans to consciously close a single eye [27]. We distinguish between **asymmetrical** (either left or right) and **symmetrical** (both eyes) gestures.

Cameras included on handheld devices and laptops do not provide enough resolution to accurately infer continuous eyelid muscle extension at distances comfortable for interaction. While it is theoretically possible to have a continuous classification when exploring eyelid gestures, in order to maintain accurate recognition, we opted to discretely classify gestures as **open**, **closed** and **half-closed**.

The possible parameters of anatomy which can be modulated by the user are therefore the selection of **eye** (both, left, right), the degree to which the eye is closed (**closedness**, which we discretize to open, half-closed, and closed), and the **duration** of the gesture (which we discretize into a short duration used to register an eyelid gesture, and a longer duration, which is used to continue or sustain one). This is summarized in Figure 3.

3.2 Perception

An interesting characteristic of this rich design space is that movement of the PS as an input signal has the effect of changing the user's ability to receive output: when we wink, squint or blink our eyes, the way we perceive visual information is affected. When performing eye gestures, the field of view might be affected – therefore, a consideration in assigning uses to eye gestures is to ensure that the gesture does not limit the ability of the user to perceive the feedback when gesturing. This effect includes a reduction in the field of view, as well as a change in depth perception [14].

Human eyes face forward with field of view of approximately 200 degrees with two eyes. The two eyes overlap for a binocular region of 120 degrees, flanked by two monocular fields of 40 degrees (See Figure 4). This field is reduced with some eyelid actions. Further, the arrangement of the eyes provides humans with stereopsis, primarily obtained by the parallax provided by the eye's different positions. While depth perception can also be assisted by secondary cues, it is nonetheless affected by the loss of binocular view.

We classify our gestures accordingly to their depth perception and field of view. Gestures can cause variable **depth perception** (monocular or binocular), as well as affecting the **field of view** (full, constrained, none).

3.3 Complexity and Sustainability

Previous work have presented classifications for direct input gestures which take into consideration the gesture complexity and how easy it is for a subject to comfortably maintain that gesture [22],[23]. Eyelid gestures suffer from equivalent anatomical limitations. Although the design space we have thus far described includes several seemingly independent dimensions, one also has to take into account the physical limitations of these parameters. For example, field of view is necessarily 'constrained' if a gesture keeps one eye open. One must also consider that, even among those parameters which are independent, users may be imprecise when

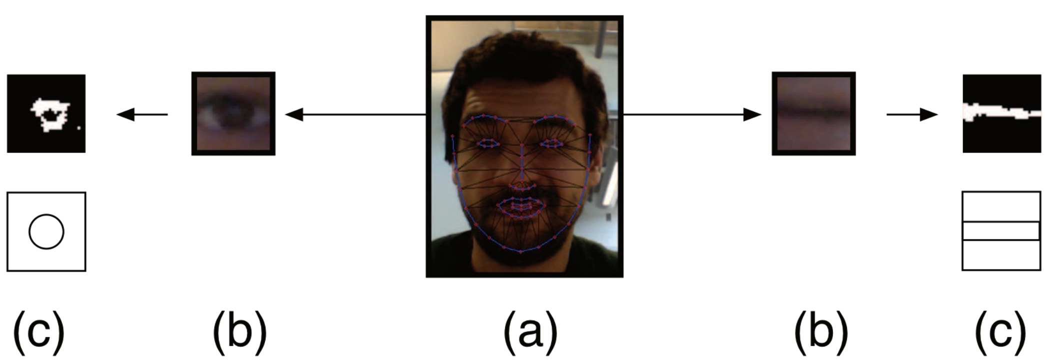

Figure 5 - Detection workflow: (a) face detection, (b) eye region of interest capture, (c) structural analysis and shape description.

controlling eyelid gestures, and therefore not all combinations of these dimensions are physically comfortable. For example, holding one eye in a squinting posture for a sustained period of time places stress on the *palpebrae superiosis*, and can cause it to twitch or cramp – a one-eyed squint is therefore difficult to maintain. Indeed, individual differences between users' abilities to control the PS must be considered. For example, it is know that not all humans can blink both eyes independently. Finally, the frequency of these gestures can create fatigue in the *palpebrae superioris,* not accustomed to high frequency of actions. This may also affect surrounding muscles such as the rectus, impacting field of view, as well as physical comfort of the user.

