Camera Canvas: Image Editing Software for People with Disabilities

Christopher Kwan and Margrit Betke

Image and Video Computing Group, Department of Computer Science,
Boston University, 111 Cummington St, Boston, MA 02215, USA
{ckwan,betke}@cs.bu.edu

Abstract. We developed Camera Canvas, photo editing and picture drawing software for individuals who cannot use their hands to operate a computer mouse. Camera Canvas is designed for use with camera-based mouse-replacement interfaces that allow a user with severe motion impairments to control the mouse pointer by moving his or her head in front of a web camera. To make Camera Canvas accessible to as wide of a range of movement abilities as possible, we designed its user interface so that it can be extensively tailored to meet individual user needs. We conducted studies with users without disabilities, who used Camera Canvas with the mouse-replacement input system Camera Mouse. The studies showed that Camera Canvas is easy to understand and use, even for participants without prior experience with the Camera Mouse. An experiment with a participant with severe cerebral palsy and quadriplegia showed that he was able to use some but not all of the functionality of Camera Canvas. Ongoing work includes conducting additional user studies and improving the software based on feedback.

Keywords: Assistive Technology, Camera Mouse, Human Computer Interaction, Image Editing, Mouse Replacement System, Photo Editing, Video-based Interface.

1 Introduction

Individuals who cannot speak and cannot use their hands to operate a computer mouse are extremely limited in their means of communication. The goal of our work was to design image-editing software that these individuals can use for communication and as a canvas for expression. The process of developing Camera Canvas helped us gain general knowledge about interface design for people with severe motion impairments. We gained insights and developed techniques that may be applied to future projects for this user group.

We built upon our experience in developing software for camera-based mouse-replacement interfaces; in particular, the Camera Mouse input system [2], which enables a user to control the mouse pointer by moving his or her head in front of a camera. The Camera Mouse [2] is software that was developed at Boston College and Boston University and is freely available on the web for download at www.cameramouse.org. With this version of the Camera Mouse, only left-click events can be simulated. A left-click event is registered when the user "dwells," or keeps the mouse pointer within a small radius of the item to be selected, for a certain amount of time, e.g.: one second. We suggest that Camera Canvas can also be used with other camera-based mouse-replacement interfaces [5], [10].

There have been various other kinds of applications specially designed for use with Camera Mouse [1]. In particular, software has been developed to enable people with motor impairments to create drawings: Eagle Paint [1] is a program designed for use with the Camera Mouse that allows users to draw freeform lines, EyeDraw [7] is a drawing program designed for use with an infrared eye tracker, and VoiceDraw [6] is a drawing program that allows users to draw freeform lines by making different sounds with their voices. There has also been work to create customizable [9] and automatically generated [4] user interfaces for people with motor impairments.

Our goal was to develop a program that gives users with motor impairments photo-editing and drawing capabilities and that has a user interface that can be extensively customized to meet the needs of each user. Many interactions in image-editing involve complex mouse actions, such as clicking and dragging, which are not possible with the Camera Mouse input system. To make such interactions possible, we needed to redesign such actions completely. We also had to take into account the wide range of movement abilities of individuals with motor impairments. Some users have better control of their movements along certain axes [3], some users have good control of their movement but only within a certain range [11], and some users can only click buttons of a certain minimum size. We also considered how our software can remain usable for users whose abilities degrade over time. To meet the needs of as many users as possible, we designed our software to be highly configurable.

2 Methods

2.1 Sliding Toolbar

The main user interface element of Camera Canvas is the Sliding Toolbar. It consists of two panels: a tool menu panel containing specific image editing tools and a navigation panel containing navigation buttons. When a tool from the tool menu panel is selected, if that tool has a submenu, the buttons of that submenu will replace the current buttons in the tool menu. The user can get back to the previous menu of buttons by pressing the Back button in the navigation panel or go back to the topmost tool menu by clicking the Main Menu button in the navigation panel.

