EyeDraw: A System for Drawing Pictures with Eye Movements

Anthony Hornof, Anna Cavender, and Rob Hoselton
Department of Computer and Information Science
University of Oregon
Eugene, OR 97403 USA
+1 541 346 1372
{hornof, acavende, robhoz}@cs.uoregon.edu

ABSTRACT
This paper describes the design and development of EyeDraw, a software program that will enable children with severe mobility impairments to use an eye tracker to draw pictures with their eyes so that they can have the same creative developmental experiences as nondisabled children. EyeDraw incorporates computer-control and software application advances that address the special needs of people with motor impairments, with emphasis on the needs of children. The contributions of the project include (a) a new technique for using the eyes to control the computer when accomplishing a spatial task, (b) the crafting of task-relevant functionality to support this new technique in its application to drawing pictures, and (c) a user-tested implementation of the idea within a working computer program. User testing with nondisabled users suggests that we have designed and built an eye-cursor and eye-drawing control system that can be used by almost anyone with normal control of their eyes. The core technique will be generally useful for a range of computer control tasks such as selecting a group of icons on the desktop by drawing a box around them.

Categories & Subject Descriptors
H.5.2 [Information Interfaces and Presentation]: User Interfaces - input devices and strategies, interaction styles

General Terms
Design, Human Factors

Keywords
Art, children, drawing, eye tracking, input devices, interaction techniques, universal access

1. INTRODUCTION
New software is needed to enable people to control their computers with eye movements. This need is especially acute for people with severely impaired motor abilities, who cannot move their limbs or speak, such as people with partial paralysis resulting from Amyotrophic Lateral Sclerosis (ALS, or "Lou Gehrig's disease"), brain injury, or cerebral palsy. These people are severely limited in their ability to interact and communicate with the rest of the world. Despite these severe disabilities, many of these users retain normal control of their eyes, which opens the door to perhaps the best and most noninvasive means for these people to interact and communicate with the world—with eye movements.

A number of eye tracking systems have been specifically designed to assist people with severe motor impairments. Systems include Quick Glance (eyetechds.com), VisionKey (eyecan.ca), and the LC Technologies Eyegaze Communication System (eyegaze.com). These systems offer computer control via eye-typing. Hundreds of people use these systems to communicate and function in life. The Eyegaze Communication System offers perhaps the most functionality, with software for uttering phrases via a speech synthesizer, making telephone calls, controlling lights and appliances, and turning pages in electronic books.

Overall, few software applications have been specifically designed to be controlled with eye movements. Exceptions include software for typing with the eyes by moving the gaze across a keyboard displayed on the computer screen [10]. However, eye-controlled software is not available for the vast majority of the activities that nondisabled people accomplish on their computers or with pencil and paper.

There is a particular need for new eye tracking software applications to be developed for children. Children have special interaction and communication needs that, if not met, will impede their social, emotional, educational, and creative development, and further reduce the ability of children with complex physical disabilities to function as an individual in society. There is a special need for eye-controlled software to be developed to accommodate the special needs of children.

This paper discusses the design and implementation of EyeDraw, a software program that will enable children with severe mobility impairments—children who can only move their eyes—to draw pictures with their eyes, and thus benefit from the same creative and social activities as nondisabled children. The system is already demonstrated to be useful and usable by nondisabled children with no prior experience using an eye tracker. User observation studies with disabled children are currently in progress.

ASSETS'04, October 18–20, 2004, Atlanta, Georgia, USA.

2. RELATED WORK

For over twenty years, researchers have been building systems that use the eyes as a direct input to the computer [2, 7, 13]. There has been more recent interest in finding ways to use eye position in some secondary, useful manner, such as monitoring a user's attention to find opportune times for interruptions [12], or to jump the mouse cursor to the gaze region when making manual mouse movements [15]. Overall, success has been limited in part because eye tracking is technically challenging and labor-intensive, and because eye movement data are noisy and difficult to interpret [8]. However, there have also been successes [8], and improvements in the accuracy and ease-of-use of eye trackers make it feasible to build software applications tailored for eye control. One such interface zooms in when selecting an item, to compensate for the noise in eye tracking that makes it very difficult trying to select small targets with the eyes [1].

Previous research on how and why children draw pictures with crayons, or with mouse-controlled computer drawing programs, is useful in the design and development of EyeDraw. The research provides (a) evidence that a computer-based drawing program can provide important developmental experiences, (b) guidance for designing the most beneficial eye-drawing experience, and (c) a framework for evaluating the progression of drawings that children will make with their software. Children have been observed progressing through a series of five qualitative stages when beginning to draw with paper and pencil: random scribble, controlled scribble, basic forms, early pictorial, and later pictorial. Children follow the same stages of development when learning to draw on computers [5]. Besides suggesting that important developmental processes can be achieved through drawing with the eyes, this taxonomy provides a framework for categorizing the drawings of children using EyeDraw. EyeDraw emphasizes freehand, open-ended, spontaneous drawing because this has been shown to be more successful at inspiring creativity and self-expression than recipe art lessons and coloring-in drawings [4].

