A Self-initializing Eyebrow Tracker for Binary Switch Emulation

Jonathan Lombardi and Margrit Betke*
Computer Science Department
Boston University

Abstract

We designed the Eyebrow-Clicker, a camera-based human computer interface system that implements a new form of binary switch. When the user raises his or her eyebrows, the binary switch is activated and a selection command is issued. The Eyebrow-Clicker thus replaces the "click" functionality of a mouse. The system initializes itself by detecting the user's eyes and eyebrows, tracks these features at frame rate, and recovers in the event of errors. The initialization uses the natural blinking of the human eye to select suitable templates for tracking. Once execution has begun, a user therefore never has to restart the program or even touch the computer. In our experiments with human-computer interaction software, the system successfully determined 93% of the time when a user raised his eyebrows.

1 Introduction

A more natural interface for computer control to supplement the traditional input sources would make human-computer interaction easier and more pleasurable. While the mouse and keyboard allow users to explicitly enter data and interact with the computer under strict rules, the computer cannot respond to the more natural gestures and motions of the user. If this artificial barrier between user and computer could be minimized, then the amount of learning and conscious effort needed to control the machine might be decreased as well. This, in turn, could lead to a more efficient and enjoyable interaction with the computer.

We have developed a method for interpreting eyebrow motion as mouse clicks. Our system tracks the user's eyes and eyebrows and measures the distance between these features. If the distance increases beyond a certain value, the system generates a mouse click. The system, named the *Eyebrow-Clicker*, is fault tolerant and able to re-initialize itself if the tracker becomes lost. Additionally, it does not require the user to wear any devices or obtain equipment more specialized than a video camera.

The *Eyebrow-Clicker* is user independent, i.e., it provides a general scheme for tracking any user's features. The *Eyebrow-Clicker* initializes itself using the naturally recurring blinks of a user to determine the locations of the eyes and eyebrows. Each of these features is tracked individually, and if the user raises his or her eyebrows, the program notes that the distance between the eyes and eyebrows increases and sends the selection command (a mouse click) to the computer (see Fig. 1).

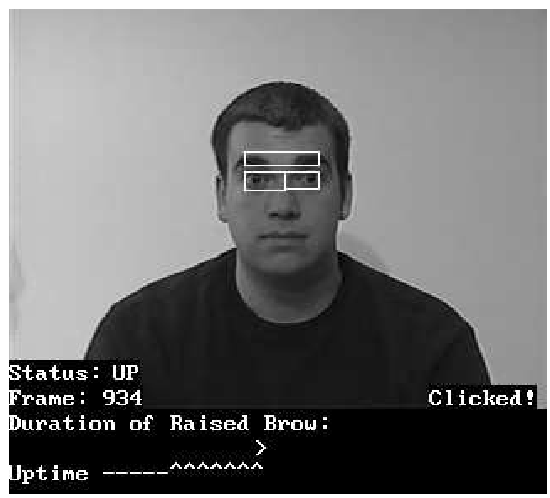

Figure 1: A user of the *Eyebrow-Clicker* immediately after of a issuing a "click" as a selection command. The bar on the bottom displays the duration of the current eyebrow-raising action.

Many other camera-based techniques have been created for human-computer interfaces. References [1, 5, 8, 11, 10, 12, 19, 20, 22] explain various face and head tracking techniques previously employed. There are many projects about tracking eye gaze for human-computer interaction, e.g., [16, 23, 24]. Eyebrow tracking is often used for determining the facial expression of the user [6, 13, 14]. Ref. [15] describes a system for

*The support of the Office of Naval Research and the National Science Foundation (IIS-0093367) is acknowledged. Email: betke@cs.bu.edu, http://www.cs.bu.edu/faculty/betke.

detecting and tracking the location of the eyes to create a communication system for computer users with disabilities. Ref. [2] uses facial tracking to emulate a mouse pointer. Refs. [9, 18] describe a binary system based on nods.

A primary consideration in the development of the *Eyebrow-Clicker* was the minimization of false positives. Under no circumstances should the system falsely deliver a mouse-click to the operating system when the user did not raise his eyebrows. With this goal in place, the current system, as tested on users has generated only one false positive, and only a marginal number of false negatives.

