2010

Doctorate thesis

Natural and Smooth Pen-based Interaction Utilizing Multiple Pen Input Channels

1108008 Chuanyi LIU

Advisor Xiangshi REN

February 2010

Course of Information Systems Engineering
Graduate School of Engineering, Kochi University of Technology

Abstract

Natural and Smooth Pen-based Interaction Utilizing Multiple Pen Input Channels

Chuanyi LIU

Pen devices such as PDAs and Tablet PCs, are being used more and more widely because of the naturalness of pen input. Pen-based interaction is an important study field in human computer interfaces. Most pen-based computing devices support multiple input channels. These input channels enable pen-based devices to possess the potential of supporting natural and smooth operation. Natural interfaces allow users comfortably and easily interact with computing devices. Smooth operation has the potential of enhancing users' performance of performing interaction tasks on pen-based devices. WIMP (Windows, Icons, Menus and Pointing devices) interfaces are prevalent for decades in GUIs. However, WIMP interfaces are not completely suitable for pen-based systems, since WIMP interfaces are initially designed for mice and keyboards. This thesis focuses on exploring a pen suitable operation semantic and the related interaction techniques. A seamless and continuous operation semantic is proposed to enhance the traditional WIMP interfaces for pen-based applications. The proposed operation semantic is suitable for a common computer task, which can be typically divided into three phases, i.e. object selection, command selection and property setting phases. With the proposed operation semantic, a whole computer task from object selection to property setting can be performed in one stroke of a pen. An experiment was done to compare the proposed operation semantic and the standard operation methods in MS-Word 2007 in both speed and accuracy. Another experiment was done to explore the suitable techniques of switch from selection (object selection) to action (command selection and property setting). In the experiment, three *pen-tip-originated* mode switching was introduced and compared with other three traditional mode switching techniques. In the proposed operation semantic, multiple pen input parameters need to be combined in performing a computer task. The utmost number and combination of pen input parameters in a pen-based application were studied with a prototype system. To evaluate the proposed operation semantic in application systems, an application paradigm of fluid and natural pen-based interaction techniques by utilizing multiple input parameters was studied. A drawing prototype system is introduced as a pen suitable application with seamless and continuous operation semantic. In the outline of this dissertation, I introduce my research background knowledge, motivation, dissertation structure and brief introductions of each part of the dissertation.

key words Pen-based system, Pen-based interaction, Selection-action pattern, Pressure-based crossing selection, Pressure, Twist angle, Tilt angle, Azimuth, Seamless, Focused attention, Multiple parameters, Continuous interaction, Mode switching.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background Knowledge

Electronic pens possess the physical properties (e.g. barrels, tips and dimensions) of the traditional drawing or writing pens and extend some electronic properties (e.g. stroke, pressure, tilt angle, twist angle and azimuth inputs) from the traditional drawing or writing pens. Human beings have used various pens (e.g. Chinese writing brushes, quills, fountain pens, ball pens, pencils) for thousands of years. Therefore, the traditional drawing or writing pens are natural tools for human beings. Electronic pens have the potential in natural human-computer interaction, since they inherit some physical properties from the traditional drawing or writing pens.

Pen-based devices such as PDAs and Tablet PCs, are being used more and more widely because of the naturalness of pen input (see Fig. 1.1). However, current operating systems (OS) and applications for pen-based devices continue to reflect operating styles that were designed for mice and they generally do not reflect or exploit the unique characteristics of pens. Many studies on pen-based interfaces have been exploited in recent decades for pen suitable interact techniques and methodologies. Ivan Sutherland [1] first demonstrated his pen-based interaction techniques in *Sketchpad*, where visible objects on the screen are directly manipulated with a light-pen, early in 1970s. *Sketchpad* supported the manipulation of objects using a light-pen, including grabbing objects, moving them, changing size, and using constraints. It contained the seeds of myriad important interface ideas. The system was built at Lincoln Labs with support from the Air Force and NSF. Early attempts to use pen input were limited by the touch screen technologies.

Fig. 1.1 Various pen-based devices.

In the last two decades, pen-based computing devices and the studies were pushed forward greatly. Nowadays, commercial electronic pens commonly possess multiple input parameters (e.g. stroke, pressure, tilt angle, twist angle and azimuth, Fig. 1.2). The utilization of pen input parameters can widen the human-computer interaction bandwidth. Therefore, many researchers foucus their attention on pen-based devices. Some of the studies have explored the human ability to apply pen input parameters in human-computer interaction. Others have focused on pen parameter-enabled applications or techniques. These recent studies are summarized below according to the parameters explored.

Up to now, pressure parameter has been explored extensively. Herot and Weinzapfel [2] first studied the human capability of the finger to apply pressure and torque to a computer screen early in 1978. Buxton [3] investigated the use of touch-sensitive technologies and the potential for interaction that they suggested in 1985. Ramos et al. [4] explored the human ability to vary pen-tip pressure as an additional channel of access to information in 2004. Ramos and Balakrishnan introduced pressure marks [5] and Zliding [6]. Pressure marks can encode selection-action patterns in a concurrent, parallel interaction. In pen strokes, variations in pressure make it possible to indicate both a selection and an action simultaneously. Zliding explores integrated panning and zooming by concurrently controlling input pressure while sliding in X-Y space. Li et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interactions in 2005. Harada et al. [8] presented a set of interaction techniques that leveraged the combination of human voice and pen pressure and position input when performing both creative 2D drawing and object manipulation tasks in 2007.

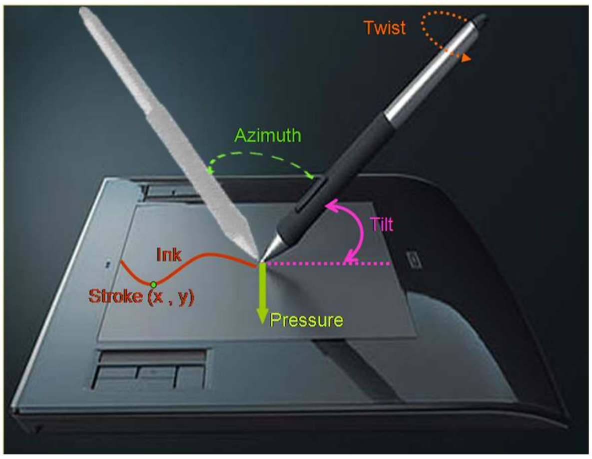


Fig. 1.2 Pen input parameters.

Input angles (i.e. tilt angle, twist angle and azimuth) are often used as UI clues for natural and intuitive interaction. Balakrishnan et al. [9] introduced the Rockin'Mouse in 1997. The Rockin'Mouse is a promising device for both 2D and 3D interaction that uses tilt input to facilitate 3D manipulation on a plane. Tian et al. [10] explored the Tilt Menu in 2008. The Tilt Menu is implemented by using 3D orientation information of pen devices for better extending selection capabilities of pen-based interfaces. Some other studies such as TiltType [11] and TiltText [12] focus on using the tilt information of mobile phones to affect text entry tasks in mobile devices. Bi et al. [13] explored rolling angle on general human being control ability in 2008. They suggested that both rolling amplitude and speed should be taken into account for rolling-based interact techniques.

As for sketch-based techniques, Apitz and Guimbretire [14] presented their CrossY, in which pen stroke did all the drawing operations in 2004. Davis et al. [15] introduced their SketchWizard in 2007, which is about wizard of Oz prototyping of pen-based user interfaces.

1.2 Research Motivation

With the development of computers, computing devices have gone beyond a tool for supporting knowledge workers in an office or science calculation in a laboratory. As they have been becoming smaller and still less expensive, they become ubiquitous in people's daily life. Amoung the ubiquitous computing devices, pen-based digital, intelligent products are taking a very important role in supporting people's study, work and life. Applications running on these pen-based devices are mostly built up from the traditional interfaces, i.e. WIMP GUIs. WIMP GUIs (graphical user interfaces based on windows, icons, menus, and a pointing device, typically a mouse) have been the predominant interfaces for decades. However, WIMP GUIs are not completely suitable for pen-based devices, since they were initially designed for mice and keyboards and do not reflect the unique characteristics of pen-based devices. Because a pen-based device inherits some physical properties from a writing or drawing pen, which is the most natural tool for people, it has the potential to make human-computer interaction be more natural and intuitive than the traditional way. Meanwhile, natural user interface is an important study field in human computer interaction. Natural interfaces have the potential to reduce users' learning time and effort, and lower the learning difficulty for novices. Fluid operation semantics have the potential to enhance users' operation efficiency. However, few studies have explored the simultaneous utilization of more than one pen input parameter with the intention of making operation more fluid and natural than traditional interfaces. The motivation of this dissertation is to explore natural and smooth pen-based operation semantic with multi-parameter of pen input and the related interaction techniques and paradigms.

1.3 Dissertation Structure

The dissertation logical structure is shown in Fig. 1.3. In brief, this dissertation exploits natural and smooth pen-based operation semantics and the supporting UI designs. A seamless and continuous operation (SC) semantic and UI design have been evaluated in a drawing prototype. The corresponding operation in MS-Word 2007 served as a baseline. From the experiment, we found out that the SC operation semantic outperformed MS Word in both operation speed and cursor footprint length. In the SC operation semantic, we divided the operation into three phases, i.e. object selection, command selection and object property setting phases. Each operation phase is corresponding to an operation mode. How to switch smoothly between different operation modes (e.g. ink and gesture, selection and action) is an open question. To compare and

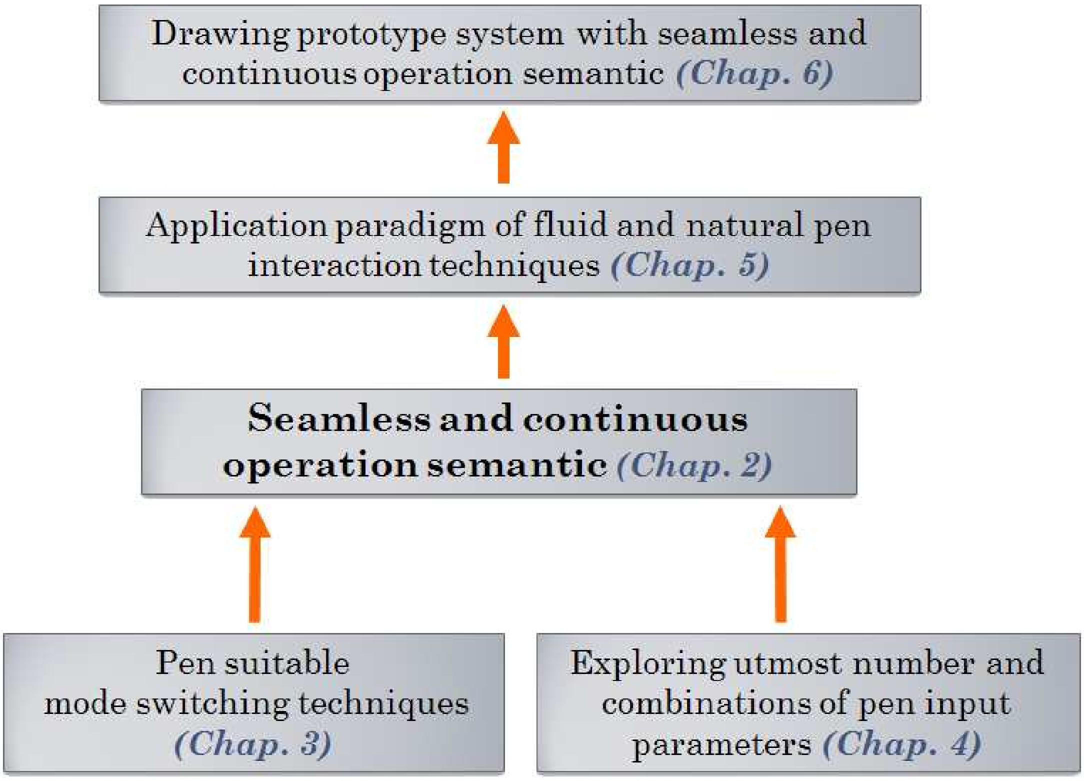


Fig. 1.3 The dissertation logical structure

evaluate different pen suitable mode switching techniques, we did another experiment. Some studies have exploited human beings' control ability on a single input channel of a pen. However, no such studies have been performed to explore the ability of human beings on manipulating more than one input parameters of a pen. We conducted a study to explore human beings' ability on manipulating multiple input parameters of a pen in 3D object operation. From the study, we established that an utmost parameter number of pen input that can be simultaneously employed in one application depends on computer tasks and the mapping between functions and input parameters. We illustrate an application level paradigm of natural and smooth pen-based operation by implementing a 2D and 3D drawing prototype system. Finally, a drawing prototype system is introduced as a pen suitable application with seamless and continuous operation semantic.

1.4 Brief Introductions of Each Part of the Dissertation

Each part of the dissertation is introduced briefly in the following subsections.

1.4.1 A seamless and continuous operation semantic with multiple input channels

Continuous interaction in pen-based systems is essential for optimum efficiency while a seamless mode switch can effectively enhance the fluency of interactions. Any interface that incorporates the advantages of seamless and continuous operation has the potential to enhance the users' operation efficiency and concentration. In this chapter, we present a seamless and continuous (SC) operation semantic and a UI design, which supports the SC operation, based on the pen's multiple-input parameters. A novel UI design evaluation factor, focused attention, which is a promising degree that takes into account the users' ability to focus on the targets themselves, and its measurement criterion, cursor footprint length (CFL), are proposed in this chapter. A prototype that can support our SC operation semantic and UI design was implemented to compare performance of SC operation with the MS Word 2007 system. The subjects were requested to select target components, activate the command menus and color the targets with a given flowchart in the two systems. The results show that the SC operation semantic outperformed MS Word in both operation speed and CFL.

1.4.2 Study on pen suitable mode switching techniques

In pen based interfaces, fluid and continuous interaction is a very important feature. How to switch smoothly between different operation modes (e.g. ink and gesture, selection and action) is an open question. For pen-based UI designers, especially for these who are seeking fluid multi-mode operation in pen-based interfaces, it is necessary to select a suitable mode switching technique for the UI design. Mode switching techniques are used widely on some pen-based devices, e.g. PDAs and Tablet PCs. Up to now, some mode switching techniques have been proposed by researchers, such as Non-Preferred Hand and Barrel Button Mode Switch. However, current switching techniques usually impose intervals (called switching intervals hereafter) on operational sequences, e.g. Timeout imposes an extra time interval on operational sequences while Non-Preferred Hand and Barrel Button Mode Switch impose not only time but also space intervals on operational sequences. The intervals not only take extra operational time but also distract the user's attention from the targets. Therefore, our basic motivation is to explore some pen-tip-originated mode switching techniques, which allow fluid and continuous switch by eliminating or reducing switching intervals. Fortunately, electronic pens commonly possess multiple input parameters (e.g. stroke, pressure, azimuth and tilt angle). However there is no study which comprehensively compares different mode switching techniques with multiple pen input parameters. Thus, this study comprehensively investigates mode switching techniques that utilize pen input parameters, so as to obtain some UI design guidelines for pen-based UI designers. Three pen-tip-originated mode switching techniques (i.e. Pressure, Tilt and Azimuth Mode Switch) have been designed for fluid pen-based operation. To perform these mode switching techniques, the pen tip does not need to be moved away from the targets, and no time interval is needed. The key feature of pen-tip-originated mode switching is that it has the potential to make pen-based operations much more fluid.

1.4.3 Exploring utmost number and combination of pen input parameters in 3D operation

3D transformation is an important study topic in HCI field. Many researchers pay attention to how to manipulate 3D objects with 2D sketches. The exiting sketch-based systems allow a user to perform some complex 3D operations by using relatively simple 2D strokes. But these sketch-based operations are not natural or intuitive enough for most users; furthermore, a sketch-based system typically imposes a gesture learning process on the users. A digital pen typically posses multiple input channels, e.g. stroke, pressure, tilt angle, twist angle and azimuth. Being different from a mouse and keyboard, a digital pen can provide physical feedback to each of its input channels, e.g. a pen's barrel tilt angle to the screen is equivalent to its input tilt angle. This chapter explores the potential of natural and fluid 3D transformation with multiple input channels of a pen. Unlike mice or keyboards, pens connately possess physical feedback to their input channels. Therefore, pens outperform other indirect input devices in some natural and intuitive operations. This chapter proposes four kinds of operations regarding 3D objects, i.e. selecting 3D objects, rotating 3D objects, scaling 3D objects, translating 3D objects. A prototype system was implemented to test the proposed 3D operations. From an informal user study, we find out that the subjects performed better with the prototype system than with the standard Maya widgets. We also find out that the utmost number of input channels that a user can concurrently manipulate depends on the mapping between pen input channels and the controlled parameters.

1.4.4 An application paradigm of fluid and natural pen interaction techniques by utilizing multiple input parameters

Pen-based interfaces have been explored extensively in recent years. The utilization of pen input parameters can widen the human-computer interaction bandwidth. Therefore, various studies have been performed on these input parameters. Some studies have explored the human ability to apply pen input parameters in human computer interaction. Others have focused on pen parameter-enabled applications or techniques. However, few studies have explored the simultaneous utilization of more than one pen input parameter with the intention of making operation more fluid and natural than traditional interfaces. Commonly, a computer task is performed through three phases in the following order: object selection, command selection and object manipulation. Switching between these phases can be performed by tapping on menu items or by pressing down predefined hot keys in the traditional WIMP (Windows, Icons, Menus and Pointing devices) interfaces. However, in most pen-based interaction environments, the hot keys are not available and it is tiring to move the pen tip repeatedly over long distances. Therefore, it is worthwhile to enhance the continuity of pen-based operations. Continuous interaction is a very important feature in pen-based user interfaces. Liu and Ren have comprehensively evaluated six pen-suitable mode switching techniques and proven that smooth operation with a pen-suitable switching mode is more efficient than the traditional interfaces used in pen-based systems. Through the studies of Liu and Ren, we found that pen input parameters have the potential to make operation more natural and intuitive than traditional interfaces. Based on these considerations, we designed four techniques in 2D and 3D drawing prototype system. The techniques integrate and exploit multiple pen input parameters and allow users to operate fluidly and naturally throughout the whole process from object selection to object manipulation.

1.4.5 A drawing prototype system with seamless and continuous operation semantic

Current studies on the utilization of pen input parameters are typically focused on the human ability to control pen parameters or on novel techniques which exploit a certain parameter (e.g., pressure or tilt angle). In this chapter, we present a versatile 2D and 3D drawing system (called PenOpera) which supports seamless and continuous operation semantic by utilizing more than one pen input parameter. Some pen-suitable techniques and possible guidelines on how to design and implement multi-parameter pen input systems are also introduced in this part.

Chapter 2

A seamless and continuous operation semantic with multiple input channels

Continuous interaction in pen-based systems is essential for optimum efficiency while a seamless mode switch can effectively enhance the fluency of interactions. Any interface that incorporates the advantages of seamless and continuous operation has the potential to enhance the users' operation efficiency and concentration. Continuous interaction in pen-based systems is essential for optimum efficiency while a seamless mode switch can effectively enhance the fluency of interactions. Any interface that incorporates the advantages of seamless and continuous operation has the potential to enhance the users' operation efficiency and concentration. In this paper, we present a seamless and continuous (SC) operation semantic based on the pen's multiple-input parameters. A whole interaction task can be performed in one stroke using SC operation semantic. This paper proposes a novel UI design evaluation factor (called *focused attention*) which is a promising factor that takes into account the users' ability to focus on the targets themselves. The evaluation factor's measurement criterion, cursor footprint length (*CFL*), is proposed in this paper. A prototype that can support our SC operation semantic was implemented to compare performance of SC operation with the MS Word 2007 system. The subjects were requested to select target components, activate the command menus and color the targets with a given flowchart in the two systems. The results show that the SC operation semantic outperformed MS Word in both operation speed and *CFL*.

2.1 Introduction

Pen devices such as PDAs and Tablet PCs, have been used more and more widely because of the natural pen input. However, the current operation systems (OS) and applications for pen devices still remain the style of OS initially designed for Mice. There are various studies on exploring pen-suitable UI design. In these studies, how to improve the switch efficiency in selection-action patterns is an important research topic. Various techniques and paradigms on selection-action patterns have been presented lately (e.g., [5,16,17]). Most of these studies utilizing the same input channel for inking and gesturing. In some cases, it is rather difficult to eliminate the ambiguity of stroke recognition completely. And the use of these proposed techniques in pen-based systems is greatly limited for the lack of flexibility and ubiquity. On the other way, a commercial electronic pen commonly possesses multiple input channels. So our basic motivation is to find out an unambiguous and ubiquitously applicable method utilizing extra pen input channels with which users can perform selection-action patterns continuously, fluidly and unambiguously.

In this chapter, we present a pen-suitable operation paradigm, under which fluid and continuous operations and seamless switch between different types of operation become possible throughout a computer task. To evaluate the proposed methods, a drawing prototype system was implemented as a Java^{TM} program. And a comparative experiment was done to compare the operation paradigm and the corresponding ways in

MS Word 2007 system. In the experiment, the subjects were asked to select the target components of a given flowchart, activate the command menus and color the targets. The results show that the proposed operation methods outperform MS Word in both speed and CFL, despite a little higher error rate.

2.2 Related Work

In this section, we discuss related work regarding both the studies on pen input parameters and these on seamless and continuous operations in pen-based systems.

2.2.1 Previous Work on Pen Input Parameters

To date, there are many studies on the utilization of pen input parameters. These studies can be roughly divided into two categories. One category investigates the general human ability to control pen input parameters; the other category aims at enhancing performance of human and computer interaction by implementing novel applications or techniques which exploit particular input parameters.

Up to now, pressure parameter has been explored extensively. Herot and Weinzapfel [2] studied the human capability of the finger to apply pressure and torque to a computer screen. Buxton [3] investigated the use of touch-sensitive technologies and the potential for interaction that they suggested. Ramos et al. [4] explored the human ability to vary pen-tip pressure as an additional channel of access to information. Ramos and Balakrishnan introduced *pressure marks* [5] and *Zliding* [6]. *Pressure marks* can encode selection-action patterns in a concurrent, parallel interaction. In pen strokes, variations in pressure make it possible to indicate both a selection and an action simultaneously. *Zliding* explores integrated panning and zooming by concurrently controlling input pressure while sliding in X-Y space. Li et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interactions. Harada et al. presented a set of interaction techniques that leveraged the combination of human voice and pen pressure and position input when performing both creative 2D drawing and object manipulation tasks [8].

Input angles (i.e. tilt angle, twist angle and azimuth) are often used as UI clues for natural and intuitive interaction. Balakrishnan et al. [9] introduced the *Rockin'Mouse*. The *Rockin'Mouse* is a promising device for both 2D and 3D interaction that uses tilt input to facilitate 3D manipulation on a plane. Tian et al. explored the *Tilt Menu* [10]. The *Tilt Menu* is implemented by using 3D orientation information of pen devices for better extending selection capabilities of pen-based interfaces. Some other studies such as *TiltType* [11] and *TiltText* [12] focus on using the tilt information of mobile phones to affect text entry tasks in mobile devices. Bi et al. [13] explored rolling angle on general human being control ability. They suggested that both rolling amplitude and speed should be taken into account for rolling-based interact techniques.

As for sketch-based techniques, Davis et al. introduced their *SketchWizard* [15], which is about wizard of Oz prototyping of pen-based user interfaces. Apitz and Guimbretire [14] presented their *CrossY*, in which pen stroke did all the drawing operations.

2.2.2 Previous Work on Seamless and Continuous Operations

Hinckley et al. presented their pigtail delimiters [17], with which selection-action patterns can be performed in one continuous fluid stroke. A pigtail is created explicitly by intersecting one stroke itself and an action is specified or an object manipulated by the stroke's direction. Pigtails provide a way to integrate an explicit command invocation in a fluid stroke following the selection specification. But it is rather difficult to manipulate multiple targets in an irregular layout, since the targets are selected by a lasso. Furthermore, there is ambiguity between pigtail delimiters and freeform drawings.

Baudisch et al. introduced *marquee menus* [18], which are a technique where the selection-action pattern occurs concurrently. The *marquee menu*'s selection is specified by a rectangular area, which is defined by the start and the end points of a straight stroke; its action is determined by one of four movement directions of the stroke. *Marquee menus* are sensitive to both a mark's point of origin and direction while providing a compact interaction phase. The technique is promising for web browsing in small screens. But it has not been elaborated to show whether and how this technique scales for non-straight strokes with arbitrary orientations. Regardless of these considerations, this kind of technique is not suitable for multiple targets in an irregular layout and ambiguity between gesture strokes and freeform drawings limits its practical applicability in other scenarios.

Ramos and Balakrishnan introduced their *pressure marks* [5], where variations in pressure are used as metaphors for actions. The marks of pressure variation are integrated into selection strokes, and then the selection-action patterns can be performed concurrently and seamlessly. However, there are some limitations with *pressure marks'* variation, e.g. once the user begins to slide a pen slightly then the HL (pressure variation signature, high-low, defined in the original) or HH (high-high) pressure mark may not appear in the following stroke. Furthermore, the number of simple pressure marks is limited, and compound marks are difficult to memorize and control. Again, this kind of technique is only useful for targets arranged in a regular layout.

2.3 The Proposed Operation Semantic

From the previous work, we can see that selection-action patterns have been explored extensively, but the uses of these techniques are limited to specific and rather narrow scenarios. Furthermore, it is rather difficult to eliminate the ambiguity between gesture strokes and freeform drawings, since both are based on the same input channel. In this chapter, we present an operation semantic with extra input channels, which allows fluid target selection and continuous and seamless switching from selection to action. Commonly, a computer task includes three phases, i.e. object selection, command selection and object property setting phases. Under the operation semantic, a computer task can be performed in one continuous and fluid stroke. In the target selection phase, users can string and select targets with a pen stroke. Pen pressure input is used as a delimiter to distinguish between selection strokes and freeform drawings. When all the targets have been selected by a pen stroke, users can activate a pie menu by rolling the pen. If the rolling angle and speed exceed the respective thresholds, the pie menu will be activated and displayed with its center under the cursor. The user then slides the pen tip, and an action is performed when the pen tip crosses a menu item. Throughout the whole process, the pen tip need not be lifted from the screen. All the operations can be performed in one continuous and fluid stroke. The design of the three phases with the operation semantic will be introduced in detail in the next section.