Previous interaction techniques for drawing with the eyes use *free-eye drawing*. In free-eye drawing, pixels are colored-in wherever the eye tracker records the gaze on the computer screen. Figure 1 shows free-eye drawing from Tchalenko [11] and from EaglePaint [6], another system for free-eye drawing with the eyes. EaglePaint puts randomly-colored digital ink wherever the user looks on the computer screen. Both systems have produced drawings that would be categorized in the "scribble" stages of drawing, but not in the basic forms or pictorial stages. Children cannot use the systems to draw recognizable objects and scenes such as houses, people, cars and trees.

The difficulties in free-eye drawing can be explained in part based on the characteristics of human visual perception and oculomotor (eye movement) processing. First, free-eye drawing jams together two task activities that are usually independent when drawing a picture: eye movements to view the drawing, and motor movements to draw lines. Second, people do not have the same control over their eyes as over their hands and other limbs. People can move their eyes in short, quick bursts, but not slow adjusting movements. Alternative input techniques are needed for drawing with the eyes.

3. HOW EYEDRAW WORKS

3.1. Eye Tracking Terminology

Drawing with the eyes must be designed and accomplished at both the unit-task level of analysis [3] as well as at the the visual-perceptual and oculomotor subtask level. To understand how EyeDraw works, a few terms pertaining to eye movements and eye tracking must be defined.

The *gaze* is the vector that goes from the eye to the *gazepoint*, which is the point in a scene where a person is looking. The gaze moves around a static scene with a series of quick jumps called *saccades*, each of which lasts roughly

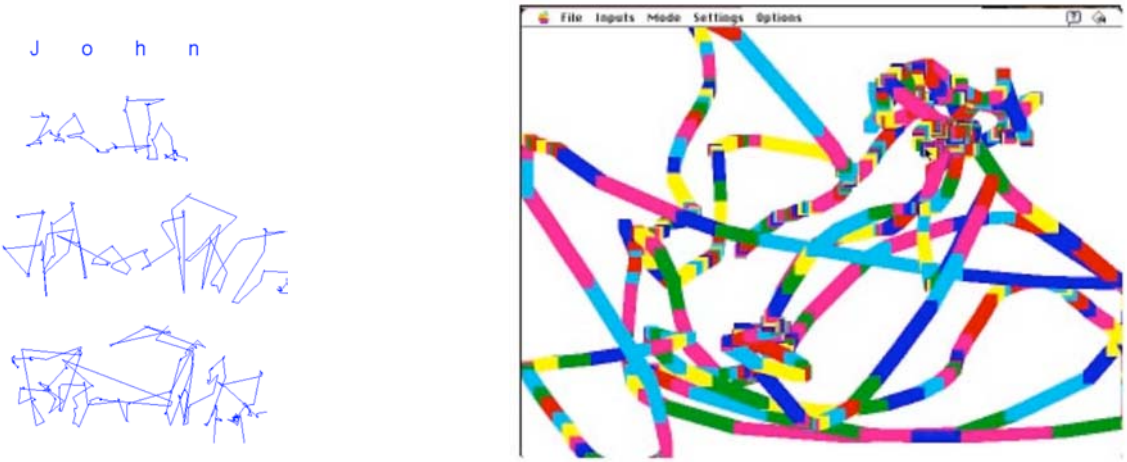


Figure 1. On the left, three attempts to free-eye draw the name "John" from Tchalenko [11]. On the right, a screenshot from EaglePaint, which displays randomly-colored ink wherever the user looks. [Adapted from 6]. Free-eye drawing appears to be difficult and not the best way to draw with the eyes.

