The PadMouse: Facilitating Selection and Spatial Positioning for the Non-Dominant Hand

Ravin Balakrishnan[1,2] **and Pranay Patel**[2]

[1]Dept. of Computer Science
University of Toronto
Toronto, Ontario
Canada M5S 3G4

[2]Alias|wavefront
210 King Street East
Toronto, Ontario
Canada M5A 1J7

ravin@dgp.toronto.edu{ravin | ppatel}@aw.sgi.com

ABSTRACT

A new input device called the PadMouse is described and evaluated. The PadMouse consists of a two degree-of-freedom touchpad mounted on a regular mouse base. Like the regular mouse, the PadMouse allows for spatial positioning tasks to be performed by moving the device on a planar surface. In addition, when coupled with an interaction technique we call Marking Keys, users can use the touchpad to activate modifiers and commands. An experiment shows that up to 32 modifiers/commands can be quickly and accurately activated using this technique, making it a viable device for the non-dominant hand in two-handed user interfaces. Other uses for the PadMouse and design alternatives are also discussed.

Keywords

Input devices, marking-menus, bimanual input, touchpad, mouse, interaction techniques, gestures, hot-keys, toolglass.

INTRODUCTION

Several user interface researchers over the past decade, having recognized that in the physical world people often use both hands to cooperatively perform many tasks, have explored the possibility of using both hands simultaneously in the computer interface. In an early study, Buxton and Myers [3] showed that in a compound task, a one-handed interface (i.e. the status-quo) was inferior to a two-handed interface which split the compound task into subtasks that could be performed in parallel by both hands. Kabbash, Buxton, and Sellen [10] came to a similar conclusion, however, they also showed that two hands could be worse than one if an inappropriate interaction technique is employed, particularly when cognitive load is increased.

Building partly on this empirical work, Bier et al. [1, 2] developed the click-through Toolglass and Magic Lenses interface which utilized both hands for its operation. More recently, Kurtenbach et al. [13] described an interface called "T3" which effectively integrated the Toolglass and Magic Lenses concepts with other two-handed techniques for performing direct manipulation operations in a 2D drawing program. Zeleznik, Forsberg, and Strauss [15] incorporated and extended some of these techniques into their "Sketch" 3D modeling system, demonstrating the use of two cursors controlled by two hands to enhance 3D interaction.

While the existing body of research has investigated a variety of two-handed interaction techniques, the issue of what constitutes an appropriate input device for the non-dominant hand remains unexplored. Furthermore, these systems have limited the non-dominant hand to coarse positioning tasks, relying on the dominant hand for almost everything else including selection of tools, commands, and modes.

In this paper, we first explore the various tasks that could (or should) be performed using the non-dominant hand within a two-handed user-interface. We then describe a new device, the PadMouse (Figure 1), coupled with an interaction technique called Marking Keys. Together, these enhance the role of the non-dominant hand by allowing it to activate modifiers and commands in addition to performing spatial positioning. Finally, we present the results of an experiment to investigate the performance of Marking Keys on the PadMouse.

Published in Proceedings of 1998 ACM Conference on Human Factors in Computing Systems (CHI'98), pp 9-16.

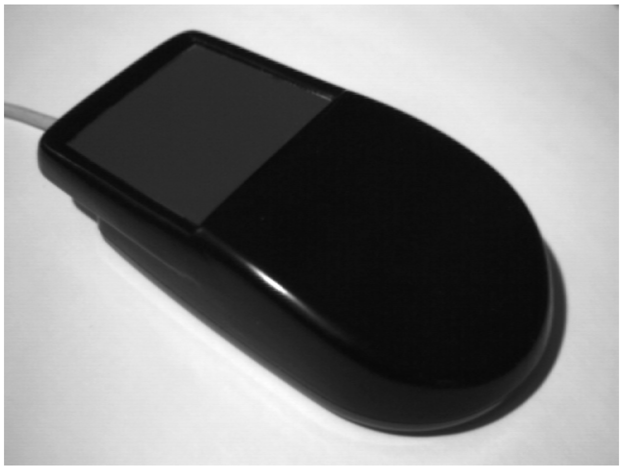
Figure 1. The PadMouse

TWO-HANDED INTERACTION

Much recent work in two-handed user interfaces [1, 2, 5, 7, 10, 13, 15] has been guided by the theoretical work of Guiard [6]. In his Kinematic Chain model of skilled bimanual action, the two hands are thought to be two abstract motors assembled in a serial linkage, thus forming a cooperative kinematic chain. Three general principles emerge from this model:

1. *Dominant-to-Non-Dominant Spatial Reference:* The non-dominant hand sets the frame of reference relative to which the dominant hand performs its motions.

2. *Asymmetric Scales of Motion:* The two hands operate in asymmetric spatial-temporal scales of motion. For instance, when writing on a piece of paper, the motion of the non-dominant hand controlling the position of the paper is of lower temporal and spatial frequency than the writing movements of the dominant hand which nonetheless depends on the non-dominant hand's movement for spatial reference.