Therefore, the notions of gesture registration and relaxation, previously discussed in [22], are of particular importance in designing eyelid gestures. Possible eyelid gesture registration postures must take into account human body limits. Further, for those eyelid gestures which are difficult to maintain, allowing for a relaxation of the gesture following registration is of great importance. Therefore, we include gesture activation in our exploratory design of eyelid gestures. We argue that gestures that only require **registration** serve a different purpose and can include higher penalties in both ergonomics and perceptive dimensions that gestures that require **continuation**.

4 Detecting Eyelid Gestures

Having explored the design space of eyelid gestures, we now consider the mechanisms by which they can be detected. Our approach can be divided into 4 parts (see Figure 5). Detecting the face, calculating the eye position, detecting the state of each eyelid, and finally classifying the gesture. The algorithm has a frame rate of around 30hz from a 720p camera and a 1.8Ghz Core i5 processor.

To detect faces we use the algorithm described in [15]. This method outputs a point mesh, out of which each eye position corresponds to a mesh sub-set and can be trivially captured (see Figure 5b). We opted for this algorithm, as opposed to a feature detector (for example, Haar features available in the OpenCV framework), because we found it to be more consistent in terms of the detected region position relative to the eye in the video image, therefore allowing us to assume that the eye pupil was always in the center of the eye's region of interest.

Once the face is detected, we calculate eye pupil position (the center point for each eye's position) and create a rectangular region for each eye for detecting its state. The size of this region is computed according to the size of the detected face to achieve scale-invariance.

For each eye, we then binarize the image of the region of interest. To make our algorithm robust to changes in lightning in particular to shadows due to face movements or gestures, we apply a threshold value based on the actual color range of each region. This operation produces a binary picture with only the 10% darkest pixels (Figure 5c).

Figure 6 - Left: web browser GUI accordingly to eyelid gestures. Right: When a call is received by a user wearing insulating gloves, they cannot give touch input. Squinting at the phone activates accelerometer gestures.

The binary regions are compared against pre-defined shapes using shape descriptions. In particular, we calculate image moments for each region and compare the results against two pre-defined templates: a circle (open eye) and wide rectangle (closed eye).

To further avoid false-positives, the output of the detection algorithm is input to a rolling average that stores the last thirty events. An eye is considered closed if 45% of the events classify the state as closed. An eye is considered open if at least 55% of the events classify the eye open. Finally, based on the state for each eye, the correspondent gesture is issued.

5 DESIGNING FOR EYELID GESTURES

We now demonstrate the power of eyelid gestures by demonstrating their use in 3 device form factors: handheld devices, desktop, and multi-touch surfaces. These scenarios were chosen as to show a spread of applications where eyelid gestures may be of use, further study is required to conclude to what scenarios best fit eyelid gestures.

5.1 Handheld Devices

We demonstrate the use of eyelid gestures to solve known problems with mobile devices. First, due to their physical size, handheld devices have limitations on the amount of information that they can provide onscreen at a time. Second, everyday situations, such as holding other artifacts [3] or wearing insulating gloves [36] can significantly reduce the ability to perform input.

The use of eyelid gestures can help interaction by providing an input channel that does not make use of fingers or facilitates hand motions. We demonstrate this with two examples: web browsing and phone call application.

5.1.1 Mobile Web Browser

When browsing with mobile devices, it is ideal to use as much display space as possible to render the content. On the other hand, actions such as entering a URL, and going back and forward usually take-up valuable display space. Applications solve this by either reserving display space for icons (reducing the amount of information on screen) or supporting a full screen that makes use of gestures or physical buttons to 'pop-up' this functionality.

We propose the use of eyelid gestures to allow the user to maintain an unobstructed view of the content, while also providing immediate access to control surfaces. When both eyes are open, a full-screen view of the webpage is provided. When the user closes their left eye, the URL bar appears. When the user closes their right eye, controls such as "Forward", "Back", and "Stop" are shown. Input may be made to these control surfaces in the usual way.

Because each view allows for the whole screen to be dedicated to the task (either displaying content or making input), the tension in allocating pixels between display and control surfaces is eliminated. Thus, controls may fill the whole display, as in Figure 6. Larger controls reduce input errors by providing large targets for touch input.

