Movement and Recovery Analysis of a Mouse-Replacement Interface for Users with Severe Disabilities

Caitlin Connor, Emily Yu, John Magee, Esra Cansizoglu, Samuel Epstein, Margrit Betke

Department of Computer Science, Boston University, Boston, MA 02215, USA

Abstract. The Camera Mouse is a mouse-replacement interface for users with movement impairments. It tracks a selected body feature, such as the nose, eyebrow or finger, through a web camera and translates the user's movements to movements of the mouse pointer. Occasionally, the Camera Mouse loses the feature being tracked, when the user moves quickly or out of frame, or when the feature is occluded from view of the web camera. A new system has been developed to recognize when the tracked feature has been lost and to locate and resume tracking of the originally selected feature. In order to better understand the directions of movement which are most and least comfortable for users with disabilities, a game interface was developed to test the accuracy and speed of users across different trajectories. The experiments revealed that trajectories most comfortable for a user with severe cerebral palsy were along diagonal axes.

Keywords: HCI, Assistive Technology, Camera Mouse, Video-based Interface

1 Introduction

Approximately 0.3 percent of the population worldwide suffers from a severe disability which can cause movement impairment [1]. This includes individuals with Cerebral Palsy, Spinal Muscular Atrophy, Amyotrophic Lateral Sclerosis, Multiple Sclerosis, and various neurological disorders. They would benefit from computer access but lack the ability to manipulate a traditional mouse. The "Camera Mouse" was developed to provide computer access to such users, who often are able to produce voluntary motions with their head or some part of their face [2,3]. The system uses a video camera or webcam to track a body feature, such as the nose, eyebrow, or finger, translating the user's movements into movements of the mouse pointer on the computer screen. For several years, the Camera Mouse has been helping users with disabilities successfully access a computer for purposes of communication, education, and entertainment [4].

In order to improve the Camera Mouse experience for users, two issues have been studied in depth. One issue is that for many users, use of the Camera Mouse requires the constant presence of a caretaker. When the system loses the selected feature during tracking, either by tracking a different feature or by attempting to track a piece of background, it is necessary to reselect the original feature manually, which in most

cases requires the intervention of a caretaker. For users who can operate independently otherwise, this can be a frustrating experience, especially for users who exhibit spastic motions, since such motions can cause the loss of the tracked feature. In response to this first issue, a new Camera Mouse system has been developed which automatically recognizes the loss of a tracked feature and automatically reselects that feature, so that the user can operate independently throughout the duration of his or her computer session.

There are many software applications which work well with or are designed to work in conjunction with Camera Mouse [4]. They are designed with the needs of users with disabilities in mind. Generally, non-disabled users are able to move their heads comfortably in all directions when using the Camera Mouse to interact with a software application. However, it has been noticed over previous testing sessions for the Camera Mouse that many users with disabilities have difficulty moving their heads in certain directions. In response to this issue, the second part of this study aims to identify which trajectories are consistently more and less comfortable for the users with disabilities than others. Once the most comfortable trajectories have been determined, software applications can be created which require the most movement over the most comfortable axis.

Our work relates to efforts by the Computer Vision community to develop facial feature tracking systems [5]. Many existing systems require some manual initialization, and few are able to recover from loss of track, for example due to occlusion [5-8]. The approach that is most similar to our own uses a threshold for correlation to identify loss of feature [6].

2 Methodology

The Camera Mouse software is an accessible and inexpensive mouse replacement interface, as it is available online for free download and works in conjunction with a standard USB webcam [3]. Upon starting the system, the interface displays the video feed and prompts the user to manually select a feature in the image frame to track. Once a feature has been selected, tracking of that feature begins, as shown in Figure 1 left. The Camera Mouse interface allows toggling between mouse pointer control via the tracked feature and via the standard mouse. The transition from tracked feature to standard mouse control can be initiated by pressing the “Num” key or simply by moving the standard mouse. Such a transition is necessary if the user wants to select a different feature to be tracked or if the tracker lost the initially selected feature.