3. *Precedence of the Non-Dominant Hand:* Contribution of the non-dominant hand to a cooperative bimanual task starts earlier than the dominant hand. In the handwriting example, the dominant hand starts writing *after* the paper has been oriented and positioned by the non-dominant hand.

The two-handed interfaces developed to date have by and large adhered to these principles, although perhaps not exploiting them to maximum advantage. In the Toolglass and Magic Lenses interface [1, 2] the non-dominant hand controls the spatial position of the Toolglass sheet, setting up a context for the dominant hand which performs precise positioning and drawing tasks in their 2D drawing application. In their "T3" concept application, Kurtenbach et al. [13] use the non-dominant hand for a greater variety of tasks: positioning and orienting the artwork, positioning a Toolglass, and cooperating with the dominant hand to scale, orient, and position graphical objects. Zeleznik et al. [15] rely on the non-dominant hand cooperating with the dominant hand to translate and rotate 3D objects, control the virtual camera, and perform a variety of other editing tasks.

I've got two hands, but lost my hot-keys!

A characteristic feature of today's ubiquitous WIMP (windows, icons, menus, and pointer) user interface is the use of the non-dominant hand to activate commands and modifiers using the keyboard (sometimes referred to as 'hot-keys' or 'keyboard shortcuts') while the dominant hand operates the mouse. We use the term "commands" to mean operations that are performed once when a key or combination of keys is pressed (e.g., Control-C for Copy, Control-V for Paste in many applications); and "modifiers" to mean operations that put the application into a particular mode only for the duration of the keypress (e.g., on a Mac, holding down the Shift key allows for multiple selections to be made with the pointer. Releasing the key returns the system to its default single selection mode).

In high-end 2D and 3D graphics applications, there are in the order of five hundred available commands and modifiers: the most frequently used ones are usually rapidly accessible via hot-keys. Ironically, the speed and functionality of this albeit limited form of two-handed interaction has not been incorporated in the more recent two-handed interfaces discussed in the previous sections. For example, the Toolglass and Magic lenses interface [1, 2] uses the non-dominant hand only to control the spatial position of the Toolglass sheet via a trackball or touchpad, while the dominant hand selected tools from this sheet. Also, the *contents* of the sheet were selected from a master sheet using the dominant hand. Similarly, Kurtenbach et al.'s "T3" application [13] relied on the dominant hand to select the contents of the Toolglass via a Marking Menu [11]. Thus, the dominant hand is required to serially select the tool and then perform the operation with the tool. The non-dominant hand merely keeps the tools within easy reach of the dominant hand. We believe that a system in which the non-dominant hand selects the tool or mode which is then used by the dominant hand would be more in line with Guiard's Kinematic Chain model [6] where the actions of the non-dominant hand naturally precedes and sets the stage for the actions of the dominant hand. One way to achieve this while retaining the favourable aspects of the Toolglass and T3 style two-handed interfaces would be to provide the non-dominant hand with an input device that allowed for rapid activation of commands and modifiers *in addition* to performing spatial positioning.

THE PADMOUSE

The PadMouse (Figure 1) is a new input device that integrates a two degree-of-freedom touchpad with the ubiquitous two degree-of-freedom mouse. This allows for the input of gestures by the user's middle or index finger on the touchpad, while the user's hand controls the 2D spatial position of the device on the work surface (like a regular mouse).

In our prototype implementation, we mounted a Synaptics T1002D touchpad on a generic serial/PS2 mouse base. Two serial ports (or one serial port and one PS2 mouse port) are required to interface it to a host computer. This is sufficient for evaluating our design and interaction techniques. A product would require the integration of the mouse and touchpad firmware.

While the PadMouse can foreseably be incorporated into a variety of user interface techniques, in this paper we focus on its use as an enhanced device for the non-dominant hand. In addition to using the PadMouse for spatial positioning in Toolglass and T3 style interfaces, the touchpad on the PadMouse can be used to activate commands and modifiers using a technique we call Marking Keys.

MARKING KEYS

Marking Keys is an adaptation of Marking Menus [11]: a mark based interaction technique that allows a user to select items from a menu which may be single level or hierarchical. As Kurtenbach [11] describes, selections using Marking Menus can be made in two ways (Figure 2):

1. *menu mode:* The user enters this mode by pressing a button on a stylus or mouse and pausing for a short time (less than 1 second). A radial menu then appears on the display, centered around the cursor position. The user selects a menu

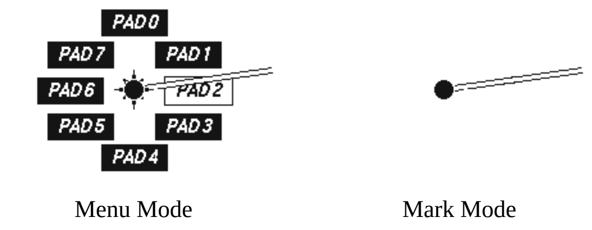

Figure 2a. Selecting from a single level marking menu. In menu mode, the menu is displayed and the user moves to the desired item. Users who are familiar with the menu layout select items in mark mode where no menu is displayed.