5.1.2 Answering the Phone

When the user is wearing insulating gloves, they cannot provide touch input to a capacitive touchscreen. Under such conditions, it is desirable to use other input modalities for simple actions, such as accepting or refusing a call. Current phones often have several sensors (such as accelerometers) that can be used to infer gestures of the device, such as "answer the phone". These, however, still require user activation so that false positives (in this case, 'pocket answering' due to accelerometer variations when walking) can be avoided. Eyelid gestures can be used as a modality for delimiting

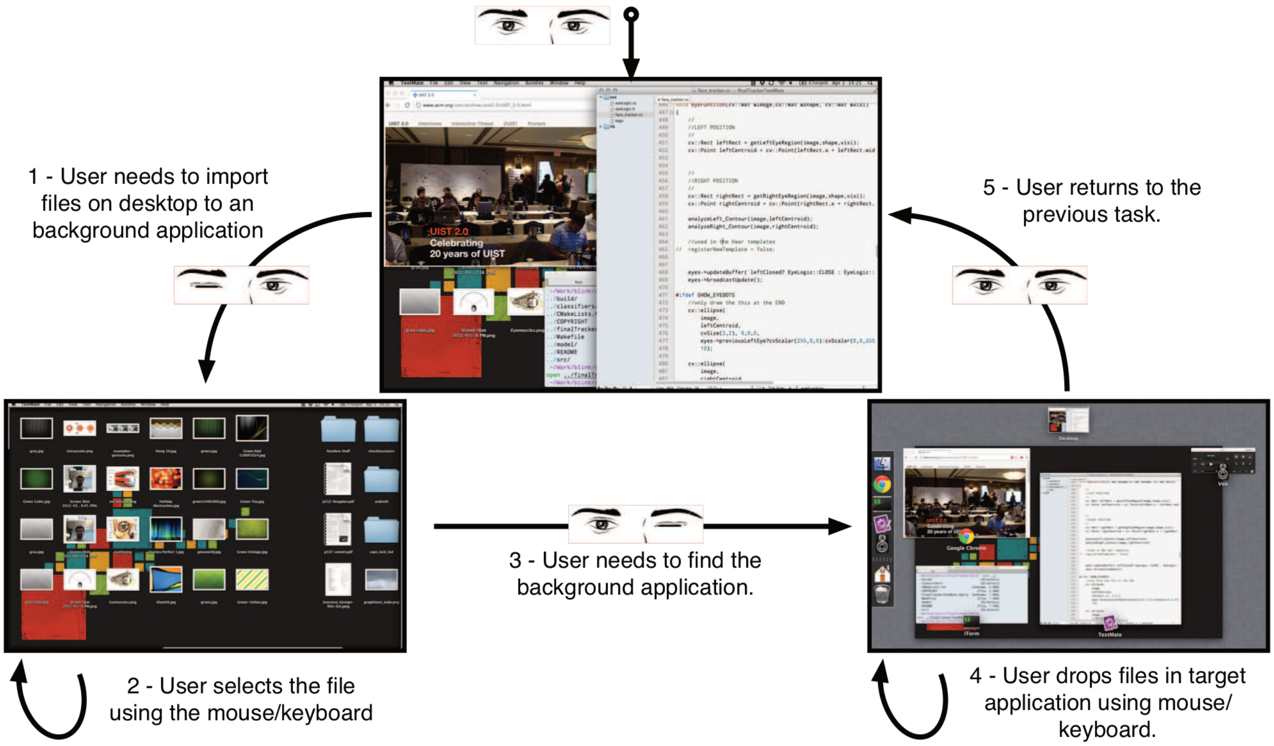

Figure 7 - Desktop system-wide eye gestures example.

input via other sensors. For example, if a call is received while wearing gloves, the user can take the phone out of their pocket to see the identity of the caller. They can then *squint* both eyes to activate the accelerometer gestures (and receive an on-screen indication of the available gestures, as shown in Figure 6). They can then perform the desired accelerometer gesture. The mode terminates upon completion of the gesture.

5.2 Desktop Applications

Eyelid gestures can also be useful in desktop computing, where, to support the complexity required to manage a multi-task system, operating systems often allow users to assign shortcuts to actions, system-wide or application-based. We explore this use of eyelid gestures by presenting both system-wide and application specific gestures. In particular, we examine their use for window management, engaging modes in image editing software, and 3D navigation.

5.2.1 Window Management

To manage overlapping windows and to switch from one window to another, operating systems provide functionality such as tabs or switchers. Some operating systems also provide visual effects to access the desktop or an overview of the active windows, activated using the keyboard or mouse gestures. Eyelid gestures that change the user perception of information are adequate shortcuts for the visual effects. We activate the windows overview mode when users close the left eye, and activate the desktop when users close the right eye. By gesturing in sequence, users can quickly navigate between windows and drag files from their desktop to non-visible windows without the need to resize windows (Figure 7).