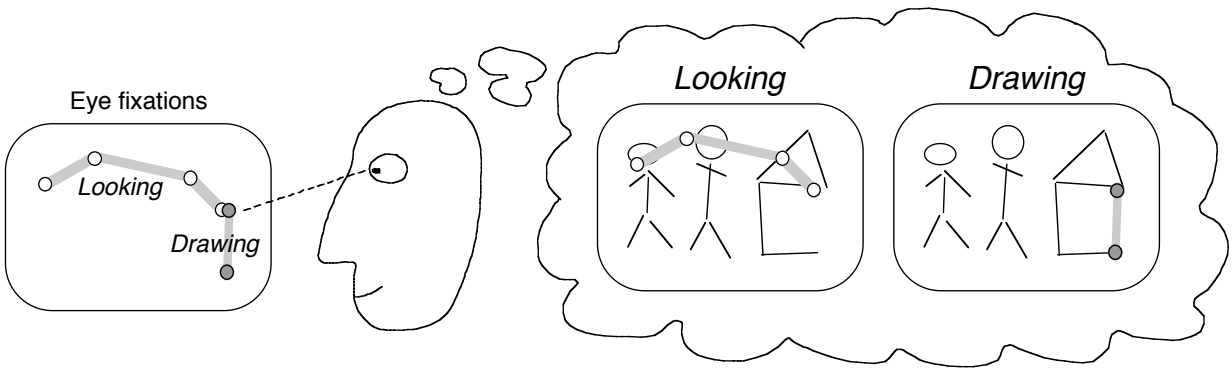


Figure 2. Without moving the gaze from the drawing, the EyeDraw user can shift between looking and drawing. The thick gray lines are eye movements. The circles are fixations. Gray circles are longer fixations that exceed a dwell threshold. Given this user input, EyeDraw would draw a line creating the right wall of the house.

30 ms. Between saccades, the gazepoint stays at the same location (with a slight tremor) for a *fixation* that lasts roughly 100 to 400 ms. A *dwell* is a long fixation. The reason that the eyes move, in short, is so that people can put items of interest into the high resolution vision which is at the center of their gaze.

An eye tracker generally reports the gazepoint on the computer screen 30 to 1000 times per second. EyeDraw uses the LC Technologies Eyegaze eye tracker, which reports the gazepoint 60 times per second, or once every 16.7 ms. The system uses the pupil-center corneal-reflection technique.

EyeDraw averages the location of every six consecutive gazepoints reported by the eye tracker and displays them on the screen as the *eye cursor*. The eye cursor is a colored square (seven pixels wide) that dances around the screen wherever the user puts their eyes, with a small, roughly 133 ms delay.

3.2. Alternating between Looking and Drawing

Figure 2 shows the basic idea of how the EyeDraw user can, while keeping their gaze on the picture, shift between using their eyes to (a) look at or study the drawing and (b) add to the drawing. This smooth subtask-switching is one of several differences between EyeDraw and previous software technology developed for drawing with the eyes. In both Tchalenko's free-eye drawing system and in EaglePaint, the ink effectively poured from the user's gaze. What resulted was a case of the "Midas touch" problem, in which anything the user looked at became activated. The user could not examine alternative spaces in which to draw or pick up the pen to move to the next character without putting down more ink all along the way. EyeDraw does not have such a problem.

Figure 3 shows how a user controls the drawing process in EyeDraw. The design departs from free-eye drawing by providing control that is one level removed from the direct coloring-in of pixels. Rather than draw directly, the user manages a drawing process. It is somewhat analogous to using a tool in drawing software for the general public. The user defines the starting and ending point of a line rather than drawing the line pixel by pixel. The EyeDraw user is still,

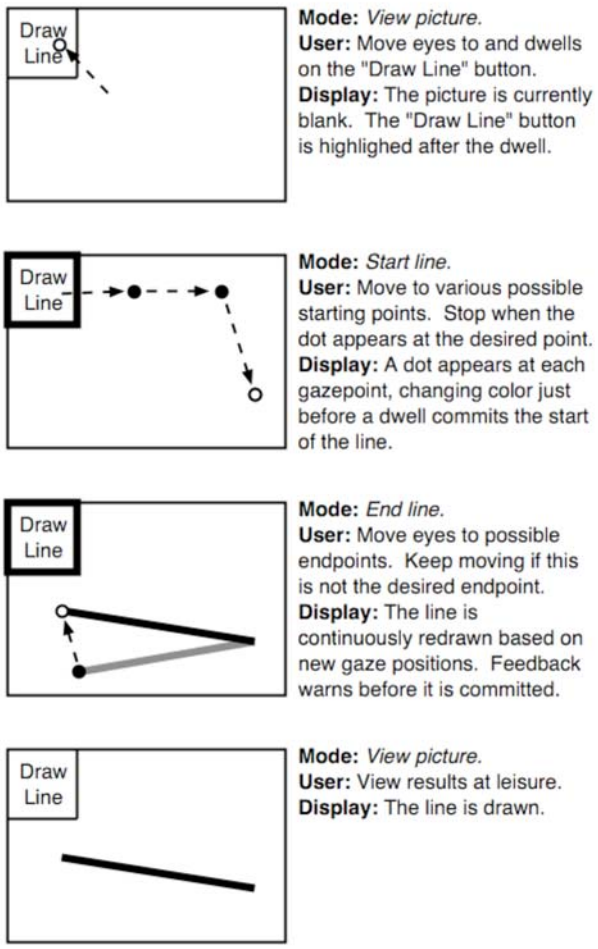


Figure 3. A storyboard of how the user controls the drawing of a line through a series of eye movements and dwells. A key feature in the interaction can be seen in the third frame, in which the line is continuously redrawn in new positions until the user selects the final position with a dwell. The same basic drawing technique is also used to draw circles and squares.

