Human–Computer Interaction — INTERACT'99
Angela Sasse and Chris Johnson (Editors)
Published by IOS Press, © IFIP TC.13, 1999

An Imprecise Mouse Gesture for the Fast Activation of Controls

Martin S Dulberg, Robert St Amant & Luke S Zettlemoyer

Department of Computer Science, North Carolina State University,
EGRC-CSC Box 7534, Raleigh, NC 27695–7534, USA.
{msdulber,stamant,lszettle}@eos.ncsu.edu

Abstract: A common task in graphical user interfaces is selecting or activating a single control from a small group of plausible candidates. This task ordinarily requires the same precise movements as any other; if we could reduce the precision needed, however, we might also reduce the target acquisition and activation time. This paper describes the flick gesture, designed for this purpose. Our experimental results demonstrate that the gesture performs well with regard to speed, accuracy, and variability, compared to conventional gestures in a laboratory setting. We also describe a testbed we have implemented in Microsoft Windows that lets us explore and evaluate the use of such gestures in settings with more ecological validity.

Keywords: direct manipulation, marking menus, intelligent user interfaces, interaction techniques.

1 Introduction

Modern graphical user interfaces (GUIs) rely heavily on precise movements of the mouse. Immediately after Microsoft Word 6.0 starts up on the Macintosh, for example, almost eighty distinct visible targets for the mouse pointer appear. These include menu titles, buttons, palettes of tools, text boxes, pop-up menu indicators, scroll bar gadgets, window decorations, and mouseable documentation. Without our ability to position the pointer precisely, we would find manipulation of such densely clustered objects extremely difficult, no matter how good the overall organization.

Such precision, however, should not always be necessary. In many situations, only one or two controls in the interface have any relevance for an interaction. For example, when the user selects the Print command from a menu, a dialog box appears that allows the modification of settings and the initiation of the printing action. The OK button must be activated at some point to successfully print — but clicking on the button requires the same effort as changing any one of the less frequently used controls. We propose that in such a situation, when a specific control is inevitably part of a given interactive task, the system can arrange for an imprecise activation gesture, for faster, more streamlined operation.

We have developed a gesture, which we call a *flick*, or flying click, for this purpose. To carry out a flick gesture, the left mouse button is briefly pressed and held while the mouse is quickly moved a short distance. Releasing the mouse button completes the gesture. (Some users have described this gesture as *dragging toward a target*.)

The flick gesture has a few attractive features. The most obvious advantage is that precise target acquisition, the zeroing-in phase of conventional Fitts' Law tasks, is bypassed. The gesture can thus in theory be faster than clicking directly on a control. Once a control has been activated through a flick, the mouse pointer also remains very close to its previous location, minimizing the repositioning effort needed to resume a previous activity. Flicking is directional, and can be applied, within limits, to any visible target in the interface. Though not intended to replace any existing gestures, flicking is also a useful alternative to keyboard shortcuts, such as pressing the Enter key to indicate confirmation. It does not require the user's hand to leave the mouse. Unlike keystrokes, the flick gesture also has the potential to be applied dynamically to different targets, rather than statically associated with specific commands.

This paper has two general themes. One theme concerns the feasibility of an imprecise gesture: how well does it work? We describe two experiments that explore the potential uses and limitations of the flick gesture. Our first experiment assumed an unambiguous goal of activating a single control, with no other legal actions possible. We compared the

flick gesture to two alternatives: the conventional selection of a control with the mouse, and pressing the Enter key. Encouraging results prompted a follow-on experiment, in which we tested whether the system could distinguish between flicks to alternative targets and could differentiate the flicking gesture from other types of high speed mouse movement. In both cases, our expectations were largely met.

The other theme concerns evaluation: can we confidently apply our laboratory results to the real world? We briefly discuss the implications for implementing the flick gesture in conventional user interface, and describe a general-purpose testbed for incorporating novel gestures into an existing interface and testing their impact.