The most common reasons that a feature being tracked is lost are that the user moves the feature out of the image frame, that an object moves in front of the feature, blocking it from view, or that the user makes a sudden movement so that the feature has moved far from where it was positioned in the previous frame. This last issue is of particular concern for users with disabilities because many exhibit spastic motions that can cause loss of the tracked feature. In all cases of loss, the Camera Mouse software will continue to track some part of the image, tracking either another feature

on the user or a piece of the background and using the movement of that feature to direct the movement of the mouse pointer.

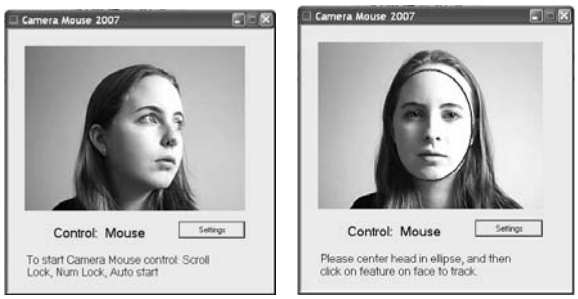

Fig. 1. The Camera Mouse interface which appears on the computer screen. Left: The interface during tracking. The green square appears over the feature being tracked. Right: The New Camera Mouse interface upon startup with oval graphic.

Our new system recovers a lost feature with a two-stage process. In the first stage, the system performs periodic checks on the tracked feature in order to recognize when the feature is lost. The second stage, the re-initialization stage, involves finding and reselecting the original feature to continue tracking it. An overview of the system is shown in Figure 2.

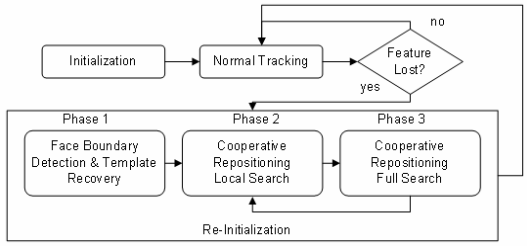

Fig. 2. Overview of new Camera Mouse system.

2.1 Recognition of Breakdown of Interaction

At the time the new Camera Mouse system starts, an oval graphic is inscribed on the currently viewed video feed, and the user is prompted to center his or her head in this shape before manually selecting the facial feature to track, as shown in Figure 1 right. Once the user does so, the coordinates of the pixel selected are stored for use in the re-initialization phase later. Once the feature has been selected manually, that feature is tracked.

The system prepares for the possibility of tracked feature loss by saving a template, a sub-image of x by x pixels around the point which is manually selected, at the time of selection. The system detects that the original feature has been lost, or

that the feature being tracked is not the originally selected feature, by periodically comparing the area of x by x pixels around the point being tracked in the current frame to the previously stored template every t image frames. The template and current sub-image are compared by calculating the normalized correlation coefficient (ncc) between them, based on the brightness values of their corresponding pixels [9]. If the normalized correlation coefficient falls below a threshold r_n, then the feature being tracked and the originally selected feature are judged to be different, and the original feature is considered lost.

To account for the possibility that the system is tracking a piece of background with brightness patterns similar to that of the original feature, the new Camera Mouse system also compares color values. The idea behind this comparison is that skin and hair colors have relatively more red and less blue tones than do most background colors, so comparing the color values should distinguish between them. At the time of manual feature selection, when the template is saved, the averages of the red, green, and blue (RGB) color values of the pixels in the template are computed and normalized. The normalization is computed for the red component as follows

Normalized average red value = $\sum$(red values of each pixel) / [$\sum$(red values of each pixel) + $\sum$(green values of each pixel) + $\sum$(blue values of each pixel)],

and for the green and blue components similarly.

