HUMAN–COMPUTER INTERACTION, 2010, Volume 25, pp. 324–388

ISSN: 0737-0024 print / 1532-7051 online
DOI: 10.1080/07370024.2010.499839

Taylor & Francis
Taylor & Francis Group

Direct Pen Interaction With a Conventional Graphical User Interface

Daniel Vogel[1] and Ravin Balakrishnan[2]
[1]Mount Allison University, Sackville, New Brunswick, Canada
[2]University of Toronto, Ontario, Canada

We examine the usability and performance of Tablet PC direct pen input with a conventional graphical user interface (GUI). We use a qualitative observational study design with 16 participants divided into 4 groups: 1 mouse group for a baseline control and 3 Tablet PC groups recruited according to their level of experience. The study uses a scripted scenario of realistic tasks and popular office applications designed to exercise standard GUI components and cover typical interactions such as parameter selection, object manipulation, text selection, and ink annotation. We capture a rich set of logging data including 3D motion capture, video taken from the participants' point-of-view, screen capture video, and pen events such as movement and taps. To synchronize, segment, and annotate these logs, we used our own custom analysis software.

We find that pen participants make more errors, perform inefficient movements, and express frustration during many tasks. Our observations reveal overarching problems with direct pen input: poor precision when tapping and dragging, errors caused by hand occlusion, instability and fatigue due to ergonomics and reach, cognitive differences between pen and mouse usage, and frustration due to limited input capabilities. We believe these to be the primary causes of nontext errors, which contribute to user frustration when using a pen with a conventional GUI. Finally, we discuss how researchers could address these issues without sacrificing the consistency of current GUIs and applications by making improvements at three levels: hardware, base interaction, and widget behavior.

1. INTRODUCTION

Given our familiarity with using pens and pencils, one might expect that operating a computer using a pen would be more natural and efficient. Yet even though

Daniel Vogel is an Adjunct Professor of Computer Science at Mount Allison University, Canada. **Ravin Balakrishnan** is an Associate Professor of Computer Science and Canada Research Chair in Human-Centred Interfaces at the Department of Computer Science, University of Toronto.

CONTENTS

---

the second generation of commercial *direct pen input* devices such as the Tablet PC, where the input and output spaces are coincident (Whitefield, 1986), are now cost effective and readily available, they have failed to live up to analysts' prediction for marketplace adoption (Spooner & Foley, 2005; Stone & Vance, 2009). Relying on text entry without a physical keyboard could be one factor, but part of the problem may also be that current software applications and graphical user interfaces (GUIs) are designed for *indirect input* using a mouse, where there is a spatial separation between the input space and output display. Thus issues specific to direct input—such as hand occlusion, device stability, and limited reach—have not been considered (Meyer, 1995).

The research community has responded with pen-specific interaction paradigms such as crossing (Accot & Zhai, 2002; Apitz & Guimbretière, 2004), gestures (Grossman, Hinckley, Baudisch, Agrawala, & Balakrishnan, 2006; Kurtenbach & Buxton, 1991a, 1991b), and pressure (Ramos, Boulos, & Balakrishnan, 2004); pen-tailored GUI widgets (Fitzmaurice, Khan, Pieké, Buxton, & Kurtenbach, 2003; Guimbretière & Winograd, 2000; Hinckley et al., 2006; Ramos & Balakrishnan, 2005); and pen-specific applications (Agarawala & Balakrishnan, 2006; Hinckley et al., 2007). Although these techniques may improve pen usability, with some exceptions (e.g., Dixon, Guimbretière, & Chen, 2008; Ramos, Cockburn, Balakrishnan, & Beaudouin-Lafon, 2007), retrofitting the vast number of existing software applications to accommodate these new paradigms is arguably not practically feasible. Moreover, the popularity of *convertible* Tablet PCs, which operate in laptop or slate mode, suggests that users may prefer to switch between mouse and pen according to their working context (Twining et al., 2005). Thus any pen-specific GUI refinements or pen-tailored applications must also be compatible with mouse and keyboard input. So, if we accept that the fundamental behaviour and layout of conventional GUIs are unlikely to change in the near future, is there still a way to improve pen input?

As a first step toward answering this question, one needs to know what makes pen interaction with a conventional GUI difficult. Researchers have investigated lower level aspects of pen performance such as accuracy (Ren & Moriya, 2000) and speed (MacKenzie, Sellen, & Buxton, 1991). However, like the pen-specific interaction paradigms and widgets, these have for the most part been studied in highly controlled experimental situations. Examining a larger problem by testing isolated pieces has advantages for quantitative control, but more complex interactions between the pieces are missed.

Researchers have suggested that more open-ended tasks can give a better idea of how something will perform in real life (Ramos et al., 2006). Indeed, there are examples of qualitative and observational pen research (Briggs, Dennis, Beck, & Nunamaker, 1993; Fitzmaurice, Balakrishnan, Kurtenbach, & Buxton, 1999; Haider, Luczak, & Rohmert, 1982; Inkpen et al., 2006; Turner, Pérez-Quiñones, & Edwards, 2007). Unfortunately, these have used older technologies like indirect pens with opaque tablets and light pens, or focused on a single widget or a specialized task. To our knowledge, there has been no comprehensive qualitative, observational study of Tablet PC or direct pen interaction with realistic tasks and common GUI software applications.

In this article, we present the results of such a study. The study includes pen and mouse conditions for baseline comparison, and to control for user experience, we recruited participants who are expert Tablet PC users, power computer users who do not use Tablet PCs, and typical business application users. We used a realistic scenario involving popular office applications with tasks designed to exercise standard GUI components and covered typical interactions such as parameter selection, object manipulation, text selection, and ink annotation. Our focus is on GUI manipulation tasks other than text entry—text entry is an isolated and difficult problem with a large body of existing work relating to handwriting recognition and direct input keyboard techniques (Shilman, Tan, & Simard, 2006; Zhai & Kristensson, 2003, have provided overviews of this literature). We see our work as complementary; improvements in pen-based text entry are needed, but our focus is on improving direct input with standard GUI manipulation.