2 Related Work

Practically speaking, the flick gesture is little more than a shortcut for entering a command. A comparable shortcut, common in conventional GUIs, is the keyboard accelerator. A keyboard accelerator allows the user to execute a command by recalling and executing a particular sequence of keystrokes that are mapped to a particular function. Accelerators suffer from two limitations: they require the user to retain and recall information related to specific commands, and they require the user to drop out of a direct manipulation interaction style in favour of a command/response conversational style. Novice users can find large vocabularies of keystroke accelerators difficult to acquire (Shneiderman, 1997). Further, even when keystrokes have been completely internalized, so that they can be recalled without conscious effort, the cognitive burden of switching between the direct and indirect style of interaction remains (Norman, 1986). The alternative of pop-up menus presents similar difficulties (Norman, 1991). The flick gesture is not subject to these limitations. It is easily learned, and because flick targets are generally visible, it remains within the direct manipulation framework.

Though our presentation of the flick gesture from a precision perspective is novel, the action itself is not. The WebBook (Card et al., 1996) implemented a gesture called a ruffle, similar to the flick, for rifling through a set of web pages like a book. The use of marking menus, a more extensively examined technique, is also very similar to flicking. Pie menus are circular menus that pop up in the interface directly where the user has either clicked or moused down. The user then moves to the desired wedge of the pie menu and either clicks again or mouses up to activate an item. A drawback of pie menus, despite their speed and accuracy, is their inefficient use of screen space (Callahan et al., 1998; Hopkins et al., 1988). With marking menus, in contrast, an expert user can quickly perform the same dragging gesture without waiting for the appearance of the pie menu (Kurtenbach & Buxton, 1991). Our work differs from these earlier efforts in the consideration of flicking in a novel context and by a more detailed evaluation in some regards, of the gesture in use (Kurtenbach & Buxton, 1993).

3 Experiments

We conducted two formal experiments to explore the characteristics of the flick gesture, both on a personal computer running Windows 95, using a 17″ monitor in SVGA (1024 × 768) mode. Participants used a Dell two-button mouse and standard keyboard. Only right-handed participants were used for the experiment. All participants were un-compensated undergraduate or graduate Computer Science students. The experimental software logged all mouse clicks, mouse up, and mouse down events with the corresponding time and position of the mouse cursor during the event. We also conducted a number of informal observations, as a part of the development described in Section 4.

3.1 Experiment 1

Eighteen participants were presented with a screen (Figure 1) that contained two command buttons, each 20 pixels square, one located in the centre of the screen and one randomly positioned between 20 and 300 pixels away from the centre button. This 'outer' button was repositioned after each trial using polar coordinates to uniformly distribute its position in terms of distance and angle. The experiment consisted of three different tasks, presented to participants in blocks of 50 trials.

Figure 1: Screen shot of first experiment.

- Click: The participant was instructed to click on the outer button, then click on the centre button as quickly as possible.

- Enter: The participant was instructed to click on the outer button, then press Enter with the same hand, as quickly as possible to activate the centre button. The Enter gesture was used to determine whether allowing a control to be activated by a shortcut key was as efficient as the flick gesture.

- Flick: The participant was instructed to click on the outer button, then perform a flick in the general direction of the centre button.

The order of presentation was varied in all possible permutations across the participants. The participants were given three trials of training with on screen instructions for each condition to practice before the block began. No data was collected during the training. A within-subjects design was used for data analysis.

3.2 Results

Figure 2 shows a comparison of mean duration (task completion time) over the three conditions. Each point in the plot represents the value for one of the 18 participants in each of the conditions. To conserve space we have included the data for all of the conditions on one plot. We used two-tailed t-tests for all the pairwise comparisons discussed here*. The bars across the centre of Figures 2 & 3 represent the mean across all three conditions. There was no order effect observed between the three conditions.

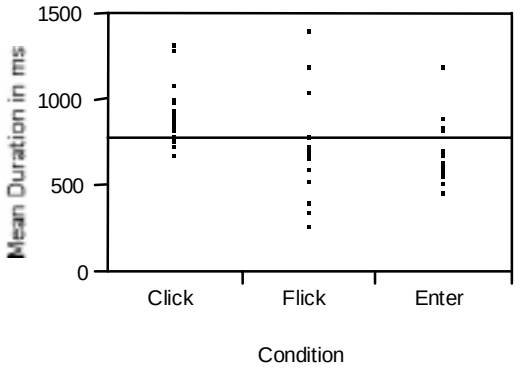

Figure 2: Mean duration by condition.

