PAPER Special Section on Knowledge, Information and Creativity Support System

Pen-Based Interface Using Hand Motions in the Air

Yu SUZUKI[†a)], ***Student Member***, **Kazuo MISUE**[†b)], ***and*** **Jiro TANAKA**[†c)], ***Members***

SUMMARY A system which employs a stylus as an input device is suitable for creative activities like writing and painting. However, such a system does not always provide the user with a GUI that is easy to operate using the stylus. In addition, system usability is diminished because the stylus is not always integrated into the system in a way that takes into consideration the features of a pen. The purpose of our research is to improve the usability of a system which uses a stylus as an input device. We propose shortcut actions, which are interaction techniques for operation with a stylus that are controlled through a user's hand motions made in the air. We developed the Context Sensitive Stylus as a device to implement the shortcut actions. The Context Sensitive Stylus consists of an accelerometer and a conventional stylus. We also developed application programs to which we applied the shortcut actions; e.g., a drawing tool, a scroll supporting tool, and so on. Results from our evaluation of the shortcut actions indicate that users can concentrate better on their work when using the shortcut actions than when using conventional menu operations.

key words: *human-computer interaction, pen-based interface, menu presentation, hand motions*

1. Introduction

Because the pen is a tool used by almost everyone from an early age, it is one of the most familiar creative activity tools. Most people can handle a pen correctly when given one. Therefore, we think that compared to devices used only for computers, like the mouse and keyboard, the pen-shaped stylus is a more suitable interface for creative activities.

Computers employing pen-based devices as an input device include tablet PCs, PDAs, large screen displays with touch panels and so on. As the use of computers with such devices has spread, people increasingly use computers to engage in creative activities like writing and painting. Especially, large screen displays with touch panels are being used instead of a chalkboard in education and instead of a whiteboard in meetings. As pen-based systems are made more practicable in this way, functions of application programs based on the premise of operating with a pen-based device, such as digital ink and handwriting recognition, have become increasingly available.

When we consider application programs in terms of the interface, though, we find most application programs do not provide an interface which strongly supports pen-based input. A function menu typically follows one of two styles: the menu bar style or the small icon style. With the menu bar style, the user has to follow a tree-structured hierarchical menu from a menu bar at the top of a window. In the small icon style menu, the user has to correctly click on a small icon. Therefore, the user has to correctly select the menu item or icon to operate the menus. When using a mouse, the user can click a certain target with the pointing position rarely deviating from the intended movement of the pointing cursor to a target to click it. On the other hand, when a stylus is used, the process of bringing the pen close to the display will often cause the actual position to deviate from the intended position. In other words, the pointing position is often inaccurate when the pen tip touches the display. This is generally because of parallax effects due to the display panel thickness or unsteady hand motion because the wrist or elbow does not rest on a stable surface. In creative activities, the user should concentrate on his or her essential work. However, the difficulty of correctly pointing with a stylus interrupts the user's mental concentration. These problems occur because the WIMP interface, designed on the premise of mouse operation, is applied to stylus operation.

When the user uses a large screen display with a touch panel, the operating menu bar style menu or small icon style menu is particularly difficult. This is because the user has to move from his or her standing position to operate menus since these menus are displayed in the extreme upper-left corner of a window. Therefore, for usage of a large screen display, it is desirable that the user is able to operate menus without depending on the standing position.

In addition, use of the WIMP interface creates a problem with input operations by a stylus. The problem is that input operations by a mouse must be duplicated through input operations by a stylus. Basic stylus operations are tapping, stroking, double tapping, pushing the barrel button + tapping, holding, and so on. Among these, tapping and stroking are essential operations as a pen. In contrast, the others are input operations used to implement mouse operations. When they were designed, these input operations did not take into consideration that a stylus is a pen-shaped device, and this accounts for much of difficulty in using a pen-based input system.

As mentioned, a stylus is suitable for creative activities, but the current stylus interface is not always easy to use and interrupts the user's mental concentration in creative activities. The purpose of our research is to develop a stylus inter-

Manuscript received April 4, 2008.
Manuscript revised July 4, 2008.