2 Method

The *Eyebrow-Clicker* has two phases: the initialization/recovery phase and the tracking phase. A flowchart of the system is given in Fig. 2. During startup, the system begins in the initialization phase. The user remains still and blinks. Difference images of the current and previous frames are created, $D(x,y,t) = I(x,y,t) - I(x,y,t-1)$, and the Motion Energy Image (MEI) is computed [7]. The MEI $E_\tau(x,y,t)$ is defined to be the union of a sequence of τ binary difference images:

$$E_\tau(x,y,t) = \bigcup_{i=0}^{\tau-1} D(x,y,t-i). \tag{1}$$

A sequence length of $\tau = 60$, which corresponds to 2 seconds of video, is used. The *Eyebrow-Clicker* extracts eye and eyebrow templates from the MEI. It tracks the locations of each of the templates and uses the relative difference to determine the state of the eyebrows. In the event that the features are lost, the system restarts the initialization phase to recover from the error.

2.1 Initialization and Recovery

The initialization phase requires that the user remains still, except for blinking. During this process, thresholded image differencing occurs, generating a MEI. In the MEI, there should be two large regions of motion energy where the blinking took place, and scattered noise caused by slight motion of edges in the scene. Subsequently, the image is "opened" [17] to remove the noise. After removal of the noise, the two remaining regions yield the locations of the user's eyes in the scene. This method is discussed in greater detail in [15].

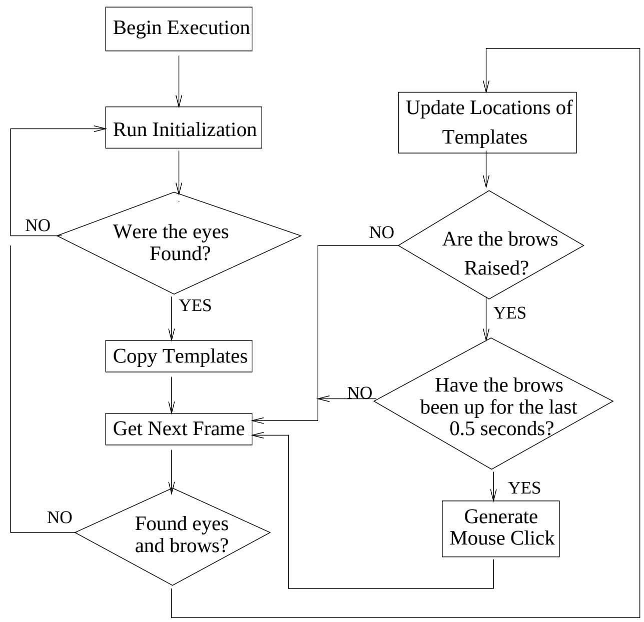

Figure 2: A flowchart of the algorithm.

2.2 Tracking

After the templates for the eyebrow pair and each of the eyes are obtained, the *Eyebrow-Clicker* enters its tracking phase. We use the last known location of each of the templates to begin the search in the current frame. The system searches the local area using the normalized correlation coefficient [3] to determine the best match (see Fig. 3). If the best match is below the threshold 0.5, we assume that the tracker is lost and the system re-initializes to obtain better templates.

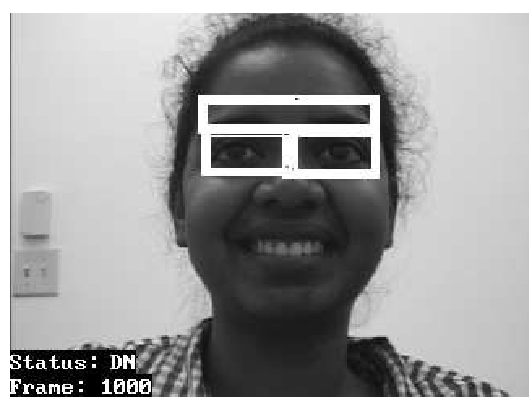

Figure 3: The white boxes denote the localized areas where the search proceeds for each of the templates.

If all three templates are successfully captured, the program uses anthropomorphic information about the face to further protect against any incorrect tracking.

If the relative locations of the templates are anthropomorphically infeasible, e.g. one eye template is above the eyebrow template and the other below, the tracking stage ends and re-initialization occurs. Significantly overlapped or spaced templates will also force the program to recover itself.

When the tracker determines that the eyes and eyebrows are in a state consistent with human facial structure, it computes the ratio

$$T_{\text{current}} = \frac{(y_{\text{left eye}} + y_{\text{right eye}})/2 - y_{\text{eyebrows}})}{(h_{\text{left eye}} + h_{\text{right eye}})/2}, \quad (2)$$

where $y_{\text{eyebrows}}, y_{\text{left eye}}$, and $y_{\text{right eye}}$ represent the y-coordinates of the respective templates, $h_{\text{left eye}}$ and $h_{\text{right eye}}$ the height of the respective templates. If T_{current} is larger than some threshold T, the system determines that the eyebrows are raised. A threshold of $T = 1.25$ is sufficient to remove most jitter caused by eye movement, yet still small enough so that all users are able to surpass this bound easily.