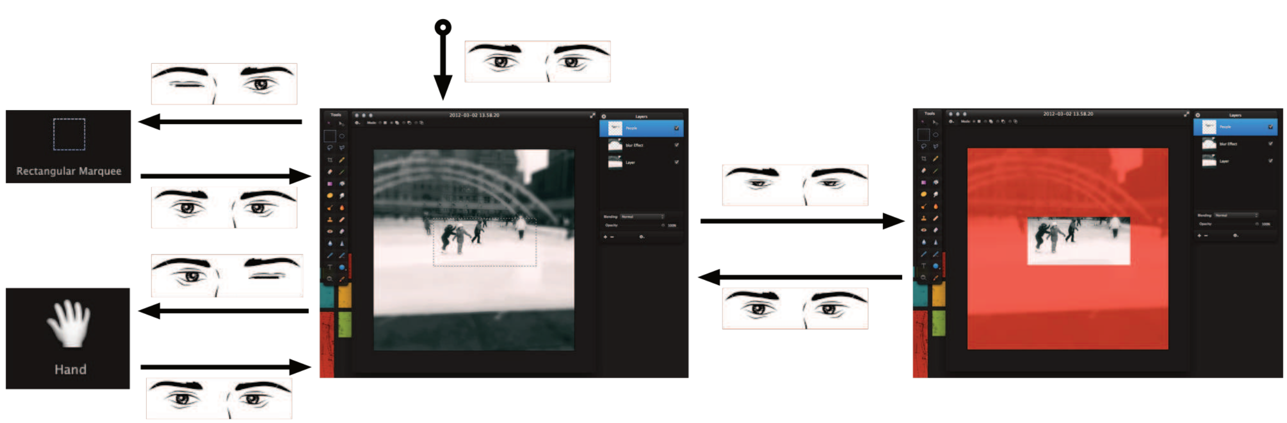

Figure 8 - Application specific gestures: macro system for image editing software

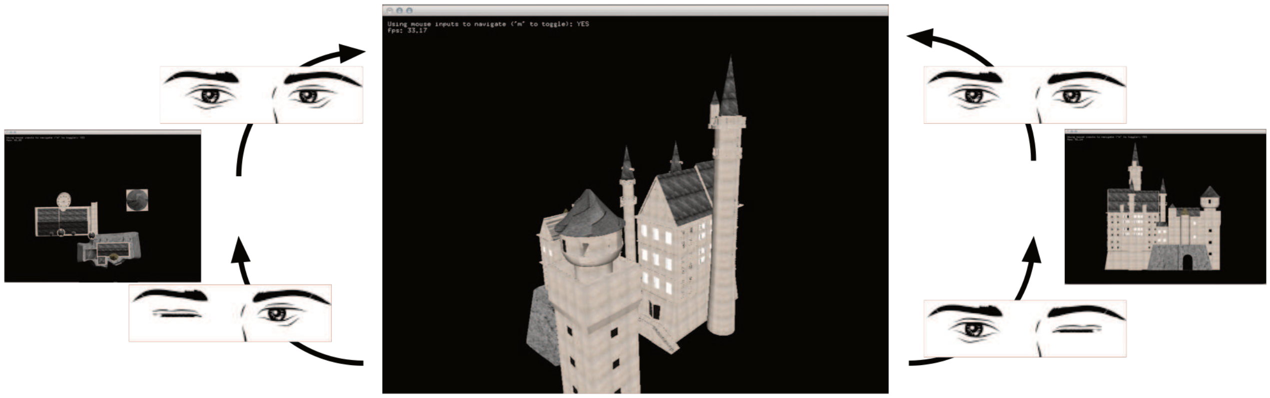

Figure 9 - Eye gestures associated to changes in perception. Whenever a gesture is detected the model switches from a perspective projection to a orthogonal projection.

5.2.2 Quick Mode Changes for Image Editing

Professional image editors provide complex editing tools. The underlying workflow generally consists of navigating to a portion of the image, selecting the area to be edited, and then selecting an action to perform. This action requires repeatedly switching between navigation, region specification and command execution, and can be tedious and repetitive to perform.

We use eye gestures to reduce this tedium. By closing the left eye, the navigation tool is activated (allowing users to drag the document), closing the right eye empties the current selection and enables the selection tool (allows users to select new regions) and squinting both provides a selection mask for easy identification of the current selection region. Figure 8 illustrates the new workflow. While we demonstrate this for image editing, the use of eye gestures for mode changes is generally applicable, and is also a particularly salient kinesthetically-held mode.