We base our analysis methodology on *Interaction Analysis* (Jordan & Henderson, 1995) and *Open Coding* (Strauss & Corbin, 1998). Instead of examining a broad and open-ended social working context for which these techniques were originally designed, we adapt them to analyze lower level interactions between a single user and software. This style of qualitative study is more often seen in the Computer-Supported Cooperative Work (CSCW) community (e.g., Ranjan, Birnholtz, & Balakrishnan, 2006; Scott, Carpendale, & Inkpen, 2004), but CSCW studies typically do not take advantage of detailed and diverse observation data like the kind we gather: video taken from the user's point-of-view; 3D positions of their forearm, pen, Tablet PC, and head; screen capture; and pen input events. To synchronize, segment, and annotate these multiple streams of logging data, we developed a custom software application. This allows us to view all data streams at once, annotate interesting behavior at specific times with a set of annotation codes, and extract data for visualization and quantitative analysis.

We see our methodology as a hybrid of typical controlled human–computer interaction (HCI) experimental studies, usability studies, and qualitative research.

2. BACKGROUND AND RELATED WORK

Using a pen (or stylus) to operate a computer is not new. Tethered light-pens were introduced in the late 1950s (Gurley & Woodward, 1959) and a direct pen version of the RAND tablet was described in the 1960s (Gallenson, 1967). Pen-based personal computers have been commercially available since the early 1990s with early entrants such as Go Corporation's PenPoint, Apple's Newton, and the Palm Pilot. In fact, Microsoft released Windows for Pen Computing in 1992 (Bricklin, 2002; Meyer, 1995). Yet regardless of this long history, and in spite of Microsoft's high expectations with the announcement of the Tablet PC in 2000 (Markoff, 2000; Pogue, 2000), pen computing has not managed to seriously compete with mouse and keyboard input (Spooner & Foley, 2005). Recent reports show that Tablet PCs accounted for only 1.4% of mobile PCs sold worldwide in 2006 (Shao, Fiering, & Kort, 2007).

Part of the problem may be that current software applications and GUIs are not well suited for the pen. Researchers and designers have responded by developing new pen-centric interaction paradigms, widgets that leverage pen input capabilities, and software applications designed from the ground up for pen input.

One popular interaction paradigm is using gestures: a way of invoking a command with a distinctive motion rather than manipulating GUI widgets. Early explorations include Buxton, Sniderman, Reeves, Patel, and Baecker (1979), who used elementary gestures to enter musical notation; Buxton, Fiume, Hill, Lee, and Woo (1983), who used more complex gestures for electronic sketching; and Kurtenbach and Buxton's (1991a) Gedit, which demonstrates gesture-based object creation and manipulation. Later, completely gesture-based research applications appeared such as Lin, Newman, Hong, and Landay's (2000) DENIM; Moran, Chiu, and Melle's (1997) Tivoli; and Forsberg, Dieterich, and Zeleznik's (1998) music composition application. Note that these all target very specific domains, which emphasize drawing, sketching, and notation.

Although these researchers (and others) have suggested that gestures are more natural with a pen, issues with human perception (Long, Landay, Rowe, & Michiels, 2000) and performance (Cao & Zhai, 2007) can make the design of unambiguous gesture sets challenging. But perhaps most problematic is that gestures are not self-revealing and must be memorized through training. Marking Menus (Kurtenbach & Buxton, 1991b) addresses this problem with a visual preview and directional stroke to help users smoothly transition from novice to expert usage, but these are limited to menulike command selection rather than continuous parameter input.

Perhaps due to limitations with gestures, several researchers have created applications which combine standard GUI widgets and gestures. Early examples include

Schilit, Golovchinsky, and Price's (1998) Xlibris electronic book device; Truong and Abowd's (1999) StuPad; Chatty and Lecoanet's (1996) air traffic control system; and Gross and Do's (1996) Electronic Cocktail Napkin. These all support free-form inking for drawing and annotations but rely on a conventional GUI tool bar for many functions.

Later, researchers introduce pen-specific widgets in their otherwise gesture-based applications. Ramos and Balakrishnan's (2003) LEAN is a pen-specific video annotation application that uses gestures along with an early form of pressure widget (Ramos et al., 2004) and two sliderlike widgets for timeline navigation. Agarawala and Balakrishnan's (2006) BumpTop uses physics and 3D graphics to lend more realism to pen-based object manipulation. Both of these applications are initially free of any GUI, but once a gesture is recognized, or when the pen hovers over an object, widgets are revealed to invoke further commands or exit modes.

Hinckley et al.'s (2007) InkSeine presents what is essentially a pen-specific GUI. It combines and extends several pen-centric widgets and techniques in addition to making use of gestures and crossing widgets. Aspects of its use required interacting with standard GUI applications in which the authors found users had particular difficulty with scrollbars. To help counteract this, they adapted Fitzmaurice et al.'s (2003) Tracking Menu to initiate a scroll ring gesture in a control layer above the conventional application. Fitzmaurice et al.'s (2003) *Tracking Menu* is designed to support the rapid switching of commands by keeping a toolbar near the pen tip at all times. The scroll ring gesture uses a circular pen motion as an alternative to the scrollbar (Moscovich & Hughes, 2004; G. Smith, schraefel, & Baudisch, 2005). Creating pen-specific GUI widgets has been an area of pen research for some time; for example, Guimbretière and Winograd's (2000) FlowMenu combines a crossing-based menu with smooth transitioning to parameter input. In most cases, compatibility with current GUIs is either not a concern or unproven.

Another, perhaps less radical pen interaction paradigm is selecting targets by crossing *through* them rather than tapping *on* them (Accot & Zhai, 2002). Apitz and Guimbretière's (2004) CrossY is a sketching application that exclusively uses a crossing-based interface including crossing-based versions of standard GUI widgets such as buttons, check boxes, radio buttons, scrollbars, and menus.