2.3.1 Target selection

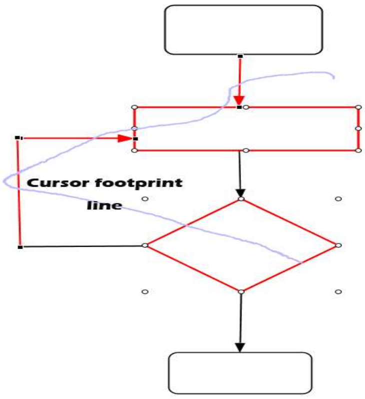
As suggested by [14, 19, 20] crossing could serve as a potential substitute for the classic point-and-click interaction in UI design. Crossing performs better than pointing-and-clicking in some application scenarios, especially for pen-based input devices. In the prototype system, we present a pressure-based line-string selection method. While a pen is slid across the screen, the objects strung together by the stroke are selected when the pen input pressure exceeds a given threshold. Yin and Ren [21] investigated three novel line-based techniques for multi-target selection. Out of the three techniques, *Line-string* employs a line stroke to "string" and select targets. The technique is promising under some conditions. However, it is difficult to select scattered targets in an irregular and dense layout since all targets that are crossed are “strung together” and are thereby selected. In the next sections, we introduce a more flexible pressure-based crossing selection technique, by which an object can easily be “jumped over” by a pen stroke. In traditional UI design, to select scattered targets in an irregular layout, a point-tap-lift-point movement cycle of a pen tip is necessary while a specific key (e.g. *Ctrl* or *Shift*) is being pressed. In our implementation of the prototype, all the targets can be selected in a fluid and continuous stroke.

Pressure Coupling Normal Stroke and Line-string Selection

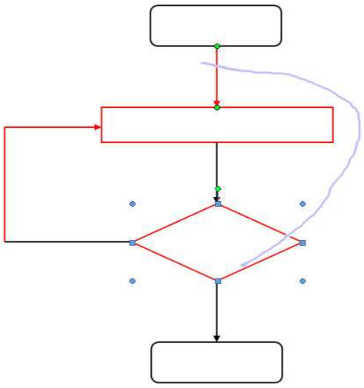
In the prototype system, pressure is used as a delimiter to couple normal stroke and line-string selection. A pilot study was done to determine the right pressure spectrum for normal stroke and line-string selection. In our experiment, 12 participants were asked to draw with light pressure, normal pressure and heavier pressure alternately on a WACOM tablet-display, which has 1024 levels of pressure. The results showed a statistically significant difference on the *maximum average pressure* of a stroke between the light, the normal and the heavier pressure conditions. In our implementation, the heavy spectrum of pressure was employed for line-string selection, and the normal spectrum for a drawn line; for low end pressure, the spectrum is more difficult to control [6], therefore, it was omitted from the technique design.

Object Selection

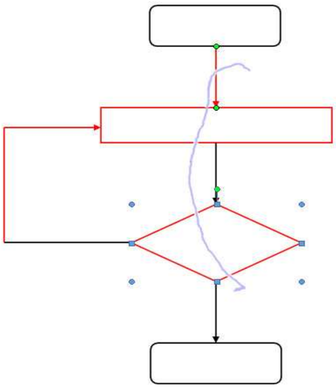
The user strokes the pen starting from a *blank area*, where there is no object. If the pressure input exceeds the specified threshold, the stroke will affect pressure-based line-string selection; otherwise it will produce a normal stroke. Under this selection mode, the user only needs to stroke the pen on a screen and all the objects connected



(a) String & select objects with one stroke.



(b) Steer clear of an object.



(c) Ignore an object crossed by the stroke.

Fig. 2.1 Pressure-based line-string selection (the objects with *sizing handles* are selected).

by the pen will be selected (see Fig. 2.1a). A blue footprint line following the path of the pen is used as visual feedback for the selection state. If there are some objects that the user does not want to select in the path of the selection stroke, s/he can steer clear of them (Fig. 2.1b) or reduce the pressure on the pen to below the threshold without lifting the pen tip from the screen, until the blue footprint line disappears. Then the figure will be crossed by the stroke without being selected (Fig. 2.1c).

Undoing Selection

The user can stroke the pen back and across the footprint line on a selected object to undo its selection. If the user lifts the pen and taps in a *blank area*, selection of all the items will be canceled.

2.3.2 Activating the Menu

Although, there are various studies on selection-action patterns, most of these techniques use the same pen input channel for both command gesture and freeform drawings. It is rather difficult to eliminate the recognition ambiguity completely. In the following section, we introduce a smooth and unambiguous technique for switching smoothly between selection and action by introducing extra pen input channels.

Li et al. [7] investigated five different mode switching techniques in pen-based UI design. They suggested that *non-preferred hand* mode switch is the most promising. In their experiment, a physical button mounted at the top-left corner of a Tablet PC screen was employed as a *mode switching button.* In their study, they did not explore pen angle related input channels, e.g. tilt angle, azimuth or twist angle. To determine the most suitable extra input channel that can serve as a switching trigger to activate the menu, we performed a pilot study to investigate all the possible input channels of a pen for mode switching techniques. After the first block of tests using the non-preferred hand section of the trials, we noticed that the subjects tended to keep one finger of their non-preferred hands on the *mode switching button.* Taking into account the practical application scenarios, it is impossible to keep the non-preferred hand on a specific button all the time. And under most conditions, the keyboard or such a button is not available in a pen-based system. Taking all the above factors into account, the pilot study indicated that angles were more suitable for pen-based interfaces than other channels in mode switching. Based on the pilot study, we used twist angle as an extra input channel to activate the menu in our implementation.

Bi et al. presented their study on rolling (twist) angle for pen input [13]. They suggested that the rolling can be identified as incidental if the rolling speed of a data event is between -30$^\circ/s$ and 30 $^\circ/s$ or the rolling angle is between -10° and 10°. And -90° to 90° can be exploited as the usable rolling range. Based on their study results, rolling is employed in our experiment design to activate the pie menu if the rolling speed exceeds the range of [-50$^\circ/s$, 50 $^\circ/s$], and rolling angle exceeds [-50°, 50°]. After selecting all the targets, the user intentionally rolls the pen. If the rolling angle and speed exceed the specific thresholds, the pie menu will be activated and displayed with its center under the cursor.

2.3.3 Performing an Action

When the pie menu is activated, the user slides the pen tip across a menu item and the corresponding action is performed.

2.4 Experiment

To evaluate the performance of the SC operation semantic, a quantitative experiment was conducted. The corresponding operation in MS Word 2007 served as a baseline.

2.4.1 Apparatus

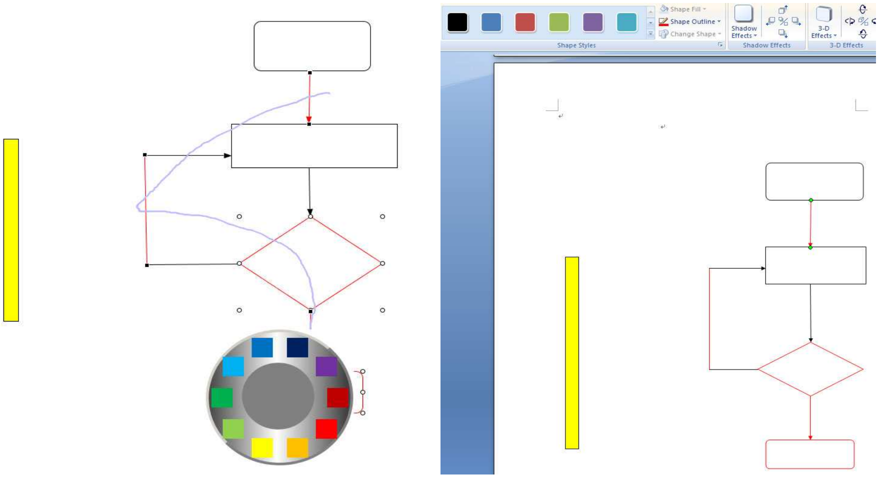
The hardware used in the experiment was a WACOM Cintiq 21UX flat panel LCD graphics display tablet with a resolution of 1,600 × 1,200 pixels (1 pixel= 0.297mm), using a wireless pen with a pressure, tilt angle, azimuth and twist angle sensitive isometric tip (the width of the pen-tip is 1.76mm). It reports 1024 levels (ranging from 0 to 1,023, the minimum unit is 1) of pressure and 360°(ranging from 0° to 359°, the minimum unit is 1°) of twist angle. The experimental program was implemented with Java^{TM} 6.0 running on a 3.2 GHz P4 PC with the Windows XP SP2 Professional operating system.

2.4.2 Participants

Six participants (two female and four male, ranging in age from 27 to 36 years old) were all volunteers from the local university community. All of them were right-handed. They were all regular computer users and expert in MS Word 2007, but none of them had previous experience using our drawing prototype system. One of them had two years experience using a digital pen and the other five have no such experience.

2.4.3 Task

In the experiment, the subjects were asked to perform with both types of interface (SC operation UI and Word operation UI). For each trial in both types of interface, the subjects were given a flowchart (Fig. 2.2) composed of 10 components. Five out of the



(a) The experiment UI design of SC operation semantic.

(b) The experiment UI in Word 2007.

Fig. 2.2 The experiment UI design.

10 components were randomly chosen as targets (displayed in red). The target color was shown as a rectangular bar to the left side of the flowchart. For each corresponding trial, the flowchart size, component number, location on the screen as well as the targets were kept the same in both kinds of interface. The subjects were requested to color the outlines of the target components with the given target color. Each trial includes three operation phases, i.e. object selection phase (called as *selection phase*), menu trigger phase (called as *trigger phase*) and object property setting phase (called as *setting phase*). With the proposed semantic, the subjects selected the targets using pressure-based line-string selection (this process is computed as its *selection phase*), rolled the pen to activate the pie menu (this process is computed as its *trigger phase*) and slid the pen tip across a menu item to color the targets (this process is computed as its *setting phase*). The experimental program recorded the time and accuracy of each phase, and the CFL per trial. With Word 2007, the subjects tapped the pen tip on each target to select it (with the *Shift* or *Ctrl* key being pressed, this process is computed as its

selection phase), moved the pen tip from the last target and pointed to the toolbar (this process is computed as its *trigger phase*) and tapped the pen tip to color the targets (this process is computed as its *setting phase*). Running in the background, the experimental program analyzed and recorded the time and accuracy of each phase, and the CFL per trial.

2.4.4 Procedure and Design

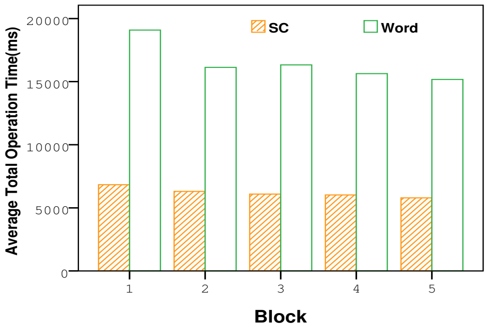
Each subject was requested to complete 5 blocks of trials. Each block consisted of 6 selection-action trails. The program recorded one *selection phase* error if any target component was omitted or any other non-target component was selected. One *trigger phase* error was recorded when the menu was activated accidentally in the SC trials, or when the wrong toolbar was tapped in Word 2007. If the target components were not colored with the target color, one *setting phase* error was recorded. The errors caused in the *selection phase*, *trigger phase* and *setting phase* are called *selection error*, *trigger error* and *setting error* respectively. The time elapsed in each phase, the *selection phase*, *trigger phase* and *setting phase*, was computed respectively as *selection time*, *trigger time* and *setting time.* When an error occurred, a beep was played to remind the participant to improve accuracy. The subjects were required to correctly perform each operation in the three phases. If an error occurred, the current phase's operation should be repeated until it was successful. The request aimed at enhancing the participants' learning effects of each phase's operation. And then the next phase's operation proceeded. When the whole three phases were finished, the current trial ended. The number of errors for each phase was included in the experimental result analysis, but the operation time for each phase was only calculated based on the time of correct completion. From the calculation on the experimental results, we can see the participants' learning effects for each experimental phase's operation. A within-subject design was used. The dimensions of all the flowcharts were displayed at a resolution of 297×622 pixels, so that the flowcharts could be displayed in a normal size and SC and Word operations could be fairly compared. In the SC operation UI, there are ten standard colors arranged in the same order as the standard color arrangement in the color toolbar of Word 2007. Before the task in Word 2007 began, the subjects were directed to activate the standard color toolbar as a *quick access toolbar*, and to scroll the Word page to keep the flowchart directly under the toolbar. The dependent variables were trial time, CFL, error rate and subjective preference. Prior to the study, the experimenter explained and demonstrated the task to the participants. The participants were requested to do the trials as quickly and accurately as possible. At the end of the experiment, the participants were instructed to give their subjective comments by completing a questionnaire, which consisted of four questions regarding “usability”, “fatigue”, “preference” and “focused attention” on a 1-to-7 scale (1=lowest preference, and 7 =highest preference).

2.4.5 Results

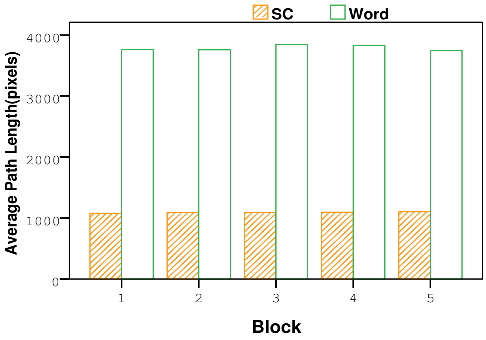
Trial time for each participant averaged thirty minutes. A RM-ANOVA (repeated measures analysis of variance) was used to analyze the performance in terms of operation time, CFL, accuracy and subjective preference.

Total Operation Time and CFL

There was a significant difference in the overall mean operation time (F(1, 5) =41.832, p=0.001) and CFL (F(1, 5) =50.394, p=0.001) between the two operation semantics. The overall mean operation time per trial was 6309.945 ms of SC operation, 16562.46 ms of operation in Word 2007. And the overall CFL per trial was 1084.172



(a) The average total operation time.



(b) The average CFL.

Fig. 2.3 The average total operation time and CFL.

pixels for SC operation, 3805.964 pixels for the operation in Word 2007. There were no main effects for blocks on overall mean operation time for either SC operation (F(4, 20) =1.718, p=0.186) nor Word operation (F(4, 20) =1.663, p=0.198). There were no main effects for blocks on CFL for either SC operation (F(4, 20) =0.247, p=0.908) or Word Operation(F(4, 20) =0.058, p=0.993). However, as Fig.2.3a illustrates, we observed a little improvement in speed. No significant effect was found for semantic∗block on overall mean time (F(4,20) = 1.029, p = 0.417) or overall CFL (F(4,20) = 0.094, p = 0.983), which indicated that the improvement in learning did not significantly affect relative performance on the two kinds of operation semantic.

Selection Time

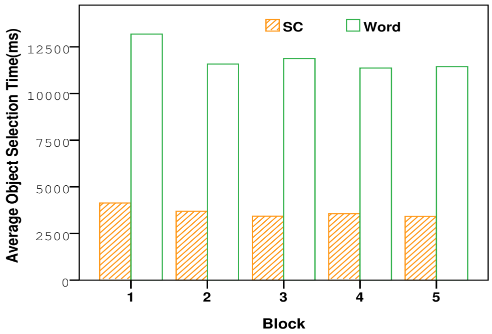


Fig. 2.4 The average selection time.

There was a significant difference in the overall mean selection time (F(1, 5) =88.284, $p < 0.0001$) between the two different kinds of operation semantics. The overall mean selection time per trial was 3700.110 ms for SC operation and 11955.45 ms for Word operation. There were no main effects for blocks for the operation of SC (F(4, 20) =1.164, p=0.356) or Word 2007 (F(4, 20) =0.625, p=0.650), on overall mean selection time. A small speed improvement in selection time for both SC and Word operation was also observed in Fig. 2.4. No significant effect was found for semantic∗block on the overall mean selection time (F(4,20) = 0.307, p = 0.870), which indicated the learning improvement did not significantly affect the relative performance of the two kinds of operation semantic on selection time.

Trigger Time

There was a significant difference (F(1, 5) =6.991, p=0.046) in the overall mean trigger time per trial between the two different kinds of operation semantics. The

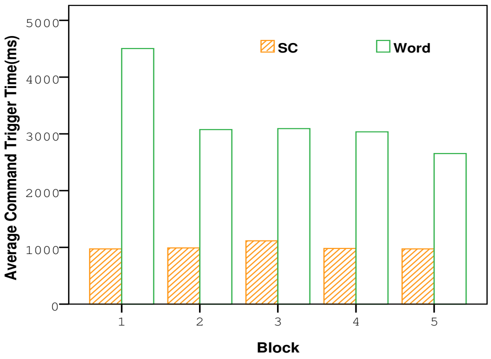


Fig. 2.5 The average trigger time.

overall mean trigger time per trial was 1030.373 ms for SC operation and 3297.632 ms for Word operation. There was no main effect for the operation of either SC (F(4, 20) =0.885, p=0.491) or Word (F(4, 20) =1.570, p=0.221), for blocks on overall mean trigger time. Fig.2.5 also illustrates a small improvement in selection time for both SC and Word operation. No significant effect was found for semantic∗block on overall mean trigger time (F(4,20) = 1.562, p = 0.223), which indicated learning improvement did not significantly affect the relative performance of the two kinds of operation semantic on trigger time.

Setting Time

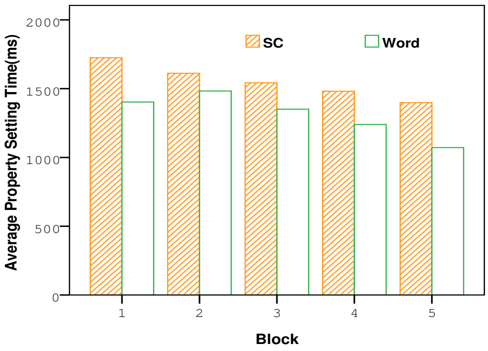


Fig. 2.6 The average setting time.

There was a significant difference (F(1, 5) =12.973, p=0.016) in the overall mean setting time per trial between the two different kinds of operation semantics. The overall mean setting time was 1579.463 ms for SC operation and 1309.381 ms for Word operation. For the operation of both SC(F(4, 20) =2.896, p=0.048) and Word (F(4, 20) =2.994, p=0.043), there were main effects for blocks on overall mean setting time. Fig.2.6 illustrates a little improvement in setting time for both SC and Word operation. No significant effect was found for semantic∗block on the overall mean trigger time (F(4, 20) = 0.417, p = 0.794), which indicated the learning improvement did not significantly affect the relative performance of the two kinds of operation semantic on setting time.

Errors

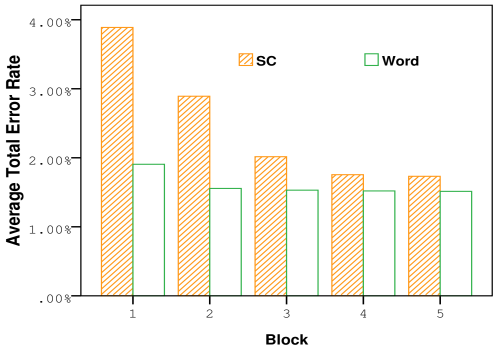


Fig. 2.7 The average total error rate.

The results showed a significant difference in the overall mean error rate (F(1, 5) =24.306, p=0.014) between the two different kinds of operation semantic. The overall mean error rate was 2.458% of SC operation and 1.606% of Word operation. There were main effects for blocks on overall mean errors for SC operation (F(4, 20) =6.332, p=0.002), but no main effects for blocks on overall mean errors for Word operation (F(4, 20) =1.010, p=0.043). As Fig. 2.7 illustrates, we observed a significant decrease in errors for SC and a marginal one in Word operation. Significant effects were found for semantics∗block on the overall mean errors (F(4, 20) = 5.588, p = 0.003), which indicated the learning improvement significantly affected the relative performance of the two kinds of operation semantic regarding errors.

Selection Error

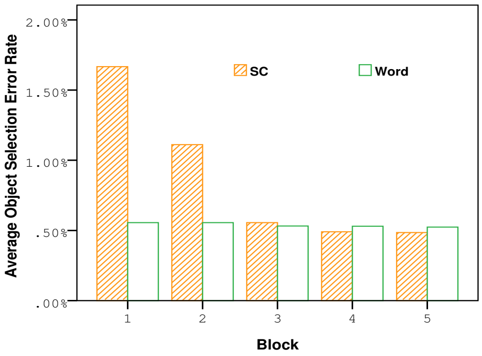


Fig. 2.8 The average selection error rate.

The experimental analysis reported a significant difference in the overall mean selection error rate (F(1, 5) =9.423, p=0.028) between the two different kinds of operation semantic. The overall mean selection error rate was 0.864% of SC operation, 0.540% of Word operation. There were main effects for blocks on overall mean selection error rate for SC operation (F(4, 20) =1.650, p=0.021), but no main effects for blocks on the overall mean selection error rate for Word operation (F(4, 20) =0.625, p=0.650). Fig. 2.8 illustrates a big improvement in selection errors for SC operation and a marginal improvement for Word operation. Significant effect was found for semantic$*$block on the overall mean trigger time (F(4, 20) = 5.058, p = 0.037), which indicated the learning improvement significantly affected the relative performance of the two kinds of operation semantic on selection errors.

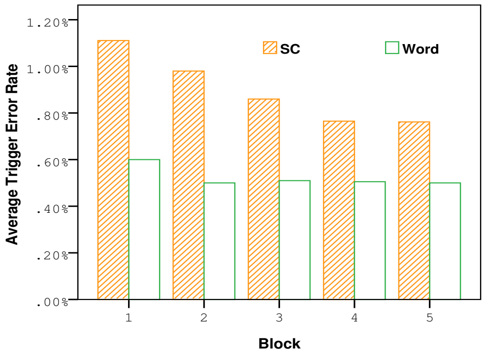


Fig. 2.9 The average trigger error rate.

Trigger Error

There was a significant difference in the overall mean trigger error rate (F(1, 5) =20.000, p=0.007) between the two different kinds of operation semantic. The overall mean trigger error rate was 0.896% for SC operation and 0.524% for Word. There were main effects for blocks on overall mean trigger error rate for SC operation (F(4, 20) =17.857, p=0.001), but no main effects for blocks on overall mean trigger error rate in Word 2007 (F(4, 20) =0.250, p=0.906). Fig. 2.9 illustrates a significant decrease in trigger error rate for SC operation and a little decrease for Word 2007. Significant effect was found for semantic∗block on the overall mean trigger time (F(4, 20) = 9.062, $p < 0.0001$), which indicated the learning improvement significantly affected the relative performance of the two kinds of operation semantic on trigger error rate.

Setting Error

There was no significant difference in the overall mean setting error rate (F(1, 5) =5.000, p=0.076) between the two operation semantics. The overall mean setting error rate was 0.7% for SC operation and 0.534% for Word operation. There were main effects for blocks on overall mean setting error rate for SC operation (F(4, 20) =5.000, p=0.006), but no main effects for operation in Word 2007 (F(4, 20) =2.742, p=0.057).

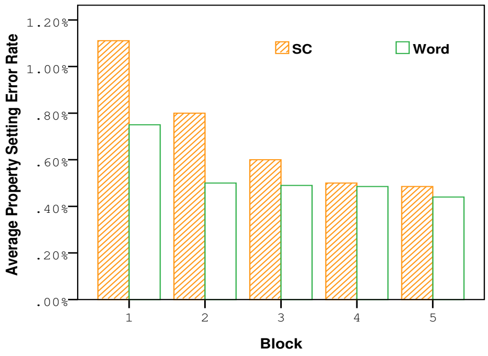


Fig. 2.10 The average setting error rate.

Fig. 2.10 illustrates the improvement in setting errors of both SC and Word operation. No significant effect was found for semantic∗block on the overall mean setting error rate (F(4, 20) = 2.619, p =0.066), which indicated the learning improvement did not significantly affect the relative performance of the two kinds of operation semantic on trigger errors.

Subjective Comments

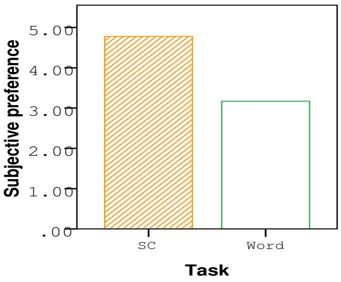


Fig. 2.11 The subjective preference.

Fig. 2.11 shows the subjective ratings for the two kinds of operation semantic. These ratings were based on the average value of the answers given by the subjects to the four questions. Significant main effects were observed between the two operation semantics (F(1, 5) =9.365, p=0.028). The average preference for SC operation semantic is 4.8, and for MS Word it is 3.2.

2.5 Discussion

Various contrastive techniques (e.g. *lassoing + pigtailing* [17], *pressure marks* [5]) were taken into account, but none of the presented techniques for pen-based systems is suitable for the wide range of common computer tasks. Although these techniques are rather promising for targets in a regular layout, in most application scenarios, the targets are scattered in irregular layouts. Under these conditions, the techniques fall short. Thus, MS Word was chosen as the baseline because it is the most widely used semantic paradigm. At the beginning of the experiment, we noticed that the participants stroked the pen rather cautiously and slowly to select the targets, rolled the pen nervously to activate the pie menu, and wanted to lift the pen tip to tap the target menu item. But after several trials, they stroked and rolled the pen fluidly and confidently. They commented that the SC operation was enjoyable; some of them said that performing the SC operation was like playing games.

Among the six participants, there was only one person who had experience using a digital pen. At the beginning of the experiment, the participants were given five minutes to master the manipulation of a digital pen. The results of the expert of digital pen use were compared with the means of those of the five novices on speed, accuracy and CFL. No significant difference was reported between the results of the expert and those of the other participants. This indicates that novices of digital pen use could master the SC operation rather quickly.