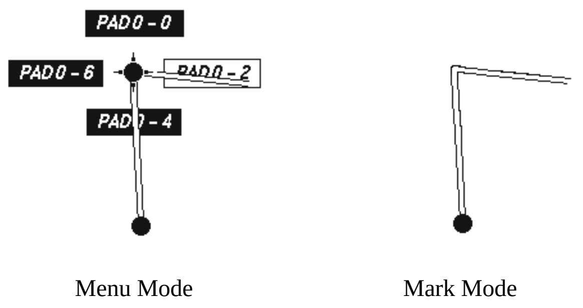

Figure 2b. Selecting from a hierarchical marking menu. The first segment of the mark selects a submenu and the second segment selects the desired menu item. In menu mode, the current (sub)menu is displayed and the user moves to the desired item. Users who are familiar with the menu layout select items in mark mode by making a compound mark.

item by moving the cursor into the desired sector of the radial menu, the selected item is highlighted, and selection confirmed when the mouse/stylus button is released.

2. *mark mode:* The user enters this mode by pressing a button on a stylus or mouse and *immediately* moving the cursor in the direction of the desired menu item. Instead of displaying the radial menu, the system displays an ink-trail following the cursor. When the button is released, the menu item that corresponds to the direction of movement is selected. If the user stops moving but does not release the button, the system assumes that the user is unsure of the menu layout and displays the menu, effectively providing guidance by self-revelation only when required. Studies have shown that making selections in mark mode can be 3.5 times faster than when the menu is displayed [11].

The beauty of Marking Menus is that a user who is unfamiliar with the menu layout only has to wait a split second before the menu is displayed. The mark made to select an item from this menu is similar to the mark required to make a selection when in mark mode. Each time a user makes a selection in menu mode, they are rehearsing the mark that would be required when in mark mode. This helps the user learn the markings. Once the user is familiar with the menu layout, selection is achieved very quickly by simply making a mark without waiting for the menu to be displayed. Thus, Marking Menus effectively utilizes self-revelation, guidance, and rehearsal to aid the user in making a smooth transition from novice to expert behaviour.

Our Marking Keys interaction technique, whose behaviour is illustrated in Figure 3, retains the favourable characteristics of Marking Menus while introducing several changes which result in a gesture based alternative to hot-keys:

- Instead of using a stylus or mouse to drive a cursor which then makes the required mark, users simply use their index finger to create the mark on the PadMouse's touchpad.
- No button presses are required. Rather, the menuing system is activated when the user's finger first touches the touchpad. If the user pauses for 0.4 seconds, a radial menu of options is displayed and the system is in menu mode. If the user starts moving immediately after touching the pad, the system is in mark mode and no menu is displayed.
- Recognition of the mark is achieved by pausing for 0.2 seconds after the mark is made. If a valid mark results in the selection of a command, that command is performed and the system returns to an idle state. If the mark results in the selection of a modifier, that modifier is activated and remains active as long as the finger remains on the touchpad. When the finger is lifted off the pad, the modifier is inactivated and the system returns to its default