In spite of the activity in the pen research community, commercial applications for the Tablet PC tend to emphasize free-form inking for note taking, drawing, or mark-up while relying on standard GUI widgets for commands. Autodesk's Sketchbook Pro (Autodesk, 2007) is perhaps the most pen-centric commercial application at present. It uses a minimal interface, it takes advantage of pen pressure, and users can access most drawing commands using pen-specific widgets such as the Tracking Menu and Marking Menu. However, it still relies on conventional menus for some commands.

One reason for the lack of pen-specific commercial applications is that the primary adopters of pen computing are in education, healthcare, illustration, computer-aided design, and mobile data entry (Chen, 2004; Shao et al., 2007; Whitefield, 1986) in spite of initial expectations that business users would be early adopters. These vertical

markets use specialized software that emphasizes handwritten input and drawing, rather than general computing. For the general business and consumer markets, software applications and GUIs are designed for the mouse, thus usability issues specific to direct pen input have not always been considered.

In this section we review what has been done to understand and improve pen input with controlled experiments evaluating pen performance and device characteristics, field work examining in situ usage, and observational usability studies of realistic scenarios.

2.1. Experiments Investigating Performance and Device Characteristics

Researchers have examined aspects of pen performance such as speed and accuracy in common low-level interactions like target selection, area selection, and dragging. Not all studies use direct pen input with a Tablet PC-sized display. Early studies use indirect pen input where the pen tablet and display are separated, and other researchers have focused on pen input with smaller handheld mobile devices. Pen characteristics such as mode selection, handedness, tactile feedback, tip pressure control, and barrel rotation have been investigated thoroughly, but other aspects like pen tilt and hand occlusion are often discussed but not investigated in great detail.

Target Selection

MacKenzie et al. (1991) are often cited regarding pen pointing performance, but because they used indirect pen input, one has to be cautious with adopting their results for direct pen input situations such as the Tablet PC. In their comparison of indirect pen input with mouse and trackball when pointing and dragging, they found no difference between the pen and mouse for pointing but a significant performance benefit for the pen when dragging. All devices have higher error rates for dragging compared to pointing. The authors concluded that the pen can outperform a mouse in direct manipulation systems, especially if drawing or gesture activities are common. Like most Fitts' law style research, they used a highly controlled, one-dimensional, reciprocal pointing task.

Ren and Moriya (2000) examined the accuracy of six variants of pen-tapping selection techniques in a controlled experiment with direct pen input on a large display. They found very high error rates for 2 and 9 pixels targets using two basic selection techniques: *Direct On*, where a target is selected when the pen first contacts the display (the pen down event), and *Direct Off*, where selection occurs when the pen is lifted from the display (the pen up event). Note that in a mouse-based GUI, targets are typically selected successfully only when *both* down *and* up events occur on the target, hence accuracy will likely further degrade. Ramos et al. (2007) argued that accuracy is further impaired when using direct pen input because of visual parallax and pen tip occlusion—users can not simply rely on the physical position of the pen tip. To compensate, their Pointing Lens technique enlarges the target area with

increased pressure, and selection is trigged by lift off. With this extra assistance, they find that users can reliably select targets smaller than 4 pixels.

In Accot and Zhai's (2002) study of their crossing interaction paradigm, they found that when using a pen, users can select a target by crossing as fast, or faster, than tapping in many cases. However, their experiment uses indirect pen input and the target positions are in a constrained space, so it is not clear if the performance they observe translates to direct pen input. Hourcade and Berkel (2006) later compared crossing and tapping with direct pen input (on a PDA) as well as the interaction of age. They found that older users have lower error rates with crossing but found no difference for younger users. Unlike Accot and Zhai's work, Hourcade and Berkel used circular targets as a stimulus. Without a crossing visual constraint, they found that participants exhibit characteristic movements, such as making a checkmark. The authors speculated that this may be because people do not tap on notepads but make more fluid actions like writing or making checkmarks supporting the general notion of crossing. Forlines and Balakrishnan (2008) compared crossing and pointing performance with direct and indirect pen input. They found that direct input is advantageous for crossing tasks, but when selecting very small targets indirect input is better.

Two potential issues with crossing-based interfaces are target orientation and space between targets. Accot and Zhai (2002) suggested that targets could automatically rotate to remain orthogonal to the pen direction, but this could further exacerbate the space dilemma. They noted that as the space between the goal target and nearby targets is decreased, the task difficulty becomes a factor of this distance rather than the goal target width. Dixon et al. (2008) investigated this "space versus speed tradeoff" in the context of crossing-base dialogue boxes. They found that if the recognition algorithm is relaxed to recognize "sloppy" crossing gestures, then lower operation times can be achieved (with only slightly higher error rates). This relaxed version of crossing could ease the transition from traditional click behavior, and, with reduced spatial requirements, it could coexist with current GUIs.

Mizobuchi and Yasumura (2004) investigated tapping and lasso selection on a pen-based PDA. They found that tapping multiple objects is generally faster and less error prone than lasso circling, except when the group of targets are highly cohesive and form less complex shapes. Note that enhancements introduced with Windows Vista encourage selecting multiple file and folder objects by tapping through the introduction of selection check boxes placed on the objects. Lank and Saund (2005) noted that when users lasso objects, the "sloppy" inked path alone may not be the best indicator of their intention. They found that by also using trajectory information, the system can better infer the user's intent.

Mode Selection

To operate a conventional GUI, the pen must support multiple button actions to emulate left and right mouse clicks. The Tablet PC simulates right-clicking using dwell time and visual feedback by pressing a barrel button, or by inverting the pen to use the "eraser" end. Li, Hinckley, Guan, and Landay (2005) found that using

dwell time for mode selection is slower, more error prone, and disliked by most participants. In addition to the increased time for the dwell itself, the authors also found that additional preparation time is needed for the hand to slow down and prepare for a dwell action. Pressing the pen barrel button, pressing a button with the nonpreferred hand, or using pressure are all fast techniques, but using the eraser or pressing a button with nonpreferred hand are the least error prone.