Once the threshold T has been crossed, a timer is set, and if the user's eyebrows remain up for a constant number of milliseconds (we used 500 ms), the selection command is delivered to the system (see Fig. 4). This additional timing threshold prevents occasional jitter from issuing a false click.

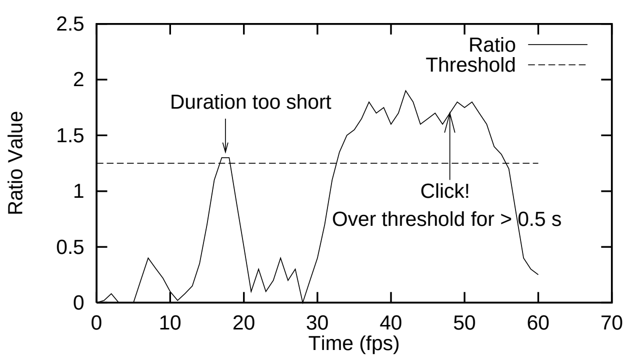

Figure 4: Values of Equation 2 are plotted while a user raises and lowers her eyebrows. The region where the graph crosses the threshold indicates jitter caused by a user blink.

After a click has been issued for the current eyebrow motion, the system prevents any additional clicking actions until the user's eyebrows return to a relaxed state. This avoids a situation whereby one eyebrow movement can result in a potentially infinite number of clicks in rapid succession (see Fig. 4).

3 Experiments

A Windows NT Workstation with an Intel Pentium III-866 MHz processor, 1 GB RAM, and a Matrox II video capture board was used for system development and testing. Video was captured with a Sony EVI-D30 CCD camera. Only grayscale images are used by the *Eyebrow-Clicker*. The Intel OpenCV library was used to interface the Matrox frame grabber with Intel's Image Processing Library [21].

During the experiments, the camera was placed on the table directly in front of the monitor. Each of the users were seated approximately 2–3 feet from the front of the monitor (see Fig. 5). Regular overhead fluorescent lighting was used.

Figure 5: A user in front of the system. The camera is placed in front of the monitor, but below the screen.

Five subjects tested the system. The users practiced with the system before testing. Each user began by watching himself raise and lower his eyebrows and determining how to use the *Eyebrow-Clicker*. During this training, the users simply executed mouse clicks whenever they desired, thereby adjusting to the system.

We also asked the users to experiment with the amount of yaw, pitch and roll rotations they could perform with their head and still have the *Eyebrow-Clicker* function as expected.

After the user felt comfortable using the *Eyebrow-Clicker*, usually within 2–3 minutes or less of training, we started the actual test. For the test, we used the "Frog 'N Fly Demo" by Simtech Publications (see Fig. 6). The user of the game pretends to be a frog, and the goal is to "catch a fly" that appears on the screen. The game is timing based – the user is supposed to activate a mouse click as soon as a circle appears around the fly. With the *Eyebrow-Clicker* as the HCI interface, the user must raise his eyebrows to cause the frog to catch the fly. Each user was asked to catch the flies that appeared during 3 minutes of the game.