5.2.3 Navigating 3D Models

When building 3D models, users often evaluate their edits by going back and forward between views, to get a sense of the overall model, or by hiding specific parts of the model, in order to understand the contribution of the change.

People jokingly close one eye or the other in order to change their perspective of nearby objects. We map this behavior onto gestures supporting the changing of camera viewpoints. The user switches between multiple cameras by closing one eye or the other, or keeping both eyes open (Figure 9). We extend the ocular dissonance to the extreme, by giving users complete control over the cameras assigned to each eye configuration. Users can, for example, edit in perspective projection and assign orthogonal views to the camera associated with one of their eyes (to facilitate planar restricted movement, for example).

We also facilitate 'x-ray vision' through squinting. In perspective view, objects closer to the camera often occupy a significant portion of the viewing angle and often occlude other objects. We provide an eyelid gesture (squint) that provides a visualization mode that displays objects close to the camera as wireframes, thus enabling the user to see objects otherwise occluded. This gesture is time dependent: the longer the user maintains the gesture, the more objects are displayed in wireframe (see Figure 10).

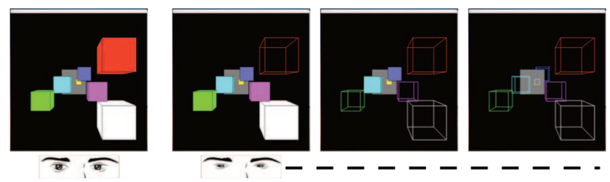

Figure 10 - Squint gesture renders objects close to the camera as wire-frames. If the user sustains the gesture, more objects will be wire-framed.

5.3 Surface Computing

Finally, we consider the use of eyelid gestures in surface computing, where gestures are used as means to interact with available information. The teaching of these gestures has been the focus of previous research [1],[7]. We introduce eyelid gestures as actions to activate learning systems.

5.3.1 Learning Available Gestures

As previous projects note, users need to be taught which gestures are available to be executed on each object. We implemented a system that uses eyelid gestures to activate a learning system which overlays objects with descriptions of the available gestures. Although intrusive, in that the overlay occludes content, it is transient, and activated only upon the user performing the eye gesture (see Figure 11c).

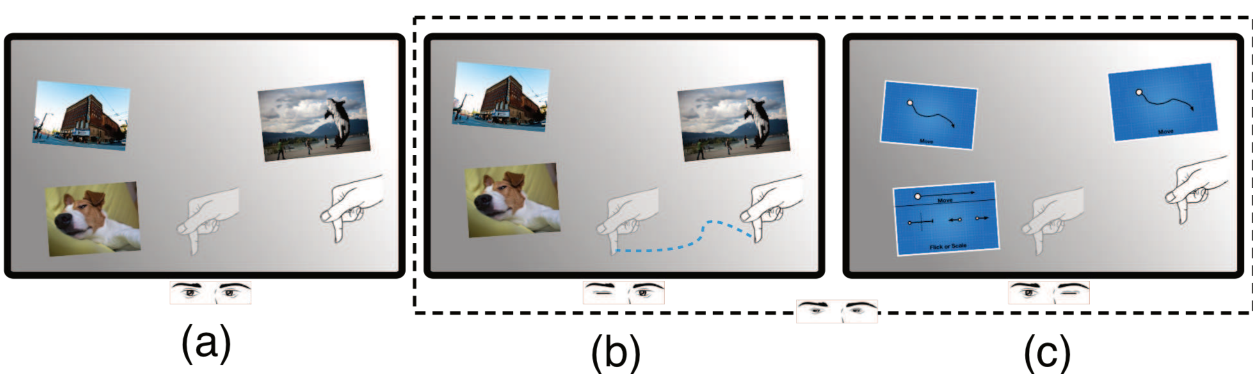

Figure 11 – Gesture learning system. The two sub-systems can be activated separately, by closing the corresponding eye, or together by squinting both eyes.

5.3.2 Visualizing Gestures

As previously described, there is an additional need in surface computing to provide even more visual feedback than in other media [7]. This is particularly true for gestures, where novices may activate actions without realizing what gesture they executed [1]. By closing the left eye, users activate a learning system that outputs any touch or gesture path, so that they can see exactly what of their inputs is actually registered as a gesture to induce a given effect (see Figure 11b).