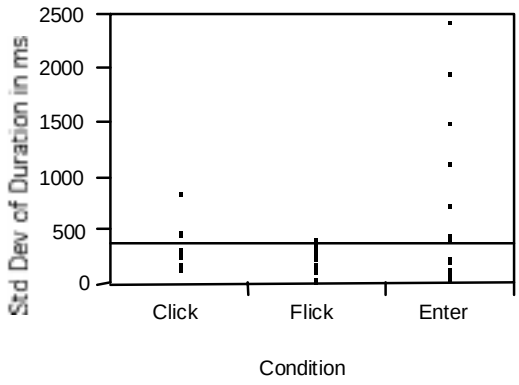

Figure 3: Standard deviation of duration by condition.

Flicking is significantly faster than clicking ($t(34) = 3.053, p = 0.0044$), taking 692ms on average compared to 940ms. The Flick condition is not significantly faster than the Enter condition ($t(34) = 0.294, p = 0.7708$), at 716ms. Note that the spread in the data for each condition in Figure 1, especially for the Flick condition, reflects the expected variability between participants. The lower values, for example, are associated with participants who were on average faster than the others.

Figure 3 shows a comparison of the standard deviation of duration over the three conditions. That is, here each data point measures the variability in duration for a *single* participant (rather than between participants, as in Figure 2.) Flicking shows less variability than clicking, at 227ms vs. 314ms, but the difference is not significant ($t(34) = 1.497, p = 0.1437$). Flicking is however significantly less variable than pressing Enter, at 793ms ($t(34) = -2.112, p = 0.0421$).

We are aware of no work that explains in detail the differences between the Flick and Click conditions at the motor level. Research into the nature of targeting movements does however give insight into our results (MacKenzie, 1992). Such movements do not exactly follow the smooth log curve suggested by the simple form of Fitts' law. Rather, an initial imprecise movement is followed by a few more precise targeting movements that end on the target (Meyer, 1988). This first movement or primary sub-movement may correspond to the flick gesture. The variability in the duration of the flick gesture might be accounted for by neuro-motor noise.

For similar reasons, we found less variability in the Flick condition than in the Enter condition.

*Note that we are not comparing the three conditions against one another in a single analysis of variance. The relative speed of Click vs. Enter is irrelevant and its inclusion reduces the power of the test; we are interested in how the Flick condition compares with the Click and the Enter conditions independently.

Perhaps unexpectedly, however, flicking was no faster than moving the hand to the Enter key. One might intuitively expect that because the distance that the mouse moves in the Flick condition is so much smaller than the distance to the Enter key, flicking should be much faster than entering. We suspect this to be an artifact of our data collection. Moving a hand between the mouse and the Enter key actually requires two gross targeting movements, one to reach the key and the other to return to the mouse. While the participants were not instructed to complete the blocks as quickly as possible, but merely the trials, it was noticed that the average time to complete a block of flick trials was shorter than for the click blocks. In retrospect, we should have measured the entire end-to-end task time. We would then expect to find the difference between flicking and entering to be much greater.

To summarize, Experiment 1 was a pilot study into the potential effectiveness of the flick gesture. Though the number of trials per participant was relatively small, the results were unambiguous. The flick gesture is 26% faster than the click gesture, with equal variability. It is also at least as fast as the enter gesture, and significantly less variable.

3.3 Experiment 2

Twelve participants were presented with a screen (Figure 4) that had a red circle 20 pixels wide in the centre. They were instructed to move the mouse cursor to the centre circle at which time a green circle, also 20 pixel in diameter would appear somewhere at a fixed distance (400 pixels) from the centre. The participants were told to flick by initiating the mouse down event inside the red circle and complete it by moving the cursor towards the green circle while releasing the mouse button.

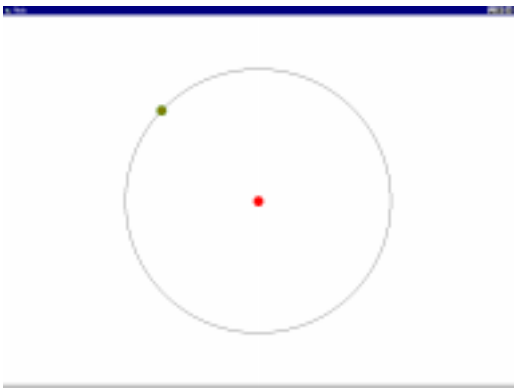

Figure 4: Screen shot of Experiment 2.