The Prev and Next buttons in the navigation panel allow the user to reposition the tool panel by sliding it sideways. This sliding ability addresses the problem of some users only having good control within a certain range. If users cannot reach a button on the edge of the screen, they can use the Prev and Next buttons to slide the buttons towards the center. The direction of movement is from the perspective of the button currently in the center position of the toolbar (above the space between the Prev and Next buttons). Pressing the Prev button will cause the button in the previous position to the center position to slide to the center position. Similarly, pressing the Next button will cause the button in the next position after the center position to slide to the center position. As long as the user keeps the mouse cursor on top of the Prev or Next button, the toolbar will continue to automatically slide on an adjustable interval.

2.2 Configuration

Camera Canvas has three configurable settings in the Settings menu: toolbar placement, button size, and toolbar sliding speed. These settings can all be changed at runtime using tools within the application. The tools are designed to be easy to use so that the user can actually modify the configuration himself. The placement and orientation of the toolbar can be changed to four settings: Horizontal-Top (Fig. 1, left), Horizontal-Bottom, Vertical-Left, and Vertical-Right (Fig. 1, right).

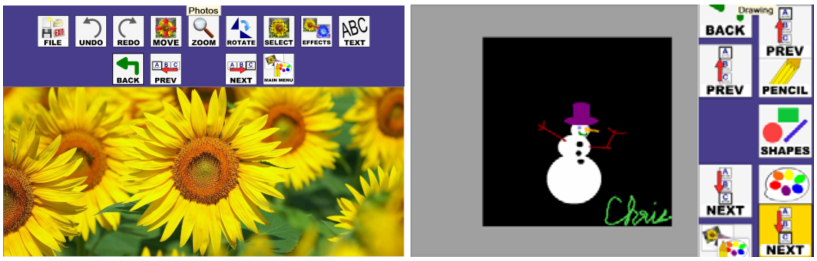
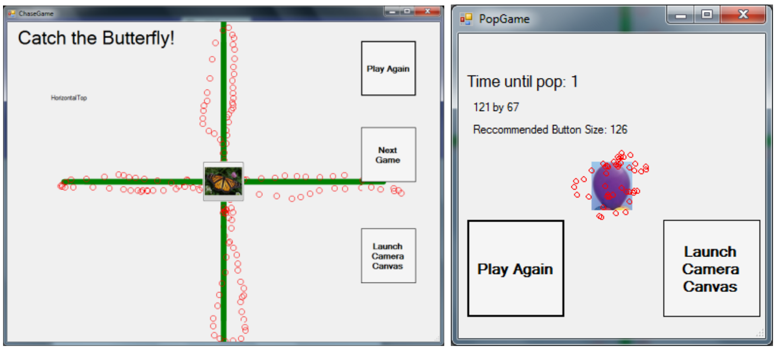


Fig. 1. Camera Canvas in Photo-editing mode with a Horizontal-Top layout and smaller buttons (left) and in Drawing mode with a Vertical-Right layout and larger buttons (right). The second set of Prev and Next buttons signify that there are more buttons off-screen.



Time until pop: 1
121 by 67
Reccommended Button Size: 126

Fig. 2. The “Catch the Butterfly” game recommends which axis and area of the screen are best for the user by having her follow a butterfly (left). Green lines show ideal mouse trajectory, red circles show actual trajectory. The “Pop the Balloon” game recommends a button size for the user by having her try to keep the mouse cursor still within a small area (right). The balloon is the ideal area; red circles show the actual mouse movement area.

Each setting aims to constrain movement primarily along a single axis and in a single area of the screen to address the challenges of users having better movement abilities along different axes and users being able to reach different areas of the screen more easily than other areas. The size of all buttons in the application can be made smaller and larger to address the challenge of different people being able to click buttons of different minimum sizes. Finally, the interval at which the toolbar buttons slide can also be changed so that the buttons slide faster or slower. The Settings menu also contains a configuration wizard for Camera Canvas in the form of two simple, easy to understand games (Fig. 2). These games look at a user's performance using the Camera Mouse and recommend settings for Camera Canvas that would make it the most usable for that user.