5.3.3 Combining Two Learning Systems

Eyelid gestures can be composed to enable a combined effect (see Figure 11b and c). In this case, both of the learning systems we have described may be activated. If users squints both eyes, instead of closing just one eye, the system activates both learning systems and shows active touch path and possible gestures for each object. The combination of two learning systems with three gestures demonstrates the possibility of composition and provides a powerful learning system that is configurable for different user profiles. It also provides the user with complete, and easy control over when information is provided, reducing the need to reserve a gesture for bringing up, or manually enable and disable, a help system through other means [1],[7].

6 Discussion & Design Recommendations

Eyelid gestures provide a significant design space, are easily detected, and have multiple uses across platforms and form factors. In this section, we discuss issues affecting the implementation of eyelid gestures. In particular, we focus on three topics demonstrated by the previous examples: detection of eyelid gestures, the application of eyelid gestures as secondary modalities and perception changes imposed by eyelids.

6.1 Issues in Detection

Although the gestures were easily detected using commodity cameras, we are still not able to detect continuous motion of gesture, thus the discretization of the closedness parameter of our design space. In particular, this is a problem when using lower resolution mobile device cameras (less than 720p). This has a strong implication in the application of eyelid gestures because it limits the available design space. On the other hand, humans do not have a fine control over the amount of closedness of the eye; but are able to maintain a position of half-close. Future implementations might explore the appropriate use of a continuous version of the closedness parameter.

In the examples presented, we explored the absence of open eyes (closed eyes). It should be noted, however, that our algorithm also detects the absence of a face. When designing for eyelid gestures, face detection is useful in avoiding false positives. Further, while we did not provide such example, one might also consider the use of two-eyes closed as an eyelid gesture.

6.2 Eyelids as a Secondary Modality

Although the design space for eyelid gestures is large, we have found that they are most useful when used in combination with other input modalities, given the limitations of the *palpebrae superiosis*. We believe that the best uses of eyelid gestures are as follows:

- Substitute the main modality for a short period of time as exemplified in the mobile device accelerometer activation
- Change behavior (mode) of the main modality, as exemplified in image editing
- Change the visible output without changing input mode, as exemplified in the surface scenario
- Provide a quick ‘ready to hand’ switch

Future improvements in detection, and combination with facial expression detectors may help to increase the range of uses for this input modality.

6.3 Changes in perception

Changes in the field of view, associated with the eyelid position, distinguish eyelid gestures from other secondary modalities, such as voice. We argue that such changes should be taken into account in order to successfully apply eyelid gestures. Indeed, in the presented scenarios, the eyelid gestures are applied to actions that:

- Restructure the objects’ position accordingly to the gesture (system-wide desktop example)
- Adapt the UI to take into account changes in perception (mobile device and surface computing)
- Take into account monocular or binocular vision when displaying information. For example, the 3D Editing scenario changes to an orthogonal projection when one eye is open and a perspective projection if two eyes are open.

As we have discussed, the effects of these changes can be minimized through the use of the principle of *gesture relaxation*, so that an eyelid modulation is used to initiate an action or mode, but is not continued in order to sustain it.

6.4 Composition of Gestures

As demonstrated in the surface-computing example, eyelid gestures can be combined to further enhance the design space. In our example, we interpret the squint gesture as a combination of both left and right gestures, but actions can also be associated with transitions between states. For example, a cursor position can be highlighted if a transition between eyes closed and eyes open is detected, or sequential switch between left and right eye interpreted as a single gesture (similar to interpreting sequential touch as one gesture or double click).

6.5 Limitations

Although we present a large number of possible eyelid gestures, not all gestures within our defined design space are feasible to execute or are comfortable to sustain over time. Further studies are required to obtain anatomically-based design guidelines for eyelid gestures.

Our recognition algorithm, though simple, is hardware dependent. Mobile applications might need to account for lower frame rate, lower resolution, and camera vibration due to user movement.

Finally, not all humans are capable of independently blink both eyes. The percentage of the population that can independently blink both eyes is not know, thus any application that leverages eyelid gestures must account for the fact that there are those that cannot use this technique.

7 Conclusions

Overall, we argue that eyelid gestures are useful as an input modality, and are useful beyond the applications in accessibility in which they have been previously described.

In this paper we initiated an exploration of the design space of eyelid gestures and present a framework to classify gestures and their usage. We then present an algorithm capable of detecting gestures within the design space. Finally, we populate the space by describing the use of eyelid gestures on three form factors: mobile devices, desktop applications, and surface computing. We presented several prototypes that demonstrate the breadth of eyelid gestures as an input technique. We follow the scenarios with a discussion on how eyelid gestures can significantly contribute to an interactive environment and conclude with limitations of the approach suggested. The adoption of eyelid gestures require further study, in particular on the effectiveness of eyelid gestures and the effect fatigue has in the adoption of such technique. Future user studies will shed a light on the applicability of eye gestures as a secondary, useful, technique.