Participants were told that they could move as far or as little as they desired towards the circle and that the accuracy and speed with which they performed the gesture was being measured. Once they understood these instructions in both written and verbal form, the gesture was demonstrated for them three times by the same experimenter and they were asked to practice the gesture seven more times. All participants were then asked if they understood the directions or wished to practice any additional trials. All of the participants indicated that they were ready to begin. The participants were then presented with three blocks of 100 trials of the flick task with a one minute rest interval between blocks. This phase recorded information about accuracy and speed.†

After completing the three blocks, participants were then asked to complete two blocks of 100 iterations of a task where they were presented with blue circles in an identical fashion and asked to click on the centre blue circle and then click on the outer blue circle. This phase gathered information about the nature of a click so that we could determine if we could easily distinguish between flicks and clicks.

3.4 Results

The median distance travelled during the flick gesture was 48.4 pixels. Figure 5 shows the distribution of flick distances. The median duration of the flick gesture was 284ms. Figure 6 shows the distribution of flick duration. The difference in duration between the two experiments is explained by the fact that the users were not required to click on the centre circle in Experiment 2; duration was measured from mouse down to mouse up events. In the first experiment we included the interval after the centre button was clicked and before the flick gesture was initiated.

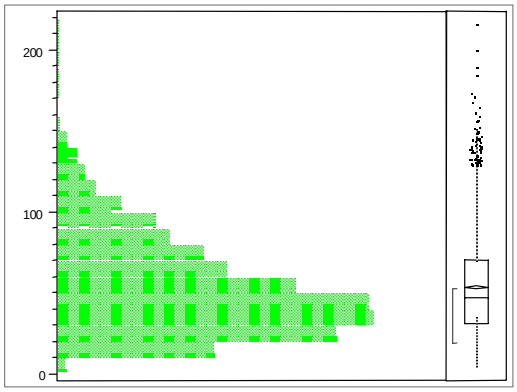

Figure 5: Distribution of flick distance in pixels.

†One participant's data was discarded for procedural reasons; the participant told the experimenter, "I wanted to see how I would be penalized if I flicked away from the circle".

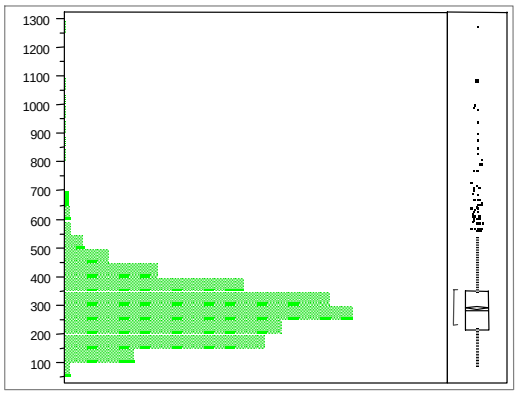

Figure 6: Distribution of flick duration in ms.

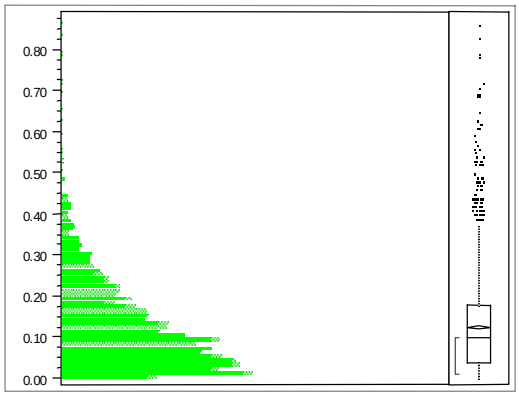

Figure 7: Distribution of flick accuracy in radians.

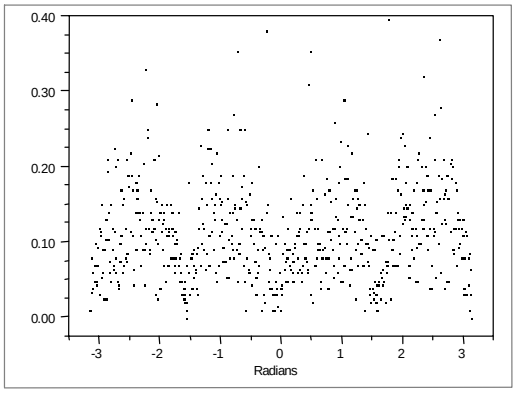

Figure 8: Flick accuracy in radians as a function of the location about a circle.