The results illustrate that the selection and trigger speed of SC operation are significantly faster than that of MS Word. But the setting speed of SC operation is a little slower than that of MS Word. This is probably due to the fact that part of the pie menu was visually occluded by the hand in SC operation in the setting phase. We observed that some of the participants tended to adjust their hands when crossing a target-menu item, others tended to hold the pen at a little higher position to facilitate crossing the menu item after the first block. Therefore, the experimental results reported that some subjects spent significantly more time in *setting phase* than other subjects. We believe that the experimental results would be more promising if all the subjects developed better pen holding postures. However, taking into account the potential for natural UI design with multiple input channels, we will find a way to overcome the problem of visual occlusion in direct pen-based devices.

From the experiment results, we also noted that error rates for the three phases of SC operation were much higher than for MS Word in the first two blocks. But the difference between SC and MS Word operation in error rates was not much different from the third block, except for the average trigger error rate. During the experiment, we observed that some participants tended to trigger the pie menu accidentally much more often than others. This is probably due to the participants' different ways of holding the pen.

Seamless and continuous (SC) operation and UI design have the potential to enhance the users' *"focused attention"*. *Focused attention* is a promising factor that takes into account the users' ability to focus on the targets themselves. From an investigation based on dozens of regular computer users, we found that most of them consider that *focused attention* is an important factor that affects their operational efficiency and sensation in UI design. Surprisingly, few studies have been performed to evaluate *focused attention* in the field of human-computer interaction. In traditional GUI's, users typically have to alternately focus their concentration on the relevant targets and commands. The repeated change in *focused attention* between targets and commands is sometimes tiresome and time-consuming, especially when the targets are small in size and situated in a complex scenario, e.g. one-pixel-wide line segments out of a complex flowchart in MS Word or small components from a complex context in an AutoCAD DWG file. In such contexts, it usually takes computer users time and effort to relocate the targets after they focus their attention on a menu or a toolbar. After the targets being selected, SC operation semantic enables activating a menu then and there. Therefore, SC operation semantic has the potential of enhancing *focused attention* throughout a whole computer task.

Fig. 2.3b illustrates that the CFL for SC is much shorter than for MS Word, which proves that the cursor needs to be moved less in SC operation then in MS Word. This further indicates that, in SC operation, the participants are able to focus their attention on the targets much better than with the standard interfaces.

Many researchers [14, 19, 20] suggested that point-and-click interfaces are not always suitable for pen-based systems. Furthermore, the repeated point-tap-lift-point cyclic movement of a pen tip tends to be tiresome. With the SC operation semantic, the whole computer task can be performed continuously, fluidly and unambiguously. We believe that our results have strong implications for smooth pen-based UI design. Firstly, the proposed SC operation semantics implied the potential of smooth pen-based operation with multiple input channels in a common application scenario. Secondly, the quantitative experiment proved that the operation semantic is rather promising beside the standard UI design for pen-based systems. Thirdly, the subjects commented that *focused attention* affected their operation efficiency and sensation significantly. However, few studies have been performed to evaluate *focused attention* in the field of human-computer interaction. A measurement criterion, *CFL*, for *focused attention* is proposed in the chapter.

2.6 Conclusion

In this chapter, we have presented an operation semantic that is suitable for seamless and continuous (SC) operation in pen-based systems. The results of SC operation are rather promising in both speed and cursor footprint length (*CFL*), and accuracy is not significantly different to standard operation in MS Word after the second block. A novel evaluation factor in the field of human-computer interaction, *focused attention*, and its measurement criterion, *CFL*, are proposed in this chapter.

Chapter 3

Study on pen suitable mode switching techniques

Mode switching is essential in pen-based systems, especially for multi-mode pen-based operation. In this study, we designed three pen suitable mode switching techniques (i.e. *Pressure*, *Tilt* and *Azimuth Mode Switch*), which utilize multiple pen parameters for pen input, and compared them with three traditional switching modes (i.e. *Timeout*, *Non-Preferred Hand* and *Barrel Button Mode Switch*). The results indicated that the techniques utilizing tilt angle and azimuth offered faster performance than the others.

3.1 Introduction

In pen based interfaces, fluid and continuous interaction is a very important feature [22]. How to switch smoothly between different operation modes (e.g. ink and gesture) is an open question. For pen-based UI designers, especially for those who are seeking fluid multi-mode operation in pen-based interfaces, it is necessary to select a suitable mode switching technique for the UI design.

Mode switching techniques are used widely on some pen-based devices, e.g. PDAs and Tablet PCs. Up to now, some mode switching techniques have been proposed by researchers, such as *Non-Preferred Hand* [7] and *Barrel Button Mode Switch* [17]. However, current switching techniques usually impose intervals (called *switching intervals* hereafter) on operational sequences, e.g. *Timeout* imposes an extra time interval on operational sequences while *Non-Preferred Hand* and *Barrel Button Mode Switch* impose not only time but also space intervals on operational sequences. The intervals not only take extra operational time but also distract the user's attention from the targets. Therefore, our basic motivation is to explore some *pen-tip-originated* mode switching techniques, which allow fluid and continuous switch by eliminating or reducing *switching intervals*.

Fortunately, electronic pens commonly possess multiple input parameters (e.g. stroke, pressure, azimuth and tilt angle). However there is no study which comprehensively compares different mode switching techniques with multiple pen input parameters. Thus, this study comprehensively investigates mode switching techniques that utilize pen input parameters, so as to obtain some UI design guidelines for pen-based UI designers. Three *pen-tip-originated* mode switching techniques (i.e. *Pressure*, *Tilt* and *Azimuth Mode Switch*) have been designed for fluid pen-based operation. To perform these mode switching techniques, the pen tip does not need to be moved away from the targets, and no space interval is needed. The key feature of *pen-tip-originated* mode switching is that it has the potential to make pen-based operations much more fluid.

3.2 Related work

Many pen-based devices use a pause as a way to provide state-transition model [23]. Time-out is a press-and-hold gesture, which requires holding the pen still for a fixed period to generate a mode switch. MacKenzie and Oniszczak [24] proposed a pressure-based technique for touch pads. The technique employs pressure value as a threshold.

When the finger pressure on the pad surface exceeds a programmable threshold, it creates an additional input state which delivers both aural and tactile feedback. GEdit [25] proposed selection-action techniques such as drawing a lasso for selection, and then ending the stroke inside the lasso to delete the selected objects, or ending the stroke outside the lasso to move a group of objects. To copy a group of objects, the user makes the move gesture and adds a “C” to the end of it. Hinckley et al. [17] employed a pigtail gesture to represent a certain command, but only recognized the pigtail after the pen was lifted.

Many pen interfaces support an ink mode for the entry of raw ink strokes, and a gesture mode for entering commands [26, 27]. Li et al. [7] showed that using the non-preferred hand to perform an explicit press of a button (the button was called a *mode switching button* in the original) on the Tablet PC’s bezel was a robust technique for ink/gesture mode switching, costing only 139ms per mode switch with about a 1% incidence of mode errors. However, in some situations when a *mode switching button* is unavailable this technique becomes impractical.

Overall, no studies have comprehensively compared different mode switching techniques using multiple pen input parameters. Thus, this is an area that is ripe for further research.

3.3 Mode Switching Techniques

In this section, we introduce the three *pen-tip-originated* mode switching techniques and compare them with the three traditional mode switching techniques.

3.3.1 Timeout Mode Switch

Timeout Mode Switch is a standard technique as a right-click equivalent. This technique is very useful for some pen-based devices (e.g. PDAs or mobile phones) where few input devices are available. This technique requires a user to press the pen tip onto the tablet, hold the pen still until mode switching visual feedback (e.g. a red circle around the pen tip) appears on the screen, before a popup menu is activated. There are two phases in the selection a command using the *Timeout Mode Switch.* Once the pen tip touches the screen, it enters the *Still Detection* phase in which the pen movement must be kept less than a *pen tip motion threshold* to be considered as *still.* In our implementation, the *Still Detection* phase was 1 second in duration and the *pen tip motion threshold* was 1.5mm. If the pen is held longer than 1 second, a green circle appears around the pen tip and then a popup menu appears beside the pen tip. After that the *command selection* phase begins, in which the user can slide the pen tip to select a menu command.

3.3.2 Non-Preferred Hand Mode Switch

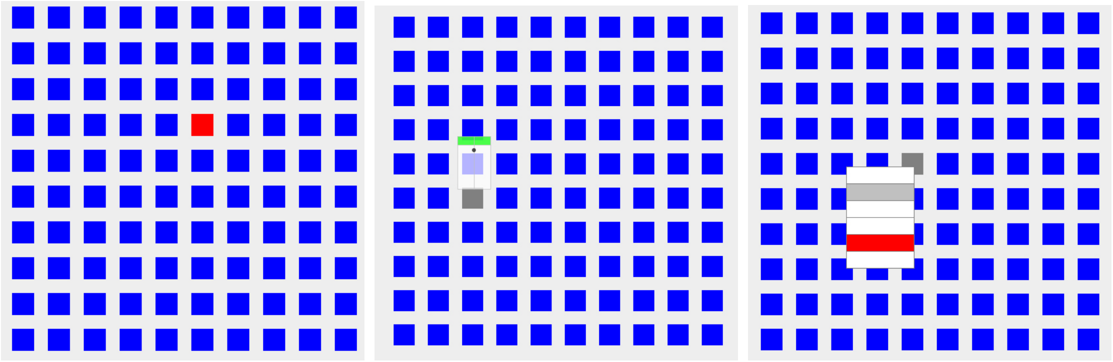
Pressing a *mode switching button* with the non-preferred hand to perform a mode switch is called *Non-Preferred Hand Mode Switch.* Two-handed interaction techniques have been explored extensively [28,29]. Appropriate UI designs for two-handed interaction can reduce operation time. *Non-Preferred Hand Mode Switch* outperformed all the other mode switching techniques in the experiment of Li et al. [7]. In our implementation, the keyboard space bar was employed as the *mode switching button.* A click on the space bar activates the mode switch and a popup menu appears on the screen, the appearance of the popup menu indicates that a mode switch has been performed.

3.3.3 Barrel Button Mode Switch

There is a barrel button on most styluses. Pressing the barrel button to present a mode switch is also a standard technique in many existing pen-based applications [7], the barrel button is used as a right-click equivalent. In our implementation, when the pen tip was pressed onto the tablet, a click on the barrel button affected a mode switch and a popup menu appeared beside the pen tip.

3.3.4 Pressure Mode Switch

Pressure as a pen input channel has been explored extensively. Ramos et al. [4] explored the human ability to vary pen-tip pressure as an additional channel of access to information with a Wacom Intuos tablet with a wireless stylus which provides 1024 levels of pressure. Ramos et al. indicated that dividing the available pressure range into six levels or less produces the best user performance. Ramos and Balakrishnan [5] introduced *pressure marks*, where pressure variations were marked as action commands. Li et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interactions. We did a pilot study to decide the best pressure



(a) Point to the target (b) Trigger a menu (c) Select a command

Fig. 3.1 Pressure Mode Switch

spectrum for mode switch. In our experiment, 12 participants were asked to draw alternately with their own sense of normal pressure and then with heavier pressure on a Wacom LCD graphics tablet display using a digital pen, which reports 1024 levels of pressure. The results demonstrated a statistically significant difference on the *maximum average pressure* of a stroke between the normal and heavier pressure conditions.

In our implementation, the heavy spectrum of pressure was employed for mode switching, because in the low end pressure spectrum, pressure distinctions are more difficult to control [4, 30]. Thus, the low end spectrum was omitted from the technique design. The participants pressed the pen tip on the active target (the red square, see Figure 3.1a), a white rectangle (with a green bar in its topmost edge and a vertical grey line in its center) appeared beside the active target (Figure 3.1b). When the participants pressed the pen heavier and heavier, a little black circle rolled up the grey line (Figure 3.1b). We call this process *press phase.* When the little black circle entered the green bar, a popup menu appeared beside the target, then a mode switch was performed and the following process was *command selection phase* (Figure 3.1c).

3.3.5 Tilt Mode Switch

In order to determine the best tilt angle spectrum for mode switching, a pilot study, similar to the one for *Pressure Mode Switch*, was performed. In our experiment, 12 participants were asked to draw a freeform drawing on a WACOM tablet display, in which tilt angle ranges from 22° to 90°. The collected data indicated that, when a pen is naturally held and slid, the tilt angle range for input fell mostly into the median range. In other words, the very high and very low tilt angle range was rarely used (called the *rarely-used spectrum*). *Tilt Mode Switch* can be produced by the position transition of users' holding pens from the often-used spectrum to the *rarely-used spectrum.*

In our implementation, the *rarely-used spectrum* of tilt angle was employed to per-

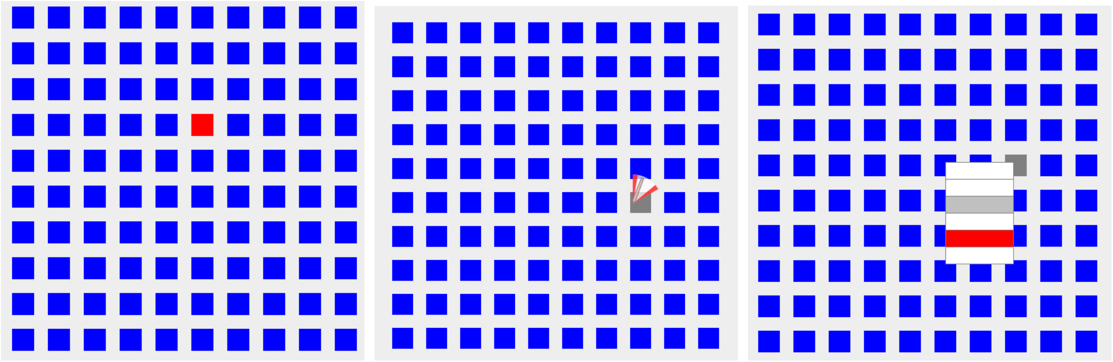

(a) Point to the target (b) Trigger a menu (c) Select a command

Fig. 3.2 Tilt Mode Switch

form a mode switch. In the experiment, the participants pressed the pen tip onto the target square, then a white sector, two small red sectors and a pink bar appeared beside the pen tip. The two red sectors were positioned beside the white sector, the pink bar floated on the sectors with one end at the circular center (Figure 3.2b). The participants tilted the pen; and the pink bar rotated together with the pen, indicating changes in the tilt angle. The tilting process is called *tilting phase*. When the pink bar entered either of the two red sectors, a popup menu appeared beside the pen tip. This indicated that a mode switch had been performed and the next process was *command selection phase*.

3.3.6 Azimuth Mode Switch

A pilot study (similar to that in subsection 3.5), was done to detect the possible azimuth spectrum for *Azimuth Mode Switch* on a Wacom tablet display with a digital pen, which reports an azimuth range from 0° to 360°. The experiment results indicated that there was also a *rarely-used azimuth range* just as for the tilt angle. The mechanism of *Azimuth Mode Switch* is similar to that of *Tilt Mode Switch*, and the corresponding introduction is omitted in this section.

3.4 Experiment

To investigate the performance of each of these six mode switching techniques, a quantitative experiment was conducted.

3.4.1 Apparatus

The hardware used in the experiment was a WACOM Cintiq 21UX flat panel LCD graphics display tablet with a resolution of 1,600 × 1,200 pixels (1 pixel= 0.297mm), using a wireless pen with a pressure, tilt angle, azimuth and twist angle sensitive isometric tip (the width of the pen-tip was 1.76mm). It reports 1024 levels (ranging from 0 to 1,023, the minimum unit is 1) of pressure, 360° (ranging from 0° to 359°, the minimum unit is 1°) of azimuth, 69° (ranging from 22° to 90°, the minimum unit is 1°) of tilt angle and has a binary button on its barrel. The experimental program was implemented with Java^{TM} 6.0 running on a 3.2 GHz P4 PC with the Windows XP SP2 Professional operating system.

3.4.2 Participants

Sixteen participants (two female and fourteen male, ranging in age from 21 to 33 years old) were all volunteers from the university community. All of them were right-handed. Ten of them had two years experience of using a digital pen and the other six had no such experience.

3.4.3 Procedure

The experiment included a warm-up session, six experimental sessions with one session for each technique, and a post-study questionnaire. The experiment took each participant about 30 minutes in total. A 6×6 Latin Square was used to counterbalance the order of the techniques. In one experimental session, each participant was given 3 blocks of trials. The participants could take a break between blocks. In the experiment, the participants were asked to select a target square from a selection region which consisted of 100 squares arranged in a 10×10 grid (Figure 3.2a). The target square (a red square, see Figure 3.2a) appeared randomly in the 10×10 grid 10 times in one block for each technique. In total, the experiment consisted of:

16 participants ×
6 mode switching techniques ×
3 blocks ×
10 repeats
= 2880 trials

In each trial, the participants were asked to press the pen tip onto the target square (this process is called *pointing phase* hereafter). The participants used one of the six mode switching techniques to trigger a popup menu (hereafter, this process is named *triggering phase*) and slid the pen to select the red target menu item (see Figure 3.1c, this process is called *selecting phase*). One mode switch was performed in each *triggering phase*, therefore, 2,880 mode switches were performed during the experiment. We collected data on all events in the experiment including pen down/up, pen drag, pen move, barrel button down/up, *mode switching button* down/up as well as the stylus' current position, pressure value, tilt angle, azimuth and the time of each event.

3.5 Results

A RM-ANOVA (repeated measures analysis of variance) was used to analyze the performance in terms of performance time, error rate and subjective preference.

3.5.1 Operation time

The total operation time, triggering time (the time elapsed in *triggering phase*) and selection time (the time elapsed in *selecting phase*) are analyzed in this section. A significant difference between the six techniques in total mean operation time, $F(5, 287) = 52.1, p < 0.001$, was observed in the results. Figure 3.3 illustrates that *Timeout Mode Switch* clearly took the longest total operation time. *Azimuth Mode Switch* performed the fastest of all six mode switching techniques. Further analysis of the experimental time is reported in the following sections.

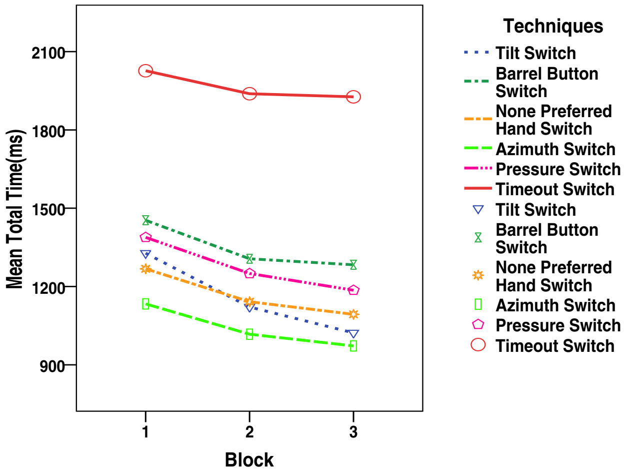


Fig. 3.3 Total Time.

There was also a significant difference between the six techniques in triggering time (mode switching time), $F(5, 287) = 143.2, p < 0.001$. Figure 3.4 illustrates that the order in which the techniques performed regarding triggering time was similar to the order for total time. We can see that *Azimuth Mode Switch* and *Tilt Mode Switch* outperformed the other techniques more distinctly in triggering time than in the total time.

The results also show a significant difference, $F(5, 287) = 5.9, p < 0.001$, between the six techniques in selection time. Figure 3.5 illustrates an interesting thing that the

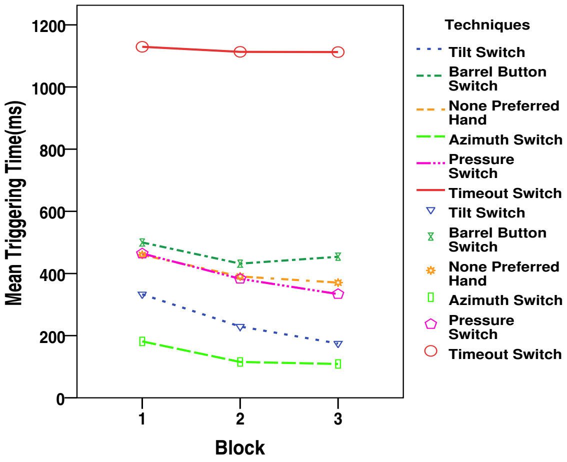


Fig. 3.4 Triggering Time.

order of the six techniques sorted in mean selection time is significantly different from that of the mean total time and that of the mean triggering time. The Tilt, Azimuth, Barrel Button and Pressure Mode Switch took distinctly more selection time than the other two.

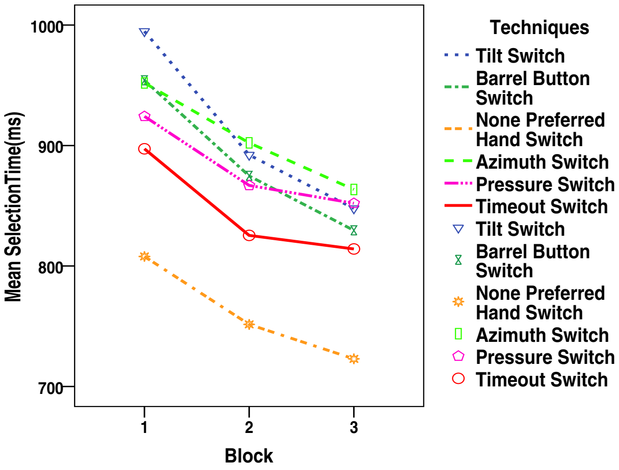


Fig. 3.5 Selection Time.

3.5.2 Errors

Three kinds of errors occurred in the three corresponding experimental phases. The errors occurred in *pointing phase*, *triggering phase* and *selecting phase* are respectively called as *pointing error*, *triggering error* and *selecting error. Pointing error* was caused by a participant when s/he pointed to a non-target square. When a participant lifted the pen tip from the target without activating a popup menu, a *triggering error* occurred. When the pen tip was lifted from the popup menu without having been pointed to the target menu item, a *selecting error* occurred. The highest *pointing error* rate, 10.00%, occurred in *Barrel Button Mode Switch*; the lowest, 3.75%, occurred in *Pressure Mode Switch.* The *pointing error* rate of *Tilt Mode Switch*, 4.79%, was a little higher than that of *Non-Preferred Hand Mode Switch*, 4.38%. The *triggering error* rate of each mode switching technique was rather low. The highest *triggering error* rate, 1.67%, occurred in *Tilt Mode Switch*; and the lowest, 0.21%, occurred in *Pressure Mode Switch.* The highest *selecting error* rate, 4.79%, occurred in *Azimuth Mode Switch*; and the lowest, 0.62%, occurred in *Timeout Mode Switch.*

3.5.3 Subjective evaluation

The six techniques were rated by the participants according to three criteria: ease of use, fatigue of the eyes and hands and subjective preference. Participants were required to rate each technique on a 7-point scale (1=lowest preference, and 7 =highest preference). No significant difference was found in the effect of the six techniques regarding subjective preferences.

3.6 Discussion

The experimental results indicate that *Tilt Mode Switch* and *Azimuth Mode Switch* outperformed *Non-Preferred Hand Mode Switch* in both speed and accuracy. This is probably due to the fact that the participants spent extra time dividing their attention between two hands for *Non-Preferred Hand Mode Switch.* Li et al. [7] showed that *Non-Preferred Hand Mode Switch* outperformed the other four techniques in their experiment but the Tilt mode was not included. The results of our experiment indicate that *Non-Preferred Hand Mode Switch* performed as well as *Tilt Mode Switch* in the first two blocks, but was surpassed by *Tilt Mode Switch* in the last block. This is probably because *Non-Preferred Hand Mode Switch* was more familiar to the subjects, thus the subjects performed well with it from the beginning. As for *Tilt Mode Switch*, it is a novel mode switching technique, therefore its learning effect is better than *Non-Preferred Hand Mode Switch.* In the experiment, we observed that the participants adjusted the pen tilt angle or azimuth to an abnormal position in *Tilt Mode Switch* and *Azimuth Mode Switch.* Before they moved the pen tip to select the target menu item, the participants tended to adjust the pen to a normal posture, with which they could comfortably hold the pen. In *Barrel Button Mode Switch*, the participants moved their fingers from their normal holding positions to the barrel button. They tended to move their fingers back to their normal holding positions prior to the following *selecting phase.* The participants also reported that they tended to release the pen tip pressure on the tablet before performing the following *selecting phase* in *Pressure Mode Switch.* All these adjustments prior to the *selecting phase* took the participants extra time. *Non-Preferred Hand Switch* outperformed all the other techniques in mean selection time in the experiment.

We also found that *Barrel Button Mode Switch* is the most prone to *pointing error.* During the experiment, we observed that the pen tip easily tended to deviate from the target when the subjects moved their fingers from their normal holding positions on the barrel to the barrel button. The deviation led to a *pointing error*. There is no significant difference between the *pointing error* rates of the mode switching techniques except for *Barrel Button*. As we anticipated, *Tilt* and *Azimuth Mode Switch* were more prone to *triggering errors* than the others except Timeout, since they were both novel mode switching techniques to the subjects. The subjects reported that they felt a little nervous when manipulating the two novel angle input channels (i.e. tilt angle and azimuth) at the beginning of the experiment, however, they said that they gained confidence after one or two blocks. This is consistent with the observation that most *triggering errors* of *Tilt* and *Azimuth* occurred in the first block. The *Timeout* has a surprisingly high *triggering error* rate. During the experiment, we noticed that some subjects tended to move the pen tip before the *Still Detection* phase ended. This usually caused a *triggering error*. The *selecting error* rates of the three *pen-tip-originated* mode switching techniques are a little higher than the others. For *pen-tip-originated* mode switching, the pen had to been adjusted to abnormal angles of *Tilt* and *Azimuth* to perform a Mode Switch, and pen input pressure had to be adjusted to a heavier scale for *Pressure Mode Switch*. The subjects tended to adjust the pen posture to a normal position and the pen tip pressure to its normal scale, with which they can comfortably hold the pen, while moving the pen tip to the target menu item. This most likely distracted their attention from the target menu item itself and made them lift the pen tip from a non-target square. Although the error rates of the three *pen-tip-originated* mode switching techniques are a little higher than the traditional methods, taking into account that the techniques were novel to the subjects, we believe that the results would have been more promising if more practice time had been taken.