[†]The authors are with the Department of Computer Science, University of Tsukuba, Tsukuba-shi, 305–8573 Japan.
a) E-mail: suzuki@iplab.cs.tsukuba.ac.jp
b) E-mail: misue@cs.tsukuba.ac.jp
c) E-mail: jiro@cs.tsukuba.ac.jp

DOI: 10.1093/ietisy/e91–d.11.2647

face and improve the usability of a system which employs a stylus as an input device. To achieve this, we have developed stylus input techniques based on the fact that a stylus is a pen-shaped device. In other words, we took advantage of the normal features of a pen, such as its shape and the way it is used, to create new input techniques. Furthermore, we developed a GUI which utilizes these input techniques. This will enable a computing environment more suitable for creative activities based on handwriting-style input.

2. Related Work

The purpose of our research is to improve the usability of a system which employs a stylus as an input device. We have taken two main approaches: develop a menu suitable for operations with a stylus, and develop an interaction technique suitable for operations with a stylus.

With regard to the first approach, Hopkins proposed Pie Menu [1] as a menu which is easy to use with a pen-based device. While the conventional pop-up menus align menu items linearly, Pie Menu aligns menu items radially. In their experiment [2], they showed that Pie Menu is advantageous compared with linear menus in terms of the selection time and the number of errors. Other research on menus includes that on Marking Menu [3], Control Menu [4] and FlowMenu [5]. A text entry method applied through FlowMenu [6] has also been developed.

Regarding the second approach, Accot et al. developed Crossing [7], which is an interaction technique. Crossing means that the user has to draw a cross on an operational object, rather than use tapping, and this enables more stable operations. As an application using Crossing, Apitz et al. developed CrossY [8], a drawing tool in which all operations can be realized by crossing. In CrossY, the user can operate GUI parts (e.g., radio buttons and scroll bars) by crossing.

Smith et al. developed the radial scroll tool [9], which includes an interaction technique for scrolling operations. With the radial scroll tool, the user lays a stylus on a display and a guide appears at the pointing point. The user can then use gestures, like drawing a circle at the pointing point, to scroll a screen. A tapping position becomes the center of a guide and the amount of scrolling depends on the distance from the center of a guide to the position where a circle is drawn.

Hinckley et al. developed Scriboli [10], which is operated through a new interaction technique called pigtail. Pigtail is a small loop at the end of a stroke. In Scriboli, the user shows a Marking Menu by drawing a pigtail after a stroke around an operation object, and then selects a command.

Siio et al. proposed an interaction technique which uses the Paperweight Metaphor [11]. They focus on a motion where people write while holding a paper with their palm. The status of the user's palm — whether it is touching the bottom of a PDA — determines the switching of modes.

Focusing on a state of the stylus, the above research utilizes a stylus whose pen tip touches a display. In contrast, our research focuses on the state of stylus whose pen tip does not necessarily touch a display. We describe this in the next section.

3. Using Hand Motions in the Air

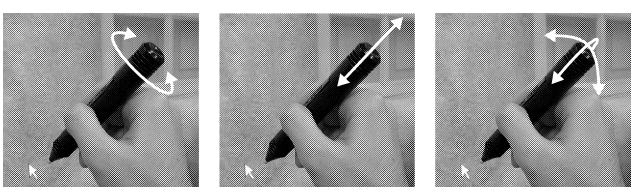
First, we considered new operating methods that took into consideration the shape of a pen, and the way a pen is used, and that would improve the usability of a system with a stylus as an input device. Then, we focused on typical motions when people use a pen. A pen's status can fall into two categories: the pen tip is touching the display, or it is not touching. The motions when the pen tip is in contact with the display have already been applied through tapping and stroking operations. In contrast, motions when the pen tip is not in contact with the display — that is, motions made in the air — have not yet been much used. However, motions in the air allow effective use of a pen; for example, forcing out the graphite lead in a mechanical pencil by shaking the pencil. Consequently, we considered utilizing motions in the air as a means of stylus interaction. By implementing this interaction technique, we hope that conventional operations based on mouse operations can be replaced with more pen-like and natural operations.

3.1 The Shortcut Actions

The shortcut actions are interaction techniques which use hand motions made in the air. A person holding a pen can make various motions in the air. For interaction application, such motions should be possible while holding a stylus and be natural. In this research, we chose to use three motions (Fig. 1).