At the time of the system's periodic comparisons between template and tracked feature, the normalized RGB values of the tracked feature area are calculated, and so is the difference between them and the stored, normalized RGB values. If this difference is greater than d for any color, then the feature being tracked and the originally selected feature are judged to be different, and the original feature is considered lost.

2.2 Re-initialization of Interaction

When the originally selected feature is considered to be lost, tracking is terminated and the re-initialization phase begins. The phase consists of three sub-phases which each implement a different searching strategy. The first sub-phase finds the lost feature in most cases, and the third sub-phase is rarely necessary. To find the lost feature, the new Camera Mouse tries to isolate the facial region by computing areas with changes in brightness from one frame to the next. The idea behind this is that if the user moves his or her head, the greatest brightness changes occur to the right and left of the user's head. To detect the left and right sides of the face, the system compares the two most recent image frames, taking the difference in brightness values of the corresponding pixels between the images to create a difference image. Summing the differences in brightness for the pixels in vertical columns of the image that are w pixels wide, the system detects the two columns with the greatest total difference mark the boundaries of movement. The area between the two columns with greatest brightness difference delineates the region of the image where the face is positioned. If the feature became lost because of a sudden spastic movement of the user to the side, the face should be especially easy to find with this method.

Once the left and right boundaries of the face are established, the top and bottom boundaries are determined to be *y* pixels above and below the y-coordinate of the originally selected feature. The y-coordinates are determined in this way because a user's vertical range of motion with a facial feature is much more limited than his or her horizontal range of motion, so the feature is almost always within this vertical range. Once the vertical and horizontal boundaries of the face have been established, the bounded region is searched for the lost feature. The search is executed by calculating the normalized coefficient of the original feature template and the *x* by *x* pixel area that has its corner positioned every *s* pixels to the right and down from the upper left corner of the bounded region. Of these calculations, the ncc and location of the area with the maximum ncc are saved, and if the ncc is greater than the threshold r_n, the system considers the feature to be found and tracks that area.

If the ncc falls below the threshold r_n, the feature is still considered lost, and the second sub-phase of search is initiated, which requires user cooperation. The oval graphic reappears, and the user is prompted to center his or her head in the oval. This way, the feature should be in the same area of the image that it was at the time of initial selection. After a pause in which the user can reposition his or her face, a region, extending *u* pixels left, right, above and below the coordinates of original selection is searched for the feature in the way described above. The user is given 3 seconds by default to center his or her head in the oval, but the amount of time can be changed in the Camera Mouse settings.

If the feature is still not found, then the user may not be centering his or her head correctly, and, in the third and final sub-phase, the region bounded by the dimensions of the oval is searched. If at this point the feature is still not found, then the user did not position their face in the oval, and the prompt reappears. The user is given another three seconds the center his or her head, and the process repeats with the search described for the second sub-phase.

If at any point, there is a manual mouse click on the image, which could happen if the user wants to switch which feature is being tracked, tracking is terminated, the oval graphic and prompt reappear, and then the user can select the new feature. The coordinates of the new selection are saved over the coordinates of the previous selection.

2.3 Movement Analysis

In order to improve the Camera Mouse experience for users, we studied the directions of movement that are most and least comfortable for users with disabilities. We considered eight possible directions of movement: up, down, left, right and diagonally up/right, down/right, up/left, and down/left. We developed an interface to use in testing which features twelve targets spaced evenly around the edge of a square interface, as depicted in Figure 3 left. The user is prompted to select each target in turn. To make the testing an enjoyable experience, we reveal a humorous graphic on the target area to be selected (picture of face in Figure 3 right). The user selects a target by clicking on it anywhere in the target area, which in the Camera Mouse system is accomplished by dwelling over a small area for a pre-specified dwell time

(e.g., 0.5 s). The order in which targets are revealed is predetermined so that each of the eight directions of movements are followed by the user (Figure 3 right).