2.3 Photo-Editing Tools

For the Photo-editing mode of Camera Canvas, we developed several interaction techniques to make common photo-editing tasks possible with camera-based interfaces. The Rotate tool uses a custom user interface component called a Choice Form (Fig. 3, left) that is an alternative to components such as sliders or small increment arrows, which are challenging for users who have difficulties controlling the mouse pointer. The middle of the Choice Form contains a preview of the rotated image so that the user can see the effects of the rotation before actually committing the change. The Choice Form is also used by many other tools in Camera Canvas.

The Move and Zoom tools place four translucent arrows in the middle of the screen. To pan around the image, the user puts the mouse pointer over one of the arrows and the image automatically moves until the user moves the mouse pointer off of the arrow. No matter the size of the image, the user only needs to make small movements between the arrows to pan, instead of having to physically move the mouse pointer around the entire image.

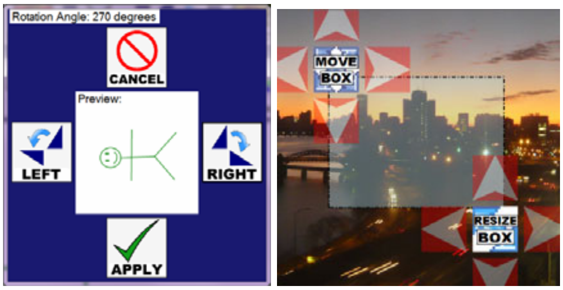


Fig. 3. Rotating a drawing using a Choice Box (left). Selecting a portion of an image (right).

Instead of the traditional click and drag method of selecting a portion of an image, the Select tool uses two sets of arrows similar to the ones used in the Move and Zoom tools. When using Select, a translucent blue rectangle (representing the selection) and two sets of arrows appear in the center of the image. The set of arrows on the left control the position of the top-left corner of the selection box and the set of arrows on the right control the position of the bottom-right corner of the selection box. By moving the mouse cursor in each of the arrows, the user can control the position and size of the selection box. Once the selection box is of the desired position and size, the user can then cut, copy, paste, or crop the selection. The two sets of arrows never change positions so no matter the size of the selection, the user can control it using only small movements between the two sets of arrows.

2.4 Drawing Tools

The Camera Canvas interaction for drawing straight lines and geometric shapes was inspired by the drawing process in EyeDraw [7]. To address the “Midas touch” problem for drawing (how to differentiate looking at the picture versus actually drawing the picture) the researchers of EyeDraw created a system where if the user looked at one spot for some amount of time, the cursor would change colors to signify that drawing was about to begin; if the user was just looking and did not want to actually start drawing they would just need to look elsewhere.

In Camera Canvas, to start drawing, the user must first dwell on the area where she would like to place the starting point of her drawing. After a click is registered, a green helper box appears where she clicked to signal that drawing is about to begin. If the user would actually like to start drawing at that point, she keeps the mouse cursor in the green Helper Box long enough for another click to register and then drawing begins. If the user does not want to place the starting point at that location, she only needs to move the mouse cursor out of the green Helper Box and it disappears, resetting the process. As the user is drawing, the line or shape is continuously redrawn with the ending point at the current position of the cursor. When the user wants to end the drawing, she dwells where she would like to end the drawing and a red Helper Box appears. If she would in fact like to place the end point of the drawing at that point, she just needs to keep the cursor inside of the red Helper Box. If she does not want to place the end point there and instead wants to continue drawing, she just needs to move the cursor out of the red Helper Box and it disappears. The sizes of the Helper Boxes are the same size as the toolbar buttons and will change if the button size is changed. The drawing process is outlined in (Fig. 4, left).