- Rolling a pen around the pen-axis direction (called rolling)
- Shaking a pen in the pen-axis direction (called shaking)
- Swinging a pen in a direction perpendicular to the pen-axis (called swinging)

We use two kinds of rolling (clockwise rolling and counterclockwise rolling, depending on the rotation direction), and four kinds of swinging (north, south, east and west). We assume that each of these input operations is a separate operation. Swinging can be classified into more than four types, but we consider four adequate given the tradeoff between the potential number of input operations and ease of use. Because each element is separated by 90°, the user can intuitively understand the swinging direction.



(a) Rolling. (b) Shaking. (c) Swinging.

Fig. 1 Three actions used in shortcut actions.

The four-color pen metaphor we have adopted, as we will explain later, is another reason we decided to use four types of swinging.

By combining these three actions, plus the conventional techniques of tapping and stroking, the user can enter various inputs.

3.2 Adopting Pen Metaphors

We have adopted two pen metaphors in the shortcut actions: the pendulum mechanical pencil metaphor and the four-color pen metaphor.

The pendulum mechanical pencil is a pen that has a mechanism which allows a user to force out the graphite lead by shaking. This means shaking a pen up and down has a purpose. In this research, we assimilated this feature as the pendulum mechanical pencil metaphor and applied it to the design of shaking. The pendulum mechanical pencil is already widely used so many people have experience using it. The need to repeatedly force out the lead by shaking a pendulum mechanical pencil makes this action a common practice and users learn to do it automatically without thinking. We hope, by using shortcut actions, to implement such automatic forms of interaction that the user does not need to think about consciously.

A four-color pen has four buttons at the top of the pen, so each perpendicular direction from a pen implies one color. We incorporated this feature as the four-color pen metaphor and applied it in the design of shortcut actions.

3.3 Providing New Input Types in Stylus Operations

Our intention was to replace menu bar and small icon menus with shortcut actions. Although such menus show some constant menu items that are context independent, the menus are not easy to use with a stylus as explained in Sect. 1. Therefore, we wanted to enable users to operate a menu without depending on context by using shortcut actions and pop-up menus.

With conventional input methods, a user can view a menu by tapping while pushing a barrel button or by tap & hold. However, because these operations have already been assigned an important role in displaying a context menu, we treat the barrel button as a previously "reserved" input method.

The user will be able to view non-context menus and context menus by using both a shortcut action and the barrel button. We expect this to improve the ease of using a stylus.

4. Development of a Context Sensitive Stylus

We have to obtain the hand motions made in the air and identify the actions to implements the shortcut actions. To do this, we developed a stylus by which we can obtain the hand motions made in the air, called Context Sensitive Stylus. In this section, we describe the way to obtain the hand motions made in the air and identify the hand motions as three actions and other motions.

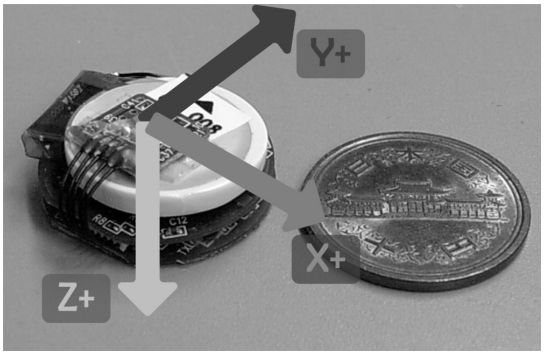


Fig. 2 Cookie sensor and three-axis detection direction.

Fig. 3 Context Sensitive Stylus.

4.1 Use of an Accelerometer

We attached an accelerometer to the stylus in order to obtain hand motions made in the air while the user holds the stylus. In our research, we used a Cookie sensor[†] as the accelerometer. We show the Cookie sensor in Fig. 2. The Cookie can detect both three-axis acceleration and magnetic direction by 10 Hz. In addition, the Cookie has a Bluetooth module which lets it communicate with a computer wirelessly.