Hinckley, Baudisch, Ramos, and Guimbretière's (2005) related work examining mode delimiters also found dwell timeout to be slowest, but in contrast to Li et al., found that pressing a button with the nondominant can be error prone due to synchronization issues. However, Hinckley et al.'s (2006) Springboard shows that if the button is used for temporary selection of a kinaesthetic quasi mode (where the user selects a tool momentarily but afterward returns to the previous tool), then it can be beneficial.

Grossman et al. (2006) provided an alternate way to differentiate between inking and command input by using distinctive pen movements in hover space (i.e., while the pen is within tracking range above the digitizing surface but not in contact with it). An evaluation shows that this reduces errors due to divided attention and is faster than using a conventional toolbar in this scenario. Forlines, Vogel, and Balakrishnan's (2006) Trailing Widget provides yet another way of controlling mode selection. The Trailing Widget floats nearby, but out of the immediate area of pen input, and can be "trapped" with a swift pen motion.

Handedness

Hancock and Booth (2004) studied how handedness affects performance for simple context menu selection with direct pen input on large and small displays. They noted that identifying handedness is an important consideration, because the area occluded by the hand is mirrored for left- or right-handed users and the behavior of widgets will need to change accordingly. Inkpen et al. (2006) studied usage patterns for left-handed users with left- and right-handed scrollbars with a direct pen input PDA. By using a range of evaluation methodologies they found a performance advantage and user preference for left-handed scrollbars. All participants cited occlusion problems when using the right-handed scrollbar. To reduce occlusion, some participants raised their grip on the pen or arched their hand over the screen, both of which are reported as feeling unnatural and awkward. Their methodological approach includes two controlled experiments and a longitudinal study, which lends more ecological validity to their findings.

Pressure

Ramos et al. (2004) argued that pen pressure can be used as an effective input channel in addition to X-Y position. In a controlled experiment, they found that participants could use up to six levels of pressure with the aid of continuous visual feedback and a well-designed transfer function creating the possibility of pressure activated GUI widgets. Ramos and colleagues subsequently explored using pressure in a variety of applications, including an enhanced slider that uses pressure to change

the resolution of the parameter (Ramos & Balakrishnan, 2005), a pressure-activated Pointing Lens (Ramos et al., 2007) that is found to be more effective than other lens designs, and a lasso selection performed with different pressure profiles used to denote commands (Ramos & Balakrishnan, 2007).

Tactile Feedback

Lee, Dietz, Leigh, Yerazunis, and Hudson (2004) designed a haptic pen using a solenoid actuator that provides tactile feedback along the longitudinal axis of the pen and showed how the resulting "thumping" and "buzzing" feedback can be used for enhancing interaction with GUI elements. Sharmin, Evreinov, and Raisamo (2005) investigated using vibrating pen feedback during a tracing task and found that tactile feedback reduces time and number of errors compared to audio feedback. Forlines and Balakrishnan (2008) compared tactile feedback with visual feedback for direct and indirect pen input on different display orientations. They found that even a small amount of tactile feedback can be helpful, especially when standard visual feedback is occluded by the hand. Current Tablet PC pens do not support active tactile feedback, but the user does receive passive feedback when the pen tip strikes the display surface. However, this may not always correspond to the system registering the tap: Consider why designers added a small "ripple" animation to Windows Vista to visually reinforce a tap.

Barrel Rotation and Tilt

Bi, Moscovich, Ramos, Balakrishnan, and Hinckley (2008) investigated pen barrel rotation as a viable form of input. They found that unintentional pen rolling can be reliably filtered out using thresholds on rolling speed and rolling angle and that users can explicitly control an input parameter with rolling within 10 degree increments over a 90 degree range. Based on these findings, the authors designed pen barrel rolling techniques to control object rotation, simultaneous parameter input, and mode selection. Because most input devices do not support barrel rotation, the authors used a custom-built, indirect pen. Tian, Ao, Wang, Setlur, and Dai (2007) and Tian et al. (2008) explored using pen tilt to enhance the orientation of a cursor and to operate a tilt-based pie-menu. The authors argued for an advantage to using tilt in these scenarios, but they used indirect input in their experiments, so applicability to direct input is unproven.

Ergonomics

Haider et al.'s (1982) study is perhaps one of the earliest studies of pen computing and focused on ergonomics. They recorded variables such as eye movement, muscle activity, and heart activity when using a light pen, touch screen, and keypad with a simulated police command and control system. The authors found lower levels of eye movement with the light pen but high amounts of muscle strain in the arms and shoulders as well as more frequent periods of increased heart rate. They noted

that because the display was fixed, participants would bend and twist their bodies to reduce the strain.

In another study using indirect pen input, Fitzmaurice et al.(1999) found that when writing or drawing, people prefer to rotate and position the paper with their nondominant hand rather than reach with their dominant hand. In addition to setting a comfortable working context and reducing fatigue, this also controls hand occlusion of the drawing. They found that using pen input on a tablet hampers this natural tendency because of the tablet's thickness and weight; hence the additional size of a Tablet PC will likely only exacerbate the problem.

2.2. Field Studies of In Situ Usage

Because field studies are most successful when investigating larger social and work related issues, researchers have focused on how pen computing has affected general working practices. For example, a business case study of mobile vehicle inspectors finds that with the addition of handwriting recognition and wireless networking, employees can submit reports faster and more accurately with pen computers (Chen, 2004). However, specific results relating to pen-based interaction—such as Inkpen et al. (2006), who include a longitudinal field study in their examination of handedness and PDAs—are less common.

Two field studies in education do report some aspects of Tablet PC interaction. Twining et al. (2005) reported on Tablet PC usage in 12 British elementary schools, including some discussion of usage habits. They found that staff tends to use convertible Tablet PCs in laptop mode, primarily to allow typing with the keyboard; although many still used the pen for GUI manipulation. However, when marking assignments or working with students, they use the pen in slate mode. The students were more likely to use the pen for making notes, though they used the onscreen keyboard or left their writing as digital ink instead of using writing recognition. Pen input enables the students to do things which would be more difficult with a mouse, such as create art work and animations. In fact, comments from several schools indicate that the pen is more natural for children. They also noted problems with the initial device cost, battery life, screen size, glare, and frequently lost pens.