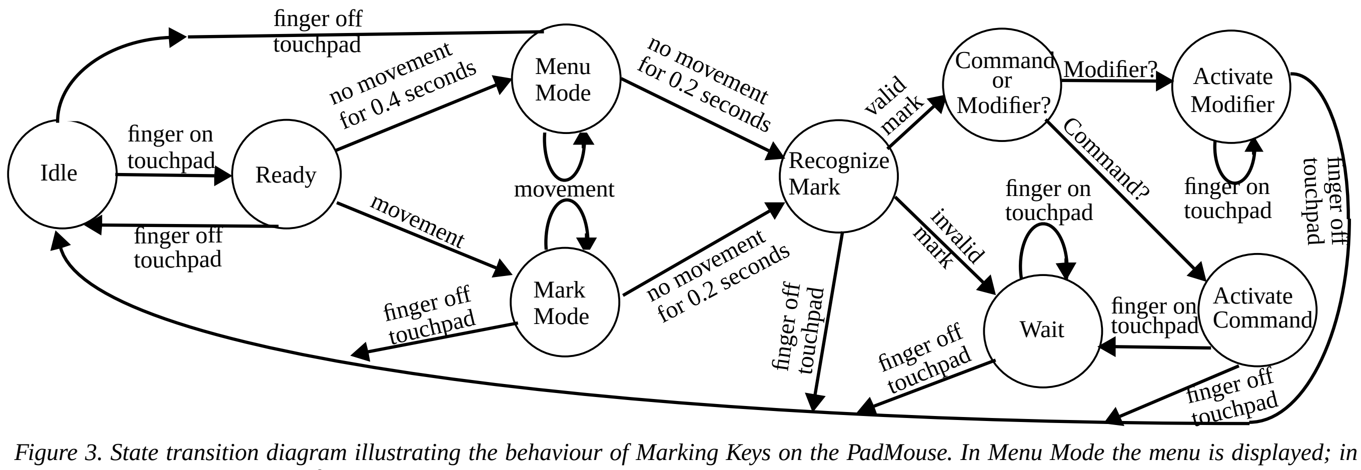

Figure 3. State transition diagram illustrating the behaviour of Marking Keys on the PadMouse. In Menu Mode the menu is displayed; in Mark Mode only the ink-trail of the mark is displayed. Once in the "Recognize Mark" state, no menu or marks are displayed.

mode. This is where Marking Keys differs from traditional Marking Menus: in Marking Menus there is no mechanism for activating modifiers; only commands can be selected. Thus, Marking Keys mimic the action of hot-keys, using gestures to select a command or modifier from a menu instead of pressing a key on a keyboard.

With hot-keys, there is no built-in mechanism for revealing the assignment of keys to functions. Users have to rely on on-line help pages or other documentation to discover the key mappings. This can be frustrating, particularly for novice users. Users of the PadMouse with Marking Keys get all the functionality of hot-keys with the added benefit of self-revelation, as well as a built-in spatial positioning device. Also, the actions required of the user when activating Marking Keys are conceptually and physically distinct from the actions required to control the spatial position of the PadMouse. As Jacob et al. [9] have shown, tasks which are conceptually independent benefit from input devices that allow the tasks to be performed separately. Users of the PadMouse and Marking Keys should likewise benefit from the separation of the two tasks it is designed to perform.

Our implementation of Marking Keys on the PadMouse allows users to activate commands and modifiers using simple marks (i.e., using a single level menu – Figure 2a) as well as compound marks (i.e., using a two level hierarchical menu – Figure 2b). A question of interest, therefore, is the bounds on how many items can be in each level, before the use of a mark for item selection becomes too slow or error prone. Kurtenbach and Buxton [12] explored this question in the context of using a stylus or mouse to select from a Marking Menu. They found that users could select from menus with up to two levels and eight items per level with an error rate under 10%. With menus more than two levels deep and greater than eight items, selection became error prone. Other research [8] also indicates that menus with up to eight items result in acceptable performance. Unfortunately, since Marking Keys on the PadMouse relies on the index finger for performing the marks rather than a mouse or stylus, this previous body of research can only serve as a guideline to us. The range and type of movement afforded by the index finger is very different from that afforded by the whole hand. Thus, it is possible that the optimal number of items that can be selected using our technique will differ from that selectable using a mouse or stylus. Furthermore, it is also likely that the finger will find marks in some directions easier to perform than others. In order to explore these issues, we conducted an experiment.

This is the first of a series of planned experiments with this device and interaction technique. At this early stage, we are mainly concerned with how users perform using Marking Keys on the PadMouse. We are therefore evaluating this technique without including the spatial positioning capabilities of the PadMouse. Issues concerning the ability to use Marking Keys while performing spatial positioning tasks is left to be formally evaluated at a later date; however, a discussion and informal observations are presented later in this paper.

EXPERIMENT

Goal

We had two primary goals for this experiment. Firstly, we wanted to determine the differences in performance when selecting using Marking Keys as the number of items and levels are increased. Secondly, we were interested in the issue of whether marks in certain directions are easier to perform than others.

In order to determine the limits of performance, we needed to measure expert behaviour. Like Kurtenbach and Buxton [12], we defined expert behaviour to occur when the user is completely familiar with the layout of the menu and knew the exact mark required to select a particular item. Research on Marking Menus [11, 12] has shown that users eventually reach this level of expertise, selecting using marks alone (i.e., without displaying the menu) over 90% of the time. The cognitive aspects of this previous work is directly applicable to our technique. Where our technique differs is in the motor skills required to perform the mark, and this is what we sought to measure.

Method

Subjects

12 right-handed volunteers participated as subjects in the experiment. All used their non-dominant hand to perform the experiment.

Apparatus

The experiment was conducted on a Silicon Graphics Indy workstation with a 19 inch colour display. The PadMouse was used as the input device. The mark recognition software was identical to that of a commercial implementation of Marking Menus and has been extensively tested in both experimental and real world settings. The workstation ran in single-user mode, disconnected from all network traffic. Subjects were seated approximately 60 cm in front of the display with their non-dominant hand manipulating the PadMouse with the index finger used to activate Marking Keys on the PadMouse's touchpad. As discussed earlier, the spatial positioning feature of the PadMouse was not used in this experiment.