References

[1] Andrew Bragdon, Arman Uguray, Daniel Wigdor, Stylianos Anagnostopoulos, Robert Zeleznik, and Rutledge Feman. Gesture play: motivating online gesture learning with fun, positive reinforcement and physical metaphors. In ACM International Conference on Interactive Tabletops and Surfaces (ITS '10). ACM, New York, NY, USA, 39-48.

[2] Anthony J. Hornof and Anna Cavender. EyeDraw: enabling children with severe motor impairments to draw with their eyes. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '05). ACM, NY, USA, 161-170.

[3] Antti Oulasvirta and Joanna Bergstrom-Lehtovirta. Ease of juggling: studying the effects of manual multitasking. In Proceedings of the 2011 annual conference on Human factors in computing systems (CHI '11). ACM, New York, NY, USA, 3103-3112.

[4] Arie E. Kaufman, Amit Bandopadhay, Bernard D. Shaviv. An eye tracking computer user interface. In Virtual Reality, 1993. Proceedings., IEEE 1993 Symposium on Research Frontiers in , vol., no., pp.120-121, 25-26 Oct. 1993

[5] Ba Linh Nguyen, Youssef Chahir, Michèle Molina, Charles Tijus, and François Jouen. Eye gaze tracking with free head movements using a single camera. In Proc of the 2010 Symposium on ICT. ACM, NY, USA, 108-113.

[6] Colin Ware and Harutune H. Mikaelian. An evaluation of an eye tracker as a device for computer input2. In Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface (CHI '87) ACM, Vol. 46, 2003 ,

[7] Daniel Wigdor, Sarah Williams, Michael Cronin, Robert Levy, Katie White, Maxim Mazeev, and Hrvoje Benko. Ripples: utilizing per-contact visualizations to improve user interaction with touch displays. In Proceedings of the 22nd annual ACM symposium on User interface software and technology '09. ACM, NY, USA, 3-12.

[8] David Fono and Roel Vertegaal. EyeWindows: evaluation of eye-controlled zooming windows for focus selection. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '05). ACM, New York, NY, USA, 151-160.

[9] David W. Patmore and R. Benjamin Knapp. Towards an EOG-based eye tracker for computer control. In Proceedings of the third international ACM conference on Assistive technologies (Assets '98). ACM, New York, NY, USA, 197-203.

[10] Ed Kaiser, Alex Olwal, David McGee, Hrvoje Benko, Andrea Corradini, Xiaoguang Li, Phil Cohen, and Steven Feiner. Mutual disambiguation of 3D multimodal interaction in augmented and virtual reality. In Proceedings of the 5th international conference on Multimodal interfaces (ICMI '03). ACM, NY, USA, 12-19.

[11] Eli L. Chang and Peter A. D. Rubin. Upper and lower eyelid retraction. International Ophthalmology Clinics, 42(2), 2002.

[12] Henry Gray. Anatomy of the Human Body. Philadelphia: Lea & Febiger, 20th edition, 1918.

[13] Hiroyuki Manabe and Masaaki Fukumoto. 2006. Full-time wearable headphone-type gaze detector. In CHI '06 extended abstracts on Human factors in computing systems (CHI EA '06).

[14] J. J. Marotta, T. S. Perrot, D. Nicolle, P. Servos, and M. A. Goodale. Adapting to monocular vision: grasping with one eye. Experimental Brain Research, 104:107–114, 1995.

[15] Jason M. Saragih, Simon Lucey, and Jeffrey F. Cohn. Deformable Model Fitting by Regularized Landmark Mean-Shift. Int. J. Comput. Vision 91, 2 (January 2011), 200-215.

[16] Jayson Turner, Andreas Bulling, and Hans Gellersen. Combining gaze with manual interaction to extend physical reach. In Proceedings of the 1st international workshop on pervasive eye tracking & mobile eye-based interaction (PETMEI '11). ACM, New York, NY, USA, 33-36.

[17] Jingtao Wang, Shumin Zhai, and Hui Su. Chinese input with keyboard and eye-tracking: an anatomical study. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '01). ACM, New York, NY, USA, 349-356.