In a field study of high school students and Tablet PCs, Sommerich, Ward, Sikdar, Payne, and Herman (2007) found that the technology does affect schoolwork patterns, but more relevant for our purposes is their discussion of ergonomic issues. For example, they found that the students used their Tablet PC away from a desk or table 35% of the time, 50% reported assuming awkward postures, and 69% experienced eye discomfort. No specific applications or issues with interaction are discussed.

2.3. Observational Usability Studies of Realistic Scenarios

There are fewer examples of research examining pen interaction and ergonomics in more complex tasks with real applications. Although evaluating low-level aspects

of pen input is certainly important, we contend that understanding how pen-based interaction functions with real applications and realistic tasks is equally if not more important, but unfortunately currently less understood. Briggs et al. noted this in 1993:

> While there has been a great deal of prior empirical research studying pen-based interfaces, virtually all prior research has examined the elementary components of pen-based interfaces separately (cursor movement, software navigation, handwriting recognition) for very elementary subtasks such as pointing, cursor movement, and menu selection. (p. 73)

Most pen-based research applications have been evaluated informally or with limited usability studies. In an informal study of the Electronic Cocktail Napkin, Gross and Do (1996) found that although there was no negative reaction to gestures, users had difficulty accurately specifying the pen position and encountered problems with hand occlusion when using marking menus. With CrossY (2004), no substantial user evaluation was reported, but initial user feedback found that, in spite of some difficulty discovering how the interface worked, users commented that "CrossY was much more intuitive and better suited for the task (i.e. drawing) and the tool (i.e. a pen)" (Apitz & Guimbretière, 2004, p. 11). Agarawala and Balakrishnan (2006), authors of BumpTop, conducted a small evaluation and found that participants were able to discover and use the functionality but that crossing widgets are awkward near display edges and noted problems with hand occlusion.

Briggs et al. (1993) compared user performance and preference when using indirect pen input and mouse/keyboard for operating business applications: word processing, spreadsheets, presentations, and disk management. Only the presentation graphics application and word processor supported mouse input in addition to keyboard commands. The experiment tested each application separately and the authors recruited both novice and expert users. They used custom-made, physical digitizer overlays with "buttons" to access specific commands for each application in addition to devoting digitizer space for controlling an onscreen cursor. Overall, they found that task times for the pen are longer for novice users with the word processor, and for all users when using the spreadsheet and file management. Much of their focus was on handwriting recognition, because at that time it was suggested that the pen was a viable, and even preferred, alternative to the keyboard for novice typists. However, the authors stated that "once the novelty wore off, most of the users hated the handwriting recognition component of the pen-based interface" (p. 79). For operations other than handwriting, the participants said that they preferred the fine motor control of the pen over the mouse when pointing, selecting, moving, and sketching. They also preferred selecting menus and buttons using the digitizer tablet.

A more recent study by Turner et al. (2007) compared how students revise and annotate UML diagrams using pen and paper, the Tablet PC, and a mouse and keyboard. They found that more detailed editing instructions are given with pen and paper and the use of gestural marks such as circles and errors were more common with pen and paper and Tablet PC. However, with mouse and keyboard, their participants

made notes with more explicit references to object names in the diagram. Their evaluation included only writing and drawing actions with a single application.

In spite of these researchers attempting to answer Briggs et al.'s call for more realistic pen input studies, one must remain cautious regarding their results because only the Turner et al. study uses direct pen input with a modern Tablet PC device and operating system. Moreover, only Turner et al. evaluate behavior with a conventional GUI.

2.4. Summary

Although many researchers have invented brand-new pen-centric interaction techniques like gestures and crossing, it is not clear if current GUIs must be abandoned, significantly altered, or merely enhanced at the input layer to better support pen interaction. The experimental results for raw pen performance seem encouraging, but there appear to be issues with accuracy, ergonomics, and occlusion. Researchers have examined aspects of pen input with conventional GUI widgets in the process of designing and evaluating alternative widgets and techniques, but their investigations and solutions have been evaluated in experimental isolation with synthetic tasks.

Although a controlled experiment gives some indication of raw performance, researchers such as Ramos et al. (2006) have argued that a more open-ended study can give users a better idea how a new tool will perform in real life. Using a tool in real life often equates to a field study, such as the pen-related field studies previously discussed. But these ethnographic inquiries are more suited to addressing general aspects of Tablet PC usage in a larger work context. In contrast, the observational studies of Briggs et al. (1993) and Turner et al. (2007) focus on specific tasks. Recent work from the CSCW community (Ranjan et al., 2006; Scott et al., 2004) have combined aspects of traditional field research methodologies with more specific inquiries into lower level interaction behavior with controlled tasks in a controlled setting—an approach we draw upon.

3. STUDY

Our goal is to examine how usable direct pen input is with a conventional GUI. For our study, we imagine a scenario where office workers must complete a presentation while away from their desk using their Tablet PC. They use typical office applications like a web browser, spreadsheet, and presentation tool. Because our focus is on GUI manipulation, the scenario could be completed without any text entry. Rather than conduct a highly controlled experimental study to examine individual performance characteristics in isolation, or, at the other extreme, an open-ended field study, we elected to perform a laboratory-based observational study situated between these two ends of the continuum with real tasks and real applications. By adopting this approach, we hope to gain a better understanding of how pen input performs using the status-quo GUI used with current Tablet PC computers.

Users primarily interact with a GUI through *widgets*—the individual elements which enable direct manipulation of underlying variables (see Figure 4 for examples). The frequency of use and location of widgets is not typically uniform. For example, in most applications, *menus* are used more often than *tree-views*. *Buttons* can appear almost anywhere, whereas *scrollbars* are typically located on the right or bottom. Also, some widgets provide redundant ways to control the same variable, enabling different usage strategies. For example, a scrollbar can be scrolled by either dragging a handle or clicking a button. To further add variability, a series of widgets may be used in quick succession forming a type of phrase (Buxton, 1995). For example making text "10 pt Arial Bold" requires selecting the text, picking from drop-down menus, and clicking a button in quick succession. Controlling all these aspects in a formal experiment would be difficult and we would likely miss effects only seen in more realistic contexts.