The accuracy of the flick gesture was computed by measuring the angle between the vector formed by the centres of the circles and the point at which the mouse was released at the end of the flick gesture. The median difference between the optimal vector, which would point in the same direction as the vector between the circles and the actual flick vector, was 0.10 radians or 5.73 degrees. Figure 7 shows the distribution of flick accuracy. Note that 90% of the flicks lie within .27 radians (15.48 degrees) and 97.5% of the flicks lie within. 41 radians (23.5 degrees). Since the differences were calculated as absolute values they should be doubled to allow for errors in either a positive or negative direction. Finally, the magnitude of errors was not distributed uniformly over the range of $-\infty$ to ∞ radians (-180 to 180 degrees.) The variance in errors for non-vertical and non-horizontal motion is also noticeably greater. We suspect that this pattern is due to inherent behaviour of the human motor system based on Jagacinski and Monk's work (Jagacinski & Monk, 1985). Figure 8 shows that the differences between the optimal vector and the actual vector were found to be smaller in either the horizontal or vertical direction and larger along the diagonals.

It is possible that a user might flick over an area that will actually accept a click. For this reason it is important to be able to distinguish between the two gestures. Based upon the data we collected in the last 2 blocks (a total of 400 clicks per participant) we found that it was relatively easy to disambiguate flicks from clicks by looking at the distance travelled by the mouse between the mouse down and mouse up events. We observe that 90% of the clicks made by the user last no longer than 157ms and cause the mouse pointer to move no more than 3 pixels. For flicking, 90% of the cases last longer than 163ms and are associated with mouse movements of more than 21 pixels. We can set one user-modifiable threshold for flicking based on the distance between a mouse down/mouse up combination. One possible source of bias is the possibility that the distance of the flick gesture might have been skewed by the requirement that we placed on the participants to start the flick inside the centre circle and finish outside. This will not increase the difficulty of distinguishing flicks from clicks, though, since more than 75% of the flicks were longer than the diameter of the centre circle.

3.5 Summary

We have demonstrated that where precision is not required the flick gesture is 26% faster than the conventional click gesture (692ms vs. 940ms) on a button. While the flick gesture is comparable in duration to the enter gesture, the flick gesture is superior to the enter gesture in terms of the time required for the user to 'recycle' or get ready for the

next task.

Results for the second experiment indicate that about eight targets can be presented to the user if they are spaced at 45 degree intervals from the user's current mouse position. Our findings on flick accuracy suggest that the most frequently used targets should be placed on the midpoints of the left and right edges of the interface, followed by the midpoints of the top and bottom edges. Less frequently used controls should be placed along the diagonals (in the corners). This would result in the best frequency of selection to accuracy ratio. This is a very different strategy than what is currently taken in the placement of items in a linear menu where items are generally placed from top to bottom in reverse order of frequency to minimize the serial order effect of selection (Norman, 1991).

Users found the flick gesture potentially useful. There was some difficulty training participants to perform the flick without giving explicit examples, since it is a relatively novel gesture. Most participants in the first experiment did not distinguish between the instructions 'click then drag' and 'drag'. In the second experiment the users did not suffer from confusion because the click was not required. Surprisingly over 3300 trials, the 11 participants made just 4 errors while flicking. An error was considered to be any mouse events other than a mouse down inside the red (centre) circle followed by a mouse up outside of the red circle. This indicates that the flick gesture is an easily learned skill that maps well to previous knowledge of graphical user interfaces, and suggests that there should be little difficulty, from the user's perspective, in adding the flick to the conventional set of gestures.

4 Real-world Integration

Graphical user interfaces conventionally use an object-oriented, event-driven model in which a dispatcher directs events to objects in order to be processed. Objects in the interface are treated as hierarchical groups of geometrical containers: windows contain buttons, for example, because the button occupies a region in the window's interior (Olsen, 1998). Gestures are dispatched in either top-down or bottom up fashion to the appropriate container object. The flick gesture breaks this convention, however; the gesture may occur over an object that can in principle interpret the event but is not the intended target (the target may be a control in an entirely unrelated application.)

We can address this problem in two ways. First, we ensure that the system can reliably distinguish between flicking and simple clicking, dragging for selection, mousing down on a button but then cancelling by moving away before a mouse up, and so forth. The solution we describe above is a reasonable first step. Second, we dissociate the processing of the flick gesture from a given application, and give the responsibility to the operating system.