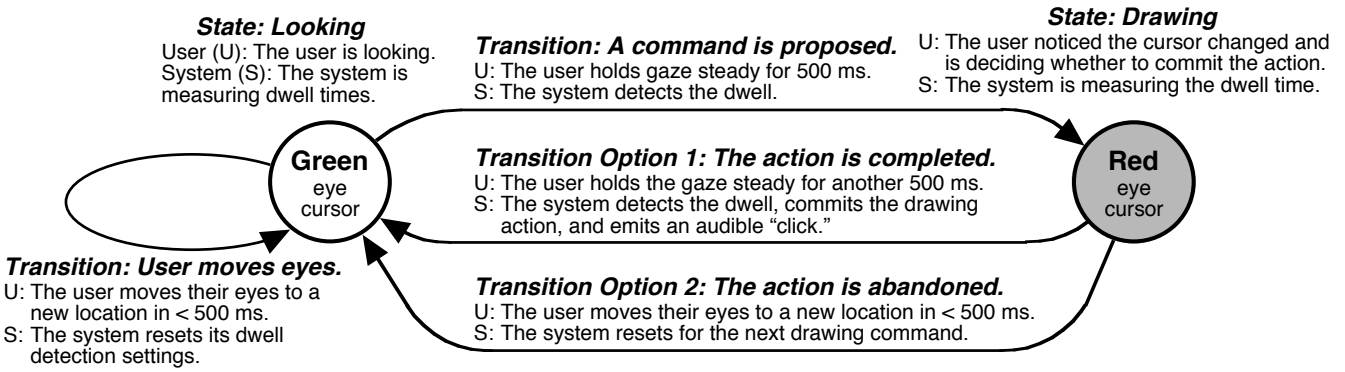


Figure 4. A state transition diagram showing the two eye-cursor states (green and red) and the various transitions between the two states while a user is drawing with their eyes using EyeDraw.

however, faced with the challenge of using the visual modality to both determine where to place the start and end of the line, and to place the points. This problem, which is resolved by a tight control and feedback loop around the eye cursor, is discussed next.

3.3. Issuing Drawing Commands

This section describes how the user controls the state of the eye-cursor, and thus the drawing process. Figure 4 shows the two states that the cursor will move through along the way to the user issuing a drawing command. The first *Looking* state uses a green cursor. As long as the user keeps moving their eyes around, the cursor will stay green.

If the gaze dwells at a location for a minimum amount of time, the program enters a *Drawing* state and the cursor changes to red. The dwell time is initially set to 500 ms, but is adjustable to accommodate different levels of ability. To stop the command from being issued, the user needs to move his or her eyes from the current location within 500 ms. This returns the user to the green *Looking* state without issuing a command. To issue the command, the user continues dwelling for another 500 ms, at which point EyeDraw executes the drawing command. Auditory feedback is also provided to confirm the drawing command was executed. The program then automatically returns to *Looking* state.

This transition between the looking and drawing states can be applied to a wide variety of drawing tools, including a line, square, and circle. The same basic control technique can be used to position and "stamp" clip-art onto a drawing.

3.4. Process-Interactive Control

EyeDraw incorporates detailed analysis and consideration of the perceptual, cognitive, and motor processes that users will apply to the eye-drawing process. One of many interesting and subtle interactions among the processes occurs when the user decides to move from the red *Drawing* state back to the green *Looking* state without executing a command.

Figure 5 shows the interaction among the device and various human processors when the user decides to *not* draw a line after the appearance of the red *Drawing* cursor. The user is following Transition Option #2 in Figure 4. The graph used in Figure 5 is a form of a CPM-GOMS analysis [9].

GOMS stands for Goals, Operators, Methods, and Selection Rules [3]. GOMS is a language and procedure for representing human procedural knowledge, and is typically used to analyze human-computer interfaces. CPM-GOMS is a form of GOMS that separates out the Cognitive, Perceptual, and Motor processes, and allows an analyst to analyze a task using the Critical Path Method.

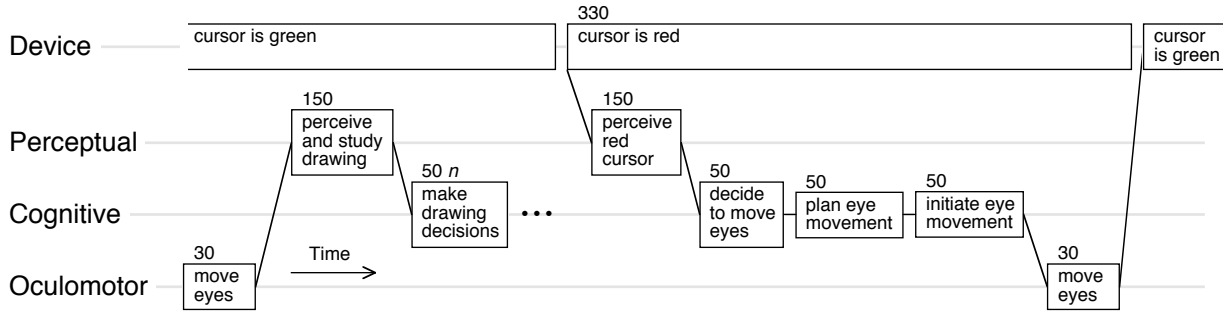


Figure 5. A CPM-GOMS model of the interaction among the device, perceptual, cognitive, and oculomotor processors when a user goes into the red *Drawing* state and then decides to go back to the green *Looking* state. Diagonal lines show dependencies among operators. Estimates of operator durations (in milliseconds) are shown.