We had one group of users complete the tasks with a mouse as a control condition for intradevice comparison. We also recruited three groups of pen participants according to their level of computer and Tablet PC experience. To support our observational analysis, we gathered a rich set of logging data including 3D motion capture, video taken from the participant's point of view, screen capture video, and pen events such as movement and taps. We use these data to augment and refine our observational methodology with high-level quantitative analysis and visualizations to illustrate observations.

3.1. Participants

Sixteen volunteers (5 female, 11 male), with a mean age of 30.8 years (SD = 5.4) were recruited. All participants were right-handed, had experience with standard office applications, and used a computer for at least 5 hr per day on average. In a prestudy questionnaire, participants were asked to rate their experience with various devices and applications on a scale of 0 to 3, where 3 was a high amount of experience and 0 no experience. All participants said their experience with a mouse was 3 (*high*).

3.2. Design

A between-participants design was used, with the 16 participants divided into four groups of 4 people each. One group used a mouse during the study and acted as a baseline control group. The remaining three groups all used the Tablet PC during the study, but each of these groups contained participants with different level of Tablet PC or conventional computer experience. In summary, the four groups were as follows:

- *Mouse*. This was the control group where participants used a conventional mouse. Participants in this group said they used a computer for between 8 and 9 hr per day.
- *Pen1-TabletExperts*. These were the only *experienced Tablet PC users*. Unlike the other pen groups, they all reported a high amount of experience with Tablet

PC pen-based computing in the prestudy questionnaire. They also reported using a computer for between 6 and 10 hr per day.

- *Pen2-ConventionalExperts*. These were *experienced computer users* who used a wide range of hardware, software, and operating systems, but they *did not* report having any experience with Tablet PC pen-based computing. They also reported that, on average, they used a computer between 9 and 10 hr per day.
- *Pen3-ConventionalNovices*. These were *computer users with limited experience* who used a single operating system and had a limited range of software experience (primarily standard office applications like word processors, spreadsheets, web browsing, and presentation tools). As with the *Pen-2-ConventionalExperts* group, they did not have any experience with Tablet PCs. They reported using a computer between 5 and 7 hr per day, which is less than all other groups.

3.3. Apparatus

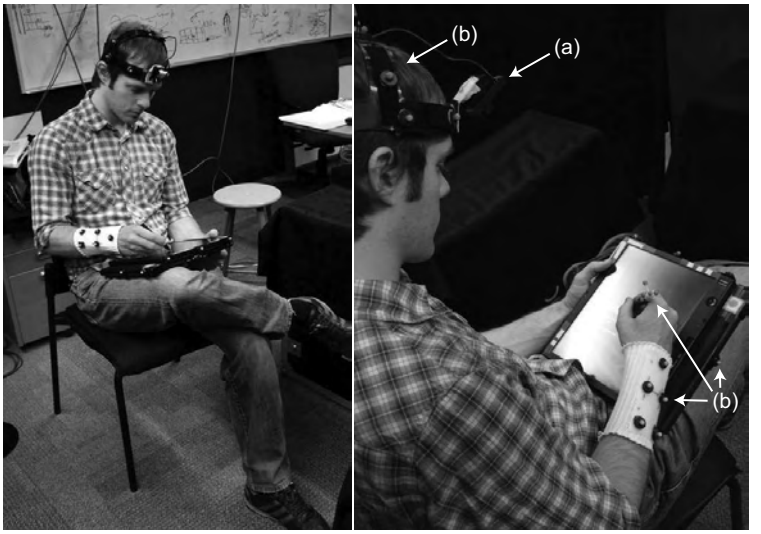
The study was conducted using a Lenovo X60 Tablet PC with Intel L2400 @ 1.66GHz and 2GB RAM. It has a 1050 × 1400 pixel display measuring 184 × 246 mm for a pixel resolution of 5.7 pixels/mm. We used the Windows Vista operating system and Microsoft Office 2007 applications because they were state of the art at the time (we conducted this experiment in 2007) and both were marketed as having improvements for pen computing. The scenario applications were Microsoft PowerPoint 2007 (presentation tool), Microsoft Excel 2007 (spreadsheet), and Internet Explorer 7 (web browser). Since the completion of this study, Microsoft released Windows 7. It includes all of Vista's pen computing improvements and adds only two improvements for text entry (Microsoft, 2009), thus our results remain as relevant for Windows 7 as they do for Windows Vista.

We gathered data from four different logging sources:

- *Screen capture*. The entire 1040 × 1400 pixel display was recorded as a digital screen capture video at 22 frames per second (fps).
- *Pen events*. Custom logging software recorded the pen (or mouse) position, click events, and key presses.
- *User view*. A head-mounted 640 × 480 pixel camera recorded the user's view of the tablet at 30 fps (Figure 1a). A microphone also captured participant comments and experimenter instructions.
- *Motion capture*. A Vicon 3D motion capture system (www.vicon.com) recorded the position and orientation of the head, forearm, tablet, and pen using 9.5 mm markers (Figure 1b) at 120 fps. These data were filtered and down sampled to 30 fps for playback and analysis.

At the most basic level, these logs provided a record of the participant's progress through the scenario. However, by recording their actions in multiple ways, we hoped we could discern when an intended action was successful or not. Moreover, capturing 2D and 3D movements would enable us to visualize characteristic motions. We also

FIGURE 1. Experimental setup and apparatus. *Note.* In the Tablet PC conditions, the participant was seated with the tablet on their lap: (a) a head-mounted video camera captured their point-of-view; (b) 9.5mm passive markers attached to the head, forearm, pen and tablet enabled 3D motion capture.



felt that the user view log, showing the hand, pen, and display together, would be particularly useful for analysing direct input interactions.

The Motion Capture and User View logs ran on dedicated computers. The Screen and Pen Event logs ran on the tablet without introducing any noticeable lag. Although the Vicon motion tracking system supports submillimetre tracking, the captured data can be somewhat noisy due to the computer vision-based reconstruction process. To compensate for this noise, we applied a low pass filter using cut-off frequencies of 2Hz for position and 6Hz for rotation before down sampling to 30 fps.