Our implementation is based on VisMap (Zettlemoyer et al., 1999; Zettlemoyer & St. Amant, to appear), a visual manipulation system that allows an application to control the Windows user interface at the event level. Flick is implemented as a VisMap controller that monitors the event queue for candidate sequences of events. Sequences are disambiguated from clicks and drag-and-drop gestures by two user-setable parameters. The radius parameter controls the minimum number of pixels that must be travelled between the mouse down and mouse up to trigger a flick, allowing the controller to differentiate between clicks and flicks. The velocity parameter allows the controller to disambiguate between flick and drag-drop gestures. The velocity is equal to the number of pixels that must be travelled in the last 110ms of the gesture. Our informal findings suggest that the end of a drag-drop gesture requires a more precise targeting effort, hence a slower velocity at the end of the gesture.

Figure 9: A flick gesture being performed with the VisMap flick controller (graphical annotations added).

An informal usability test was conducted on the VisMap controller. Six participants were introduced to the flick gesture and allowed to become familiar with the gesture over a five minute period. During this training period they were asked to flick towards various icons to change the focus of the icons on the desktop. Figure 9 shows the arrangement of the icons on the desktop. The participants were then asked to perform a variety of tasks such as flicking to give an icon focus or flicking a file on the desktop to a

specific target as shown in Figure 9. None of the participants had any trouble executing the tasks. Five of the six participants also indicated that they found the flick gesture to be useful and would use it if it were available in the operating system.

5 Discussion

Our current work involves testing the flick gesture in different contexts of varying artificiality. The flick controller, for example, allows the experimenter to specify one of three actions when a flick is received:

- Interpolate To: where the mouse cursor is slowly moved in a straight line to the target of the Flick.
- Click On: where the mouse cursor jumps to the target of the Flick.
- No Effect: where the target is activated by the Flick but the cursor remains under the user's control.

Currently the experimenter is required to enter the coordinates of the flick enabled controls in the flick controller. We allow a maximum of eight flick enabled areas. With these factors under our control, we can test a wide variety of possible scenarios.

As our initial observations of users suggests, our implementation will allow us to test the integration of flick into real work environments. The simplest case is the most obvious: if the interface presents a modal dialog window containing information for the user to confirm, a flick gesture can substitute for clicking OK or pressing the Enter key. An extension of this approach could establish a convention that flicking to the lower right portion of the screen always signifies a confirmation, whenever the context is appropriate. A flick to the lower left signifies a cancellation. These are arbitrary choices that will require practical experience to validate.

More generally, we propose that any row or column of icons alongside the general workspace of an application might provide targets for a flick gesture. This requires some analysis into geometrical relationships between the selectable controls and the majority of user activities. Some of the necessary design guidelines are already available, however, in the results we have presented. For an accuracy of 90% in targeting with the flick gesture, for example, we must position the controls at least. 54 radians (30 degrees) apart, as measured from the origin of the gesture.

For conventional interfaces, adoption of the flick gesture could lead to some interesting possibilities for more efficient and less distracting interactions by the user. We envision an alternative to the message box that would provide the user with a faster, less intrusive alternative to the current style. Message boxes often interrupt the users work-flow and require them to either accept, cancel, or request help from the system at some critical juncture of a task. The standard commands, OK, CANCEL and HELP, could be mapped to flick gestures to the left, right and bottom respectively. As long as the standard commands are presented in a clear and consistent manner the user should have no trouble dispatching the message in a clear and efficient manner. Novice users could be presented with a set of tool tips presented in the correct orientation to provide them with an appropriate mapping of flick direction to command location. In addition to potentially increasing accuracy by giving the user a close target to aim at when performing the flick, displaying tool tips in this fashion would mimic some of the advantages of pie menus or marking menus. The message could be displayed as a transparent or semi-transparent window so that the user doesn't become disoriented in the workspace.