The CPM-GOMS analysis helps the designers to understand and explain why they should not set the dwell times for transitioning between modes to the same dwell times that current eye-tracker users have set for their eye-typing application. Some users set the eye-typing dwell times as low as 180 ms. However, when eye-typing, the user does not need to perceive the position of the eye cursor after it arrives at the target and before deciding whether to click. Highly-proficient eye-typers learn the positions of the oversized keys on the computer screen; plan and execute fast, precise eye movements; and do not need to examine the position of the eye cursor landing on the key before making their decision to press the key.

When eye-*drawing*, however, it is necessary to decide on the starting or ending point of each line, and more perception and decision time is needed within each fixation. Though the eye-drawing dwell times should not be set to the same times used by eye-typing, it is possible that they can be set *based on* the eye-typing dwell times, allowing an extra appropriate amount of time for perceiving the position of the eye cursor. In Figure 5, once the cursor turns red, every operator is on the critical path. Based on this analysis, the eye-drawing dwell threshold cannot be set much lower than 330 ms if the user is to be able to deliberately exit a *Drawing* state.

The CPM-GOMS analysis also helps to illustrate why auditory feedback will be useful, and where it will be useful. In short, any time that the transmission of a discrete piece of information from the computer to the human's cognitive processor is on the critical path, auditory feedback should be considered. An auditory stimulus is perceived roughly 30 ms faster than a visual stimulus [14]. The 150 ms required for the *perceive red cursor operator* in Figure 5 could potentially be reduced by 30 ms if the stimulus were changed from visual to auditory. With an auditory *Drawing* warning, the user could start to decide whether to keep moving their eyes or start drawing 30 ms earlier. Dwell times for mode transitions could potentially be reduced by roughly 30 ms while maintaining the same level of interactive control. This is on the time scale that our users set their eye tracker settings, with eye-typing dwell times set as low as 180 ms.

3.5. Additional Important Features

Three additional features, in addition to the basic drawing control, were important in the initial deployment of EyeDraw Version 1.0. These include a grid of dots that the user can turn on and off while drawing, an "undo" button, and a facility for saving and retrieving drawings by just using the eyes.

The grid of dots is extremely useful to the EyeDraw users. Note that drawing a line in the manner described above requires the user to hold their eyes steady at a location in order to issue a drawing command, such as to specify the start and end points of a line. This is somewhat difficult to do when staring at a blank white field. The grid of dots, with roughly one cm between each dot, provides a screen full of visual anchors. They are particularly useful because users quickly learn, when using the system, that the eye cursor almost always appears with a slight error, not exactly where you are looking. If the user tries to fixate the eye cursor, they end up chasing it off the screen.

An "undo" feature seemed to be important so that users could back out of unintended drawing commands that would inevitably be issued while learning to use the system.

It was clearly important for users to be able to save their drawings for their own artistic expression and enjoyment—the whole point of the program—but also so that we could collect the drawings as data for evaluating the software during its development. There are actually two saving functions in EyeDraw: (a) saving and retrieving the drawings and (b) saving and retrieving *the eye-cursor commands that were used to create the drawings*. The second will be useful for analyzing how the tool is used by geographically remote users. It also turns out to be very useful for demonstrating the software in conference presentations.

EyeDraw is written in C++ using the Microsoft Foundation Class (MFC) GUI application framework within the Microsoft Visual C++ .NET programming environment, and using the LC Technologies Eyegaze application programming interface. EyeDraw is installed and runs directly on the Windows computer that is the major hardware component of the LC Eyegaze Communication System.

4. USER EVALUATION

4.1. Procedure

We evaluated EyeDraw Version 1.0 with nondisabled users. Version 1.0 includes a line-drawing and a circle-drawing tool, as well as the grid of dots, "undo," and a facility for saving and retrieving drawings. Figure 6 shows a screenshot of EyeDraw Version 1.0 with a drawing made by one of the authors. Note that the grid of dots is currently displayed.