Regarding the use of the mode switching techniques in real systems, the Tilt mode switching technique has been used in our prototype drawing system [31], which employs a pressure-based “line-string” selection method [32]. The user can select target components from a flow chart. While a pen is slid across the screen, the objects strung together by the stroke are selected when the pen input pressure exceeds a given threshold. After the target-selection, *Tilt Mode Switch* was utilized to perform a smooth switch from selection to action. Users tilt the pen from the *often-used spectrum* to the *rarely-used spectrum* to perform a *Tilt Mode Switch*; a pie menu is activated and displayed with its center under the pen tip. When the pie menu is activated, the user slides the pen tip across a menu item and the corresponding action is performed. Users can perform the whole operation from selection to action with one continuous and fluid stroke. *Tilt Mode Switch* makes the switching from selection to action smoother than in the traditional UI.

3.7 Conclusion

Switching effectively between different pen input modes is essential for most pen-based applications. Multi-parameter pen input can present pen-based UI designers with different choices in mode switching techniques. We designed three *pen-tip-originated* mode switching techniques and compared them with the traditional methods according to speed, accuracy and the participants’ subjective comments. The results indicate that mode switching with a pen input angle, i.e. azimuth or tilt angle, performed the fastest with a tolerable error rate. These techniques enhance operation fluidity and help users focus their attention on targets themselves by eliminating *switching intervals.* This can potentially improve operation efficiency. We believe that the results are valuable to researchers who are working on pen-based UI design, especially for those who are working on multi-mode pen-based operation.

Chapter 4

Exploring utmost number and combination of pen input parameters in 3D operation

3D transformation is an important study topic in HCI field. Many researchers pay attention to how to manipulate 3D objects with 2D sketches. The exiting sketch-based systems allow a user to perform some complex 3D operations by using relatively simple 2D strokes. But these sketch-based operations are not natural or intuitive enough for most users; furthermore, a sketch-based system typically imposes a gesture learning process on the users. A digital pen typically posses multiple input channels, e.g. stroke, pressure, tilt angle, twist angle and azimuth. Being different from a mouse and keyboard, a digital pen can provide physical feedback to each of its input channels, e.g. a pen's barrel tilt angle to the screen is equivalent to its input tilt angle. This chapter explores the potential of natural and fluid 3D transformation with multiple input channels of a pen. Unlike mice or keyboards, pens connately possess physical feedback to their input channels. Therefore, pens outperform other indirect input devices in some natural and intuitive operations. This chapter proposes four kinds of operations regarding 3D objects, i.e. selecting 3D objects, rotating 3D objects, scaling 3D objects, translating 3D objects. A prototype system was implemented to test the proposed 3D operations.

From an informal user study, we find out that the subjects performed better with the prototype system than with the standard Maya widgets. We also find out that the utmost number of input channels that a user can concurrently manipulate depends on the mapping between pen input channels and the controlled parameters.

4.1 Introduction

3D operations are widely used in many fields, e.g. CAD, education, chemistry, mechanics and architecture. Most of the commercial 3D software is notorious for their complicated operations, typically with abstract parameters setting dialog boxes. It was very difficult to perform 3D operations even for experts. For investigating simple and intuitive 3D operations, a wide variety of researches has been explored for decades. Most of those studies focused on free-form 3D modeling. Some of those studies (e.g. [33–36]) explored 3D modeling by suggestive sketch-based modeling systems, which sought to map rough sketches to linear geometry such as curves, planes and polyhedrons. The other of those studies (e.g. [37–41]) investigated 3D modeling by literal sketch-based modeling systems, which created 3D surfaces directly from user's strokes.

Although sketch-based modeling systems have evolved for decades, current sketch-based modeling systems still have some limitations. Sketch-based operations are modeling-independent, i.e. users have to know specific knowledge about the type of models. Sketch-based interfaces also suffer from the problem of self- disclosure [42]. Traditional WIMP interfaces are discoverable, in the sense that a user can look at the menu titles, icons, buttons, and dialog boxes, and garner some idea of what the application can do and how to use it. A sketch-based modeling system, on the other hand, may simply provide the user with a blank window representing virtual paper, with no buttons or menus what so ever. Though it maybe more usable and efficient for someone who has been given a tutorial, such an interface does not disclose any hints about how to use it. Devising elegant solutions to this problem is still a current challenge for sketch-based modeling researchers. In some sketching-based systems, various modeling operations are performed through a sketching metaphor (e.g. [36, 37, 43, 44]). Herndon, K.P., et al. [45] presented 3D interactive shadows, where the shadows of a 3D object was employed as a metaphor to performed 3D operations. Operations through a metaphor are neither natural nor intuitive enough. In fact, most sketch-based interfaces are far from natural-many require the user to draw in very specific ways to function properly, which reduces the immersion and ease of use. While sketches can be used in many facets of a modeling interface, a purely gestural sketch-based interface causes modality problems. That is, a given stroke or gesture can have different meanings in different modes of the system. As an example, the ShapeShop system of Schmidt et al. [39, 46, 47] uses gestures to initiate widgets, but also allows surficial augmentation strokes- what happens if an augmentation stroke is the same as a widget gesture? Only the user can truly know the intended meaning in this case. It is another challenge for researchers to provide a consistent and predictable interface without modality problems.

Electronic pens possess the physical properties (e.g. barrels, tips and dimensions) of the traditional drawing or writing pens and extend some electronic properties (e.g. stroke, pressure, tilt angle, twist angle and azimuth inputs) beyond the traditional drawing pens. Human beings have used various pens (e.g. Chinese writing brushes, quills, fountain pens, ball pens, pencils) for thousands of years. Therefore, the traditional drawing or writing pens are natural tools for human beings. Electronic pens have the potential in natural human computer interaction, since they inherit some physical properties from the traditional drawing or writing pens. This chapter explores how to manipulate 3D objects with multiple input parameters of a pen more natural and intuitive than the current ways.

User controlled transformation, which typically includes 3D translation, rotation and scaling, is one of the key components of those 3D modeling systems or viewing systems. The transformation typically was performed discretely through click and drag with a mouse or a pen. It is chasing by many researchers in HCI to make users closely mimic the feel of real medium with digital devices. Most 3D modeling systems aim at closely mimicking freehand drawing on paper. Digital pens, inheriting some physical properties from the traditional drawing or writing pens, permit a user convey information not just with the overall form of drawing, but also by varying stroking pressure, pen input tilt angle, azimuth and twist angle. All these information are utilized in calligraphy with a writing brush for centuries. However, the potential of a pen to naturally and intuitively operate 3D objects has been rarely exploited. Oshita [48] presented a pen-based intuitive interface to control a virtual figure interactively. Pen input parameters, i.e. the pen positions, pressure and tilt, were utilized to make a human figure perform various types of motions in response to the pen movements manipulated by the user. Kolhoff et al. [49] explored manipulating a virtual figure's gait with two pens "walked" on an indirect tablet. Although their operations were natural, intuitive and interactive, the operations were limited to manipulation of virtual figures. In this chapter, we present a general operation method that is independent to any 3D model. A pressure-based "line-string" selection method [32] is employed to select 3D targets. The pressure input is utilized to couple normal sketch and selection stroke. The selected targets can be translated according the pen tip on the screen. The selected targets can be rotated around both an axis and a pivot. The axis orientation can be set by the pen input parameters and will be roughly kept consistent with the pen's center axis. Rotating around a pivot will be further divided into tilting and orientating around the pivot. The pivot will also be able to be adjusted by the pen input parameters. Users can also scale the selected 3D objects with a pen through natural and intuitive manners. Four kinds of scaling operations, i.e. stretching or squeezing a 3D object along its center axis, stretching or squeezing a 3D object along its radii, is designed and implemented. Stretching or squeezing a 3D object along its center axis is to change its size along its center axis by employing pen input parameters. The operation effect of stretching or squeezing a 3D object along its radii is similar to that of rolling a piece of paper. The radial dimension of the paper reel could be enlarged when the paper is rolled in one direction. On the contrary, the radial dimension could be reduced when the paper is rolled in the opposite direction. Similarly, the radial dimension of an object can be enlarged when the pen is rotated in one direction in the our design; and the radical dimension of the object can be reduced when the pen is rotated in the opposite direction. The operations can be performed concurrently, e.g. to rotate an object around its center axis and translate it concurrently can simulate rolling an object on the screen. A prototype system was designed and implemented to test the mentioned 3D transformation. An informal user study shows that participants performed better with the prototype system than with the stand Maya widgets in building up a 3D chair model.

4.2 Related Work

In this section, we discuss related work regarding sketching-based 3D operations, virtual human figure manipulations with pens and pen input channels.

4.2.1 Previous Work on Sketching-based 3D Operations

Augmentation is a typical sketching-based modeling 3D operation, through which some new features can be added to an existing 3D model. Augmentations can be made in either a surficial or additive manner. With surficial augmentation, users can sketch features, e.g. sharp creases [40, 50, 51], on the surface of the model. Additive augmentations use constructive strokes to define a new part of a model, e.g. a limb or out cropping, along with additional strokes that indicate where to connect the new part to the original model [37,40]. Most sketch-based modeling systems support sketch-based editing operations. Additive augmentation uses constructive strokes to define a new part of a model, such as a limb or outcropping, along with additional stroke(s) that indicate where to connect the new part to the original model (e.g. [37, 40]). Surficial augmentation allows users to sketch features on the surface of the model, such ass harp creases (e.g. [40, 50, 51]). Besides augmentation, many sketch-based modeling systems support other sketch-based editing operations, such as cutting (e.g. [40,52,53]), blending (e.g. [37, 53–56]), twisting (e.g. [57]), tunneling (creating a hole, e.g. [39, 40]), object grouping [58], erasing and local smoothing [40], contour over-sketching (e.g. [56,59,60]), segmentation (e.g. [53, 61]), free-form deformation (e.g. [62]), and affine transformation (e.g. [63]). Sketching-based 3D operations are usually modeling-dependent. The same or similar 2D sketches can be interpreted into different 3D models in different systems. On the contrary, the same or similar 3D models can also be modeled from different 2D sketches in different systems. Therefore, a sketching-based modeling system typically aims at a specific application. The applications can be roughly classified into two groups, i.e. applications in computer-aided design (CAD) [64–74] and applications in digital content creation. Applications in digital content creation include wide topics, e.g. virtual garment design [75], animation [76–78], plant modeling [47, 79–86]. Besides the above operations with directly stroking the models, Schmidt et al. [46] use gestures not to manipulate an object directly, but simply to initiate an operation widget. The user can then interact with the widget to manipulate the object interactively. With the widget, 3D operations can be more precise than those operations with directly sketching on the objects. Severn et al. [63] present a direct manipulation method, transformation strokes, which are modeling-independent. With a single U-shaped gesture, quick transformation (i.e. translating, scaling and rotating) on an object can be performed.

4.2.2 Previous Work on Virtual Human Figure Manipulation with Pens

Manipulating virtual Human figures on computers is an elusive target since human figures have a large number of degrees of freedom (DOF) and their movements are complicated [48]. Some researchers explored more intuitive and natural virtual human figure manipulation with a pen than with a mouse and keyboard.

Oshita [48] presented a pen-based intuitive interface to control a virtual figure interactively. Pen input parameters, i.e. the pen positions, pressure and tilt, were utilized to make a human figure perform various types of motions in response to the pen movements manipulated by the user. Kolhoff et al. [49] explored manipulating a virtual figure's gait with two pens "walked" on an indirect tablet.

4.2.3 Previous Work on Pen Input Channels

To date, there are many studies on the utilization of pen input parameters. These studies can be roughly divided into two categories. One category investigates the general human ability to control pen input parameters; the other category aims at enhancing performance of human and computer interaction by implementing novel applications or techniques which exploit particular input parameters.

Up to now, pressure parameter has been explored extensively. Herot and Weinzapfel [2] studied the human capability of the finger to apply pressure and torque to a computer screen. Buxton [3] investigated the use of touch-sensitive technologies and the potential for interaction that they suggested. Ramos et al. [4] explored the human ability to vary pen-tip pressure as an additional channel of access to information. Ramos and Balakrishnan introduced pressure marks [5] and Zliding [6]. Pressure marks can encode selection-action patterns in a concurrent, parallel interaction. In pen strokes, variations in pressure make it possible to indicate both a selection and an action simultaneously. Zliding explores integrated panning and zooming by concurrently controlling input pressure while sliding in X-Y space. Li et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interactions. Harada et al. presented a set of interaction techniques that leveraged the combination of human voice and pen pressure and position input when performing both creative 2D drawing and object manipulation tasks [8].

Input angles (i.e. tilt angle, twist angle and azimuth) are often used as UI clues for natural and intuitive interaction. Balakrishnan et al. [9] introduced the Rockin'Mouse, which is a promising device for both 2D and 3D interaction that uses tilt input to facilitate 3D manipulation on a plane. Tian et al. [10] explored the Tilt Menu. The Tilt Menu is implemented by using 3D orientation information of pen devices for better extending selection capabilities of pen-based interfaces. Some other studies such as TiltType [11] and TiltText [12] focus on using the tilt information of mobile phones to affect text entry tasks in mobile devices. Bi et al. [13] explored rolling angle on general human being control ability. They suggested that both rolling amplitude and speed should be taken into account for rolling-based interact techniques.

As for sketch-based techniques, Davis et al. [15] introduced their SketchWizard, which is about wizard of Oz prototyping of pen-based user interfaces. Apitz and Guimbretire [14] presented their CrossY, in which pen stroke did all the drawing operations.

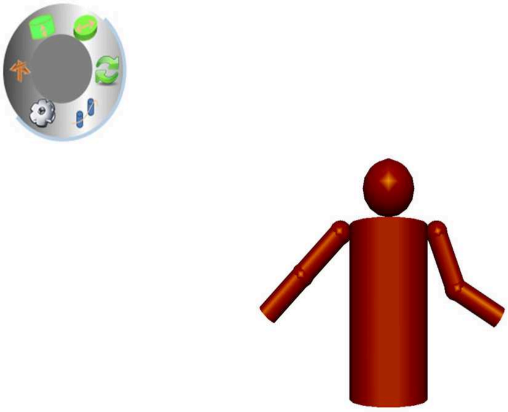


Fig. 4.1 The user interface of the prototype system.

4.3 The user interface of the prototype system

Four 3D operations are designed in the prototype system, i.e. 3D objects selection, rotation, translation and scaling. The scaling can be subdivided into scaling along an axis or a radius. A pie menu (see Fig. 4.1) with six menu items, which are corresponding to the 3D actions, is utilized in the prototype system. Bi et al. [13] presented their study on rolling (twist) angle for pen input. They suggested that the rolling can be identified as incidental if the rolling speed of a data event is between -30°/s and 30°/s or the rolling angle is between -10° and 10°. And -90° to 90° can be exploited as the usable rolling range. Based on their study results, rolling is employed in our experiment design to activate the pie menu if the rolling speed exceeds the range of [-50°/s, 50°/s], and rolling angle exceeds [-50°, 50°]. After selecting all the targets, the user intentionally rolls the pen. If the rolling angle and speed exceed the specific thresholds, the pie menu will be activated and displayed with its center under the cursor. The user slides then pen tip and cross a menu item to select an operation command. The menu disappears from the screen after an action has been certified, so as to keep the interface simple and leave more space for 3D manipulation.

4.4 Manipulating a 3D object like holding it in hand

In contrast with a decades-long history of digital devices, people have been using pen and paper to express graphical ideas for centuries [87]. A digital pen inherits some physical properties from a drawing or writing pen. Therefore, a digital pen is more natural and intuitive than a mouse and keyboard in some interactive tasks. Most of the user interfaces use artificial digital widgets as feedback to users. However, a pen's own physical status can serve as feedback to users of its digital input. For example, a pen's physical orientation or tilt angle gives a user visual feedback of the pen's digital input of azimuth or tilt angle; the reacting force of a pen's barrel onto a user's fingertips give the user tactile feedback. Unlike manipulating a 3D object with a mouse and keyboard, manipulating a 3D object with a pen has the potential to fully utilize the pen's physical feedback. In the design, a pen's physical feedback is exploited through keeping the coincidence of movements between the pen and a 3D target. The operation effect is just like holding and manipulating an object by hand. In this section, we introduce the 3D object manipulation from a user's point of view. The operations include selecting 3D objects, rotating 3D objects, scaling 3D objects, translating 3D objects.

4.4.1 Selecting 3D objects

In the design, a pressure-piercing "line-string" selection method is employed to select 3D targets. A user slides the pen tip on the screen starting from a blank area, where there is no object's projection. If the pen tip pressure and the stroke length exceed the predefined thresholds, a target selection process is evoked. When the pen tip enters the projecting region of a 3D object, a user can press the pen tip more heavily over a usual scale. The pressure value of the pen input is utilized as a metaphor of the depth into the screen. "Line-string" selection is like stringing an object with a needle.

When the pressure of the pen tip on the screen gets heavier and heavier, the needle is pierced into the screen deeper and deeper. If the needle is pierced deeper than the depth of the object into the screen, the object will be stringed and selected. After an object is selected, the user can continue stroke the pen tip on the screen to select another object. All the selected objects are recorded with a list, which is called *target list* in this chapter. If there are some objects that the user does not want to select in the path of the selection stroke, s/he can steer clear of them or reduce the pressure on the pen to a normal scale without lifting the pen tip from the screen, then the needle will go though the object's projecting region over its surface and the object will not be selected. During the user performing pressure-piercing "ling-string" selection, a blue footprint line of the pen tip is drawn on the screen to give the user an intuitive feedback (see Fig. 4.2). To undo the selection, the user can stroke the pen back and across the footprint line on a selected object to undo its selection. The object is removed from the *target list*. The undoing operation is similar to pulling a thread out of an object and eliminating it from a string of objects.

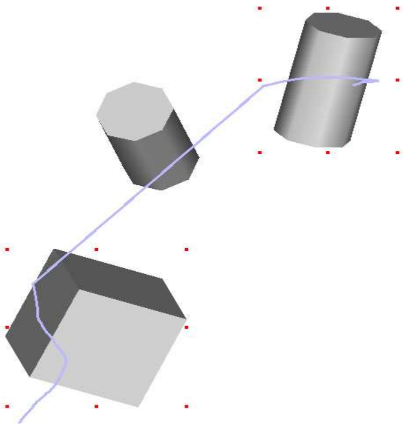
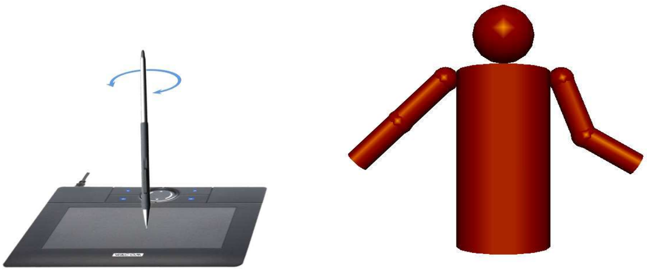


Fig. 4.2 Selecting objects pressure-piercing "line-string" selection method.

4.4.2 Rotating 3D objects

Rotation of a physical object can be classified into two types, i.e. rotation around a pivot and rotation around an axis. Both of the two types of rotation can be manipulated with a pen in an intuitive, natural, interactive and continuous way.

Rotating 3D objects around an axis



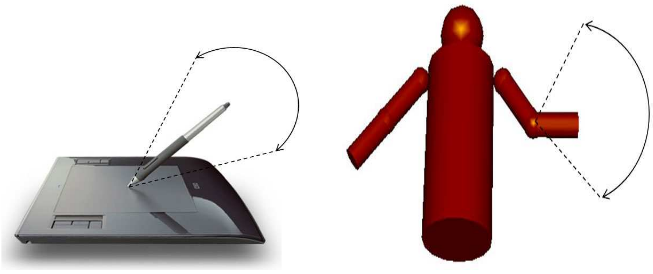
(a) Rolling the pen.

(b) The figure rotating around an axis together with the pen.

Fig. 4.3 Rotating objects around an axis.

A selected object can be rotated around an axis. The user press the pen more heavily, when the pressure input exceeds a predefined threshold and the pen keep resting, a rotation axis can be determined. The pen is considered to be resting only when its locomotion and posture transformation do not exceed the predefined thresholds. The pen's posture transformation includes tilting and orientating. Bi et al. [13] reported that the rolling of a pen could be consider as user intentional only when the rolling speed and the rolling angle surpasses the specific ranges. The user rolls the pen intentionally to distinct rotation axis setting operation from the pressure-piercing selection. When a rotation axis is determined, the axis crosses the point which is on the screen and under the pen tip, and it points to the pen's center axis. After an axis has been decided, the user can rotate an object by rolling the pen. When the pen's rolling speed and rolling angle exceeds the specific ranges, the object rotates around the axis concurrently with the pen's rotation. The user can "feel" the object's rotation by his fingertips even without seeing the screen, since the pen and the object rotate coincidentally. If the *target list* contains more than one object, all the objects rotate concurrently and maintain their relative locations. Fig. 4.3 illustrates six parts of a figures rotating with the pen coincidentally. The rotation axis can be changed interactively. Some rotation effects regarding depth can achieved when the rotation axis is not perpendicular to the screen. In other user interfaces,some types of operations respecting depth are difficult to perform. However, in our prototype system, operations regarding to depth can be performed interactively, intuitively and naturally, since rolling a pen is very familiar to most people.

Rotating objects around a pivot

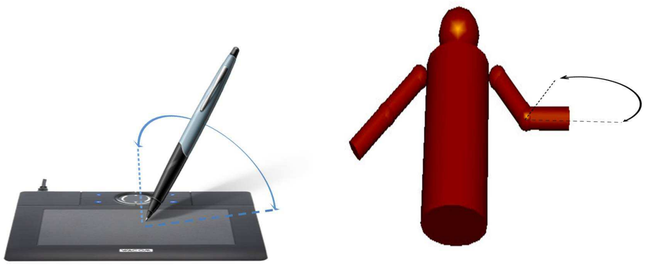


(a) Tilting the pen.

(b) The figure's forearm tilting together with the pen.

Fig. 4.4 Tilting objects around a pivot.

A pivot is set to be the intersection point of a rotation axis and the object's inner surface in the screen, which is facing the pen tip's pointing direction. Rotating around



(a) Orientating the pen.

(b) The figure's forearm orientating together with the pen.

Fig. 4.5 Orientating objects around a pivot.

a pivot includes tilting and orientating around the pivot. The user tilts the pen, and then the object tilts around the pivot. Fig. 4.4 illustrates tilting the figure's forearm around its elbow, which has been set as a rotation pivot. The user changes the pen's azimuth, and then the object's orientation is changed around the pivot. Fig. 4.5 shows orientating the figure's forearm around its elbow, which is the rotation pivot. The tilting and orientating can be performed concurrently when the user changes the pen's tilt angle and azimuth concurrently. In the illustration, the user can manipulate the figure to simulate a human being waving his arm. Similar to rotating objects around an axis, if the *target list* contains more than one object, all the objects rotate concurrently and maintain their relative locations. To filter quiver of the pen, the tilting and orientating works only when rotating angles exceed the respective thresholds. The rotation pivot can be changed through changing the respective rotation axis. When a rotation axis has been changed, the corresponding rotation pivot is determined by the intersection point of the object's inner surface. If there is more than one object in the *target list*, the pivot is determined by the rotation axis and the inner surface of the object crossed by the axis.

Rotating objects around a pivot and around an axis can be performed concurrently. The user rolls, tilts and orientates the pen concurrently, then the object(s) will roll, tilt and orientate together with the pen.

4.4.3 Scaling 3D objects

Four kinds of scaling operations, i.e. stretching or squeezing a 3D object along its center axis, stretching or squeezing a 3D object along its radii, is designed and implemented. Stretching or squeezing a 3D object along its center axis is to change its size along its center axis by employing pen pressure input. Ramos et al. [4] explored the capability of human being in controlling pressure input with a Wacom Intuos tablet with a wireless pen, which provides 1024 levels of pressure. They reported that dividing the pressure range into 6 levels or less produced the best performance. The pressure input is divided into three layers (Fig. 4.6), i.e. stretching, spacing and squeezing layers, to perform the axis-scaling operation. Stretching layer is related to the lowest levels of pressure input and squeezing layer to the highest. The stretching layer is assigned to the largest pressure bandwidth, given the pen is too sensitive to control at the low end of pressure value input [4]. A color wedge is used as visual feedback of pressure input levels. The user adjusts the pressure input into stretching layer and then an isotonic stretching operation along its center axis will be performed on the 3D object. When the user maintains the pressure in the stretching layer, the object will be bigger and bigger along its center axis. If there is more than one object has been registered to the *target list*, each object will be stretched along their respective center axis. The spacing layer is utilized as an interval to separate stretching and squeezing layers. When the user press the pen tip more heavily and make the pressure input enter squeezing layer, an isotonic squeezing operation will be performed along the object's center axis. When the user maintains the pressure in the squeezing layer, the object will be smaller and smaller along its center axis. If the *target list* contains more than one object, each object will be squeezed along its center axis concurrently.

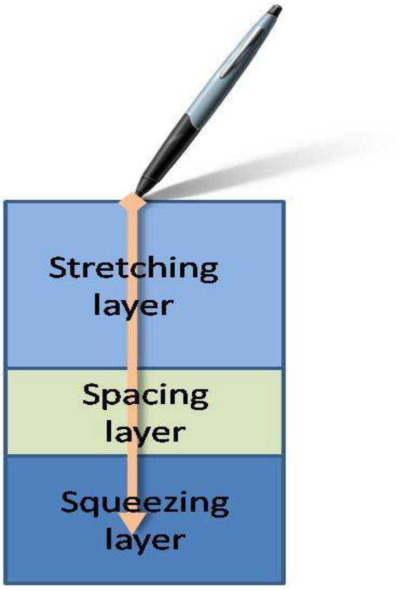


Fig. 4.6 Pressure layers in scaling 3D objects along an axis.