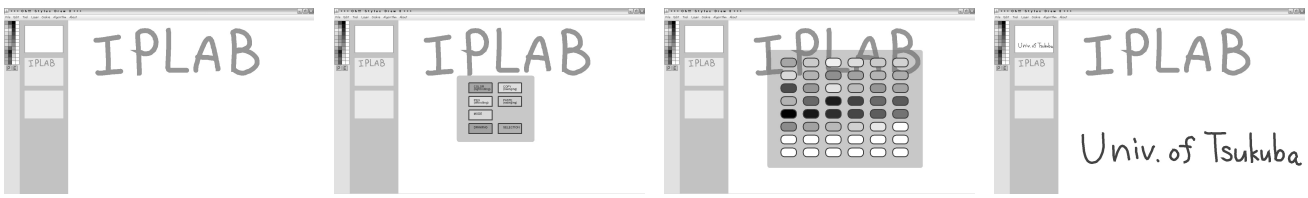
We show a stylus with an attached Cookie in Fig. 3. We call this stylus with an accelerometer attached a context sensitive stylus (CS stylus).

A concern was that the weight balance of the stylus would be changed by attaching the accelerometer to the top of the stylus. However, the lightness of the Cookie sensor prevents any such detrimental effect on usability. In addition, a key feature of the CS Stylus is that a wire connection is not used, so a user can use the CS Stylus as comfortably as a conventional stylus.

4.2 Identification of Hand Motions in the Air

Our system has to identify which hand motions correspond

[†]Cookie sensor from Nokia Research Center Tokyo.



(a) A state of drawing mode.

(b) When the user performs a shaking, the Top Menu FlowButton is shown.

(c) When the user performs a clockwise rolling while the Top Menu FlowButton is shown, the Color Palette FlowButton appears.

(d) When the user chooses a color from the color palette by tapping, the drawing color is changed.

Fig. 4 Screen shots of Oh! Stylus Draw. We show an example of the user changing the drawing color, and then writing "Univ. of Tsukuba". The user does not have to change the standing position to operate menus because FlowButton is always shown near the pointer.

to rolling, shaking, swinging, or other motions. As identification methods, we tried to use comparison of the acceleration difference and comparison of the angle of rotation calculated using an arctangent function. However, the action identification accuracy did not increase because these methods caused many false identifications.

Consequently, we applied pattern matching for action identification. In this research, we used DP matching which has high general versatility and high matching accuracy. DP matching needs two patterns. One is a registration pattern which we register in advance and the other is a matching pattern which is the target of matching. Because the system needs matching patterns for all actions, we have seven registration patterns in total: two rolling patterns, one shaking pattern, and four swinging patterns. When DP matching is used, the accuracy of identifying shortcut actions is about 90% for shaking and 60∼70% for rolling and swinging. We discuss the identification accuracy in Sect. 6.3.

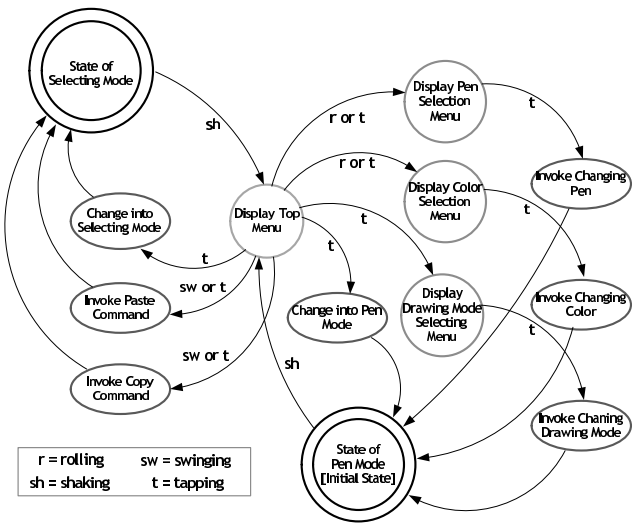


Fig. 5 Flow of operation when using shortcut actions.

5. Applications of the Shortcut Actions

As application software which uses the shortcut actions, we developed a new drawing tool and tools to extend the interface of existing application software. In this paper, we introduce a drawing tool, Oh! Stylus Draw, and a scrolling supporting tool, Oh! Stylus Scroll.