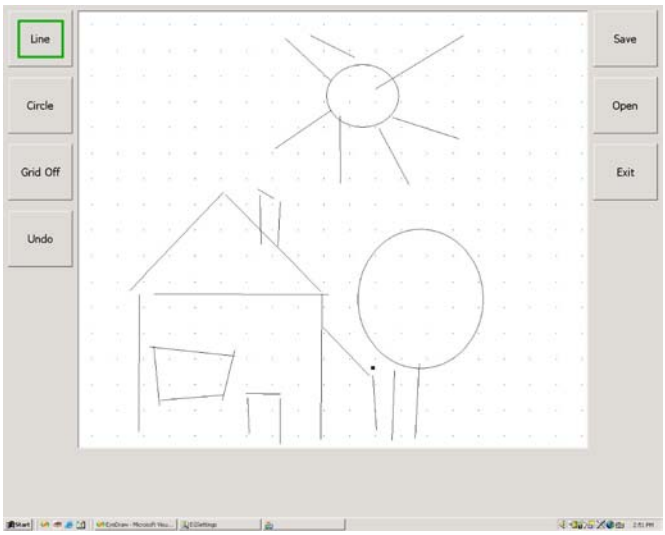


Figure 6. A screenshot of EyeDraw Version 1.0 and a drawing made by one of the authors (AC).

The primary driving question in our user observation study was simply whether users would be able to successfully use EyeDraw to draw recognizable pictures. Secondary questions included (a) which aspects of the drawing program were easier or harder, (b) what were the preferred settings for issuing drawing commands (for example, if 500 ms is a good dwell threshold), and (c) what were the participants' subjective impressions of using the software.

EyeDraw is ultimately intended for children with severe mobility impairments, such as those who currently use the LC Technologies Eyegaze Communication System and who could thus benefit from the software as soon as it is demonstrated that EyeDraw really works. However, there are no such users in our immediate geographical vicinity, so we conducted our initial evaluation with nondisabled users. This also allowed us to identify and resolve potential software issues with users that might find it easier to discuss problems they encounter, and for whom it is perhaps less important that the initial user experience is smooth and flawless.

Ten nondisabled participants were recruited. Four were female and six were male. Half were children (under eighteen years of age), with ages of 7, 10, 13, 14, and 16. The other half were 21 to 36 years of age, with an average age of 26.

Each session lasted a little under an hour. After preliminary paperwork and a brief questionnaire, the eye tracker was calibrated to the participant. Two participants were excused from the study at this point (ages 27 and 36) because the eye tracker failed to calibrate correctly for these participants.

For the remaining eight participants, each participant was briefly introduced to the basic functionality of the software, and asked to make some drawings. The four older participants were first presented with a simple pattern of lines and circles and asked to trace them (we thought this would be a little tedious for the younger participants). All participants were then asked to draw a picture or two of their choosing. Between drawing sessions, a few settings for controlling the eye drawing were sometimes manipulated to determine preferred settings, such as to evaluate dwell time of 500 ms.

4.2. Results

The preferred dwell time was consistently found to be 500 ms. We tried 250, 500, 750, and 1000 ms. Users had difficulty keeping their gaze at a single location, as was needed to issue a drawing command, for much longer than 1000 ms. The ideal setting appears to be somewhere around 500 ms, perhaps slightly faster for practiced users. The transition from the green to red cursor to indicate the transition from *Looking* to *Drawing* can optionally be set to include an additional intermediary yellow cursor state. Users, however, preferred the simpler two-state control.

After their drawing session, participants were asked to discuss their impressions of using the software. Participants found the grid of dots to be very useful, at least early in the experimental session. Most participants volunteered that they learned to compensate for the slight constant in the eye tracker by not looking directly at the eye cursor and by planning drawing movements taking the error into account.

Participants were asked to rate the ease of use and ease of learning of the program on a scale from 1 to 4, with 1 as "very easy" and 4 as "very hard." Participants generally found the program easy to learn (mean=1.6) and easy to use (1.7). The easiest tasks were clicking on the buttons with the eyes (1.2) and saving the drawings (1.3). The hardest tasks were controlling the eye cursor (2.2) and controlling the drawing (2.3).

Three of the four participants presented with lines and circles traced them quite well. The fourth had some difficulty with the circles, we believe because it took some practice to learn where the circle would start and stop based on the start and end point chosen. To trace the circle, the correct start and end points were actually well outside of the circle itself.