[18] John R. LaCourse, Francis C. Hludik. An eye movement communication-control system for the disabled. In Biomedical Engineering, IEEE Transactions on , v.37, no.12, p1215-1220, 1990

[19] Kohei Aai and Ronny Mardiyanto. Comparative Study on Blink Detection and Gaze Estimation Methods for HCI, in Particular, Gabor Filter Utilized Blink Detection Method. In Information Technology: New Generations (ITNG), 2011 Eighth International Conference on , vol., no., pp.441-446, 11-13 April 2011

[20] Kwon, S.H.; Kim, H.C.. EOG-based glasses-type wireless mouse for the disabled. In Engineering in Medicine and Biology, 1999. 21st Annual Conf. and the 1999 Annual Fall Meeting of the Biomedical Engineering Soc. BMES/EMBS Conference, 1999. Proceedings of the First Joint , vol.1, no., pp.592 vol.1, 1999

[21] M. G. Saslow. Effects of Components of Displacement-Step Stimuli Upon Latency for Saccadic Eye Movement. In JOSA, Vol. 57, Issue 8, pp. 1024-1029 (1967)

[22] M. Wu and R. Balakrishnan. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In Proceedings of the 16th annual ACM symposium on User interface software and technology, UIST '03, pages 193–202. ACM, 2003.

[23] M. Wu, C. Shen, K. Ryall, C. Forlines, and R. Balakrishnan. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In Horizontal Interactive Human-Computer Systems, TABLETOP'06, pages 185–192, 2006.

[24] Margrit Betke, James Gips, and Peter Fleming. The Camera Mouse: visual tracking of body features to provide computer access for people with severe disabilities. Neural Systems and Rehabilitation Engineering, IEEE Transactions on [see also IEEE Trans. on Rehabilitation Engineering], 10(1):1–10.

[25] Mark Ashdown, Kenji Oka, and Yoichi Sato. Combining head tracking and mouse input for a GUI on multiple monitors. In CHI '05 extended abstracts on Human factors in computing systems(CHI EA '05). ACM, New York, NY, USA, 1188-1191.

[26] Norris, G.; Wilson, E. The Eye Mouse, an eye communication device. In Bioengineering Conference, 1997., Proceedings of the IEEE 1997 23rd Northeast , vol., no., pp.66-67, 21-22 May 1997

[27] P. R. N. Sutton. The development of independent muscular control of separate eyelids in two racial groups, european and Polynesian. In British Journal of Psychology. Volume 49, Issue 1, 65–69, 1958.

[28] Robert J. K. Jacob. What you look at is what you get: eye movement-based interaction techniques. In Proceedings of the SIGCHI conference on Human factors in computing systems '90, Jane Carrasco Chew and John Whiteside ACM , NY, USA, 11-18.

[29] Roel Vertegaal. A Fitts Law comparison of eye tracking and manual input in the selection of visual targets. In Proceedings of the 10th international conference on Multimodal interfaces (ICMI '08). ACM, New York, NY, USA, 241-248.

[30] S. J. Bayless, M. Glover, M. J. Taylor, and R. J. Itier. Is it in the eyes? dissociating the role of emotion and perceptual features of emotionally expressive faces in modulating orienting to eye gaze. Visual Cognition, 19(4):483–510, 2011.

[31] Shaw, R.; Crisman, E.; Loomis, A.; Laszewski, Z.The eye wink control interface: using the computer to provide the severely disabled with increased flexibility and comfort. In Computer-Based Medical Systems, 1990., Proceedings of Third Annual IEEE Symposium on , vol., no., pp.105-111, 3-6 Jun 1990

[32] Shinjiro Kawato, Nobuji Tetsutani, Detection and tracking of eyes for gaze-camera control, Image and Vision Computing, Volume 22, Issue 12, 1 October 2004, Pages 1031-1038

[33] Takehiko Ohno and Naoki Mukawa. A free-head, simple calibration, gaze tracking system that enables gaze-based interaction. In Proceedings of the 2004 symposium on Eye tracking research & applications (ETRA '04). ACM, New York, NY, USA, 115-122.

[34] Thomas E. Hutchinson, K. Preston White, Worthy N. Martin, Kelly C. Reichert, Lisa A. Frey. Human-computer interaction using eye-gaze input. Systems, Man and Cybernetics, IEEE Transactions on , vol.19, no.6, pp.1527-1534, Nov/Dec 1989

[35] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. ACM Comput. Surv. 35, 4 (2003).

[36] Wilcox, Dominic. Finger-NoseTM Stylus for Touch Screens, Apr 12, 2012: http://tinyurl.com/nosepoint