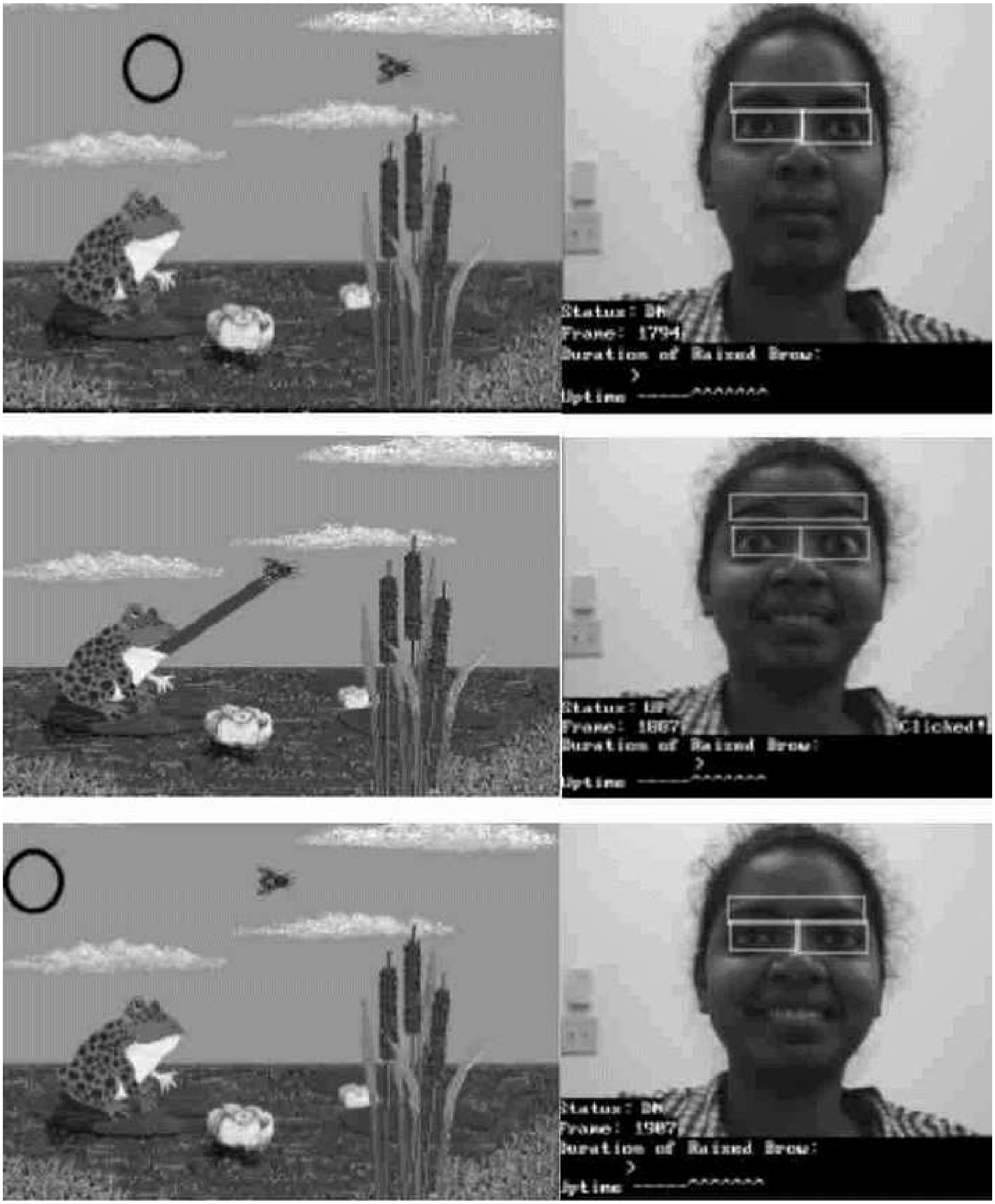

Figure 6: Screenshots of the Frog 'N Fly Demo and a user. Courtesy of Simtech Publications.

A video demonstrating the *Eyebrow-Clicker* during a "Frog 'N Fly Demo" session is available on our web site [4].

4 Results and Discussion

The five subjects caught 153 out of 170 flies, which represents a 90.0% success rate. Two re-initializations occurred during the middle of the game. The re-initialization caused the number of misses to be inflated, because the "Frog 'N Fly Demo" could not be paused during re-initialization. About 50% of the misses were due to the system re-initializing itself. Discounting these additional misses, the success rate becomes 158/170=92.7%. When the tracker became lost, the user missed two or even three flies during the time the tracker needed to recover.

Only one click was falsely generated throughout all trials. We therefore achieved one of our primary goals in creating the system – an extremely low false-positive rate.

The *Eyebrow-Clicker* functions at 29.5 frames per second during all segments of execution on our test machine. It never used more than 21% of the CPU resources, demonstrating the potential for concurrent use with other programs without tremendous burden on the machine.

The *Eyebrow-Clicker* places significant restrictions on the freedom of movement of the user. Our results indicate that the algorithm does not tolerate shaking and turning of the user's head, i.e., rotations measured by the yaw and pitch angles. However, bending the head forward or raising it, measured by the roll angle, is tolerated up to approximately 20 degrees in most users, as long as the eyebrows do not become occluded due to head wear or hair.

Head translation parallel to the image plane is allowed with complete freedom. In addition, the system will detect when the user moves towards the camera or away from it. The system then reinitialize itself to obtain new templates. This is desirable, since the size of the face changes in the image and the templates must be updated appropriately.

If the user is in the middle of a blink when the system obtains copies of the templates, the *Eyebrow-Clicker* experiences quite a lot of jitter. This does not generate false clicks, except in the unlikely case that a user closes his eyes and keeps them shut.

5 Conclusions

The *Eyebrow-Clicker*, in its current form, provides a valuable interface to replace the mouse in HCI applications that solely require selection commands. The high percentage of correctly issued commands in the experiments demonstrates the practicality of our system.