Unlike most controlled experiments with Tablet PC input, we intentionally did not place the tablet in a fixed location on a desk. Instead participants were seated in a standard chair with the tablet configured in slate mode and held on their lap (Figure 1). This was done to approximate a working environment where tablet usage would be most beneficial (e.g., while riding a subway, sitting at a park bench, etc.). If the device is placed on a desk, then using a mouse becomes practical, and perhaps users in the environment would opt to use a mouse instead of the pen. Mouse participants were seated at a standard desk with the same Tablet PC configured in open laptop mode. A wired, 800 DPI, infrared mouse was used with the default Windows pointer acceleration (dynamic control-display gain) setting.

3.4. Protocol

The study protocol required participants to follow instructions as they completed building a short Microsoft PowerPoint presentation slide deck using a web browser,

spreadsheet, and presentation tool. The slide deck was partially constructed at the beginning of the study, and users could complete all tasks without entering any text. Participants were told that the study was not about memorization or application usability. They were instructed to listen closely to instructions and then complete the task as efficiently as possible. As they worked, they were told to "think aloud" by saying what they were thinking, especially when encountering problems. If they forgot what they were doing altogether, the experimenter intervened and clarified the task.

Each session was conducted as follows: First, each of the participants in the three tablet groups completed the pointing, dragging, and right-clicking sections of the standard Windows Vista Tablet PC tutorial. Then all participants completed a brief PowerPoint drawing object manipulation training exercise, as early pilot experiments revealed that these widgets were not immediately intuitive. This training period took between 5 and 10 min. Once training was completed, the main portion of the study began. The experimenter used a prepared script to issue tasks for the participant to complete. The participant completed each task before the experimenter read the next one. At the conclusion of the study, we conducted a brief debriefing interview.

Tasks

In total, our seven-page study script contained 47 tasks that had to be completed. Because reproducing the entire script here is not feasible, we summarize the main sections and give a small example from the script.

The first eight tasks asked the participant to open a PowerPoint file and correct the position and formatting of labelled animal thumbnails on a map (Figure 2). The text for Tasks 4 through 8 are shown next to convey the general style:

Task 4: Correct the position of the polar bear and owl thumbnails: the polar bear's habitat is in the north and the owl is in the south.

Task 5: Make the size, orientation, and aspect ratio of all the thumbnails approximately the same as the beaver. It's fine to judge this "by eye."

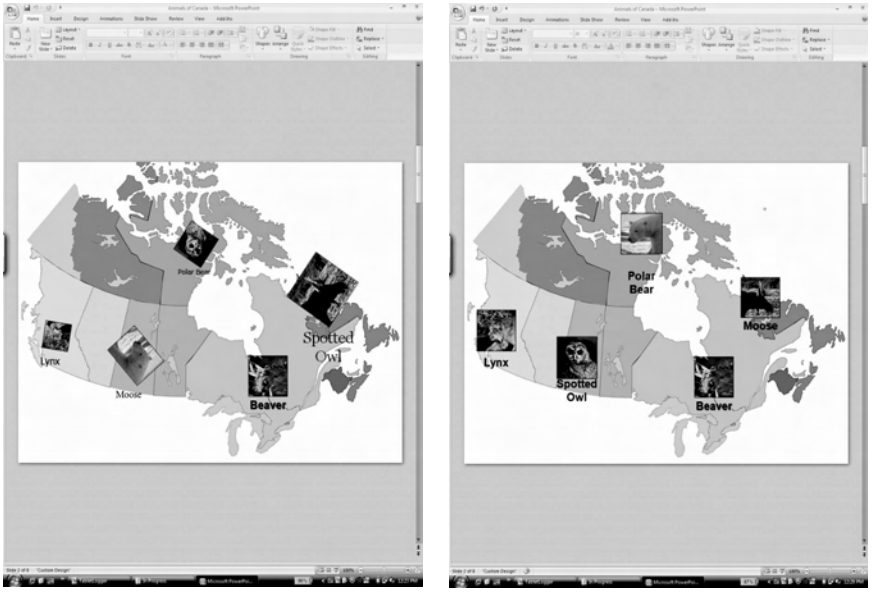
Task 6: Change all labels to the same typeface and style as the beaver (20pt Arial Bold). You may have to select the "Home" tab to see the font controls.

Task 7: Now, we'll make all animal names perfectly centered *below* the thumbnail. Select the label and the thumbnail *together* and pick "Align/Align Center" from the "Arrange" button in the "Drawing" toolbar.

Task 8: This is a good time to save your presentation. Press the save document icon located at the top left of the application.

Tasks 9 through 20 continued with the participant completing a slide about one of the animals by copying and pasting text from Wikipedia and inserting a picture from a file folder (Figure 3a). Tasks 21 to 26 repeated the same steps with two more partially completed animal slides. In Tasks 27 to 37, the participant used Microsoft Excel to create a chart of animal population trends from a preexisting data table, copied and pasted it into the final slide of the presentation, and added ink annotations such as written text and colored highlighting (Figure 3b). Finally Tasks 38 to 47 asked the

FIGURE 2. Study screen captures taken from initial task sequence where the participant corrects the formatting of labelled animal thumbnails on a map: (a) before task 4 where some thumbnails are in the wrong position, scaled or rotated incorrectly, or have text labels rendered in different fonts and sizes; (b) at the conclusion of task 8 when the participant said all requested corrections to the thumbnails are complete.