The user can scale an object along each of its radii by rolling the pen. The operation effect of stretching or squeezing a 3D object along its radii is similar to that of rolling a piece of paper. The radial dimension of the paper reel could be enlarged when the paper is rolled in one direction. On the contrary, the radial dimension could be reduced when the paper is rolled in opposite direction. The prototype system allows the user to enlarge or diminish an object along each of its radii when s/he rolls the pen anticlockwise or clockwise. The object scaling along its radii is isometric to the rolling angle of the pen, i.e. the object keep scaling along its radii when the user rolls the pen, the user stop rolling and the object stop scaling. For both radial stretching and squeezing, a specific threshold was set to filter incidental rolling input [13] of the pen. If there is more than one object contained in the *target list*, each object scales along its radii concurrently remaining their relative locations. The radical scaling size of each object is directly proportional to the rolling angle of the pen and the object's own radical dimension.

4.4.4 Translating 3D objects

When the user wants to move the selected object(s), s/he slides the pen tip and cross the locomotion menu item. The select object(s) can be translated by the pen tip locomotion on the screen. X-y locomotion of an object on the screen is similar to drag an object. The user tap the pen tip on the object and then slides it on the screen, the object will move together with the pen tip. In most of 3D operation systems,the transformation regarding to depth is a typically difficult action since the screen is actually 2D and a common mouse can only input planar points. A more intuitive depth-operation is implemented in our prototype, where the pressure input of the pen is utilized as metaphor of object depth into the screen. Similar to the axial scaling, the pressure input is divided into three levels, i.e. pulling layer, spacing layer and pushing layer, for z-locomotion. Pulling layer is related to the lowest levels of pressure input, while pushing layer to the highest. When the pressure input is in the pulling layer, an isotonic z-locomotion towards the screen is performed on the object. In other words, if the user maintains the pen pressure input in pulling layer, the object will continuously approach the screen. On the contrary, when the pressure input is maintained in the pushing layer, the object isotonically moves away from the screen. To filter incidental inputs, thresholds are predefined for both x-y and z-locomotion. The locomotion takes effect only when the inputs exceed the thresholds.

4.4.5 Combined transformations

Some of the mentioned operations can be performed concurrently. The user crosses the combined manipulation menu item to begin a combined transformation. The combined transformation sometimes can produce some special operation features, e.g. to rotate an object around its center axis and translate it concurrently can simulate rolling an object on the screen. The user rolls then pen and move the pen tip on the screen at the same time, the object rolls on the screen. Moving and tilting a figure can make it “lurch” on the screen. Rolling and translating a whipping top can make it spin and slide on the screen. In the prototype system, all the rotating and translating related transformations can be combined together and performed concurrently.

4.5 Informal user study

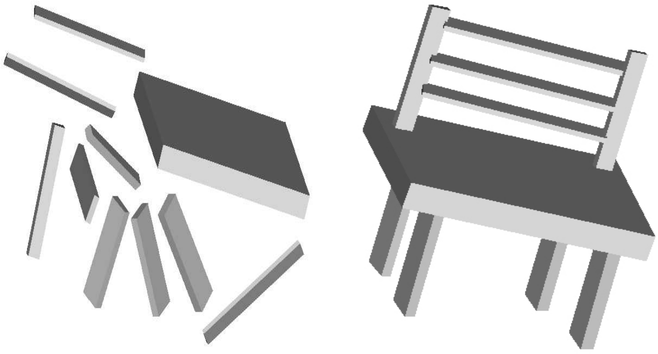
4.5.1 Apparatus

Participants were permitted to employ different input devices, including a direct input display-tablet, indirect tablet, mouse and keyboard. The direct and indirect tablets both possess a wireless pen with a pressure, tilt angle, azimuth and twist angle sensitive isometric tip (the width of the pen-tip is 1.76mm). The pen reports 1024 levels (ranging from 0 to 1,023, the minimum unit is 1) of pressure, 360°(ranging from 0° to 359°, the minimum unit is 1°) of twist angle,360° (ranging from 0° to 359°, the minimum unit is 1°) of azimuth and 69° (ranging from 22° to 90°, the minimum unit is 1°) of tilt angle. The experimental program was implemented with Java^{TM} 6.0 running on a 3.2 GHz P4 PC with the Windows XP SP4 Professional operating system.

4.5.2 Participants

Six participants (two female and four male ranging in age from 21 to 26 years) were all volunteers from a university community. All of them were graduate students and right-handed, and they all had graphics backgrounds. Two of them had years of experience on standard widgets in Maya and they were both regular users of Maya. Other two participants were familiar with 3D manipulation concepts, but did not regularly use 3D modeling tools. The other two subjects were novices who had no familiarity with

3D modeling.



(a) A set of parts. (b) A chair model.

Fig. 4.7 The subjects were asked to assemble the parts into a chair model.

4.5.3 Evaluation

Prior to the evaluation, the subjects are conducted to practice the prototype system and the standard 3D widgets in Maya each for about thirty minutes. After the practice, the subjects are requested to assemble the parts (Fig. 4.7a) into a chair model (Fig. 4.7b). All the six participants were requested to perform the assembly task for two times.

The two expert participants took roughly the same time with both the prototype system and the standard widgets in Maya in the first assembly task. In the second time, they only took about half as long with our prototype system as they did in Maya. However, they still thought that using our prototype was disagreeable for them. We noted that they tend to force our system to behave like the tools they were more familiar with. We think this result is very positive, taking into account their years of experience of the standard 3D widgets in Maya versus hours training with our prototype system.

These two irregular 3D model users completed the assembly task spending roughly one third as long with our system as using the Maya widgets for the first time. In the second round, they took about as half time with our system as with the standard widgets in Maya. They commented that our system was simple to use for them.

The two novice participants spent hours in trying to assemble the chair with the standard 3D widgets in Maya. But it seemed that it was too difficult for them to complete the task. Hours later, they stopped their effort. Surprisingly, they performed rather well in the assembly task with our prototype system. They overcame a little difficulty and completed the task for the first time. In the second time, they smoothly completed the assembly.

In the beginning of their training with our system, some participants reported that they feel nervous to concurrently control multiple input channels of a pen. We noted that the participants' anxiety might be caused by their concerning manipulating the pen and the object concurrently. We requested them to focus their concentration just on manipulating the pen. Following our advice, and then the participants commented that they could comfortably manipulate a 3D object as their desire. In some pen-based interfaces, interaction with multiple input channels of a pen is sometimes prone to causing users' fatigue. This may be caused by that users are forced to concern the pen and the manipulated object concurrently. In our prototype system, this problem is smoothed down in some degree due to the consistency of the pen and object's locomotion. The consistency allows participants to just focus their concentration on either the pen or the object. The participants commented that this greatly relieved their nervousness.

4.6 Discussion

The operations relating to depth are difficult to perform, for lack of real 3D input devices. A Wacom digital pen can not support z-dimension input. In this chapter, we introduce some depth-related operations, i.e. rotating objects around an axis, rotating an object around an axis which is not on the screen, or tilting object(s) around a pivot when the tilting direction is not parallel to the screen. We also exploring translating object(s) with pen input pressure by dividing the pressure into three layers. The depth-related translation is similar to pressing object(s) into the water, heavier the pressure is, deeper the objects go down. The design is somewhat natural and intuitive, but not enough, since depth is not directly mapped from a spatial input. Some researchers (e.g. [88]) employ a Vision system to catch real 3D movement. But such a Vision system is too complex and expensive for most of the users. Some digital pens can also input negative pressure values with their eraser ends; contrarily, the pen tips input positive pressure values. We can also design translation by utilizing both the positive and the negative pressure input, e.g. the negative pressure can be used to pull objects near, while the positive pressure can be employed to push objects away.

Visual occlusion is a limitation for pen-based interfaces on direct devices, especially when the target is small in dimension. It is also a limitation, but this problem has been smoothed away in some degree in our design. This is because that the pen's physical feedback can be used when the objects are visually occluded. For an expert, the user can "feel" the objects' posture even without looking at them. Through our experience of using the prototype system, we find that the mapping between input channels and controlled parameters significantly effect users' performance. When the mapping is natural and intuitive, the user can vividly anticipate the target's transformation through the pen's movement. The anticipation, which can be served as pre-feedback, significantly enhances users' performance.

It is an open question to determine the utmost number of the input parameters of a pen that the user can manipulate concurrently in a computer task. Through this study, we find that natural and intuitive mapping enable a user manipulate all the possible input channels concurrently in one operation. In the prototype system, the user can vividly simulate some objects' movement around a pivot or an axis, e.g. the swinging of a pendulum, striking a bell with hammer, spinning a whipping top.

Most of the gesture-based interfaces [44] often suffer from self-disclosure problems. WIMP(Windows, Icons, Menus, Pointing devices) GUIs possess the characteristics of self-disclosure. Users can learn how to operate the interfaces through icons and menus. Comparatively, gesture interfaces are context-sensitive. The same gestures may stand for different commands in different contexts. The difference prohibits users to transfer their training experience between gesture interfaces. This problem has not been swept away but smoothed down in our prototype system. The natural and intuitive operation helps users to transplant their experience using a physical pen to manipulating 3D operations with a digital pen.

4.7 Conclusion

This chapter introduces how to naturally and intuitively manipulate 3D objects with multiple input channels of a pen. Three kinds of 3D transformation(i.e. translating, rotating and scaling) with multiple pen input channels are presented. Manipulating 3D objects with a pen can produce depth-related operation effects. A user can vividly simulate real objects' movement by utilizing a digital pen's input channels. Through the study, we also find out that the utmost number of input parameters that a user can concurrently manipulate is dependent on the mapping between pen input channels and the controlled parameters. Natural and intuitive mapping enables a user easily and concurrently to manipulate all the possible input channels of a pen. Natural and intuitive pen-based interfaces allow users to utilize a pen's physical posture as feedback. The physical feedback can smooth down some wide existing problems(e.g. visual occlusion and self-disclosure) in sketch-based interfaces. The informal user study reports rather positive results.

Chapter 5

An application paradigm of fluid and natural pen interaction techniques by utilizing multiple input parameters

Nowadays, commercial electronic pens commonly possess multiple input parameters (e.g. stroke, pressure, tilt angle, twist angle and azimuth). Current studies on the utilization of the parameters typically focus on the human ability to control the input parameters or on novel techniques which exploit only one parameter. In this chapter, we discuss how to employ multiple parameters for pen input to make operation more fluid and natural than the current case with traditional interfaces.

5.1 Introduction

Pen-based interfaces have been explored extensively in recent years. Nowadays, commercial electronic pens commonly possess multiple input parameters (e.g. stroke, pressure, tilt angle, twist angle and azimuth). The utilization of pen input parameters can widen the human-computer interaction bandwidth. Therefore, some studies have explored the human ability to apply pen input parameters in human computer interaction (e.g. [2, 4, 89]). Others have focused on pen parameter-enabled applications or techniques (e.g. [5, 7, 10]). However, few studies have explored the simultaneous utilization of more than one pen input parameter with the intention of making operation more fluid and natural than traditional interfaces.

Commonly, a computer task is performed through three phases in the following order: object selection, command selection and object manipulation. Switching between these phases can be performed by tapping on menu items or by pressing down predefined hot keys in the traditional WIMP (Windows, Icons, Menus and Pointing devices) interfaces. However, in most pen-based interaction environments, the hot keys are not available and it is tiring to move the pen tip repeatedly over long distances. Therefore, it is worthwhile to enhance the continuity of pen-based operations. Continuous interaction is a very important feature in pen-based user interfaces [22]. Liu and Ren have comprehensively evaluated six pen-suitable mode switching techniques [90] and proven that smooth operation with a pen-suitable switching mode is more efficient than the traditional interfaces [32] used in pen-based systems. Through the studies of Liu and Ren, we found that pen input parameters have the potential to make operation more natural and intuitive than traditional interfaces.

Based on these considerations, we designed four techniques, which integrate and exploit multiple pen input parameters and allow users to operate fluidly and naturally throughout the whole process from object selection to object manipulation. All the techniques were implemented and combined in a pen-based drawing application which we developed for testing.

5.2 Related Work

To date, there are many studies on the utilization of pen input parameters. These studies can be roughly divided into two categories. One category investigates the general human ability to control pen input parameters; the other category aims at enhancing the efficiency of human-computer interaction by implementing novel applications or techniques which exploit particular input parameters.

Up to now, the pressure parameter has been explored extensively. Herot and Weinzapfel [2] studied the human ability to apply finger pressure and torque to a computer screen. Buxton [3] investigated the use of touch-sensitive technologies and their potential for interaction. Ramos et al. [4] explored the human ability to vary pen-tip pressure as an extra channel of access to information. Ramos and Balakrishnan introduced some interaction techniques that employed the pressure parameter in pen-based operations, such as *pressure marks* [5] , *Zliding* [6] and *LEAN* [91]. Li et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interaction. Ren and colleagues introduced *ZWPS* [89], where pressure was used as a switch mode to couple a standard *Point Cursor* with a zoomable technique; and the Adaptive Hybrid Cursor [92], where pressure was used as an additional control factor to widen the adjustable range of the scrolling velocity. Yin and Ren [93] presented a novel Chinese calligraphy and painting system, where pressure input was employed to enhance the realistic sense of the user's manipulation of the pen. Harada et al. presented a set of interaction techniques that leveraged the combination of the human voice, pen input pressure and pen position when performing both creative 2D drawing and object manipulation tasks [8].

Pen input angles (e.g. tilt angle, azimuth and twist angle) are often used as UI clues for natural and intuitive interaction. Oshita [48] explored the utilization of pen position, pressure and tilt angle to make a virtual human figure perform various motions in response to the movement of a pen tip on an *indirect* tablet. However, the purpose of his study was to control virtual object motions which is different from our aim which is to design continuous interaction techniques. Balakrishnan et al. [9] introduced the *Rockin'Mouse*, which is a promising device for both 2D and 3D interaction that uses tilt input to facilitate 3D manipulation on a plane. Tian et al. introduced studies on utilizing tilt angle and azimuth with an intuitive cursor [94] and menu [10]. Bi et al. [13] explored general human performance with rolling angle (twist angle) and illustrated some natural and intuitive interaction examples with it.

Stroke, as a basic input channel of pen-based devices, has been explored extensively. Davis et al. introduced their *SketchWizard* [15],which is about wizard of Oz prototyping of pen-based user interfaces. Apitz and Guimbretire [14] presented their *CrossY*, in which pen strokes triggered all the drawing operations. Yin and Ren [95] presented a quantitative analysis of both Arc and Line stroke-based techniques for scrolling in pen-based interfaces, and compared them with traditional scroll bars. Their experimental results indicated that the Line technique outperformed both the Arc technique and the traditional Scroll Bar.

In summary, our review indicates that although there are a few studies which have utilized pen input parameters for designing interaction techniques, none of the studies addressed the issue of multi-parameter pen input to make operation fluid and natural. Our basic motivation is to investigate the potential of multi-parameter pen input to make operation more fluid and natural. Thus, this study will benefit HCI literature in a number of important ways.

In the following sections, we introduce four techniques to present how to make operation fluid and natural with multi-parameter pen input. The operation and techniques are introduced according to the common order of operation (i.e. object selection, command selection and object manipulation) in a computer task.

5.3 Object Selection

The standard rubber-band selection method is effective to select multiple targets in a regular layout. However, it is difficult to select multiple targets in an irregular layout. The limitation of the standard rubber-band multi-target selection and other pen-based multi-target selection techniques [21] inspired us to find a more flexible and ubiquitous multi-target selection technique. As suggested by [14, 19] crossing performs better than pointing-and-clicking in UI design, especially for pen-based input devices. In our pen-based drawing application, we present a pressure-based crossing selection method, which is similar to "*Line-string*" [21], which employs a line stroke to "string" and select targets. Although the technique is promising under some conditions, their experiment showed that it was difficult to select scattered targets in an irregular and dense layout since all targets that were crossed were strung together and were thereby selected. Thus, we introduce a more flexible pressure-based crossing selection technique, by which an object can easily be "jumped over" by a pen stroke.

5.3.1 Design

In our pen-based drawing system, pressure is used as a switch mode to couple normal stroke and crossing selection functions in a continuous stroke. A pilot study determined the right pressure spectra for normal stroke and crossing selection. In the pilot study, 12 participants were asked to draw with their own sense of light pressure, normal pressure and heavier pressure alternately on a WACOM combined tablet-display which has 1024 levels of pressure. The results showed a statistically significant difference on the *maximum pressure scale* of a stroke between the light, normal and heavier pressure conditions. In our implementation, the heavy spectrum of pressure was employed for crossing selection and the normal spectrum for normal stroke; because low end spectrum is more difficult to control [6], it was omitted from the final design.

5.3.2 Operation

If no operation command (e.g. drawing a certain shape, setting objects' color) is registered by the system, it is *system selection phase*, then the user can select target objects. The user strokes the pen starting from a *blank area*, where there are no objects. If the pressure input surpasses the specified threshold, the stroke will switch to perform pressure-based crossing selection; otherwise it will be a normal stroke. Under this selection mode, the user only needs to stroke the pen on the screen and all the objects crossed by the pen will be selected (See Figure 5.1). A blue footprint line is used as visual feedback for crossing selection. If there are some objects that the user does not want to select in the path of the selection stroke, s/he can steer clear of them, or reduce the pressure on the pen tip to less than the threshold without lifting it from the screen. The blue footprint line will disappear and the figure will be "jumped over" (crossed by the stroke without being selected) (Figure 5.2).

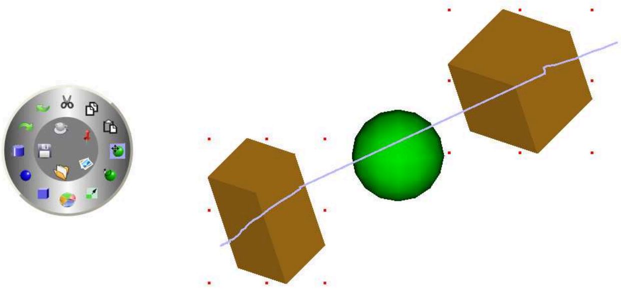
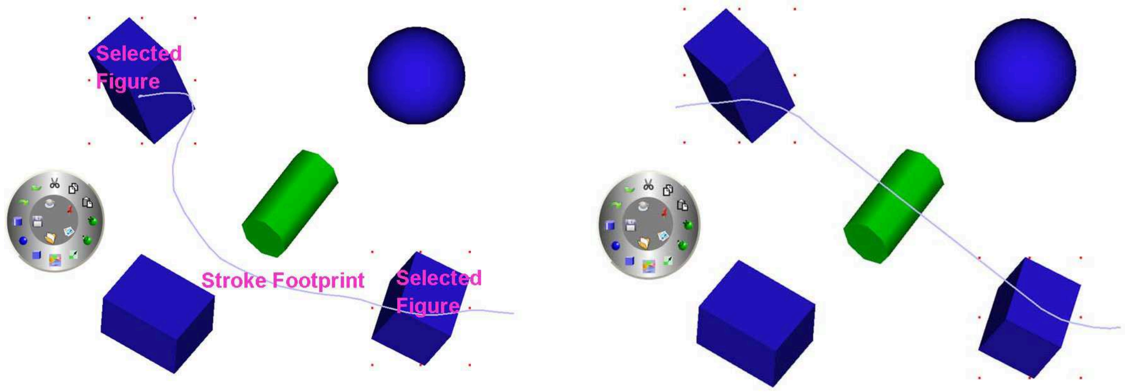


Fig. 5.1 Pressure-based crossing selection.



(a) Steer clear of an object. (b) Ignore an object crossed by the stroke.

Fig. 5.2 Flexible selection process.

5.3.3 Undo

The user can stroke the pen backwards and cross the footprint line on a selected object to undo its selection. If the user lifts the pen tip and then taps the display in a *blank area*, all selections will be canceled.

5.4 Operation command selection

The hierarchical menus, which are initially designed for mouse-based user interfaces, tend to be awkward in pen-based systems. Many researchers, e.g. [10, 96] aimed their studies at finding pen suitable menus. To improve the continuity and efficiency of operation, we designed a pen suitable menu (called *space menu* hereafter), which is tilt and azimuth based. The user can complete all editing operations on the selected figure/s by utilizing a *space menu* without lifting the pen from the screen (Figure 5.3). The *space menu* is introduced in detail in next section.

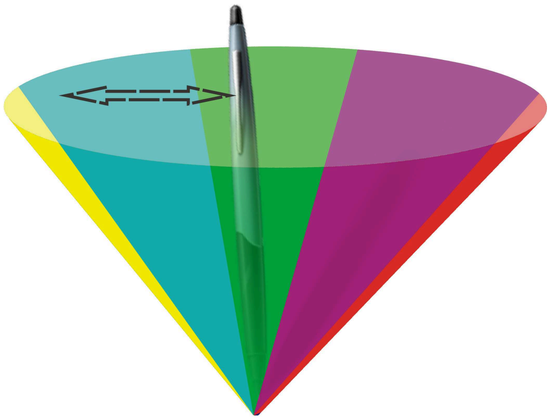


Fig. 5.3 A *space menu* (concept map). Each orientation region corresponds to a specific menu command.

5.4.1 Space vs. commands mapping

The 3D space around the pen tip is divided into regions according to the pen's tilt angle and azimuth. Each region is delegated a certain operation command. When the pen is directed to one of the regions, the corresponding operation command will be activated by the pen's pressure input if it surpasses the specified threshold [89]. According to previous research in our group, human pen rotation control performance differs in different spectra of pen tilt angle and azimuth. Therefore, more commands are mapped to the spectra in which subjects perform better. Unlike the *Tilt Menu*'s [10] indirect mapping between pen 3D orientation and menu command, for a *space menu*, each command is mapped directly with a 3D orientation region. An expert user can point the pen directly and rapidly to a command region without looking at the command prompt.

5.4.2 Fluid and continuous operation

The user directs the pen to a certain 3D orientation region (called *command sensitive region*), a text prompt of the corresponding command appears on the screen. By applying a little more pressure (surpassing the specified threshold [89]) the user can affirm the menu command selection, and a text prompt will appear beside the pen tip. By keeping the pen out of any *command sensitive region* and stroking on the selected objects, the selected command (e.g. setting the objects' color or stroke width) will be executed. If the user wants to perform another operation, s/he can adjust the pen's posture into *command sensitive region* to select the corresponding command. During the object selection phase, the command selection phase and the object manipulation phase, the pen tip need not to be lifted from the screen. All the operations can be done fluidly and continuously.

5.5 Pen suitable operations

After selecting the command, the user can manipulate the selected objects according to the command. In this section, we briefly introduce some pen suitable operations.

5.5.1 Object rotating

In the system, the user can rotate the selected 3D figures by changing the tilt angle of the pen. Firstly, the user slides the pen tip onto the figures to set a rotation axis. The user then changes the pen tilt angle and the figures rotate around the rotation axis together with the pen (Figure 5.4). In the process of rotation, the user can lift the pen tip from the screen and stroke on the figures to set another rotation axis. Visual consistency between the hand and eyes is maintained throughout the manipulation.

Under rotation operation mode, when the user puts the pen tip on the display and strokes it on the selected figure/s, the program records the stroke and approximates it into a line segment, and then sets the line as a rotation axis. Then, when the user changes the pen's tilt angle on the selected figure/s, the program maps (according to specified formulas) the difference between the tilt angle and that of the X,Y,Z coordinates of the 3D figure/s, and redraws the figure/s instantly. The user can see the rotation of the 3D figure/s as the pen is rotated (Figure 5.4).

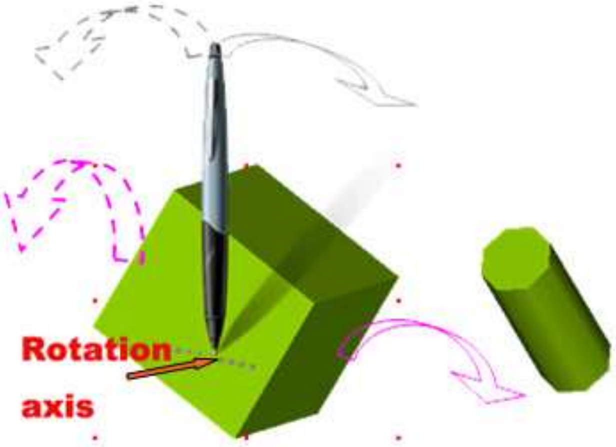


Fig. 5.4 Pen tilt angle controls 3D figure rotation. After setting the rotation axis on the figure, users rotate the pen and the figure will rotate around the rotation axis together with the pen.

5.5.2 Stroke setting

We designed a crossing-based stroke setting technique, which combined pen tip stroke and pressure input to set the stroke width of 2D figures. The user slides the pen tip on the figures, changes the stroke width by sliding the pen tip starting from the selected figures, and adjusts the changing scale by adjusting the pen pressure; the heavier the pressure, the greater the ratio of *stroke width per sliding length* will be. We introduce a simple pen gesture in this function to make the operation more flexible. If the user slides the pen tip clockwise, the selected figures' stroke width becomes wider.

Conversely, if the user slides the pen tip anticlockwise, the figures' stroke width becomes thinner (Figure 5.5).

The program senses the direction of the sliding pen (clockwise or anticlockwise) from the pen stroke, a changing scale (hereafter referred to as S) from the pen pressure and the recalculated length (hereafter referred to as L^*) form the pen stroke length. We calculate the stroke width (referred to as W) by the formula (5.1).

$$W = \begin{cases} W - L^* \times S & \text{if crossing is } \textit{anticlockwise} \\ W + L^* \times S & \text{if crossing is } \textit{clockwise} \end{cases} \tag{5.1}$$

If the pen pressure surpasses the specified threshold [89], S will be a number more than 1; otherwise, S will be a number not more than 1. The resolution of the screen is taken into account when we calculate, so that the changing velocity of the stroke will not be too high on a small screen and it will not be too low on a large screen. If the pen tip is slid anticlockwise, W will be calculated by the first equation; otherwise, it will be calculated by the second equation. The minimum of W is 1 pixel and the maximum is 30 pixels. While the user is sliding the pen tip on the selected figure/s, the figure/s is redrawn and the operation effect can be seen simultaneously (Figure 5.5).