5.1 Oh! Stylus Draw

Oh! Stylus Draw is a drawing tool which provides the user with basic functions such as changing the drawing color, pen type, or drawing mode. As the menu interface of Oh! Stylus Draw, we implemented a FlowButton menu interface which we developed for operations with a stylus as well as a conventional style menu interface consisting of a menu bar and small icons menu. FlowButton is a kind of popup menu and panel type menu as shown at the center of Figs. 4 (b) and 4 (c). FlowButton is displayed where the user is working because it pops up near the pointing cursor. A user using a large screen display with a conventional menu has to change his or her standing position to operate the menu, and operating the menu itself with a stylus is difficult. In contrast, FlowButton allows a the user to operate the menu easily without moving the standing position and the menu itself is easy to operate with a stylus.

In Oh! Stylus Draw, the user calls the FlowButton panel and chooses menu items by using the shortcut actions. The correspondence between actions and menus is as follows.

- rolling (clockwise) . . . Calling the color palette
- rolling (counterclockwise) . . . Calling the pen panel
- shaking . . . Calling the top menu panel
- swinging (North) . . . Copying an object
- swinging (South) . . . Pasting the object

We will now describe the method of operation for Oh! Stylus Draw. Using a menu bar style menu, the user taps items in the following order, which is the same as for a general menu interface: menu bar, sub-menu, and then command item. Using the shortcut actions, the user combines three actions and tapping. The flow of operations is as shown in Fig. 5. Basic operational procedures are as follows:

1. Calling a top menu FlowButton by shaking

2. Calling a sub-menu FlowButton by rolling or tapping
3. Invoking a command by tapping

As an example, we will explain how a drawing color is changed. We show the screen transition in Fig. 4. The operational procedure of this task is as follows:

1. Call a top menu FlowButton by shaking (Fig. 4 (b))
2. Call a color palette FlowButton by clockwise rolling or tapping "Color Button" (Fig. 4 (c))
3. Choose a color by tapping

When performing the same task through conventional menu operations, the user has to access a tool panel at the far left of a display and tap it. These operations require ineffectual motions.

As mentioned, in Oh! Stylus Draw, FlowButton is displayed near the pointing cursor. This solves the large screen display problem of difficult to operate menus which depend on the user's standing position. In addition, the user can operate all menus without troublesome operations, such as the need to use a barrel button, because the user can call FlowButton menus and easily choose menu items through shortcut actions. As a result, the user can concentrate on his or her work without menu operations interrupting his or her concentration.

5.2 Oh! Stylus Scroll

The user has to use a scroll bar for scrolling operation with a stylus. However, this requires precise pointing by the user. Furthermore, when using a large screen display, the user has to move from the current standing position to the scroll bar position. Thus, scrolling operation with a stylus is cumbersome.

Oh! Stylus Scroll is a scroll support tool to make scrolling with a stylus easier. If the scroll bar is vertical (horizontal), clockwise rolling invokes downwards (rightwards) scrolling and counterclockwise rolling invokes scrolling in the opposite directions. If there are both vertical and horizontal scroll bars, vertical scrolling is dealt with preferentially. Oh! Stylus Scroll allows a user to scroll without regard to the standing position because the user does not directly operate the scroll bar.

This application software, the user can operate many GUI components, such as a slider which can otherwise be operated by a mouse wheel, as well as the scroll bar. Oh! Stylus Scroll provides users with an interface which allows them to use a stylus to operate GUIs designed on the premise of a mouse being used as an interface device.

6. Evaluation

6.1 Purpose of Evaluation

The purpose of our research has been to improve the usability of a system which uses a stylus as an input device. This evaluation tested whether users could better concentrate on their work when using the shortcut actions than when using conventional menu operations. We tested two hypotheses:

Hypothesis 1: The user experiences fewer operation failures when using shortcut actions than when using conventional operations.

Hypothesis 2: The user is required to make fewer ineffectual motions when using shortcut actions than when using conventional menu operations.

Operation failures were failures caused by the user. Ineffectual motions were motions which did not directly contribute to the primary work of the user.