The ideal path that the user's movements should follow is considered the straight-line path between the coordinates clicked on the previous target and the coordinates clicked on the current target. These ideal paths between targets correspond to the possible eight directions of movement. Over the duration of the test, the coordinates of the moving mouse pointer and the location of mouse clicks are recorded, so that the path taken by the mouse pointer can be analyzed in relation to the ideal path.

The ease of movement for each direction of movement, or for each path between targets, was measured in two ways by comparing the ideal path with the actual path taken. The first measure defines the actual path as the sum of lengths of the straight line paths between each of the successive recorded mouse locations along the path the user took. To compute the second measure of ease of movement, each point on the actual path for which a mouse coordinate was recorded is projected perpendicularly onto the ideal path (Figure 4) and the distance between the point on the actual path and the projected point on the ideal path. The second measure is then defined by the average of these distances.

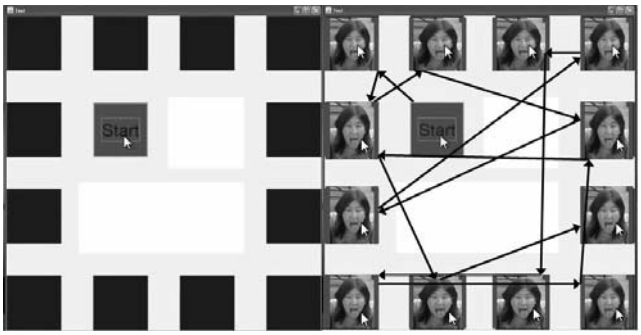

Fig. 3. The movement analysis interface. Left: The interface upon starting the test. Right: The trajectories the user follows during the test.

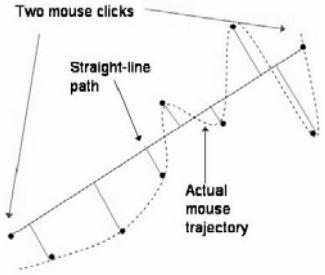

Fig. 4. Second measure of ease of movement. Mean distance from the ideal path (straight line) is calculated by averaging the shortest distances (brown lines) between each point (black disk) on the actual path (dashed line) and the ideal path.

3 Testing and Results

3.1 Recognition of Breakdown of and Recovery from HCI

The system was tested with thirteen users without disabilities and with two users with disabilities in separate testing sessions. One initially tested user was a spastic quadriplegic, non-speaking person in his mid-forties who suffers from cerebral palsy. The subject is a regular user of the Camera Mouse software. In a five-minute session in which the user played a game, Eagle Aliens, with the new Camera Mouse, the tracked feature was never lost, and the system never identified it as lost [3]. During this session, the user did not experience any spastic events which might have resulted in feature loss. We therefore designed the following experiment to simulate a common cause of feature loss.

After the system was initialized to track the tip of the user's nose, we repeatedly forced the system to track the wrong feature by waving a hand between the camera and the user, obstructing the camera's view of the feature. This user went through the obstruction and recovery trial 8 times. Out of the 8 trials, the correct feature was identified 62.5% (5/8) of the time in sub-phase 1, 25% (2/8) of the time in sub-phase 2, and 12.5% (1/8) of the time, the system selected a part of the eyebrow to track rather than the nose tip. However, within seconds, the system determined that the piece of eyebrow was not the correct feature and selected the nose tip in sub-phase 1.

We analyzed the reason that the system identifies a feature as lost. Each time the feature is identified as lost, the system records whether it was due to an ncc that was too low or due to RGB values that were too different. With this additional information, we conducted the same test involving deliberate feature occlusion. In tests consisting of 10 trials each with 13 users without disabilities, the correct feature was identified 76.1% (99/130) of the time in sub-phase 1, 20.8% (27/130) of the time in sub-phase 2, and 3.1% (4/130) of the time in sub-phase 3 (the feature was reselected correctly 100% of the time). The feature was identified as lost using the ncc 91.5% (119/130) of the time and because of color 8.5% (11/130) of the time.