5.6 User Study

To test the proposed pen suitable interaction techniques, we designed and implemented the techniques integrated into one pen-based drawing application. In the application, the user can complete all drawing operations and manipulate different 2D or 3D figures without using the keyboard or the mouse. The application fully utilizes multiple pen input parameters, and makes drawing post-manipulation with a pen more fluid and natural than traditional input interfaces.

We performed an informal user study, with six participants, on a Wacom LCD

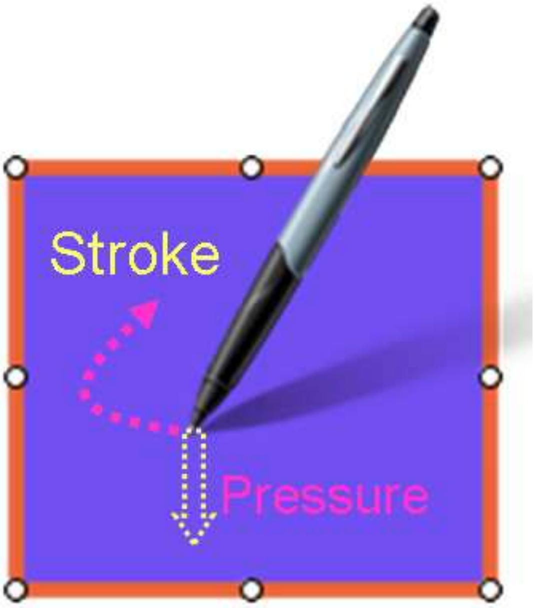


Fig. 5.5 Utilizing pen stroke and pressure to set a 2D figure's stroke width.

graphics display tablet with a digital pen, which possesses pressure, tilt angle and azimuth input. In the informal user study, we found that most of the participants had a preference for simple but effective operations (e.g., utilizing pen pressure to adjust a figure's shape). But for slightly more complex operations (e.g., utilizing pen tilt-angle to rotate a 3D figure) the participants with more experience using a pen gave better evaluations than the participants who had only a little such experience. At the beginning of the user's experience with the application, some participants with little experience using a pen felt a little confused and anxious about controlling more than one input parameter at a time. Nevertheless, they did rather well after practicing with the application for no more than five minutes. We also found that the mapping of pen input parameters and program functions has a remarkable effect on user experience and skill acquisition rate. Natural and intuitive mapping, especially mapping that has the advantage of visual consistency, can help users grasp the corresponding operation easily and quickly.

5.7 Discussion

We have presented four pen-based interaction techniques which are based on utilizing multiple pen input parameters. All these techniques integrate the operational functions of preview, display, undoing, redoing and setting in one stroke. For example, when setting a 2D figures' stroke width, the user slides the pen on the figure in one direction (clockwise or anticlockwise). Then the figures' stroke width is changed and displayed on the figures. If the pen is lifted from the figure, the figure's stroke width will be to set, and the previous display serves as preview. If the user slides the pen in the opposite direction, the operational effect will be to undo the previous operation. If the user then slides the pen in the original direction, the operation effect will be to redo the original operation. This kind of design merges command selection and direct manipulation [16], gives prominence to the features of the pen and makes it possible for all manipulation to be carried out continuously and fluidly [22].

There are a lot of studies on selection-action interaction, such as selection-action interaction with pressure marks [5], with a combination of lassos and pigtails [17] and with rubber-banding [97]. All these studies explored valuable pen suitable selection-action interaction. But in all the above techniques, only one action can be executed with each selection, and the action is not independent of a selection. For example, for *pressure marks*, variations in pressure were integrated into pen strokes to indicate both a selection and an action simultaneously. Undoubtedly, this design could significantly enhance operation efficiency by eliminating the interval between selection and action. But it was impossible to change the action after the selection, since they were both integrated into the same pen stroke. By contrast, with our *space menu*, the user can perform continuously as many actions as s/he needs to after one selection. The actions can be combined with one selection whether it is a lasso, rubber-banding, keyboard-aided or pressure-based crossing selection. Therefore, with a *space menu*, the selection-action interaction will be more flexible and efficient.

Fluid interaction in pen-based systems is critically significant. This chapter has shown that the utilization of multiple parameters can make interaction continuous, i.e. all the operations of a whole computer task can be performed by one stroke. The features of the four techniques we have presented are:

- *The pressure-based crossing selection technique* is more flexible and more promising than other multi-target selection techniques in some application scenarios, e.g., to select some scattered targets in an irregular and dense layout.
- *The space menu* is orientation-based, where the intuitive pen input angles are employed to indicate menu commands. After the target selection, more than one action can be performed continuously.
- The *object rotating* technique manipulates a 3D figure's rotation by employing pen input tilt angle, which is consistent with the pen's own physical posture. Therefore, our designed rotation operation is more intuitive than with a mouse or a keyboard.
- The *stroke setting* technique changes a 2D figure's stroke width by utilizing pen crossing combined with pen pressure input. This technique is sensitive to the pen tip's crossing space length, direction and pen tip pressure input. Thus a figure's stroke width can be adjusted rapidly and precisely.

Although the proposed operation semantic and the techniques are novel, the user study reported that users could easily master the operations.

5.8 Conclusion

In this chapter, we investigate the potential of multi-parameter pen input for fluid and natural operations. We also present an operation semantic that allows fluid and natural operation in pen-based systems. In another study of the authors [32], we have proved that fluid operation can significantly improve interaction speed and accuracy in pen-based systems. A flexible pressure-based multi-target selection technique and a natural and intuitive angle-based menu design technique are also introduced. We also illustrate two pen suitable operations in the 2D and 3D drawing applications. The user study has proved the usability of our proposed operation semantic and UI design techniques. We believe that our presentation has strong implications for fluid and natural pen-based UI design, and the implications are valuable for pen-based UI designers, especially for those who are seeking to develop crossing-based user interfaces.

Chapter 6

A drawing protosystem with seamless and continuous operation semantic

Current studies on the utilization of pen input parameters are typically focused on the human ability to control pen parameters or on novel techniques which exploit a certain parameter (e.g., pressure or tilt angle). In this chapter, we present a versatile 2D and 3D drawing system (called PenOpera) which employs more than one pen input parameter to make drawing post-manipulation more natural and flexible than traditional interfaces. Some pen-suitable techniques and possible guidelines on how to design and implement multi-parameter pen input systems are also introduced in this part.

6.1 Introduction

There are many studies on how to utilize pen input parameters (e.g., pen pressure, tilt angle, azimuth, and position). Some of these explore the human ability to apply pen parameters in human computer interaction [2–4,89,98]. Others focus on pen parameter-enabled applications or techniques [5–7,9,11,12,89,91,94]. We can derive some useful guidelines on utilizing pen input parameters and many novel applications and techniques from these studies. But we are unaware of any research that has explored the utilization of more than one pen input parameter in one application. Thus, we designed and implemented this multi-parameter pen input application, PenOpera. At this point, PenOpera is a prototype system, but it is robust enough for our research work. In PenOpera, the user can complete all drawing operations and manipulate different 2D or 3D figures without using the keyboard or the mouse. PenOpera fully utilizes some pen input parameters (e.g., pressure, tilt angle, azimuth, position), and makes drawing post-manipulation with a pen more natural and flexible than traditional input interfaces. Many novel techniques are incorporate into our prototype system. PenOpera is not only an applied system but also a case study on multi-parameter pen input software. Many helpful points for designing multi-parameter pen input can be derived from our work.

PenOpera comprises three parts: a 2D drawing subsystem, a 3D drawing subsystem and a common main software framework (6.1). In the system, we afford seven 2D shapes and three 3D shapes. The user can draw each of these shapes with just one stroke of the pen. In this chapter, we will not show how to create a 2D or 3D object. Our presentation focuses on how to manipulate the drawn figures naturally and flexibly using the pen only and continuously i.e. without using the mouse or keyboard.

According to an informal user study, our prototype system is easy for a common user to grasp most of the operations within five minutes.

6.2 Related Work

To date, there are some studies on the utilization of pen input parameters (e.g., pen pressure, tilt angle, azimuth and position). These studies can be roughly divided into two categories. One category investigates the general human ability to control pen input parameters; the other category aims at enhancing performance of human and computer interaction by implementing novel applications or techniques which exploit particular input parameters.

There are many studies on pen pressure. Herot and Weinzapfel [2] studied the human capability of the finger to apply pressure and torque to a computer screen. Buxton [3] investigated the use of touch-sensitive technologies and the potential for interaction that they suggested. Ramos, et al. [4] explored the human ability to vary pen-tip pressure as an additional channel of access to information. Ramos and Balakrishnan introduced *pressure marks* [5] , Zliding [6] and LEAN [91]. Pressure marks can encode selection-action patterns in a concurrent, parallel interaction. In Pen strokes, variations in pressure make it possible to indicate both a selection and an action simultaneously. Zliding explores integrated panning and zooming by concurrently controlling input pressure while sliding in X-Y space. LEAN employed a set of novel interaction techniques for the fluid navigation, segmentation and annotation of digital video. Li, et al. [7] investigated the use of pressure as a possible method to delimit the input phases in pen-based interactions. Ren et al. introduced their ZWPS [89] and the adaptive hybrid cursor [92]. In ZWPS, pressure is used as a switch mode to couple a standard Point Cursor and a zoomable technique; and for the adaptive hybrid cursor, pressure is used as an additional control factor to widen the adjustable range of the scrolling velocity. Harada et al. presented a set of interaction techniques that leveraged the combination of human voice and pen pressure and position input when performing both creative 2D drawing and object manipulation tasks [8].

There are also some studies on tilt angle and azimuth. Oshita explored utilizing pen position, pressure and tilt angle to make a virtual human figure perform various motions in response to the pen movements [48]. Balakrishnan et al. [9] introduced the Rockin'Mouse. The Rockin'Mouse is a promising device for both 2D and 3D interaction that uses tilt input to facilitate 3D manipulation on a plane. Tian et al. [94] explored the Tilt Cursor. The Tilt Cursor is a type of cursor that dynamically reshapes itself to provide the 3D orientation cue of a pen. Some other studies such as TiltType [11] and TiltText [12] focus on using the tilt information of mobile phones to affect text entry tasks in mobile devices.

There are also a lot of studies on utilizing pen position input in pen suitable applications, such as SketchWizard [15] and CrossY [14]. Further discussion about this problem is outside the scope of this chapter.

Up to now, we are unaware of any work which addresses the issue of multi-parameter input in one pen-based 2D/3D drawing application. Only pressure input has typically been used by a few drawing and image manipulation programs, like Adobe Photoshop, to modulate the parameters of the active brush, such as stroke thickness or color opacity. Therefore, our basic motivation is to investigate the potential of multi-parameter pen input in applied software.

6.3 User Interface

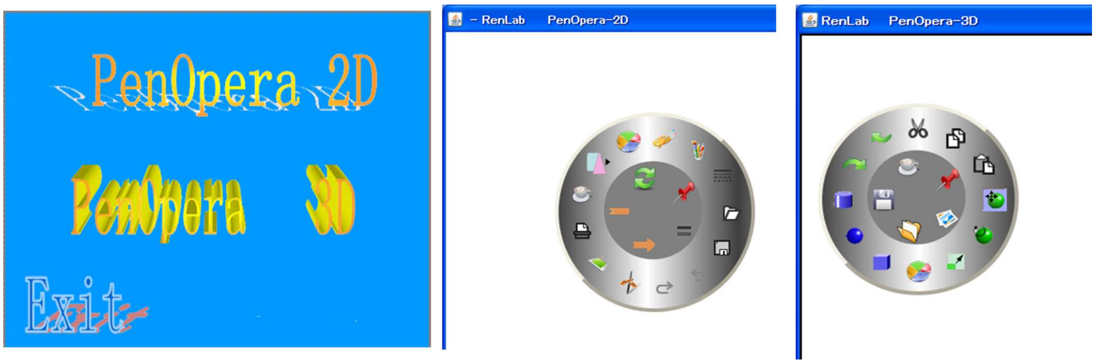


Fig. 6.1 PenOpera.

PenOpera's physical user interface is based on a touch-sensitive screen/tablet, which affords pressure, tilt angle and azimuth. PenOpera comprises a 2D drawing subsystem, a 3D drawing subsystem and a common main software framework (see Figure 6.1). In either the 2D subsystem or the 3D subsystem, the user interface includes operation space and menus. We employ two different kinds of menu. One is a pressure based pie menu (see Figures 6.2 and 6.3), called pressure menu. The other is a space menu (Figure 6.4), i.e. pen gesture menu. All effective commands could be located in space menus or pressure menus, depending on the designer's preference. All menu items are circularly arranged in pressure menus. There is a main control menu in the 3D subsystem, and there is a main control menu and a shape submenu (Figure 6.5) in the 2D subsystem. There are three different ways to arrange a pressure menu and its submenu. The first way is to spread its submenu items in the main menu i.e. to reveal the submenu in the main menu. The second way is to display the submenu beside the main menu. The third way is to display the submenu and hide the main menu at the same time. We arrange the menus in the latter way (Figure 6.5) to keep the window and menu clear and simple and to leave more work-space in the operation window. As with tracking menus [96], pressure menus also have two states: tracking state and inactive state. In each pressure menu, a pushpin button (with a red pushpin icon, see Figures 6.3 and 6.5), controls the menu's state. The user can press the pushpin to switch the pressure menu's state. This is the same with the tracking menu, but unlike the tracking menu, the user need not move the pen to press the pushpin when s/he wants to alternate the pressure menu's state. A pen's pressure change (surpassing the specified threshold, 1000 units, and then released) at any point of the operation window can activate the pressure menu switch and instantly relocate the pressure menu close to the cursor (Figure 6.6 left). When the user operates with pressure menus in a large window (e.g., whiteboard) much pen movement time can be saved. In the tracking state, a pressure menu locates itself south-west of the cursor and relocates itself near the south or west edge of the window (Figure 6.6 right). Further discussion about pressure menus is outside the scope of this chapter.

In the 2D subsystem, the user can complete all editing operations on the selected

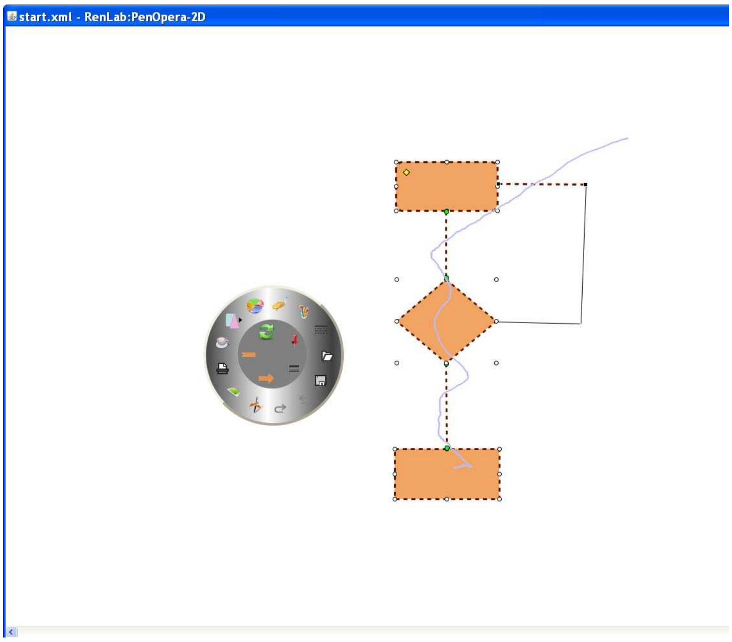


Fig. 6.2 The PenOpera 2D subsystem user interface.

figure/figures by utilizing space menus without lifting the pen from the touch-sensitive screen/tablet. The user presses a certain menu item (with a black triangle in its icon, Figure 6.2) in the pressure menu to remove the limitation of space menus. The user adjusts the pen's tilt angle to a specified spectrum, (60°, 80°), to activate the space menus. The user directs the pen to a certain orientation region (corresponding to a certain azimuth spectrum, which is related to a specific menu command) and then presses the pen a little more heavily (surpassing the specified threshold, 850 units [89]) to affirm the menu command selection.

The first and most important feature of this function is that the user can continuously perform different editing operations without lifting the pen tip from the touch-sensitive screen/tablet. Continuous interaction is a very important feature in pen-based user interfaces [22]. Therefore, much command selection time can be saved. The second feature of this function is that it is more intuitive than traditional menus, because each command menu is related to a certain space region. When the user selects a specific command, a corresponding text prompt appears on the screen. But an experienced user can choose to close the prompt information. And then space menus will occupy no space in a window, they are especially useful for small displays (e.g., screens of PDAs or intelligent mobile phones).

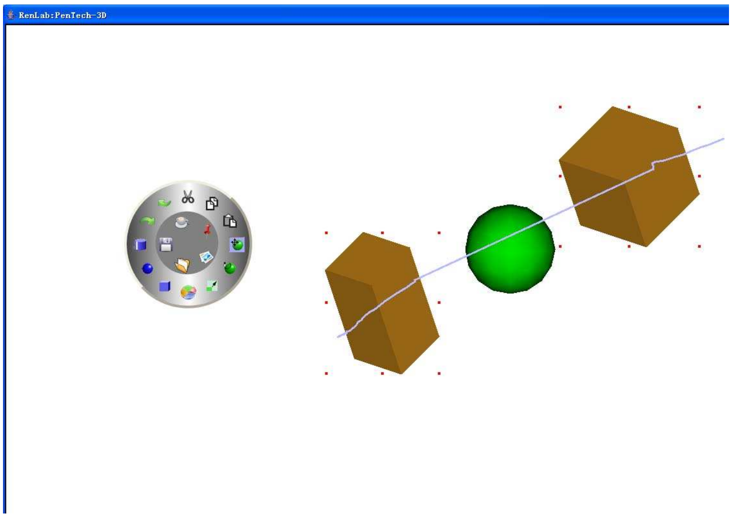


Fig. 6.3 The PenOpera 3D subsystem user interface.

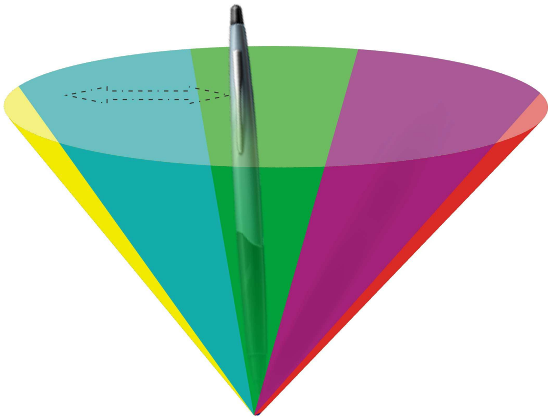


Fig. 6.4 A space menu (concept map). Each orientation region corresponds to a specific menu command.

Fig. 6.5 A pressure menu and its submenu.

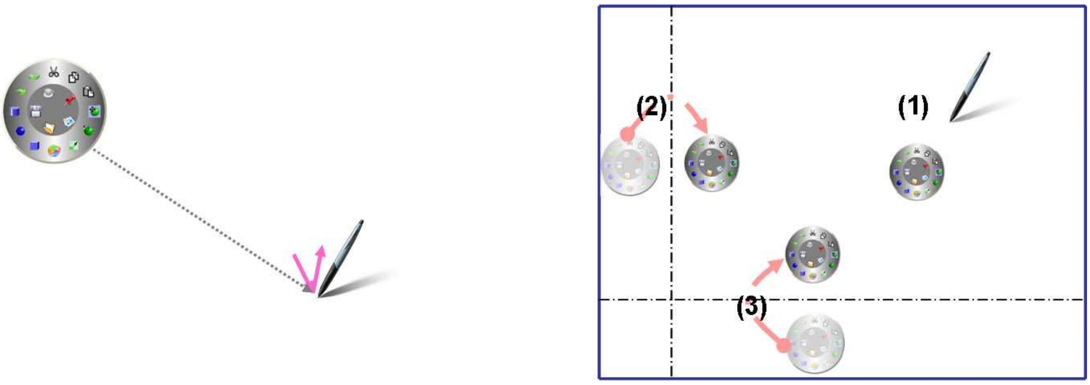


Fig. 6.6 Pressure menu. Pen pressure change can activate a pressure menu and move it near to the cursor(left);A pressure menu's common position according to the cursor(1), a pressure menu relocates itself near the window's west or south edge (2)(3) (right).

6.4 Operation from users' view

6.4.1 Pressure-based crossing selection operation

In the system, we present a pressure-based crossing selection method, keyboard-aided multi-selection method and rubber band multi-selection method. The latter two methods are both traditional and will not be discussed in this chapter. As suggested by Accot [19] and Apitz [14] crossing performs better than pointing-and-clicking in UI design, especially for pen-based input devices. Here, pressure is used as a switch mode to couple the traditional rubber band multi-selection with the pressure-based

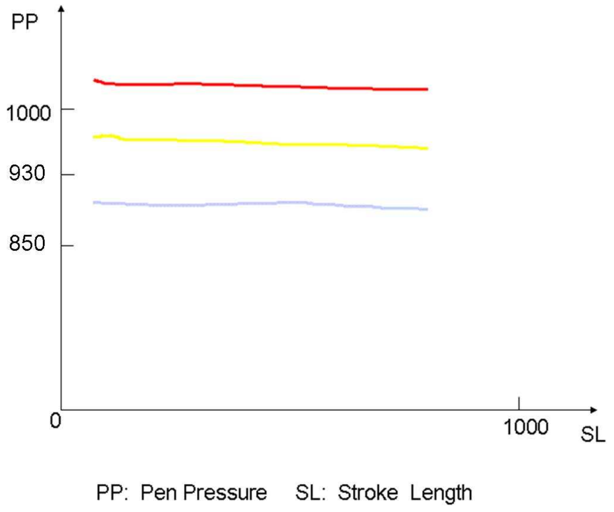


Fig. 6.7 The footprint of a pen in pressure-based crossing selection.

crossing selection. If the pressure input surpasses the specified threshold (850 units [89]), the selection method will be pressure-based crossing selection; otherwise it will be rubber band multi-selection. Contrasted with the traditional way, the pressure-based crossing multi-selection method is more flexible, easier to use and quicker when selecting a number of irregularly deployed objects from among other objects. Under this selection mode, the user only needs to stroke the pen on a screen/tablet and all the objects crossed by the pen will be selected. In the 3D subsystem, the user first presses the move & select menu item in pressure menu. The user then slides the pen a little more heavily (surpassing the specified threshold, 850 units [89], and less than 930 units) and a blue footprint line will appear in the path of the pen stroke (Figure 6.7), and all the objects crossed by the stroke will be selected. In the 2D subsystem, at first, the user should adjust the pen's tilt angle to the spectrum of (20°, 55°], then slide the pen while applying a little more pressure (surpassing the specified threshold, 850 units [89], and less than 930 units). The rest of the operations are the same as in the 3D subsystem. If there are some objects that the user does not want to select in the path of the selection stroke, s/he can reduce the pressure on the pen to less than the threshold without lifting the pen tip from the screen/tablet, until the blue stroke line disappears. Then the figure will be crossed by the stroke without being selected (Figure 6.8). In the PenOpera 2D subsystem, the user can also adjust the pen's tilt angle over the selection and drag control threshold, 55°, to ignore the crossed figure.

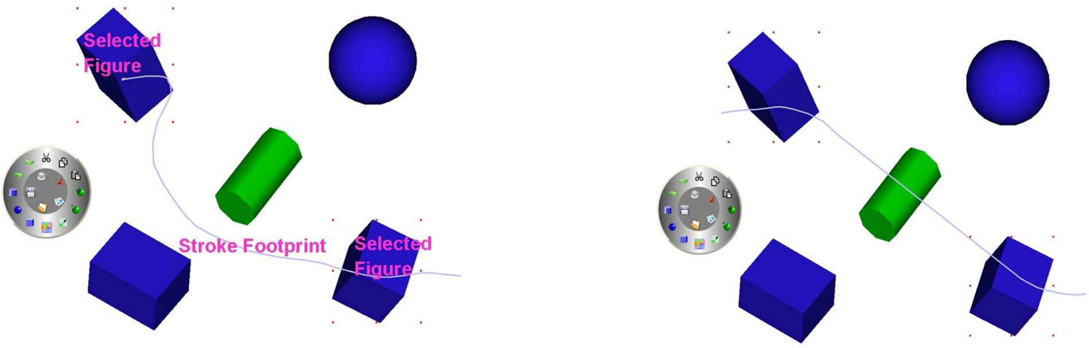


Fig. 6.8 (a)Steer clear of an object. (b) Ignore an object crossed by the stroke.

In the system, the pressure menu's state switch is also triggered by pressure. In order not to trigger the pressure menu during a pressure-based crossing selection process, we include three kinds of pen footprint each with a different color (Figure 6.7). If the footprint is blue, it means that users can perform a selection operation without triggering the pressure menu's state switch. If the footprint turns yellow, then the user should reduce the pen pressure. If the footprint turns red, it means that the pressure menu's state has been switched. In the informal user study, users generally grasped the operation in several minutes.

6.4.2 3D rotation controlled by pen tilt angle

In the 3D subsystem, the user can rotate the selected 3D figure/figures by changing the tilt angle of the pen. The user selects the figure/figures that s/he wants to rotate. The user presses the rotation menu item in the pie menu. Then the user slides the pen tip onto the selected figure/figures to set a rotation axis. Finally, the user changes the pen tilt angle, and then the figure/figures will rotate around the rotation axis together with the pen (Figure 6.9).

Fig. 6.9 Pen tilt angle controls 3D figure rotation. After setting the rotation axis on the figure, rotate the pen and the figure will rotate around the rotation axis together with the pen.

In the process of rotation, users can lift the pen tip from the screen/tablet and stroke on the selected figure/figures to set another rotation axis. Thus, the figure/figures can be rotated arbitrarily in 3D space.