6.2 Evaluation Method

In this experiment, we compared the operability of three menu operation methods:

- Menu operation using the conventional menu interface (the conventional method)
- Menu operation using a single action (the single-SA method)
- Menu operation using multiple actions (the multi-SA method)

In the conventional method, the user uses a tool panel as shown in the left side of Fig. 4. In the tool panel, the user can change a drawing color by tapping each color cell. The cell size is about the same as that for general paint tools.

In the single-SA method, the user first calls the top menu FlowButton by shaking. Next, the user calls the color palette FlowButton by tapping, and then changes the color by tapping.

In the multi-SA method, the user first calls the top menu FlowButton by shaking, and then calls the color palette FlowButton by rolling. The user then changes the color by tapping.

Five participants took part in this experiment. All were male, 22–24 years old, and right-handed. All of the participants had some experience using a pen-based interface, but none used one daily or were accustomed to using a pen-based interface.

We used Oh! Stylus Draw in this experiment. There were two tasks. In both, the participants had to join a pair of rectangles using a line whose color was the same as that of the rectangles. There were five pairs in each task, and each was drawn in a different color. In task 1, the five pairs were shown on the left side of the display, and in task 2 they were shown on the right side of the display. The tool panels were located on the left side, so it was easier to access the tool panels in task 1 than in task 2. The participants did these tasks using each of the three methods one time. We measured the time needed for a task, the number of false action identifications, delay time due to false action identification, the number of false operations, delay time due to false operations, and the number of steps. Through these measurements, we tested whether the shortcut actions improved system usability.

Table 1 Mean value of the result. ([ ] represents standard deviation)

Method and task	Time needed for a task (s)	Number of false identifications of the action (count)	Delay time due to false identification of the action (s)	Number of false operations (count)	Delay time due to false operations (s)	Number of steps (step)	Time needed for a task without delay time due to false identifications (s)
Conventional_Task1	26.7 [10.8]	0.0 [0.0]	0.0 [0.0]	4.2 [1.8]	2.2 [1.9]	0.0 [0.0]	26.7 [10.8]
Conventional_Task2	33.4 [14.6]	0.0 [0.0]	0.0 [0.0]	6.4 [3.2]	5.2 [4.8]	5.4 [0.5]	33.4 [14.6]
Single-SA_Task1	31.3 [4.6]	5.2 [1.3]	8.2 [2.6]	0.0 [0.0]	0.0 [0.0]	0.0 [0.0]	23.1 [2.5]
Single-SA_Task2	29.4 [6.1]	4.2 [2.7]	5.7 [3.6]	0.0 [0.0]	0.0 [0.0]	0.0 [0.0]	23.7 [4.7]
Multi-SA_Task1	46.3 [12.9]	11.6 [4.6]	19.7 [13.6]	0.0 [0.0]	0.0 [0.0]	0.0 [0.0]	26.6 [3.4]
Multi-SA_Task2	44.2 [7.5]	8.4 [3.0]	18.4 [8.4]	0.0 [0.0]	0.0 [0.0]	0.0 [0.0]	25.8 [3.9]

For this evaluation, we first explained to the participants how to use Oh! Stylus Draw and its interface. We then let the participants freely use Oh! Stylus Draw and CS Stylus, including the tool panel, before they had to perform the assigned tasks. We did not limit the practice time, but all participants practiced about 10 minutes. We recorded the experimental conditions for all participants, and then measured the number of false action identifications, the delay time due to false identifications, and so on. We specified in advance the order of the two tasks and the three menu operation methods to the participants. We decided this order in consideration of counterbalancing. After the evaluation, participants completed a questionnaire regarding the ease of operation.

In this experiment, we used a large screen display with a touch panel, a CS Stylus, and a conventional stylus except with Oh! Stylus Draw. The display was a 50-inch panel with a screen resolution of 1280×768. We showed Oh! Stylus Draw for the full screen. The Top Menu FlowButton size was 320×274 and the size of menu elements shown in the FlowButton was 96×41. The size of the color palette FlowButton was 640×487 and the size of menu elements shown in the FlowButton was 64×36. If the FlowButton is too large, too much of the working area will be hidden and usability will be impaired; if the FlowButton is too small, though, objects will be hard to select. Therefore, we decided on the FlowButton size with the screen size and usability in mind.