The same test was conducted with a user with cerebral palsy, consisting of 44 trials. The correct feature was identified 65.9% (29/44) of the time in sub-phase 1, 13.6% (6/44) of the time in sub-phase 2, and 20.4% (9/44) of the time in sub-phase 3. Again, the feature was reselected correctly 100% of the time. The feature was identified as lost using the ncc 93.2% (41/44) of the time and because of color 6.8% (3/44) of the time.

3.2 Movement Analysis

The game interface described above was used to test six users, four with severe motion impairments due to cerebral palsy and two users without disabilities. Of the four subjects with disabilities, the test proved too cognitively challenging for three users, and thus no meaningful data could be retrieved from those testing sessions.

The fourth subject, the same user who tested the new Camera Mouse system, ran two sessions successfully. The data from these two sessions and from the sessions run by the two subjects without disabilities were used in the analysis.

The ease of movement for each direction of movement, or for each path between targets, was determined by the two measures defined above. On average, for users without disabilities, the length difference between shortest possible and actual path was 1,352 screen pixels, whereas for the subject with cerebral palsy, the average was 21,859 pixels. The mean distance for users without motion impairments was 13 pixels, and for the user with motion impairments it was 101 pixels. Each subject tested played the game twice, so that each was forced to move along every trajectory several times. The data from the individual paths recorded were averaged for each trajectory.

Evaluating the movement patterns of subjects using these two measures, vertical and horizontal movements clearly proved the most difficult and least accurate directions of movement for the user with disabilities by having the greatest length differences between shortest possible path and actual path and having the greatest mean distance from the shortest path. The directions of movement which consistently proved the most natural and comfortable for the user with disabilities, those which had the least length differences between shortest possible path and actual path and having the least mean distance from the shortest path, were diagonal, particularly the up/right and down/left directions. On average, for users without disabilities, the mean distance from the ideal path was 8 screen pixels for horizontal and vertical trajectories and 18 pixels for diagonal trajectories, and the difference in path length from the ideal path was 1196 pixels for horizontal and vertical trajectories and 1503 pixels for diagonal trajectories.

In contrast, the user with disabilities was least comfortable with vertical and horizontal movements, and most comfortable with diagonal movements. On average, for the user with disabilities, the mean distance from the ideal path was 124 pixels for horizontal and vertical trajectories and 83 pixels for diagonal trajectories, and the difference in path length from the ideal path was 22,971 pixels for horizontal and vertical trajectories and 18,522 pixels for diagonal trajectories.

4 Discussion and Conclusions

Testing the new Camera Mouse with users with and without disabilities proved successful. The most encouraging result is that the correct feature was identified 100% of the time, rarely with a few-second delay between loss and recovery. This shows that users who have severe motion impairments can in fact have an independent session of computer use. Testing showed that in most cases, the feature was identified as lost as a result of an ncc below threshold r_n, but the RGB color difference threshold d was still necessary for those cases where the system tracked a piece of background with a brightness pattern similar to the template.

The results also showed that, of the three sub-phases of search for the original feature, the feature was identified within in sub-phase 1 in most cases (about 75% of the time). The system very rarely needed to employ sub-phase 3 (about 3% of the

time). Since the user would only have to participate in the cooperative repositioning in about 25% of the instances of loss, he or she would rarely experience any interruption in tracking.

Because the search in sub-phase 1 is based on the user's recent movements, the amount of activity that the user exhibits can alter the effectiveness of the search. For example, the second subject with disabilities that we tested had just taken medication which made him drowsy, so during the testing session, he exhibited minimal movement. As a result, the proportion of trials in which the search entered sub-phases 2 and 3 was much higher with him than with the first user or with the users without disabilities. For users who exhibit spastic movements, the opposite is likely to be true, since the system would be able to determine the facial boundaries with high accuracy if the user just made a sudden movement.