The operation is very natural since it employs the pen's tilt angle to change the 3D figure's tilt angle. The user only needs to stroke and rotate the pen on the figure/figures, and then s/he can rotate it/them arbitrarily. This operation is very flexible, natural and easy to use.

6.4.3 Utilizing pen azimuth to control 2D figures' orientation

In the PenOpera 2D subsystem, the user can change the orientation of the triangle/triangles (Figure 6.10) by adjusting the pen's azimuth. The user presses the orientation menu item in the main pressure menu of the PenOpera 2D subsystem. The user selects the triangle/triangles, for which orientation is to be set. The user puts the pen tip on the selected triangle/triangles. Finally, the user changes the pen's azimuth; and then the triangle/triangles will change its/their orientation together with the pen's rotation (Figure 6.10). Traditionally, this kind of manipulation can be performed by clicking an arrow key on the keyboard or by dragging a mouse, but employing a pen makes such manipulation more natural, flexible, intuitive and easy.

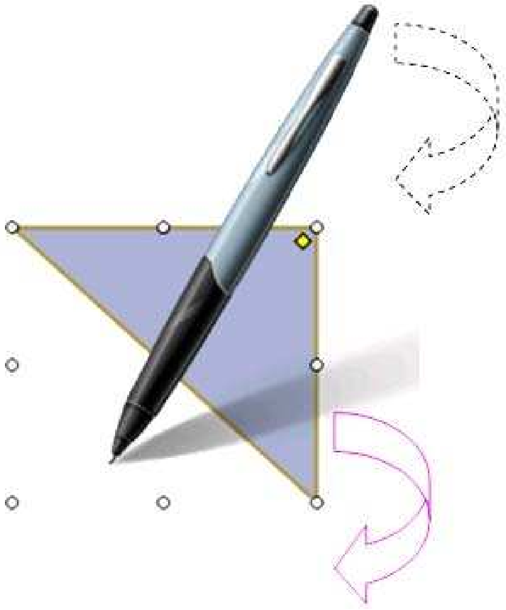


Fig. 6.10 Utilizing pen azimuth to control a triangle's orientation.

6.4.4 Utilizing pen pressure to adjust a 3D figure's shape

In the 3D subsystem, the user can press the pen a little more heavily (pressure surpassing the specified value, 850 units [89]) to slightly adjust a figure's shape. For example, the user can change a cuboid into a cube by pressing the pen on a screen/tablet more heavily when drawing a cuboid. Traditionally, users need to press a modifier key when they perform the same task with a mouse and a keyboard. Employing a pen makes such modification quicker and easier, especially when a figure is being drawn on a sensitive screen/tablet where it is difficult to reach a modifier key.

6.4.5 Utilizing stroke and pen tilt angle to set fill or stroke colors

In our prototype system, the fill color or stroke color of 2D/3D figures can be set by utilizing stroke and pen tilt angle. The user selects the 2D/3D figure/figures, for which color is to be set. The user selects the fill/stroke color menu item in the pressure menu of the 2D/3D subsystem. Then the user slides and rotates the pen on the touch-sensitive screen/tablet passing through the range of colors which are available for selection. During the process, the user can undo previous operations by sliding and rotating the pen in the reverse direction. The user stops the rotation of the pen at the desired color and selects that color by taking the pen off the screen/tablet surface.

6.4.6 Utilizing pen stroke and pressure to set the stroke width of 2D figures

In the 2D subsystem, the user can set the selected figure/figures' stroke width by stroking on it/them. The user presses the stroke width menu item in the main pie menu of the 2D subsystem. The user selects the figure/figures for which the stroke width is to be changed. Then, the user slides the pen tip on the figure/figures to adjust its/their stroke width. During this process, the user can change the rate at which the width will change by adjusting the pen pressure; the heavier the pressure, the quicker the change will be. We introduce a simple pen gesture in the function to make the operation flexible. If the user slides the pen tip clockwise, the selected figure/figures' stroke width becomes wider. Conversely, if the user slides the pen tip anticlockwise, the selected figure/figures' stroke width becomes thinner (Figure 6.11).

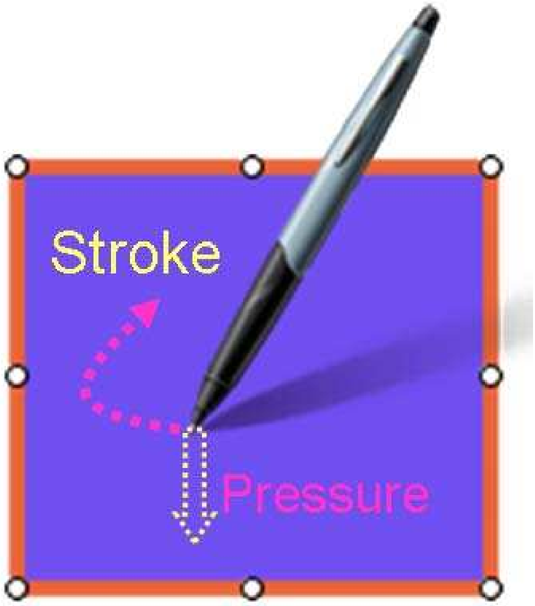


Fig. 6.11 Utilizing pen stroke and pressure to set a 2D figure's stroke width.

6.4.7 Utilizing pen stroke to set dash parameters, and beginning and/or ending arrows for 2D figures

In the 2D subsystem, the user can set dashed lines, and beginning and/or ending arrows using the pen's stroke and pressure (Figure 6.12). The user selects the figure/figures to which s/he wants to set dashed lines or beginning and/or ending arrows. The user presses the dash/arrows menu item in the main pie menu of the 2D subsystem. Then, the user slides the pen tip onto the selected figure/figures to set its/their properties.

The main features of this function are continuous operation [22] and prompt preview. Under this operation mode, the user can slide the pen tip onto the selected figure/figures to set its/their properties to different values continuously [22], i.e. in one process, and the operation effect can be seen at once. Another significant feature of this function is that it is gesture-based: the software can distinguish between two gestures, clockwise sliding and anticlockwise sliding. Clockwise sliding changes the dashes/arrows

(a) Utilizing pen stroke to set stroke dash.

(b) Utilizing pen stroke to set arrows of lines.

Fig. 6.12 Set stroke dash or line arrows.

in one sequence; anticlockwise sliding changes the dashes/arrows in the reverse sequence. Thus, the user can go back to the foregoing choice at any time during the operation.

6.5 Algorithm

6.5.1 Logical software structure

There are five layers in PenOpera (Figure 6.13). The first layer presents pen/tablet API (Application Programming Interface). Based on the first layer, many parameters (e.g., pen pressure, tilt angle, azimuth, position, etc.) can be accessed. Many fundamental empirical studies on the second layer have been done to evaluate human performance capabilities when controlling these parameters [2–4]. Most functions of PenOpera (e.g., keep tracking method, pressure-based selection method, angle rotation method, etc.) are presented in the third layer. Based on the third layer, the program draws and controls 2D/3D figures in the fourth layer. In the last layer, the program can output a 2D/3D drawing into a file.

6.5.2 Mapping between pen input parameters and drawing post-manipulation

Table 6.1 Pen input parameters and drawing post-manipulation mapping table. Here, S stands for the parameter utilized as discrete state and V stands for the parameter utilized as a continuous variable.

Parameters / Manipulation	Pressure	Tilt angle	Azimuth	Position
Pressure menus	S			
Pressure-based crossing selection	S	S		
Pen tilt angle controlled 3D figures' rotation		V		
Utilizing pen azimuth to control 2D triangle's orientation			V	
Utilizing pen pressure to adjust a 3D figure's shape	S			
Utilizing stroke and pen tilt angle to set 2D/3D figures' fill color or stroke color		V		V
Utilizing pen stroke and pressure to set 2D figures' stroke width	S			V
Utilizing pen stroke to set 2D figures' stroke dash, lines' beginning and/or ending arrows				V
Space menus	S	V	V	

Pen pressure [89], tilt angle and azimuth can be typically used to perform interactions by being mapped to several discrete states, or by controlling a continuous variable. The mapping between pen input parameters and drawing post-manipulation is shown in Table 6.1. As found by previous researches [89, 98], the degree of pen pressure perceived by human users is not consistent with that sensed by a digitizer. The previous work on pen tilt angle and azimuth in our laboratory also shows that a human performs differently when controlling the pen's rotation in different ranges of pen tilt angle or azimuth. The mapping between pen input parameters and drawing post-manipulation is well designed according to the previous work and our pilot study. In the design of the mapping between multi-parameter pen input and drawing post-manipulation, there are some problems that should be taken into account. According to the previous work, we know that a human user displays different levels of proficiency to control a pen in different spectrums of the various pen input parameters. In PenOpera, we mapped the drawing post-manipulation to the spectra of pen input parameters, in which users perform better. At the same time, the mapping should make the operations in PenOpera simple enough for a common user to grasp. Through good design, we also achieved the goal of making the manipulation natural, flexible and intuitive (i.e., with the advantage of visual consistency).

6.5.3 Algorithm of each manipulation point

Pressure-based stroke selection operation

Besides keyboard aided multi-selection, we present two stroke-based multi-selection methods. One is rubber band selection and the other is pressure-based crossing selection. The two methods can be switched by pen pressure.

In the 3D subsystem, the multi-selection method switch is controlled by pen pressure. If the pressure value surpasses the specified threshold (850 units [89]), the multi-

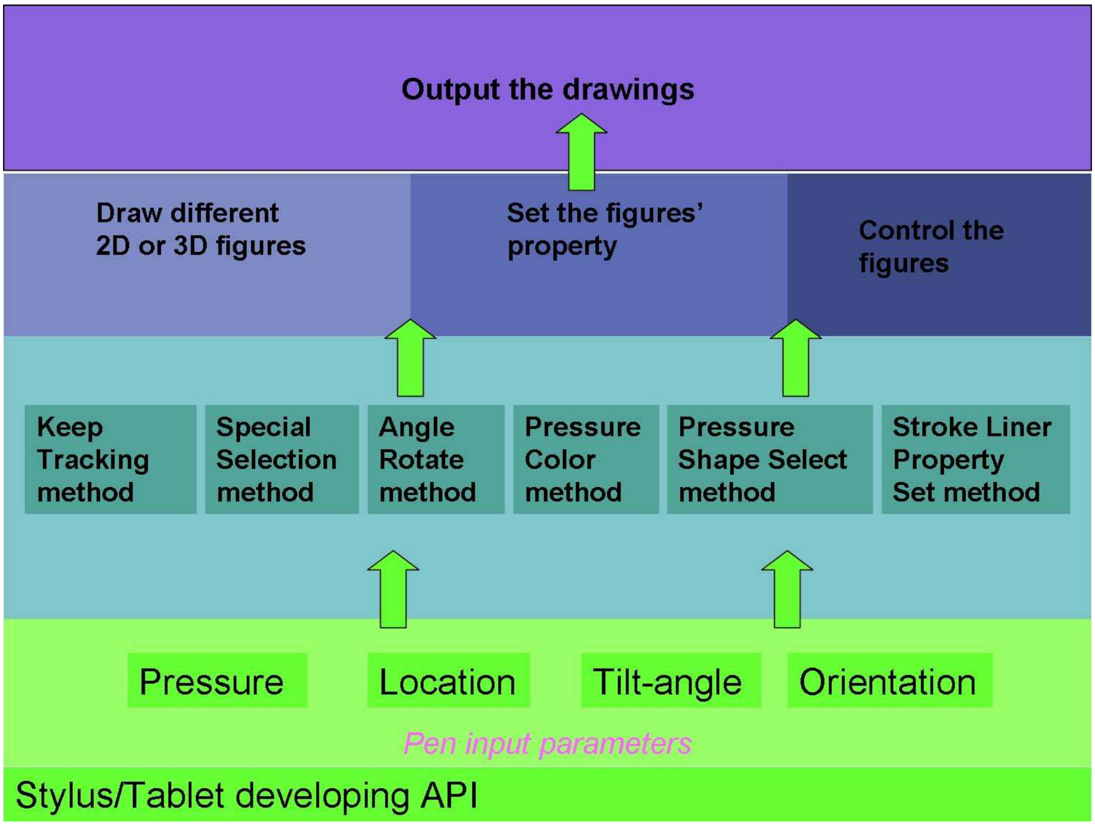


Fig. 6.13 The logical structure of PenOpera.

selection method will be pressure-based stroke selection; otherwise, the multi-selection method will be rubber band selection. In the 2D subsystem, multi-selection methods are effective only when the pen tilt angle is less then 55o. The rest of the design is the same as in the 3D subsystem.

In the process of pressure-based crossing selection, visual feedback is afforded. There are three kinds of visual feedback: light blue footprint line, yellow footprint line and red footprint line (Figure 6.7). The blue stroke line means the user can slide the pen tip to select figures without causing the pie menu to switch states (between tracking or not). The yellow stroke means that it is possible for the pie menu's state to be switched. The red line means that the pressure menu's state switch has been triggered during the selection. In our design, there is a big difference (150 units from 850 to 1000) between the pressure threshold for pressure-based crossing selection and the threshold for the pressure menu state switch. The informal user study shows that it is easy for common users to grasp the operation.

Pen tilt angle control of 3D figure rotation

Under rotation operation mode, when the user puts the pen tip on the display and strokes it on the selected figure/figures, the program records the stroke and approximates it into a line segment, and then sets the line as a rotation axis. Then, when the user changes the pen's tilt angle on the selected figure/figures, the program maps (according to specified formulas) the difference between the tilt angle and that of the X,Y,Z coordinates of the 3D figure/figures, and redraws the figure/figures instantly. The user can see the rotation of the 3D figure/figures as the pen is rotated (Figure 6.9).

Utilizing pen azimuth to control the orientation of a 2D triangle

In PenOpera, the program divides a plane into eight regions (north, northeast, east, southeast, south, southwest, west and northwest). When the user directs the pen to one of the eight regions, the program calculates the index of the pointed region from the pen's orientation, and then determines the new position of each vertex of the 2D triangle and redraws the figure. The rotation of the triangle together with the rotation of the pen is then shown on the screen (Figure 6.10).

Utilizing pen pressure to adjust a 3D figure's shape

While a 3D figure is being drawn, the program will adjust its shape factors if the pen pressure surpasses the specified threshold (850 units [89]). For example, an ellipsoid's minor axis can be lengthened to the same length as its major axis. And thus, the ellipsoid will be drawn as a sphere.

Utilizing stroke and pen tilt angle to set the fill or stroke color of 2D/3D figures

In PenOpera, the program resolves a pen's stroke and tilt-angle into integer values, and then maps (according to the specified formulas) these values to a color. The program colors the figures according to the new value and redraws them. The user can see the manipulation effect simultaneously.

Utilizing pen stroke and pressure to set the stroke width of 2D/3D figures

The program senses the sliding pen's direction (clockwise or anticlockwise) from the pen stroke, a changing scale (hereafter referred to as S) from the pen pressure and the recalculated length (hereafter referred to as) form the pen stroke length. We calculate the stroke width (referred to as W) by the formula (1). (1) If the pen pressure surpasses the specified threshold (850 units [5]), S will be a number more than 1; otherwise, S will be a number not more than 1. The resolution of the screen is taken into account when we calculate , so that the changing velocity of the stroke will not be too high in a small screen and it will not be too low in a large screen. If the pen tip is slid anticlockwise, W will be calculated by the first equation; otherwise, it will be calculated by the second equation. The minimum of W is 1 pixel and the maximum is 30 pixels. While the user is sliding the pen tip on the selected figure/figures, the figure/figures will be redrawn simultaneously and the operation effect can be seen at once (Figure 6.11).

Utilizing pen stroke to set dash parameters or beginning and/or ending arrows in 2D figures

The program predefines a number of types of dashes and arrows. The software calculates the required dash or arrowhead parameters based on the stroke of the pen which is input by the user, and it then redraws the figure accordingly. Again, the manipulation effect will be shown on the screen instantly (Figure 6.12).

Space command menus

The 3D space around the pen tip is divided into regions according to the pen's tilt angle and azimuth (Figure 6.4). Each region is delegated a certain operation command. When the pen is directed to one of the regions, the corresponding operation command will be activated by the pen's pressure input if it surpasses the specified threshold (850 units [89]). According to previous research in our laboratory, human pen rotation control performance differs in different spectra of pen tilt angle and azimuth. Therefore, more commands are mapped to the spectra in which humans perform better.

6.6 Implementation

PenOpera is implemented in Java^{TM} 6.0. It consists of approximately 76,000 lines of code spread across the two subsystems and a common main software framework. The prototype system is currently robust enough for common users to draw and manipulate 2D and 3D figures.

6.7 User Experience

The goal of the informal user study was to examine whether it is difficult for users to learn PenOpera, what can be drawn with the system and whether the participants feel that the operations are natural and flexible in a desktop environment.

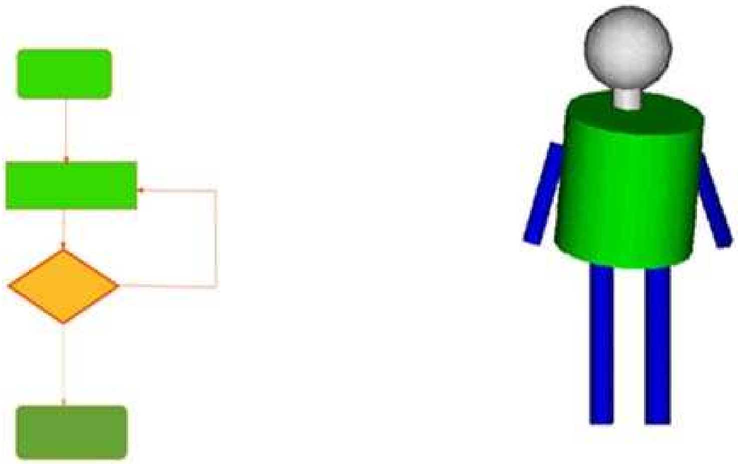


Fig. 6.14 The given drawing samples.(left)2D drawing sample;(right)3D drawing sample.

Apparatus. The hardware used in the user study was a Wacom Cintiq 21UX flat panel LCD graphics display tablet with a resolution of 1,600 × 1,200 pixels (1 pixel=0.297mm), using a wireless pen with a pressure, tilt angle and azimuth sensitive isometric tip (the width of the pen-tip is 1.76mm). It reports 512 levels (ranging from 0 to 1,024, the minimum unit is 2) of pressure, 360 degrees (ranging from 0 to 359, the minimum unit is 1 degree) of azimuth, 68 degrees (ranging from 22 to 90, the minimum unit is 1 degree) of tilt angle and has a binary button on its barrel. The test program was implemented with Java^{TM} 6.0 running on a 3.2 GHz P4 PC with the Windows XP SP2 Professional operating system. During the experiment, the mouse and the keyboard were not available to participants.

Participants. Six participants (one female and five male ranging in age from 26 to 33 years, none paid) were all volunteers from the local university community. All of

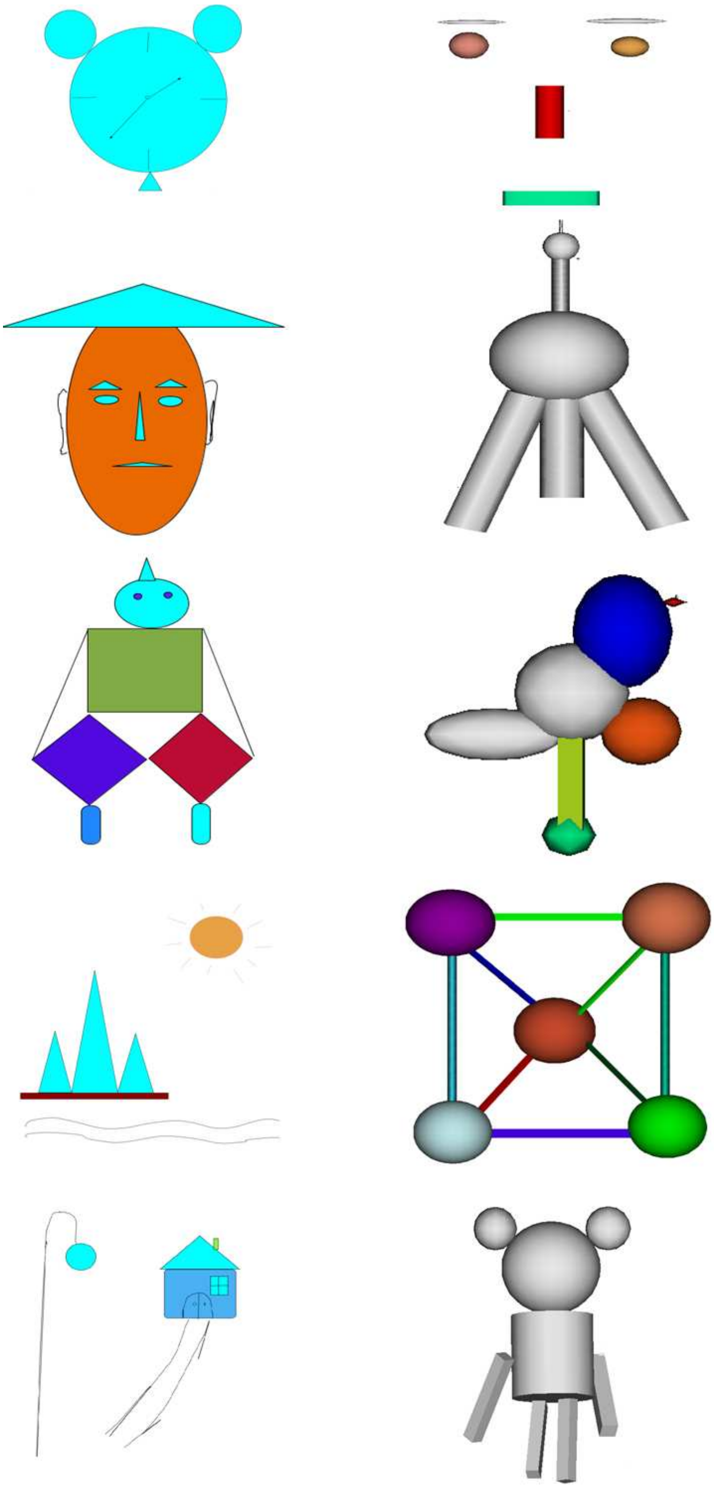


Fig. 6.15 The 2D (left) and 3D (right) figures drawn by the participants.

them were Ph.D. students and all were right-handed. Three of them have more than one year experience using the digital pen and the other three had little such experience. None had previous experience with PenOpera.

Task. In the beginning of the user study, we briefly introduced the main functions of PenOpera and showed the participants how to use it. Then the participants began to practice using PenOpera. All the participants mastered most of the operations within 5 minutes. The participants were asked to draw a 2D figure and a 3D figure by copying the given samples (Figure 6.14), an arbitrary 2D figure and an arbitrary 3D one. It took each participant no more than 5 minutes to draw either the given 2D figure or the 3D figure. And within 11 minutes, each participant drew an interesting 2D figure (Figure 6.15 left part) and a 3D one (Figure 6.15 right part) of their own design. Finally, we asked the participants to give their comments on the whole system and each of the novel techniques implemented in PenOpera.

6.8 Discussion

In the 2D/3D figures' fill or stroke color setting, and in the 2D figures' stroke width setting operations, PenOpera integrates the functions of preview, display, undoing, redoing and setting, all in one stroke. For example, when setting a 2D figures' stroke width, the user slides the pen on the figure in one direction (clockwise or anticlockwise). Then the figures' stroke width is changed and displayed on the figures. If the pen is lifted from the figure, the figure's stroke width will be set, and the previous display serves as preview. If the user slides the pen in the contrary direction, the operation effect will be to undo the last operation. If the user then slides the pen in the original direction, the operation effect will be of redoing. This kind of design merges command selection and direct manipulation [16]. This kind of design gives prominence to the features of the pen and makes it possible for all manipulation to be carried out continuously and fluently [22].

In commonly available software, only one default command can exist in a certain context. But in pen-based software, by utilizing pen input parameters, we can incorporate more than one default command in a given context. For example, we link two different default commands according to the different spectra of pen tilt angle when users manipulate 2D figure properties in the 2D subsystem. The user can slide the pen tip onto the selected 2D figures to set their properties (e.g., the fill color or stroke width) at a high spectrum of pen tilt angle; and the user can drag the figures or adjust their sizes at a low spectrum of the pen tilt angle. Much command selection time can be saved by making more than one command available as a default setting.

There are a lot of studies on selection-action interactions, such as selection-action interactions with pressure marks [5], with a combination of lassos and pigtails [17] and with rubber-banding [97]. All these studies explored valuable pen suitable selection-action interactions. But in all the above techniques, only one action can be executed with each selection, and the action is not independent of a selection. Thus, none of these techniques is flexible or efficient enough. By contrast, with a space menu, a user can perform continuously as many actions as s/he needs to after one selection. The actions can be combined with one selection whether it is a lasso, rubber-banding, keyboard-aided or pressure-based crossing selection. Therefore, with a space menu, the selection-action interactions will be more flexible and efficient.

In the informal user study, we found that most of the participants had a preference for simple but effective operations (e.g., utilizing pen pressure to adjust a figure's shape). But for slightly more complex operations (e.g., utilizing pen tilt-angle to rotate a 3D figure) the participants with more experience using a pen gave better evaluations than the participants who had only a little such experience. At the beginning of the user's experience with PenOpera, some participants with little experience using a pen felt a little confused and anxious about controlling more than one input parameter at a time. Nevertheless, they did rather well after practicing with PenOpera for no more than five minutes. We also found that the mapping of pen input parameters and program functions has a remarkable effect on user experience and skill acquisition rate. Natural and intuitive mapping can help users grasp the corresponding operation easily and quickly, especially because our mapping has the advantage of visual consistency, e.g. utilizing pen azimuth to control 2D orientation (the rotation of the pen is consistent with the rotation of the 2D figure).