Task and Stimuli

In this experiment we simulate the situation where subjects are completely familiar with the marks they need to make for a selection. We achieve this by displaying the desired mark on the screen and asking the subject to make a similar mark using the touchpad on the PadMouse. Since the user is told *a priori* what the desired mark should be, the cognitive effort required of the user is minimal and constant from trial to trial, allowing us to measure motor performance.

A trial occurred as follows. The stimulus mark was displayed on screen (Figure 4a). Using their index finger on the PadMouse's touchpad, the subject attempted to make a similar mark. The moment the subject's finger made contact with the touchpad, the stimulus mark was removed from the screen and an ink-trail of the mark the subject was making was displayed instead. At no time was the menu displayed. Identical to our implementation of Marking Keys (Figure 3), the mark was recognized and the trial ended when the sub-

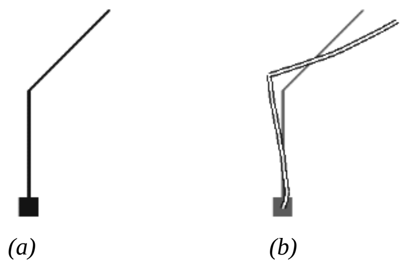

Figure 4. Stimulus and response for one experimental condition. (a) shows the desired mark presented to the subject at the start of a trial. (b) shows the subject's mark overlaid on the desired mark on completion of the trial.

ject's finger was stationary on the touchpad for 0.2 seconds. Once the mark was recognized, the stimulus mark was displayed overlaid with the ink-trail of the mark the subject made (Figure 4b). This served to reinforce learning and aid in correcting errors for future trials. If the subject's mark did not result in the same command or modifier selection as the stimulus mark, the system would beep to indicate an error.

Design

Guided by the results of previous work [8, 11, 12] as well as our informal use of the device and technique, we decided to limit our study to menus up to two levels deep and up to eight items per level as we felt that this would be the upper limit beyond which performance would definitely degrade. We chose three menu layouts (Figure 5):

*1 level, 8 item*s: this was the simplest layout studied. The eight items are laid out in a circle, each occupying a 45 degree sector. Only simple straight marks are required to select from this menu.

2 levels, 16 items: Four possible first level marks lead to a second level menu which in turn has four items, resulting in 4x4=16 items in total. All the required marks are aligned to the horizontal or vertical axes (often referred to as on-axis layout). Compound marks are required for selection.

2 levels, 64 items: Eight possible first level marks lead to a second level menu which in turn has eight items, resulting in 8x8=64 items in total. Compound marks are required for selection. Both on-axis and off-axis marks are required. This layout exhausts all possible combinations for menus up to two levels deep and up to eight items per level. The simpler layouts above were included because they allow for higher spatial variability in the required marks.

A within subjects completely counterbalanced repeated measures design was used. All subjects performed the experiment using all three menu layouts. The presentation order of the three layouts was completely counterbalanced across the subjects. For each menu layout, subjects performed 10 blocks of trials. Each block consisted of 1 trial for each of the possible marks for that menu layout, presented in random order within the block. Subjects were told in advance the menu layout for the upcoming 10 blocks and were also given several practise trials to familiarize themselves with the task, They were allowed breaks after every eight trials. After the completion of one trial within this group of eight, the next trial began after a 0.5 second pause.

The experiment consisted of 10560 total trials, computed as follows:

12 subjects x

3 menu layouts consisting of 8+16+64=88 different marks x

10 blocks of trials for each layout x

1 trial per mark per block

= 10560 total trials.

The experiment was conducted in one sitting and lasted about an hour per subject.

Results and Discussion

Figure 6 compares subjects' mean task completion time and error rates for the three menu layouts. Task completion time was measured beginning when the subject's finger first touched the touchpad and ending when the mark was recognized by the system. This effectively discounted the time taken to react to the stimulus, measuring only the cost of the motor component. Repeated measures analysis of variance showed that the number of items and levels in the menu had a significant effect on both task completion time ($F_{2,11} = 110.91$, $p < .0001$) and error rate ($F_{2,11} = 47.75$, $p < .0001$).

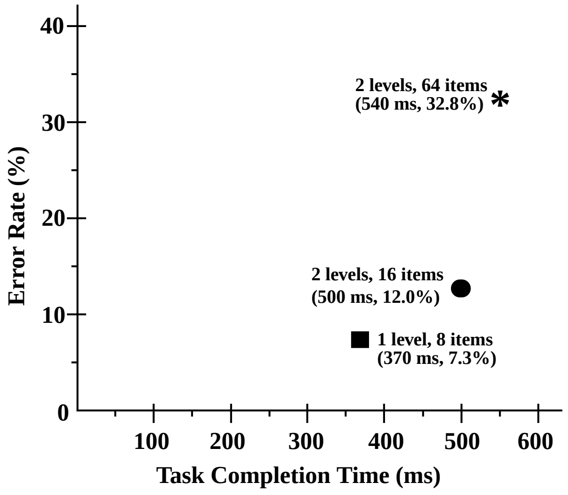

Figure 6. Comparison of the three menu layouts for error rates and movement time. Data from all 12 subjects and all trials.