Instead of using a traditional color palette which relies on sliders or clicking of a precise point in a color wheel, we implemented a simple color palette that is much more usable with Camera Mouse but still gives users a fair amount of color variety. The color menu (Fig. 4, right) first displays a set of primary colors: black, white, brown, red, orange, yellow, green, blue, and violet. When the user clicks on a primary color, nine different shades of that color are then automatically generated for the user to choose from.

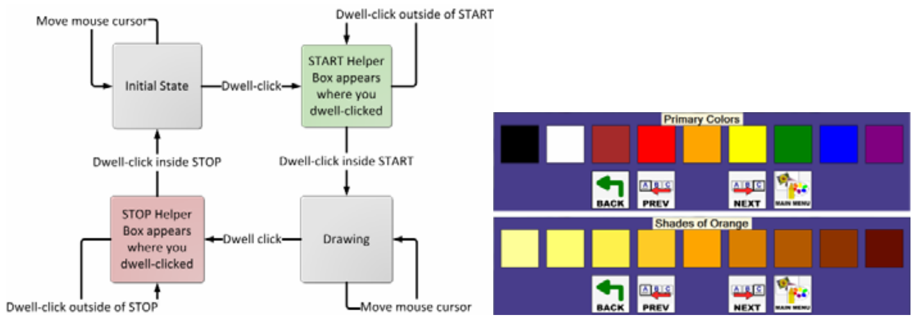


Fig. 4. The drawing process in Camera Canvas (left). The Camera Canvas color palette generating different shades of orange (right).

3 Experiments and Results

We conducted several user studies with a total of 28 users without disabilities. Their age ranges included elementary school, middle school, high school, and college age ranges. All users had never used the Camera Mouse input system before. The goal of these studies was to obtain a qualitative assessment of the program to see if it was easy to understand and use.

We asked the users to use various Photo-editing tools to manipulate a photo, use various Drawing tools to draw a shape, and then to play around with changing different configuration settings. There was no strict test plan; users were given freedom to explore the different features of the program as we observed them.

In general, the participants of our experiments found the software easy to understand and use even without prior experience using the Camera Mouse. With a little experimentation time, users were quickly able to start drawing shapes and manipulating images. We found that nearly all users enjoyed the drawing tools the most and spent most of their time with the program drawing (Fig. 5). The users gave us valuable feedback on which features needed improvement and also what features they wanted to see in future versions, common suggestions were a fill tool and clip art stamps.

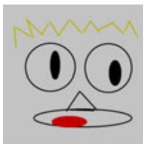


Fig. 5. A drawing created by a user without disabilities

We recently conducted experiments with a non-verbal user with severe cerebral palsy and quadriplegia (Fig. 6). This user had participated in experiments with the initial version of Camera Canvas [8]. In the prior experiments, the user was excited about the prospect of manipulating images but was unable to use the majority of the features [1]. In our test procedure, we first explained the general purpose of the program to him. We explained how the toolbar worked and that we could change its position, button size, and sliding speed if necessary. From an earlier experiment with different software during that session, we found that this user had a difficult time reaching buttons at the top of the screen. We therefore changed the Camera Canvas toolbar to have the Horizontal-Bottom layout and made the button size larger.

In our experiments, we asked the participant to try out the different modes of the program. Originally we had a detailed test procedure planned for him but the session turned into more of a qualitative exploratory session where we observed which features he was able to actually use.

After we explained the Sliding Toolbar to the user, he understood how the interface worked and was able to use the Prev and Next buttons to slide the toolbar buttons he wanted towards the middle of the screen. The Prev and Next buttons had mixed results. Although the user could slide to buttons that he wanted to reach, often times he would slide the toolbar too much and overshoot the button he wanted or would accidentally activate the Prev or Next button when trying to click on a button in the tool menu, causing his intended target to shift. To address this problem, we tried to slow down the sliding speed setting, but the user still hit the Prev and Next buttons by accident because of their proximity to the tool menu buttons.