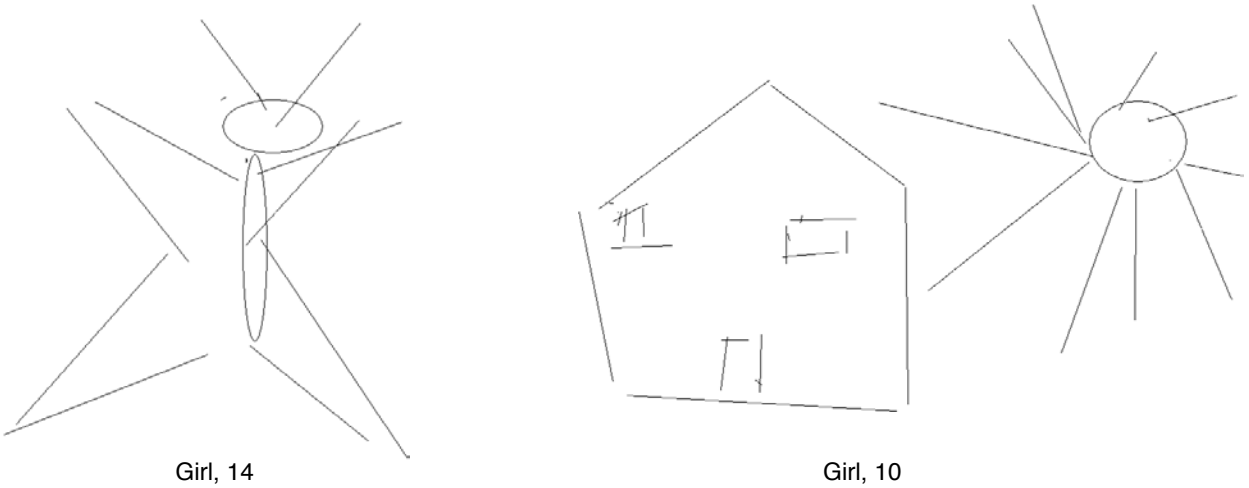
Seven out of the eight participants were able to draw a picture that was judged by the authors to be of a clearly recognizable scene. The youngest participant (seven years old) was the only one who did not draw a full picture, though he did gain control over the drawing process and was able to follow an experimenter's suggestions to draw lines from one region of the screen to another.

Figure 7 shows drawings created by the four other children who used the EyeDraw system. The drawings can be identified as a girl, a house and a car, a butterfly (the participant clarified), and a house in the sun. The adult drawings were of equivalent quality.

Though most participants volunteered that they found the activity to be fun, it was not particularly easy. This is at least in part because this was the first time the participants used the system. For those participants who created a second drawing, they liked it better than the first. This suggests that that not only using EyeDraw over time will get easier, but that drawing will improve with gaining familiarity using the eye tracker. Tchalenko observed such a practice effect with his free-eye drawing system. It does however appear as if drawing with the eyes requires a great deal of focused attention.

The quality of these drawings suggests that yes, children can use EyeDraw to draw pictures with their eyes. Participants produced these drawings the first time they sat down to use EyeDraw, which was also the first time they ever controlled a computer with their eyes. These drawings would be classified into the fourth stage of drawing, which is *early pictorial*. This is the stage at which the drawn shapes and configurations can be associated with people and things in the world. If EyeDraw supports the fourth stage, it will also likely support the first three: *random scribble, controlled scribble,* and *basic forms.* The seven-year-old participant, for example, appeared to work in the controlled scribble stage during his session with EyeDraw.

Girl, 13 Boy, 16



Girl, 14 Girl, 10

Figure 7. The drawings made by the four children in the user observation study conducted with nondisabled users. Each participant's age is shown below their drawing. This was the first time each participant ever used EyeDraw, and the first time they ever controlled a computer with their eyes.

We believe the drawings will improve, and move into the fifth and final *later pictorial* stage, as we increase the user's control over all aspects of the drawing by designing and implementing additional interactive techniques.

5. CURRENT WORK

A user observation study with disabled users who already use the Eyegaze Communication System is currently underway. We have sent a copy of EyeDraw Version 1.0 to caregivers of children and young adults who use the Eyegaze Communication System in their daily or weekly lives. We have not yet received drawings made by these users, or the eye movement recordings that will allow us to play back the steps they took to draw their pictures, but we have already learned of some enhancements to make to the software so that it would be more useful to these users. One is to display a video image of the eye on the screen so that the user and caregiver can know when the users has moved out of the camera's range.

One particular "power user" of the Eyegaze Communication System has tried out the EyeDraw software and has already made a number of suggestions—directly to the developers by eye-typed email—based on her usage of the system. Her suggestions include adding color to the drawing tools, and integrating EyeDraw into the main Eyegaze Communication System eye-controlled menu so that the user can independently decide when she wants to draw or do other computer tasks such as email.

We are already developing Version 2.0 of the EyeDraw, incorporating numerous enhancements that were desirable from the start but not critical to evaluate the feasibility of the

program, as well as some suggestions by our users. Features include additional shape tools, user-controlled color selection, and the ability to select from a series of clip-art images that the user can "stamp" down using the same basic drawing control used throughout the program. Figure 8 shows EyeDraw Version 2.0 with a drawing created by one of the authors using the tool to draw with his eyes.