6.3 Results and Discussion

We show the results for each method and each task in Table 1. All values are the mean value of all participants and rounded off to one decimal place.

First, we discuss the number of false operations; i.e., the number of times the user wanted to choose one item from the color palette on the tool panel or FlowButton, but mistakenly chose a different item. With the conventional method, the mean number of false operations was 4.2 in task 1 and 6.4 in task 2. There were more false operations in task 2 than task 1 because of the greater distance to the tool panel.

In task 2, the participants performed the task on the right side of the display. Therefore, they had to move from their standing position to operate the tool panel. This apparently caused false operations because they did not perform operations consistently. With the single-SA and multi-SA methods, no false operations occurred. This was probably because the FlowButton was used as a menu interface, and the large size of each menu item in FlowButton made it easy for the participants to choose the intended item. In addition, because the participants used the shortcut actions instead of a button displayed on the screen, they might have not been concerned about performing false operations when displaying the FlowButton. Thus, compared to the conventional method, the two methods using the shortcut actions made false operations less likely, which supports hypothesis 1.

Next, we discuss the number of steps. Video camera recording showed that the participants took a mean of 5.4 steps in front of the display to perform task 2 by the conventional method. Such walking motions are clearly unnecessary. In contrast, the mean number of steps with the two methods using the shortcut actions was 0. These results support hypothesis 2.

The above results support our two hypotheses, and we next discuss other results related to this. We first look at the number of false action identifications; i.e., the number of times the system did not correctly identify the action intended by the user. With the conventional method, the number of false identifications is 0 because this method does not need action identification. The mean number of false identifications in tasks 1 and 2 was 4.7 with the single-SA method and 10.0 with the multi-SA method. We attribute false identifications to two causes. First, the participants were not accustomed to using the CS Stylus. The questionnaire results indicate the participants felt uncertain about rolling. Although rolling is a motion the user can easily perform while holding a stylus, users do not normally roll a pen while using it. However, we expect this to become less of a problem as users become used to working with the CS Stylus. Second, the action identification accuracy was low. In this research, we used DP matching. However, we could not set a large number of elements in a pattern. The Cookie sensor resolution is 10 Hz and the time needed for each action was no more than 500 ms. Thus, we could only use about five elements for matching, and this limited number of elements could account for the low matching accuracy. Our focus was on developing interaction techniques, though, rather than on high accuracy. While we have not solved this problem, improved sensor performance in the future should make it less of a concern.

Table 2 Questionnaire results Question 1 asked which method would best allow participants to concentrate on their work. Question 2 asked which method would best allow participants to concentrate on their work if action identification accuracy was high.

Method	Question 1	Question 2
Conventional method	1	0
Single-SA method	4	5
Multi-SA method	0	0

We will now move on to the time needed for a task when there is no delay caused by false identifications. In such a case, the two methods using the shortcut actions required less time than the conventional method. A t-test showed that the difference in the required time between the conventional method and the single-SA method was significant for task 2 ($p = 0.106$). This result suggests use of the shortcut actions improves the operation speed. While our main goal in this work was to improve the usability of systems that have a stylus as an input device, improving the operation speed is expected in the future.

The questionnaire asked how well the participants could concentrate on their work when using the different methods (Questionnaire results are shown in Table 2). Question 1 asked which method allowed the participant to best concentrate when performing a task. Four participants chose the single-SA method and one chose the conventional method. The participant who preferred the conventional method said he could not concentrate because of the poor action identification accuracy with the two methods using shortcut actions. Question 2 asked which method best allowed the participant to concentrate if the action identification accuracy was high. All participants chose the single-SA method. One participant commented that the multi-SA method made concentration difficult because he was not used to the CS Stylus.

Thus, the questionnaire results suggest that users are more able to concentrate on their work more when they use the shortcut actions rather than the conventional method.

As part of our future work, we will conduct a more extensive and rigorous evaluation of our techniques.