From Figure 6, we see that the 2 level, 64 item menu had considerably higher error rates than the other two layouts. However, task completion time was not as drastically affected although the difference was statistically significant.

One possible cause of the drastic increase in errors could be that marks in some directions are particularly hard to perform and data from these trials is skewing our results. We therefore took a closer look at the performance differences between the various marks.

For the 1 level, 8 item layout, analysis of variance showed that the mark direction had a significant effect on error rates ($F_{7,77} = 2.08$, $p < .05$). A Ryan-Einot-Gabriel-Welsch pos-thoc test revealed that the mark for item 7 (refer to Figure 5a) was significantly harder to perform than the other marks.

For the 2 level, 16 item layout, analysis of variance again showed a significant effect for mark direction ($F_{15,165} = 3.31$, $p < .0001$). As expected, the easiest items were the four where both segments of the compound mark were in the same direction (items 0-0, 2-2, 4-4, and 6-6 in Figure 5b). The most error prone marks were those whose first segment was a left or right mark followed by a second segment that was an up or down mark (items 2-0, 2-4, 6-0, 6-4 in Figure 5b).

For the 2 level, 64 item layout, the situation gets more complex. Like the 2 level, 16 item layout, the marks where both segments of the compound mark were in the same direction (items 0-0, 1-1, 2-2, 3-3, 4-4, 5-5, 6-6, 7-7 in Figure 5c) performed best, with error rates under 10%. Previous research on Marking Menus with large numbers of items and levels [12] have found that when the mark segments are confined to the horizontal or vertical axes (on-axis marks), performance is better than for off-axis marks. An analysis of variance on our data grouped according to whether the mark segments were on-axis or off-axis showed a significant effect for axis ($p < .05$). Marks where both segments were on-axis performed best, followed by those where both segments were off-axis. When one segment was off-axis and the other segment on-axis, performance degraded. A possible explanation is that these "on-off" and "off-on" axis marks account for all the cases with the smallest angle between the two segments of the mark (i.e., a 45 degree turn) which are generally difficult marks to make accurately. If we consider only the 32 marks where both segments were either on-axis or off-axis, the error rate goes down to under 20%.

What constitutes an acceptable error rate for tasks like this? As Kurtenbach & Buxton [12] point out, "the answer depends on the consequences of an error, the cost of undoing an error or redoing the command, and the attitude of the user." Like Marking Menus, Marking Keys present us with the typical trade-off between speed and accuracy. A redeeming feature of Marking Keys is that if the error rate when in mark mode is too high, users can always resort to the slower but more accurate fallback technique of displaying the menu and making a selection.

Comparing our results to the prior art, we find that Marking Keys fares well. Kurtenbach [11] reports that users with 10 to 30 hours of practice are able to select from a six-item Marking Menu using a mouse at a rate between 200 and 400 ms. Nilsen [14] reports that a selection from a six-item linear menu using a mouse required on average 790 ms. Card, Moran, & Newell's [4] Keystroke-Level model indicates that a "good typist" takes 120 ms to execute a key-press on a keyboard, and 400 ms to home in on the desired key. This results in an approximate figure of 520 ms for selection using keyboard hot-keys. Our subjects, with only an hour of practice, averaged 370 ms for an eight-item menu.

This experiment has demonstrated that users of Marking Keys can quickly activate up to 32 commands and modifiers with an error rate of under 20%. This is probably more than the number of hot-keys required in a typical application. If we restrict ourselves to single level menus with a small number of items, much better performance (speed faster than hot-keys, and errors under 8%) is possible.

As mentioned earlier, this is the first of a series of planned experiments to evaluate this device and interaction technique. At this stage, we sought to determine the limits of user performance with Marking Keys and did not confound the issue by utilizing the spatial positioning capabilities of the PadMouse. An obvious question, therefore, is the extent to which spatial positioning of the mouse interferes with the ability to make correct marks on the touchpad and vice versa. Informal observations of users of the device in prototype applications (see section below) do not show evidence of interference between the finger movements on the toucpad and wrist/arm movements controlling the mouse. Indeed, experienced users (including the authors of this paper) are able to activate modifiers and commands with Marking Keys while simultaneously using the spatial positioning capabilities of the device. While it appears that interference between the two parts of the device is minimal, a formal experimental evaluation is required for confirmation and this will be conducted in the near future.