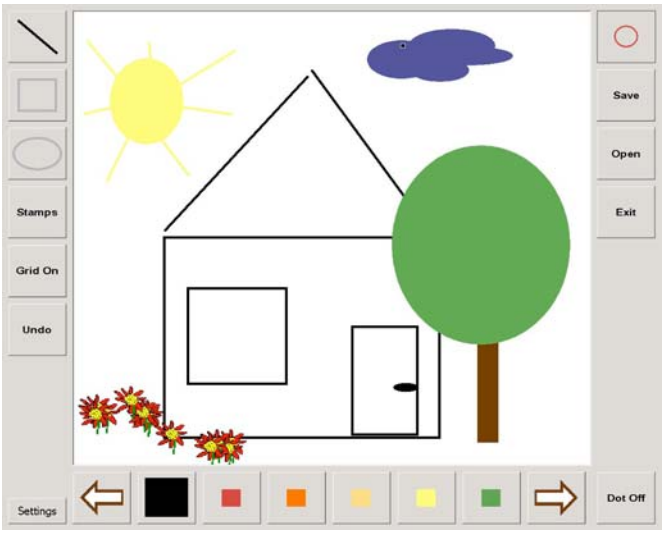


Figure 8. EyeDraw Version 2.0, currently in development, and an eye-drawing made by one of the authors (RH) using the software. The flowers were placed using a "stamping" tool.

6. CONCLUSION

This paper discusses the design and development of EyeDraw, an interactive software system that will enable children with severe motor impairments to do something that is currently impossible for them to do, to draw pictures by just moving their eyes. Our initial design and development remains focused on this one user group and task. We believe that this focus will help lead us to success. We will work directly with these users to improve the usefulness and functionality of the software as needed to better support this important childhood developmental activity.

ACKNOWLEDGMENTS

Tim Halverson assisted with eye tracking and design. Nancy Cleveland, Dixon Cleveland, and Peter Norloff, of LC Technologies, provided technical and domain expertise. Annie Zeidman-Karpinski assisted with the literature review.

The second and third authors are supported by the National Science Foundation through a Research Experiences for Undergraduates supplement to Grant IIS-0308244. The eye tracking lab used for this project is supported by the Office of Naval Research through Grant N00014-02-10440. Both of these grants are awarded to the University of Oregon with Anthony Hornof as the principal investigator.

REFERENCES

1. Bates, R., & Istance, H. (2002). Zooming Interfaces! Enhancing the Performance of Eye Controlled Pointing Devices. *Proceedings of ASSETS 2002: The fifth international ACM conference on Assistive technologies,* New York: ACM.
2. Bolt, R. A. (1982). Eyes at the interface. *Proceedings of Human Factors in Computing Systems,* New York: ACM, 360-362.
3. Card, S. K., Moran, T. P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.
4. Duncum, P. (1995). Colouring-in and alternatives in early childhood. *Australian Journal of Early Childhood*, 20(3), 33-38.
5. Escobedo, T. H., & Bhargava, A. (1991). A study of children's computer-generated graphics. *Journal of Computing in Childhood Education*, 2(4), 3-25.
6. Gips, J., & Olivieri, P. (1996). EagleEyes: An Eye Control System for Persons with Disabilities. *The Eleventh International Conference on Technology and Persons with Disabilities*.
7. Jacob, R. J. K. (1990). What you look at is what you get: Eye movement-based interaction techniques. *Proceedings of ACM CHI '90: Conference on Human Factors in Computing Systems,* ACM: New York, 11-18.
8. Jacob, R. J. K., & Karn, K. S. (2003). Eye tracking in human-computer interaction and usability research: Ready to deliver the promises (Section commentary). In J. Hyona, R. Radach, & H. Deubel (Eds.), *The Mind's Eyes: Cognitive and Applied Aspects of Eye Movements*. Oxford: Elsevier Science.
9. John, B. E., & Kieras, D. E. (1996). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction*, 3(4), 320-351.
10. Majaranta, P., & Räihä, K.-J. (2002). Twenty years of eye typing: systems and design issue. *Proceedings of the Symposium on Eye Tracking Research and Applications: ETRA 2002,* New York: ACM Press, 15 - 22.
11. Tchalenko, J. (2001). Free-eye drawing. *Point: Art and Design Research Journal*, 11, 36-41.
12. Vertegaal, R. (2003). "Attentive User Interfaces" Editorial, Special Issue on Attentive User Interfaces. *Communications of the ACM*, 46(3).
13. Ware, C., & Mikaelian, H. H. (1987). An evaluation of an eye tracker as a device for computer input. *Proceedings of ACM CHI+GI,* New York: ACM, 183-188.
14. Welford, A. T. (1980). Choice reaction time: Basic concepts. In A. T. Welford (Ed.), *Reaction Times*. New York: Academic Press, 73-128.
15. Zhai, S., Morimoto, C., & Ihde, S. (1999). Manual and gaze input cascaded (MAGIC) pointing. *Proceedings of ACM CHI '99: Conference on Human Factors in Computing Systems,* New York: ACM, 246-25.