7. Conclusions

The primary limitation of existing pen-based input interfaces is that a system using a stylus as an input device is diverging from the usage expectations on which a WIMP interface is based. To solve this problem, we propose the use of shortcut actions. These are interaction techniques based on hand motions made in the air. To implement the shortcut actions, we developed a Context Sensitive Stylus. The Context Sensitive Stylus lets a user control the system through hand motions in the air. We introduced Oh! Stylus Draw and Oh! Stylus Scroll as application programs in which the shortcut actions could be applied.

In our evaluation, we tested two hypotheses: that a user would make fewer operational failures when using the shortcut actions than with the conventional menu operations, and that the user would make fewer ineffectual motions when using the shortcut actions than with the conventional menu operations. Our results indicate that users can concentrate better on their work when using the shortcut actions. In addition, if the action identification accuracy is improved, the operation speed when shortcut actions are used is likely to be higher than with the conventional method.

We thus confirmed that hand motions made in the air can be usefully applied through new interaction techniques. This promises to make menu operations easier and enable users to better concentrate on their work.

Acknowledgments

We thank Nokia Research Center Tokyo for lending us the Cookie sensors.

References

[1] D. Hopkins, "The design and implementation of pie menus," Dr. Dobb's J., vol.16, pp.16–26, 1991.
[2] J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman, "An empirical comparison of pie vs. linear menus," Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'88), pp.95–100, 1988.
[3] G. Kurtenbach and W. Buxton, "The limits of expert performance using hierarchic marking menus," Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'93), pp.482–487, 1993.
[4] S. Pook, E. Lecolinet, G. Vaysseix, and E. Barillot, "Control menus: Excecution and control in a single interactor," Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'00), pp.263–264, 2000.
[5] F. Guimbretiére and T. Winograd, "FlowMenu: Combining command, text, and data entry," Proc. 13th Annual ACM Symposium on User Interface Software and Technology (UIST'00), pp.213–216, 2000.
[6] K. Perlin, "Quikwriting: Continuous stylus-based text entry," Proc. 11th Annual ACM Symposium on User Interface Software and Technology (UIST'98), pp.215–216, 1998.
[7] J. Accot and S. Zhai, "More than dotting the i's — Foundations for crossing-based interfaces," Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'02), pp.73–80, 2002.
[8] G. Apitz and F. Guimbretiére, "CrossY: A crossing-based drawing application," Proc. 17th Annual ACM Symposium on User Interface Software and Technology (UIST'04), pp.3–12, 2004.
[9] G.M. Smith and M.C. Schraefel, "The radial scroll tool: Scrolling support for stylus- or touch-based document navigation," Proc. 17th Annual ACM Symposium on User Interface Software and Technology (UIST'04), pp.53–56, 2004.
[10] K. Hinckley, P. Baudisch, G. Ramos, and F. Guimbretiére, "Design and analysis of delimiters for selection-action pen gesture phrases in scriboli," Proc. SIGCHI Conference on Human Factors in Computing Systems (CHI'05), pp.451–460, 2005.
[11] I. Siio and H. Tsujita, "Mobile interaction using paperweight metaphor," Proc. 19th Annual ACM Symposium on User Interface Software and Technology (UIST'06), pp.111–114, 2006.

Yu Suzuki received his bachelor's degree in Information Communication Engineering from Kyoto Sangyo University in 2006. He received his master's degree in Engineering from the University of Tsukuba in 2008. He is currently a student in the Department of Computer Science, University of Tsukuba, Japan. His research interests include human-computer interaction, pen-based interfaces, ubiquitous computing and real world oriented interfaces. He is a member of the ACM, IPSJ, and HIS.

Kazuo Misue received a BSc and a MSc from Tokyo University of Science in 1984 and 1986. He received a PhD in engineering from the University of Tokyo in 1997. He is an Associate Professor of Department of Computer Science, University of Tsukuba, Japan. His research interests include human-computer interaction, information visualization, automatic graph drawing, and supporting human creative activities. He is a member of the ACM, IPSJ, JSAI and JSSST.

Jiro Tanaka is a professor in the Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba. His research interests include visual programming, interactive programming, computer-human interaction and software engineering. He received a BSc and a MSc from the University of Tokyo in 1975 and 1977. He received a PhD in computer science from the University of Utah in 1984. He is a member of the ACM, the IEEE Computer Society and IPSJ.