Chapter 7

General Conclusion and Future Work

WIMP interfaces have been the predominant user interfaces for decades. But because the computing devices are used more and more widely, computers' main application is no longer a tool for supporting knowledge workers in office environments. As they become smaller and still less expensive, they are becoming ubiquitous and their goal is to support every aspect of human life. To support the ubiquitous computing device, besides WIMP interfaces, post-WIMP [99, 100] or non-command [101] user interfaces are deserved to be explored. In fact, more and more researchers make effort to investigate novel interfaces for the ubiquitous computing devices. Pen-based devices, e.g. PDAs, tablet PCs, intelligent mobile phones, are playing an import role in people's daily life. However, most current interfaces for pen-based devices are still WIMP-based. Because WIMP interfaces are initially designed for mice and keyboards, they have some limitations used for pen-based devices.

This thesis explores a pen suitable operation semantic and the related techniques. This operation semantic supports seamless switching between different input modes and allows users to interact with computing devices fluidly and continuously. To evaluate the proposed operation semantic, an experiment was done to compare the proposed semantic and the standard operations in MS-Word. Through the experiment, we find out that pen suitable operation semantic can significantly enhance operation speed. Three pen suitable mode switching techniques are also presented in this thesis. The utmost number of the input parameters that a user can manipulate concurrently in a computer task and the combination of these parameters are also explored in this thesis. We find out that the utmost number and combination of input parameters are dependent on the mapping between pen input channels and the controlled parameters. Natural and intuitive mapping helps users manipulate the input parameters easily and enables users manipulate all the possible input parameters concurrently. Combination utilization of multiple input parameters can produce some vivid operation effects just like manipulating physical objects by hand. An operation paradigm and a drawing prototype system are introduced to illustrate the application of the proposed operation semantic. This thesis reports that it is ripe to develop some novel operation semantics for pen-based systems.

In the future study, other novel operation semantics will be explored and more quantitative experiments will be executed to evaluate the operation semantics.

Acknowledgement

First, I would like to express my sincere appreciate to my advisor, Professor Xiangshi Ren of Kochi University of Technology, for his continuous support and patient guidance in my pursuing the Ph.D. degree.

I would like to thank my family members, for their generous love and support to my life all along.

I'd like to thank members of the Ren lab in the past and now, particularly Feng Wang, Xiaolei Zhou, Jibin Yin, Xinyong Zhang, Yizhong Xin and Minghui Sun for their friendship and contributions to my work. Many other familiar or unfamiliar friends have also afforded me great help or encouragement.

Last but not least, I would like to thank the Special Scholarship Program (SSP) Scholarship supported by KUT for providing an advantageous platform for my research.

This is a blank page.

References

[1] I.E. Sutherland, “Sketchpad: a man-machine graphical communication system,” Proceedings of the May 21-23, 1963, spring joint computer conference, Detroit, Michigan, pp.329–346, ACM, 1963.

[2] C.F. Herot and G. Weinzapfel, “One-point touch input of vector information for computer displays,” Proceedings of the 5th annual conference on Computer graphics and interactive techniques, pp.210–216, ACM, 1978.

[3] W. Buxton, R. Hill, and P. Rowley, “Issues and techniques in touch-sensitive tablet input,” Proceedings of the 12th annual conference on Computer graphics and interactive techniques, pp.215–224, ACM, 1985.

[4] G. Ramos, M. Boulos, and R. Balakrishnan, “Pressure widgets,” Proceedings of CHI 2004, pp.487–494, ACM, 2004.

[5] G. Ramos and R. Balakrishnan, “Pressure marks,” Proceedings of CHI 2007, pp.1375–1384, ACM, 2007.

[6] G. Ramos and R. Balakrishnan, “Zliding: Fluid zooming and sliding for high precision parameter manipulation,” Proceedings of UIST2005, pp.143–152, ACM, 2005.

[7] Y. Li, K. Hinckley, Z. Guan, and J.A. Landay, “Experimental analysis of mode switching techniques in pen-based user interfaces,” Proceedings of CHI 2005, pp.461–470, ACM, 2005.

[8] S. Harada, T.S. Saponas, and J.A. Landay, “Voicepen: Augmenting pen input with simultaneous non-linguistic vocalization,” Proceedings of ICMI 2007, pp.178–185, ACM, 2007.

[9] R. Balakrishnan, T. Baudel, G. Kurtenbach, and G. Fitzmaurice, “The

rockin'mouse: integral 3d manipulation on a plane," Proceedings of the SIGCHI conference on Human factors in computing systems 1997, Atlanta, Georgia, United States, pp.311–318, ACM, 1997.

[10] F. Tian, L. Xu, H. Wang, X. Zhang, Y. Liu, V. Setlur, and G. Dai, "Tilt menu: Using the 3d orientation information of pen devices to extend the selection capability of pen-based user interfaces," Proceedings of CHI 2008, Florence, Italy, pp.1371–1380, ACM, 2008.

[11] K. Partridge, S. Chatterjee, V. Sazawal, G. Borriello, and R. Want, "Tilttype: Accelerometer-supported text entry for very small devices," Proceedings of UIST 2002, pp.201–204, ACM, 2002.

[12] D. Wigdor and R. Balakrishnan, "Tilttext: Using tilt for text input to mobile phones," Proceedings of UIST 2003, pp.81–90, ACM, 2003.

[13] X. Bi, T. Moscovich, G. Ramos, R. Balakrishnan, and K. Hinckley, "An exploration of pen rolling for pen-based interaction," Proceedings of UIST 2008, pp.191–200, ACM, 2008.

[14] G. Apitz and F. Guimbretiére, "Crossy: A crossing-based drawing application," Proceedings of UIST 2004, pp.1–10, ACM, 2004.

[15] R.C. Davis, T.S. Saponas, M. Shilman, and J.A. Landay, "Sketchwizard: Wizard of oz prototyping of pen-based user interfaces," Proceedings of UIST 2008, pp.119–128, ACM, 2007.

[16] F. Guimbretiére, A. Martin, and T. Winograd, "Benefits of merging command selection and direct manipulation," ACM Trans. Comput.-Hum. Interact., vol.12, no.3, pp.460–476, 2005.

[17] K. Hinckley, P. Baudisch, G. Ramos, and F. Guimbretiere, "Design and analysis of delimiters for selection-action pen gesture phrases in scriboli," Proceedings of CHI 2005, pp.451–460, ACM, 2005.

[18] P. Baudisch, X. Xie, C. Wang, and W. Ma, "Collapse-to-zoom: viewing web pages on small screen devices by interactively removing irrelevant content," Proceedings of the 17th annual ACM symposium on User interface software and technology, Santa Fe, NM, USA, pp.91–94, ACM, 2004.

[19] J. Accot and S. Zhai, "More than dotting the i's –foundations for crossing-based interfaces," Proceedings of CHI 2002, pp.73–80, ACM, 2002.

[20] X. Ren and S. Moriya, "Improving selection performance on pen-based systems: a study of pen-based interaction for selection tasks," ACM Trans. Comput.-Hum. Interact., vol.7, no.3, pp.384–416, 2000.

[21] J. Yin and X. Ren, "Investigation to line-based techniques for multi-target selection," Proceedings of INTERACT 2007, pp.507–510, ACM, 2007.

[22] J.K. JACOB, L. DELIGIANNIDIS, and S. MORRISON, "A software model and specification language for non-wimp user interfaces," ACM TOCHI, vol.6, no.1, pp.1–46, 1999.

[23] W. Buxton, "A three-state model of graphical input," Proceedings of INTERACT'90, pp.449–456, North-Holland Publishing Co., 1990.

[24] I.S. MacKenzie and A. Oniszczak, "A comparison of three selection techniques for touchpads," Proceedings of the SIGCHI conference on Human factors in computing systems, Los Angeles, California, United States, pp.336–343, ACM Press/Addison-Wesley Publishing Co., 1998.

[25] G. Kurtenbach and W. Buxton, "Issues in combining marking and direct manipulation techniques," Proceedings of the 4th annual ACM symposium on User interface software and technology, Hilton Head, South Carolina, United States, pp.137–144, ACM, 1991.

[26] E.D. Mynatt, T. Igarashi, W.K. Edwards, and A. LaMarca, "Flatland: new dimensions in office whiteboards," Proceedings of CHI 1999, Pittsburgh, Pennsyl-

vania, United States, pp.346–353, ACM, 1999.

[27] T.R. Moran, P. Chiu, and W.v. Melle, “Pen-based interaction techniques for organizing material on an electronic whiteboard,” Proceedings of UIST 1997, pp.45 – 54, ACM, 1997.

[28] Y. Guiard, “Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model,” Journal of Motor Behavior, pp.486–517, 1987.

[29] P. Kabbash, W. Buxton, and A. Sellen, “Two-handed input in a compound task,” Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence, Boston, Massachusetts, United States, pp.417–423, ACM, 1994.

[30] J. Yin, X. Ren, and S. Zhai, “Pen pressure control in trajectory-based interaction,” Behaviour & Information Technology, 2009.

[31] C. Liu and X. Ren, “Fluid and natural pen interaction techniques by utilizing multiple input parameters,” International Journal of Innovative Computing, Information and Control, vol.6, no.4, p.9, 2010.

[32] C. Liu and X. Ren, “Making pen-based operation more seamless and continuous,” Proceedings of IFIP International Federation for Information Processing 2009, Uppsala, Sweden, pp.261–273, Springer, 2009.

[33] R.C. Zeleznik, K.P. Herndon, and J.F. Hughes, “Sketch: An interface for sketching 3d,” Proceedings of ACM SIGGRAPH 96, pp.163–170, ACM, 1996.

[34] T. Igarashi and J.F. Hughes, “A suggestive interface for 3d drawing,” Proceedings of the 14th annual ACM symposium on User interface software and technology, Orlando, Florida, pp.173–181, ACM, 2001.

[35] J.A. Jorge, N.F. Silva, and T.D. Cardoso, “Gides++,” Proceedings of 12th Encontro Português de Computcäoo Gráfica, pp.167–171, 2003.

[36] J.a.P. Pereira, J.A. Jorge, V.A. Branco, and F.N. Ferreira, “Calligraphic inter-

faces: Mixed metaphors for design," Proceedings of Interactive Systems: Design, Specification and Verification, DSV-IS 2003, pp.154–170, 2003.

[37] T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3d freeform design," Proceedings of the 26th annual conference on Computer graphics and interactive techniques, pp.409–416, ACM Press/Addison-Wesley Publishing Co., 1999.

[38] B.R.d. Araújo and J.A.P. Jorge, "Blobmaker: Free form modelling with variational implicit surfaces," Proceedings of 12th Encontro Português de Computão Gráfica (2003), pp.17–26, Springer, 2003.

[39] R. Schmidt, B. Wyvill, M.C. Sousa, and J.A. Jorge, "Shapeshop: sketch-based solid modeling with blobtrees," Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2005), Boston, Massachusetts, pp.53–62, ACM, 2005.

[40] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Fibermesh: designing freeform surfaces with 3d curves," Proceedings of ACM SIGGRAPH, San Diego, California, pp.41–50, ACM, 2007.

[41] J. Giesen, B. Miklos, M. Pauly, and C. Wormser, "The scale axis transform," Proceedings of the 25th annual symposium on Computational geometry, Aarhus, Denmark, pp.106–115, ACM, 2009.

[42] J. Joseph J. LaViola, "Sketching and gestures 101," Proceedings of ACM SIGGRAPH 2007 courses, San Diego, California, p.2, ACM, 2007.

[43] G. McCord, B. Plimmer, and B. Wuensche, "Surface manipulation using a paper sculpture metaphor," Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces, Adelaide, Australia, pp.235–238, ACM, 2007.

[44] R.C. Zeleznik, K.P. Herndon, and J.F. Hughes, "Sketch: an interface for sketching

3d scenes," Proceedings of ACM SIGGRAPH 2007 courses, San Diego, California, pp.19–24, ACM, 2007.

[45] K.P. Herndon, R.C. Zeleznik, D.C. Robbins, D.B. Conner, S.S. Snibbe, and A.v. Dam, "Interactive shadows," Proceedings of the 5th annual ACM symposium on User interface software and technology, Monteray, California, United States, pp.1–6, ACM, 1992.

[46] R. Schmidt, K. Singh, and R. Balakrishnan, "Sketching and composing widgets for 3d manipulation," Proceedings of EUROGRAPHICS 2008, pp.3–12, Blackwell Publishing, 2008.

[47] R. Schmidt and K. Singh, "Sketch-based procedural surface modeling and compositing using surface trees," Proceedings of EUROGRAPHICS 2008, pp.321–330, Blackwell Publishing, 2008.

[48] M. Oshita, "Pen-to-mime: A pen-based interface for interactive control of a human figure," Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling 2004, pp.43–52, Eurographics Association, 2004.

[49] P. Kolhoff, J. Preuß, and J. Loviscach, "Walking with pens," Proceedings of EUROGRAPHICS 2005, pp.33–36, 2005.

[50] H. Biermann, I.M. Martin, D. Zorin, and F. Bernardini, "Sharp features on multiresolution subdivision surfaces," Proceedings of Pacific Conference on Computer Graphics and Applications 2001, pp.0140–0149, 2001.

[51] L. Olsen, F.F. Samavati, M.C. Sousa, and J.A. Jorge, "Sketch-based mesh augmentation," Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling 2005, p.10, The Eurographics Association, 2005.

[52] B. Wyvill, K. Foster, P. Jepp, R. Schmidt, M.C. Sousa, and J.A. Jorge, "Sketch based construction and rendering of implicit models," Proceedings of Computational Aesthetics in Graphics, Visualization and Imaging 2005, p.8, The Euro-

graphics Association, 2005.

[53] Z. Ji, L. Liuy, Z. Chen, and G. Wang, “Easy mesh cutting,” Proceedings of EUROGRAPHICS 2006, pp.283–291, 2006.

[54] L.B. Kara and K. Shimada, “Construction and modification of 3d geometry using a sketch-based interface,” Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling 2006, The Eurographics Association, 2006.

[55] H. Wang and L. Markosian, “Free-form sketch,” Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling, Riverside, California, pp.53–58, ACM, 2007.

[56] J.J. Cherlin, F. Samavati, M.C. Sousa, and J.A. Jorge, “Sketch-based modeling with few strokes,” Proceedings of the 21st spring conference on Computer graphics, Budmerice, Slovakia, pp.137–145, ACM, 2005.

[57] Y. Kho and M. Garland, “Sketching mesh deformations,” Proceedings of ACM SIGGRAPH 2005, Los Angeles, California, pp.934–934, ACM, 2005.

[58] J.A. Landay and B.A. Myers, “Sketching interfaces: Toward more human interface design,” Computer, vol.34, no.3, pp.56–64, 2001.

[59] J. Zimmermann, A. Nealen, and M. Alexa, “Silsketch: automated sketch-based editing of surface meshes,” Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling, Riverside, California, pp.23–30, ACM, 2007.

[60] A. Nealen, O. Sorkine, M. Alexa, and D. Cohen-Or, “A sketch-based interface for detail-preserving mesh editing,” ACM Trans. Graph., vol.24, no.3, pp.1142–1147, 2005.

[61] X. Yuan, H. Xu, M.X. Nguyen, A. Shesh, and B. Chen, “Sketch-based segmentation of scanned outdoor environment models,” Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modleing (SBM’05), pp.19–26, the Euro-

graphics Association, 2005.

[62] G.D. Brigham and G.M. Draper, "A gestural interface to free-form deformation," Proceedings of Graphics Interface 2003, pp.113–120, 2003.

[63] A. Severn, F. Samavati, and M.C. Sousa, "Transformation strokes," Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling, pp.75–82, Aire-la-Ville, Switzerland, 2006.

[64] J.P. Pernot, S. Guillet, J.C. Léon, F. Giannini, C.E. Catalano, and B. Falcidieno, "A shape deformation tool to model character lines in the early design phases," Proceedings of the Shape Modeling International 2002 (SMI'02), pp.165–173, IEEE Computer Society, 2002.

[65] J.a.P. Pereira, J.A. Jorge, V.A. Branco, N.F. Silva, T.D. Cardoso, and F.N. Ferreira, "Cascading recognizers for ambiguous calligraphic interaction," Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling 2004, pp.63–72, the Eurographics Association, 2004.

[66] P. Leclercq, "Invisible sketch interface in architectural engineering," in Graphics Recognition, pp.353–363, 2004.

[67] I. Kókai, J. Finger, R.C. Smith, R. Pawlicki, and T. Vetter, "Example-based conceptual styling framework for automotive shapes," Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling, Riverside, California, pp.37–44, ACM, 2007.

[68] L.B. Kara, K. Shimada, and S.D. Marmalefsky, "An evaluation of user experience with a sketch-based 3d modeling system," Computers & Graphics, vol.31, no.4, pp.580–597, 2007.

[69] L.B. Kara and K. Shimada, "Sketch-based 3d-shape creation for industrial styling design," IEEE Comput. Graph. Appl., vol.27, no.1, pp.60–71, 2007.

[70] R. Juchmes, P. Leclercq, and S. Azar, "A freehand-sketch environment for ar-

chitectural design supported by a multi-agent system," Computers & Graphics, vol.29, no.6, pp.905–915, 2005.

[71] M.D. Gross and E.Y.L. Do, "Drawing on the back of an envelope: a framework for interacting with application programs by freehand drawing," Computers & Graphics, vol.24, no.6, pp.835–849, 2000.

[72] M.D. Gross, "The cocktail napkin, the fat pencil, and the slide library," Proceedings, Association for Computer Aided Design in Architecture (ACADIA '94), pp.103–113, 1994 National Conference, St Louis, 1994.

[73] M.J. Fonseca, A. Ferreira, and J.A. Jorge, "Towards 3d modeling using sketches and retrieval," Proceedings of EUROGRAPHICS Workshop on Sketch-Based Interfaces andModeling 2004, pp.127–136, the Eurographics Association, 2004.

[74] M. Contero, F. Naya, P. Company, J.L. Saorin, and J. Conesa, "Improving visualization skills in engineering education," IEEE Comput. Graph. Appl., vol.25, no.5, pp.24–31, 2005.

[75] E. Turquin, J. Wither, L. Boissieux, M.P. Cani, and J.F. Hughes, "A sketch-based interface for clothing virtual characters," IEEE Comput. Graph. Appl., vol.27, no.1, pp.72–81, 2007.

[76] B.Y. Chen, Y. Ono, and T. Nishita, "Character animation creation using hand-drawn sketches," The Visual Computer, vol.21, no.8, pp.551–558, 2005.

[77] J. Davis, M. Agrawala, E. Chuang, Z. Popović, and D. Salesin, "A sketching interface for articulated figure animation," Proceedings of ACM SIGGRAPH 2006 Courses, Boston, Massachusetts, pp.15–24, ACM, 2006.

[78] M. Thorne, D. Burke, and M.v.d. Panne, "Motion doodles: an interface for sketching character motion," 2007.

[79] F. Anastacio, M.C. Sousa, F. Samavati, and J.A. Jorge, "Modeling plant structures using concept sketches," Proceedings of the 4th international symposium on

Non-photorealistic animation and rendering, Annecy, France, pp.105–113, ACM, 2006.

[80] X. Chen, B. Neubert, Y.Q. Xu, O. Deussen, and S.B. Kang, “Sketch-based tree modeling using markov random field,” Proceedings of ACM SIGGRAPH Asia 2008, Singapore, pp.1–9, ACM, 2008.

[81] T. Ijiri, S. Owada, and T. Igarashi, “The sketch l-system: Global control of tree modeling using free-form strokes,” Proceedings of smart graphics’06, pp.138–146, Lecture notes on computer science, 2006.

[82] T. Ijiri, S. Owada, M. Okabe, and T. Igarashi, “Floral diagrams and inflorescences: interactive flower modeling using botanical structural constraints,” Proceedings of ACM SIGGRAPH 2005, Los Angeles, California, pp.720–726, ACM, 2005.

[83] S. Mondet, W. Cheng, G. Morin, R. Grigoras, F. Boudon, and W.T. Ooi, “Streaming of plants in distributed virtual environments,” Proceeding of the 16th ACM international conference on Multimedia, Vancouver, British Columbia, Canada, pp.1–10, ACM, 2008.

[84] B. Neubert, T. Franken, and O. Deussen, “Approximate image-based tree-modeling using particle flows,” Proceedings of ACM SIGGRAPH 2007 papers, San Diego, California, p.88, ACM, 2007.

[85] P. Tan, T. Fang, J. Xiao, P. Zhao, and L. Quan, “Single image tree modeling,” Proceedings of ACM SIGGRAPH Asia 2008, Singapore, pp.1–7, ACM, 2008.

[86] M.N. Zakaria and S.R. Shukri, “A sketch-and-spray interface for modeling trees,” Proceedings of the 8th international symposium on Smart Graphics, Kyoto, Japan, pp.23–35, Springer-Verlag, 2007.

[87] T. Igarashi, “Freeform user interfaces for graphical computing,” Proceedings of Smart Graphics 2003, pp.39–48, 2003.

[88] X. Cao and R. Balakrishnan, “Visionwand: interaction techniques for large dis-

plays using a passive wand tracked in 3d," Proceedings of ACM SIGGRAPH 2004 Papers, Los Angeles, California, pp.729–729, ACM, 2004.

[89] J. Yin and X. Ren, "Zwps: A hybrid selection technique for small target acquisition in pen-based interfaces," Proceedings of INTERACT 2007, pp.503–506, Springer, 2007.

[90] C. Liu and X. Ren, "Experimental analysis of mode switching techniques in pen-based user interfaces," International Journal of Innovative Computing, Information and Control, vol.6, no.3, p.8, 2010.

[91] G. Ramos and R. Balakrishnan, "Fluid interaction techniques for the control and annotation of digital video," Proceedings of UIST 2003, pp.105–114, ACM, 2003.

[92] X. Ren, J. Yin, S. Zhao, and Y. Li, "The adaptive hybrid cursor: A pressure-based target selection technique for pen-based user interfaces," Proceedings of INTERACT 2007, pp.310–323, Springer, 2007.

[93] J. Yin and X. Ren, "An interactive calligraphy and ink-sash painting system," international Journal of Innovative Computing, Information and Control, vol.6, no.2, p.10, 2010.

[94] F. Tian, X. Ao, H. Wang, V. Setlur, and G. Dai, "The tilt cursor: Enhancing stimulus-response compatibility by providing 3d orientation cue of pen," Proceedings of UIST 2007, pp.303–306, ACM, 2007.

[95] J. Yin and X. Ren, "An empirical study of arc and line stroke-based scrolling techniques in pen-based interfaces," International Journal of Innovative Computing, Information and Control, vol.6, no.2, p.13, 2010.

[96] G. Fitzmaurice, A. Khan, R. Pieké, B. Buxton, and G. Kurtenbach, "Tracking menus," Proceedings of UIST 2003, pp.71–80, ACM, 2003.

[97] S.N. KRISHNAN and S. MORIYA, "One stroke operations: A new pen-based user interface that can integrate or separate operand specification, menu opening

and selection, and action execution, in one or more strokes," Transactions of Information Processing Society of Japan, vol.37, no.12, pp.2419–2437, 1996.

[98] R.C. Barrett, R.S.O. Jr., and J.D. Rutledge, "Graphical user interface cursor positioning device having a negative inertia transfer function," Patent # 5,570,111, IBM Corp, 1996.

[99] A.v. Dam, "Post-wimp user interfaces," Commun. ACM, vol.40, no.2, pp.63–67, 1997.

[100] H. Stphane, D. Cedric, D. Pierre, F. Jean-Daniel, and H. Gerard, "The magglite post-wimp toolkit: draw it, connect it and run it," Proceedings of the 17th annual ACM symposium on User interface software and technology, Santa Fe, NM, USA, pp.257–266, ACM, 2004.

[101] J. Nielsen, "Noncommand user interfaces," Commun. ACM, vol.36, no.4, pp.83–99, 1993.

Appendix A

Publications

A.1 Articles in or submitted to refereed journals

1. Chuanyi Liu, Xiangshi Ren and Daniels Paul, “Mobile Devices Strengthen Classroom Management” , International Journal of Intelligent Engineering and Systems (IJIES), Vol.1, No. 3, 2008.

2. Chuanyi Liu and Xiangshi Ren, “Experiment Analysis of Mode Switching Techniques in Pen-based User Interfaces”, Accepted for publication in International Journal of Innovative Computing, Information and Control (IJICIC), Vol.6, No.3, 2010.

3. Chuanyi Liu and Xiangshi Ren, “Fluid and Natural Pen Interaction Techniques by Utilizing Multiple Input Parameters”, Accepted for publication in International Journal of Innovative Computing, Information and Control (IJICIC), Vol.6, No. 4, 2010.

A.2 Articles in full paper refereed international conference proceedings

4. Chuanyi Liu and Xiangshi Ren, “Making Pen-based Operation More Seamless and Continuous”, in Proc. INTERACT 2009, pp. 261-273, Uppsala, Sweden, 2009.

5. Chuanyi Liu, Xiangshi Ren and Dawei Li, “A Comparative Evaluation of Mode Switching Techniques”, in Proc. IPC2008, pp. 975- 981, Sydney, Australia, 2008.

A.3 Articles in abstract refereed international conference proceedings

6. Chuanyi Liu, Paul Daniels, Xiangshi Ren and Yoshimasa Kimura, “A Pen-based Classroom Management System”, International Conference on Human-Computer Interaction 2007(HCI Interaction 2007), Beijing, China, 2007.

7. Jibin Yin, Xiangshi Ren and Chuanyi Liu, “Mode Switching Techniques Based on Pen Angle Inputs”, The 8th Asia- Pacific Conference on Computer Human Interaction (APCHI2008), Seoul, Korea, 2008.

A.4 Articles in local conference proceedings

8. Chuanyi Liu, Paul Daniels, Xiangshi Ren and Yoshimasa Kimura, “Research on Using Intelligent Mobile Devices in Classroom Management”, IEEE2007, Tokushima, Japan, 2007.

9. Chuanyi Liu and Xiangshi Ren, “Mode Switching Techniques”, International Conference on Next Era Information Networking (NEINE’08), Kochi, Japan, 2008.

10. Chuanyi Liu and Xiangshi Ren, “Angles Outperform the Traditional Way”, IEEE2008, Tokushima, Japan, 2008.