In general, the user kept accidentally clicking neighboring buttons to his intended button because the buttons were too close together. This suggests that the buttons should be spaced farther apart or that this setting should also be adjustable. A particularly frustrating experience for the user was accidentally hitting the Main Menu button when he was in the middle of trying to apply an effect to the image. Accidentally hitting the button would take the user all the way to the Main Menu of the program and then he would have to click on Photos, slide down to Effects, and then click on the effect again. This happened multiple times and eventually we took control of the mouse in order to get the user back to the Effects menu again.

The observations that the user had to keep sliding to reach buttons near the edge of the screen and that he kept hitting buttons accidentally, suggest that the user might benefit from the toolbar having fewer buttons. A greater amount of buttons on the toolbar increases the chance for error and also may be cognitively overwhelming for someone using the program for the first time. Perhaps a more hierarchical approach (more levels with fewer buttons at each level) would be more usable for this user.

In general, even with larger buttons, it was difficult for this user to stay on top of one button long enough for the click to register. Shortening the time required for a click in the Camera Mouse settings helped reduce the problem slightly but in general it still persisted. In the future, we may experiment with adjusting the dwell radius setting in Camera Mouse to more closely match the button size in Camera Canvas. It is possible that even though the user placed the pointer on top of the button, he may not have kept it within a small enough radius for Camera Mouse to register a click.

The ability to configure the user interface of Camera Canvas was very important in experiments with this user. We used all three of the configuration options: toolbar placement, button size, and sliding speed to try to make the most usable layout for the user. The user also played the configuration games. He was able to understand and complete both the butterfly (toolbar layout configuration) and the balloon (button size configuration) games, although the layouts of the buttons in both games could be improved or ideally made configurable. The games recommended a Vertical-Right layout with buttons of size 160 by 160 pixels for the user. The user was satisfied with these settings and chose to keep them for the remainder of the experiment. Although the user liked these settings, we do not know if there were settings that could have made the program even easier for him to use because we stopped trying different settings after the user indicated he was satisfied.

In the Photo-editing tools, the user was able to successfully use the Move and Zoom features to zoom the image to a greater magnification and then pan the image so that a particular portion was centered on the screen. Even though the arrows of the Move feature were a fixed size, the user was still able to activate them because they were activated whenever the mouse entered the region rather than forcing the user to stay in the region for an amount of time, as is the case with buttons. This suggests that a boundary crossing or mouse touch approach instead of buttons might be more usable for this user. The user was also able to apply the Invert Colors feature to the image and then undo the change.

In the Drawing mode, the user was able to select different shapes and then draw rectangles around the image [Fig. 6, left]. We are not sure if the user intended to draw something specific or was just experimenting with the tool, as this may have been the first time that this user had interacted with a drawing interface.

We learned a great deal from our experiments with this user. We were able to see problems with the interface that did not arise when testing the software with users without disabilities. While there were many features that the user had trouble with, could not use, or did not try, in general the experiment was a major improvement over the experiment with the initial version of the software with this user.

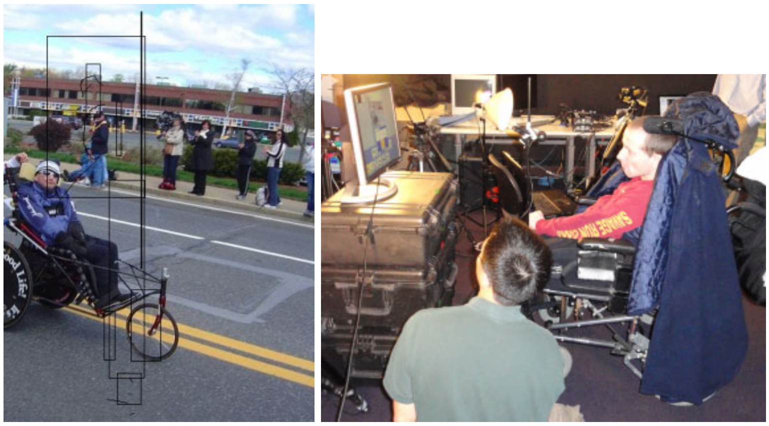


Fig. 6. An image edited by a user with severe cerebral palsy. He was able to rotate the image (presented to him upside-down) and experiment with drawing several shapes on the image (left). A user with severe cerebral palsy interacting with Camera Canvas using the Camera Mouse (right).