APPLICATIONS

We have prototyped the use of the PadMouse with Marking Keys within one of Alias|wavefront's new 3D modeling/animation applications, Maya. We use the spatial positioning capability of the PadMouse to control the virtual camera in the non-dominant hand (Zeleznik et al. [15] describe similar use of a non-dominant hand input device) while Marking Keys is used to activate frequently used commands (e.g. "undo", "redo last command", "copy"), and modifiers (e.g., activation of graphical manipulator widgets for object translation, rotation, scaling, etc.). The dominant hand uses a regular mouse to perform operations in the 3D scene using those graphical manipulator widgets. In the default unimanual interface, the dominant hand has to constantly switch between camera manipulation and object manipulation. Our bimanual interface has the advantage of having the non-dominant hand control the camera so that users can now tumble/track/dolly around the 3D scene to get the best view of their work while changing parts of the scene with their dominant hand. Preliminary feedback from those who have tried this implementation is encouraging.

We have also incorporated the PadMouse and Marking Keys in a prototype 2D drawing application with a Toolglass and Magic Lenses interface similar to the "T3" concept application of Kurtenbach et al. [13]. Here, the spatial positioning capability of the PadMouse replaces the digitizing tablet puck used in the "T3" application, while Marking Keys are used to select the contents of the Toolglass sheet (the previous implementation [13] uses the dominant hand to make this selection using a Marking Menu).

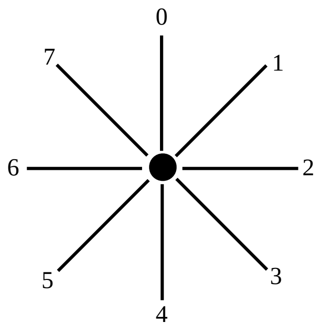

Figure 5(a). 1 level, 8 item layout

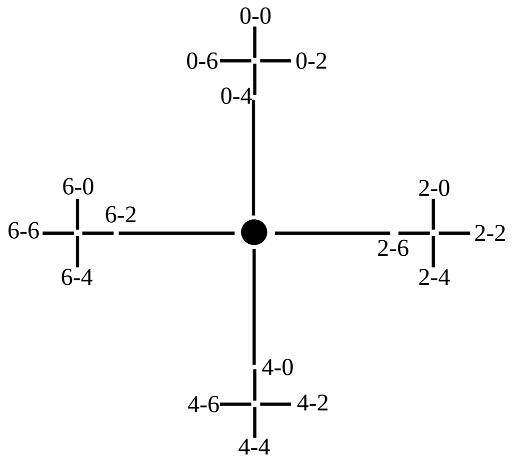

Figure 5(b). 2 level, 16 item layout. The first segment of the mark selects the submenu and the second segment selects the item from that submenu. All the marks are aligned to the horizontal or vertical axes (on-axis layout).

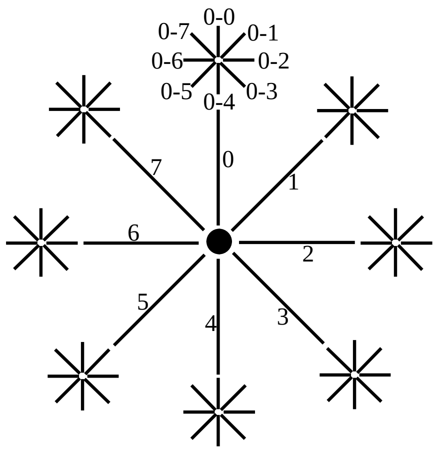

Figure 5(c). 2 level, 64 item layout. The first segment of the mark selects the submenu and the second segment selects the item from that submenu. Both on-axis and off-axis marks are included. To avoid clutter, we have only labeled one submenu. The others are labeled in a similar fashion: first segment number followed by second segment number

FUTURE DIRECTIONS

In this paper, we have focused on the use of the PadMouse as a non-dominant hand device in a two-handed user interface. Clearly, this is not the only application for this device. Other possible uses include using the touchpad to perform scrolling and other tasks. Zhai, Smith, and Selker [15] recently performed a comparison of various input devices in scrolling tasks. It would be interesting to see how the PadMouse fares in such a comparison.

We are also exploring alternative designs to the PadMouse for providing command/modifier selection and spatial positioning in an integrated device. One design we have prototyped (Figure 7) is a mouse with 10 keys in an arrangement designed to facilitate comfortable key activation without much finger movement. This design is an evolution of the obvious solution of placing a keypad on the back of the mouse – a design where it is impossible to move the mouse spatially while pressing buttons on the keypad. Preliminary use of our prototype indicates that some users find it difficult to activate buttons with the fourth and fifth fingers, but this could perhaps be overcome with improved button designs. Performance differences between this device and the PadMouse with Marking Keys will have to be evaluated.