The Camera Mouse system accepts a range of cameras and image dimensions. The size of the template and tracked feature image (x by x pixels), the width w of the columns over which brightness is summed, as well as the distance y and length u which delineate the areas in which to search were optimized in relation to an image input size of 360 by 240 pixels. These dimensions were selected because they are the image dimensions of the web camera recommended on the Camera Mouse website. For example, with too small a value for x, the pattern becomes less distinct, and the system is less able to accurately distinguish between the tracked feature and similar patterns elsewhere in the image. But, with too large a value for x, the software operates too slowly. The ideal value for x we found was 21, so the sub-image represents 0.5% of the image. For the other variables, the optimal values were found to be 50 image frames for t, 10 pixels for w, 50 pixels for y, 5 pixels for s, and 30 pixels for u.

The threshold values were also optimized so that they did not identify a part of the image that was not the original feature as the original feature but which was low enough that it allowed for variation in lighting and angle of the feature as he user moves. The threshold for the ncc, r_n, is most accurate at 0.75, and the difference threshold for the normalized RGB values, d is best at 0.1.

The most significant obstacle we encountered was the speed of the system. The more comprehensive the search becomes, the slower it is. We therefore had to compromise in the depth of the search in order to make sure that the Camera Mouse interface operates in real time. As the average user's computer becomes faster, we will be able to release newer versions of the system with a more comprehensive search procedure.

In the future, we plan to try to improve the ease and independence with which Camera Mouse users access computers. One direction we will investigate is for the system to find the user's face and initialize a feature to track on startup, eliminating the need for manual selection.

Currently, most software, including most software compatible with the Camera Mouse, has menus, scroll bars, and other interactive features which require the user to move the mouse pointer in horizontal and vertical directions. This is in keeping with our result that these directions of movement are the most natural for users without disabilities. Since our research has shown, in contrast, that horizontal and vertical trajectories are not necessarily the most comfortable directions of movement for users with disabilities, future software designed to work in conjunction with a mouse-

replacement assistive technology such as the Camera Mouse will consider relocating these interactive features along other axes. In particular, axes in diagonal directions should be considered if they turn out to be the more comfortable directions of movement for users with disabilities. Designing software this way will allow users with disabilities to have a less strenuous and time-consuming computer session.

5 Acknowledgements

The authors thank the subjects for their efforts in testing the new Camera Mouse and participating in our movement analysis experiment. Caitlin Connor was supported as a Clare Booth Luce research fellow, and Emily Yu was funded by the CRA-W Distributed Mentor Program. The paper is based upon work supported by the National Science Foundation under Grant IIS-0713229 and the Boston University Undergraduate Research Opportunities Program. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

6 References

1. World Health Organization, http://www.who.int/en/
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, March 2002, 10(1): 1-10.
3. Camera Mouse, http://www.cameramouse.org/
4. M. Betke. "Camera-based Interfaces and Assistive Software for People with Severe Motion Impairments," Proceedings of the 3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAmI 08), Patras, Greece. 21st-22nd of July 2008. J.C. Augusto, D. Shapiro and H. Aghajan (Eds.).
5. T. Castelli, M. Betke, and C. Neidle. "Facial feature tracking and occlusion recovery in American Sign Language," In A. Fred and A. Lourenço, editors, Pattern Recognition in Information Systems: Proceedings of the 6th International Workshop on Pattern Recogntion in Information Systems (PRIS 2006), Paphos, Cyprus, May 2006. INSTICC Press, pages 81-90.
6. Strom, Jacob. "Reinitialization of a Model-Based Coder," Proceedings of the EuroImage International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging (ICAV3D 2001), Mykonos, Greece, May 2001. 4 pp.
7. A. Yilmaz, O.Javed, and M. Shah. “Object tracking: A survey,” ACM Computing Surveys, 38(4):1-45.
8. J. Chen and B. Tiddeman. “A Robust Facial Feature Tracking System,” IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS), Como, Italy, September 2005, pages 445-449.
9. Open Computer Vision Library, http://sourceforge.net/projects/opencvlibrary/