The flick could also be useful in an intelligent user interface, one that anticipates user selection of specific commands. In most such arrangements, the system takes one of two paths: it generates actions autonomously, providing choices of results to the user (e.g. Letizia (Lieberman, 1995)), or it provides the user with a special command that carries out whatever action the interface predicts is appropriate (e.g. Eager (Cypher, 1993)). With the flick we have another choice that lies between these: the interface can highlight a specific control and allow the user to flick in an appropriate direction. If several commands have a reasonable probability of being the most appropriate choice, the interface can highlight all of them, provided they are sufficiently separated spatially, for the user's gesture. The flick gesture offers a partial solution to at least one of the usability problems for anticipatory interfaces. For example, if the occurrence of a selection, cut command, and mouse down action in a word processor is highly predictive of a paste command immediately to follow, then the standard alternatives for an intelligent user interface are to execute the paste autonomously or tell the user that a paste may be appropriate. Flicking supports a better possibility: the interface can pop up a small window containing the text 'Paste', near the Edit menu for the user to flick towards.

Acknowledgements

The authors wish to thank James Lester, David McAllister and Eric Wiebe for their invaluable suggestions and inspirations. We are also grateful to our participants for graciously giving of their time.

References

Callahan, J., Hopkins, D., Weiser, M. & Shneiderman, B. (1998), An Empirical Comparison of Pie vs. Linear Menus, *in* ***EDITOR*** (ed.), *Proceedings of CHI'98: Human Factors in Computing Systems*, ACM Press, pp.95–100.

Card, S. K., Robertson, G. G. & York, W. (1996), The WebBook and the Web Forager: An Information Workspace for the World Wide Web, *in* G. van der Veer & B. Nardi (eds.), *Proceedings of CHI'96: Human Factors in Computing Systems*, ACM Press, pp.111–7.

Cypher, A. (ed.) (1993), *Watch What I Do: Programming by Demonstration*, MIT Press.

Hopkins, D., Callahan, J. & Weiser, M. (1988), Pies: Implementation, Evaluation and Application of Circular Menus, HCIL Technical Report ***NUMBER***, University of Maryland.

Jagacinski, R. J. & Monk, D. L. (1985), "Fitts' Law in Two Dimensions with Hand and Head Movements", *Journal of Motor Behavior* **17**(1), 77–95.

Kurtenbach, G. & Buxton, W. (1991), Issues in Combining Marking and Direct Manipulation Techniques, *in* ***EDITOR*** (ed.), *Proceedings of the ACM Symposium on User Interface Software and Technology, UIST'91*, ACM Press, pp.137–44.

Kurtenbach, G. & Buxton, W. (1993), The Limits of Expert Performance Using Hierarchic Marking Menus, *in* S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel & T. White (eds.), *Proceedings of INTERCHI'93*, ACM Press/IOS Press, pp.482–7.

Lieberman, L. (1995), Letizia: An Agent That Assists Web Browsing, *in* ***EDITOR*** (ed.), *Proceedings of the International Joint Conference on Artificial Intelligence*, ***PUBLISHER***, p.***PAGES***.

MacKenzie, I. A. (1992), Movement Time Prediction in Human Computer Interfaces, *in* ***EDITOR*** (ed.), *Proceedings of Graphics Interface '92*, ***PUBLISHER***, pp.483–93.

Meyer, B. (1988), *Object Oriented Software Construction*, Prentice–Hall.

Norman, D. A. (1986), Cognitive Engineering, *in* D. A. Norman & S. W. Draper (eds.), *User Centered Systems Design: New Perspectives on Human–Computer Interaction*, Lawrence Erlbaum Associates, pp.31–62.

Norman, K. (1991), *The Psychology of Menu Selection*, Ablex.

Olsen, D. (1998), *Developing User Interfaces*, Morgan-Kaufmann.

Shneiderman, B. (1997), *Designing the User Interface: Strategies for Effective Human–Computer Interaction*, third edition, Addison–Wesley.

Zettlemoyer, L. & St. Amant, R. (to appear), A Visual Medium for Programmatic Control of Interactive Applications, *in* ***EDITOR*** (ed.), *Proceedings of CHI'99: Human Factors in Computing Systems*, ACM Press.

Zettlemoyer, L., St. Amant, R. & Dulberg, M. (1999), IBOTS: Agent Control Through the User Interface, *in* ***EDITOR*** (ed.), *Proceedings of the Fifth International Conference on Intelligent User Interfaces*, ***PUBLISHER***, pp.31–7.

Human–Computer Interaction — INTERACT'99
Angela Sasse and Chris Johnson (Editors)
Published by IOS Press, © IFIP TC.13, 1999

Author Index

Human–Computer Interaction — INTERACT'99
Angela Sasse and Chris Johnson (Editors)
Published by IOS Press, © IFIP TC.13, 1999

Keyword Index