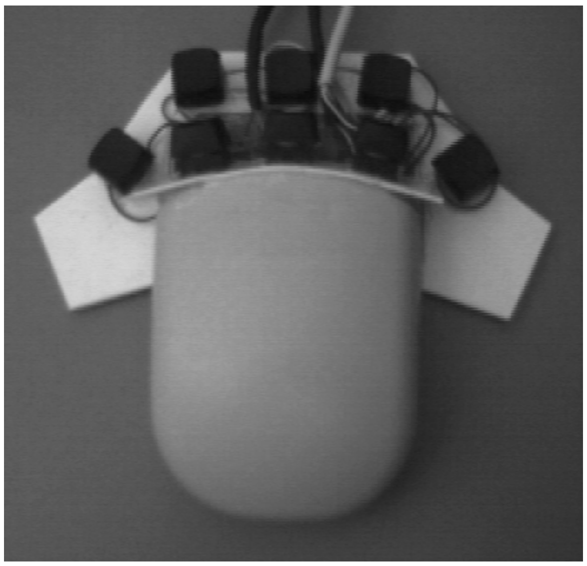

Figure 7. Prototype Ten-key mouse.

CONCLUSIONS

We have presented a new input device, the PadMouse, and an associated interaction technique called Marking Keys. This allows users to perform both spatial positioning tasks as well as activate commands and modifier using the non-dominant hand. Our experiment showed that subjects were able to effectively activate up to 32 commands/modifiers using this device and interaction technique. We believe that incorporating a device like this in two-handed interfaces can lead to more facile and efficient human-computer interaction.

ACKNOWLEDGMENTS

We thank Thomas Baudel, Bill Buxton, George Fitzmaurice, Gordon Kurtenbach, Scott MacKenzie, and Russell Owen for valuable discussions and assistance during the course of this work. We'd also like to thank the other members of the Input Research Group at the University of Toronto and all the volunteers who participated in our experiment. The support of Alias|wavefront is gratefully acknowledged.

REFERENCES

[1] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., & DeRose, T.D. (1993). Toolglass and magic lenses: The see-through interface. *Computer Graphics Proceedings, Annual Conference Series*, 73-80, New York: ACM SIGGRAPH.

[2] Bier, E. A., Stone, M. C., Fishkin, K., Buxton, W., & Baudel, T. (1994). A taxonomy of see-through tools. *Proceedings of the CHI'94 Conference on Human Factors in Computing Systems*, 358-364, New York: ACM.

[3] Buxton, W., & Myers, B. A. (1986). A study in two-handed input. *Proceedings of the CHI'86 Conference in Human Factors in Computing Systems,* 321-326, New York: ACM.

[4] Card, S. K., Moran, T.P., & Newell, A. (1980). The Keystroke-Level model for user performance time with interactive systems. *Communications of the ACM, 23, 7,* 396-410, New York: ACM.

[5] Fitzmaurice, G. W., & Buxton, W. (1997). An empirical evaluation of graspable user interfaces: towards specialized, space-multiplexed input. *Proceedings of CHI'97 Conference on Human Factors in Computing Systems,* 43-50, New York: ACM.

[6] Guiard, Y. (1987). Asymmetric division of labour in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behaviour, 19,* 486-517.

[7] Hinckley, K., Pausch, R., Proffitt, D., Patten, J., & Kassell, N. (1997). Cooperative Bimanual Action. *Proceedings of the CHI'97 Conference on Human Factors in Computing Systems*, 27-34, New York: ACM.

[8] Hopkins, D. (1991). The design and implementation of pie menus. *Dr. Dobbs' Journal*, December 1991, 16-26.

[9] Jacob, R. J. K., Sibert, L. E., McFarlane, D. C., & Mullen, M. P. (1994). Integrality and separability of input devices. *ACM Transactions on Computer-Human Interaction, 1, 1,* 3-26.

[10] Kabbash, P., Buxton, W., & Sellen, A. (1994). Two-handed input in a compound task. *Proceedings of the CHI'94 Conference on Human Factors in Computing Systems,* 417-423, New York: ACM.

[11] Kurtenbach, G. (1993). *The Design and Evaluation of Marking Menus.* Ph.D. Thesis, Department of Computer Science, University of Toronto, Toronto, Canada.

[12] Kurtenbach, G., & Buxton, W. (1993). The limits of expert performance using hierarchical marking menus. *Proceedings of InterCHI'93 Conference on Human Factors in Computing Systems*, 482-487, New York: ACM.

[13] Kurtenbach, G., Fitzmaurice, G., Baudel, T., & Buxton, W. (1997). The design of a GUI paradigm based on tablets, two-hands, and transparency. *Proceedings of the CHI'97 Conference on Human Factors in Computing Systems,* 35-42, New York: ACM.

[14] Nilsen, E. L. (1991). Perceptual-motor control in human-computer interaction. Technical Report NO. 37, Cognitive Science and Machine Intelligence Laboratory, University of Michigan.

[15] Zeleznik, R. C., Forsberg, A. S., & Strauss, P. S. (1997). Two pointer input for 3D interaction. *Proceedings of 1997 Symposium on Interactive 3D Graphics*, Providence, Rhode Island.

[16] Zhai, S., Smith, B. A., & Selker, T. (1997). Improving browsing performance: A study of four input devices for scrolling and pointing. *Proceedings of INTERACT'97: The Sixth IFIP Conference on Human-Computer Interaction*, Sydney, Australia.