The *Eyebrow-Clicker* used only a small percentage of the available computational resources, rendering it viable for use on most modern computers.

A system with a very low false positive rate, as our system, has great potential to become part of a more general interface system that incorporates a full mouse-replacement strategy.

Our system also has potential as an interface for people with disabilities. It can function simply as a binary switch, or, in a more complex interface, provide a clicking module, for example for the Camera Mouse [2]. The self-correcting portion of the *Eyebrow-Clicker* would be extremely useful because of the inability of a potential user to re-initialize the system manually. However, in order for the system to be usable in such an environment, the system will have to tolerate severe rotations and translations in all directions.

References

[1] L.-P. Bala, K. Talmi, and J. Liu. Automatic detection and tracking of faces and facial features in video sequences. In *Picture Coding Symposium*, Berlin, Germany, 1997.

[2] M. Betke, J. Gips, and P. Fleming. The Camera Mouse: Visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 10(1):1–10, March 2002.

[3] M. Betke and N. C. Makris. Recognition, resolution and complexity of objects subject to affine transformation. *Int J Comput Vis*, 44(1):5–40, August 2001.

[4] Human-Computer Interfaces web page at Boston University. http://www.cs.bu.edu/faculty/betke/research/hci.html.

[5] S. Birchfield. Elliptical head tracking using intensity gradients. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pages 232–237, Hilton Head Island, SC, 1998. IEEE Computer Society.

[6] M. J. Black and Y. Yacoob. Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motions. In *Proceedings of the Fifth International Conference on Computer Vision*, pages 374–381, Cambridge, Massachusetts, June 1995.

[7] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[8] D. Comaniciu and V. Ramesh. Robust detection and tracking of human faces with an active camera. In *IEEE Intern. Worksh. on Visual Surveillance*, pages 11–18, 2000.

[9] J. W. Davis and S. Vaks. A perceptual user interface for recognizing head gesture acknowledgements. In *IEEE PUI*, Orlando, FL, 2001.

[10] F. De La Torre, Y. Yacoob, and L. Davis. A probabilistic framework for rigid and non-rigid appearance based tracking and recognition. In *IEEE FG*, pages 491–498, 2001.

[11] D. DeCarlo and D. Metaxas. The integration of optical flow and deformable models with applications to human face shape and motion estimation. In *Proceedings of the 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 231–238, San Francisco, CA, June 1996. IEEE Computer Society.

[12] G. J. Edwards, C. J. Taylor, and T. F. Cootes. Learning to identify and track faces in image sequences. In *Proceedings of the International Conference on Face and Gesture Recognition*, pages 260–265, 1998. Best paper award.

[13] I. A. Essa and A. Pentland. Facial expression recognition using a dynamic model and motion energy. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pages 360–367, 1995.

[14] S. B. Gokturk, J.-Y. Bouguet, C. Tomasi, and B. Girod. Model-based face tracking for view-independent facial expression recognition. In *Proc Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 272–278, May 2002.

[15] K. Grauman, M. Betke, J. Gips, and G. R. Bradski. Communication via eye blinks - detection and duration analysis in real time. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, Kauai, Hawaii, December 2001. IEEE Computer Society.

[16] T. Hutchinson, K. P. White JR., W. N. Martin, K. C. Reichert, and L. A. Frey. Human-computer interaction using eye-gaze input. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1527–1533, 1989.

[17] R. Jain, R. Kasturi, and B. Schunk. *Machine Vision*. McGraw Hill, 1995.

[18] A. Kapoor and R. Picard. A real-time head and nod shake detector. In *IEEE PUI*, Orlando, FL, 2001.

[19] M. LaCascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on robust registration of texture-mapped 3D models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4):322–336, April 2000.

[20] C. H. Morimoto and M. Flickner. Real-time multiple face detection using active illumination. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 8–13, Grenoble, France, March 2000.

[21] Open Source Computer Vision Library, Intel Corporation (July 2002). http://www.intel.com /research/mrl/research/opencv.

[22] R. Stiefelhagen, J. Yang, and A. Waibel. Estimating focus of attension based on gaze and sound. In *IEEE PUI*, Orlando, FL, 2001.

[23] Y. Tian, T. Kanade, and J. Cohn. Dual-state parametric eye tracking. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pages 110–115, 2000.

[24] S. Zhai, C. Morimoto, and S. Ihde. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of CHI'99: ACM Conference on Human Factors in Computing Systems*, pages 246–253, Pittsburgh, PA, May 1999.