4 Ongoing Work

Our ongoing work involves conducting additional user studies with people with motor impairments. We are also working on improving the program based on observations from our experiments with the user with severe cerebral palsy. We hope to make modifications to the software to make it more usable for this user and conduct additional experiments with him. We are also adding additional features to the program, such as the much requested fill and clip art stamps and continuing to look into using simple games to recommend user interface settings.

Acknowledgements. We would like to thank the participants of our user studies for their valuable feedback. We would also like to thank Mikhail Breslav, Samuel Epstein, Nathan Fuller, John J. Magee, Eric Missimer, Isaac Paquette, Ashwin Thangali, Diane Theriault, Gordon Towne and Zheng Wu for their support throughout the development of this program and their assistance during user studies. We gratefully acknowledge funding from the National Science Foundation (HCC grants IIS-0910908, IIS-0713229, and IIS-0855065) and from the Boston University Undergraduate Research Opportunities Program.

References

1. Betke, M.: Intelligent interfaces to empower people with disabilities. In: Nakashima, H., Augusto, J.C., Aghajan, H. (eds.) Handbook of Ambient Intelligence and Smart Environments, pp. 409–432. Springer, Heidelberg (2009)
2. Betke, M., Gips, J., Fleming, P.: 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 (2002)
3. Connor, C., Yu, E., Magee, J., Cansizoglu, E., Epstein, S., Betke, M.: Movement and recovery analysis of a mouse-replacement interface for users with severe disabilities. In: 13th International Conference on Human-Computer Interaction (HCI International 2009), San Diego, CA (2009)
4. Gajos, K.Z., Weld, D.S., Wobbrock, J.O.: Automatically generating personalized user interfaces with Supple. Artif. Intell. 174(12-13), 910–950 (2010)
5. Gorodnichy, D., Dubrofsky, E., Ali, M.: Working with computer hands-free using Nouse perceptual vision interface. In: Proceedings of the International CRV Workshop on Video Processing and Recognition (VideoRec 2007). NRC, Montreal (2007)
6. Harada, S., Wobbrock, J.O., Landay, J.A.: VoiceDraw: a hands-free voice-driven drawing application for people with motor impairments. In: Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility, pp. 27–34. ACM, Tempe (2007)
7. Hornof, A.J., Cavender, A.: EyeDraw: Enabling children with severe motor impairments to draw with their eyes. In: Proceedings of ACM CHI 2005: Conference on Human Factors in Computing Systems, pp. 161–170. ACM, New York (2005)
8. Kim, W.-B., Kwan, C., Fedyuk, I., Betke, M.: Camera Canvas: image editor for people with severe disabilities. Department of Computer Science Technical Report BUCS-2008-010, Boston University (2008)
9. Magee, J., Betke, M.: HAIL: Hierarchical adaptive interface layout. In: Miesenberger, K., Klaus, J., Zagler, W., Karshmer, A. (eds.) ICCHP 2010. LNCS, vol. 6179, pp. 139–146. Springer, Heidelberg (2010)
10. Manresa-Yee, C., Varona, J., Perales, F.J., Negre, F., Muntaner, J.J.: Experiences using a hands-free interface. In: Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility, pp. 261–262. ACM, New York (2008)
11. Paquette, M., Betke, M., Magee, J.: IWeb Explorer: A web browser designed for use with an eye controlled mouse device. Computer Science MA Thesis Report, Boston University (2005)