(a) initial state of slide at beginning of task 4 (b) final state of slide at end of task 8

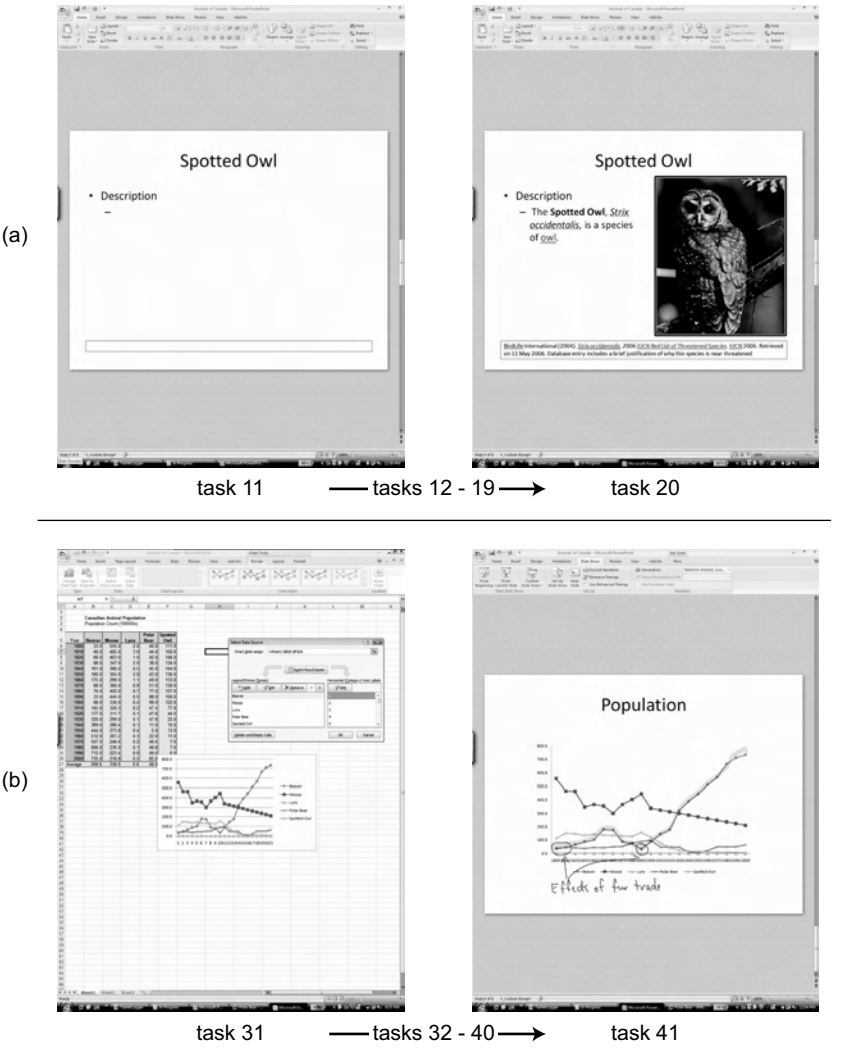
participant to configure various slide show options and save an HTML version. After viewing the final presentation in the web browser, the study was complete.

The tasks in the study covered simple interactions like pressing the Save button, as well as more complex tasks like formatting several text boxes on a presentation slide. We included two different types of tasks to reflect real world usage patterns:

- **40 Constrained tasks** had a predictable sequence of steps and a clearly defined goal for task completion. Task 8 (shown earlier), which asked the participant to press the Save button, is an example.
- **7 Open-ended tasks** had a variable sequence of steps and required the participant to assess when the goal was reached and the task complete. Task 5 (shown earlier), which asked the participant to match the size and orientation of animal thumbnails, is an example.

Participants took between 40 min and 1 hr to complete the study, including apparatus setup, training time, and debriefing. Comments from our participants suggested that the tasks were not too repetitive or too difficult, and in contrast to formal experiments we have administered, participants felt that time passed quickly—some even said the study was enjoyable.

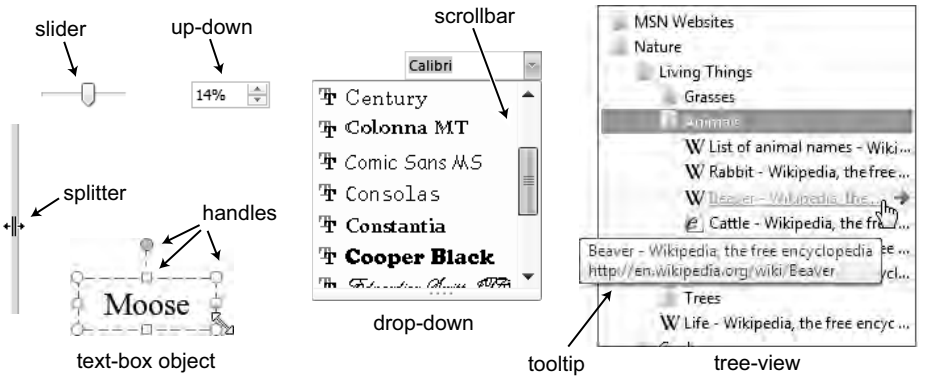
FIGURE 3. Screen captures of selected scenario tasks: (a) tasks 11 and 20, in which the participant is asked to complete a slide about one of the animals by copying and pasting text from Wikipedia, and inserting a picture from a file folder; (b) tasks 31 and 41, in which the participant uses Excel to create a chart of animal population trends from a pre-existing data table, copy and paste it into the final slide of the presentation, and add ink annotations.



Widgets and Actions

The tasks in our study were designed to exercise different widgets like menus and scrollbars, and common actions like text selection, drawing, and handwriting. Figure 4 illustrates some of the lesser known widgets with a brief explanation. For our purposes, we categorize widgets according to their functionality, visual appearance, and invocation. For example, a menu is different than a context menu because the

FIGURE 4. Illustration of some widgets: a *slider* adjusts a parameter by dragging a constrained handle; an *up-down* increments or decrements a parameter by incrementing using a button or by entering the value in a text-box; a *drop-down* selects one option from a list of choices which are shown upon invocation—it may use a *scrollbar* to access a long list; a *tree-view* enables navigation through a hierarchical collection of items by expanding and collapsing nodes; a *text-box* object is one kind of drawing object in PowerPoint—it uses *handles* for scaling and rotation; a *splitter* changes the proportion of two neighbouring regions by dragging a constrained handle.



latter is invoked with a right-click, a drop-down is different from a menu because the former has a scrollable list of items, and a tree-view is different because hierarchical items can be expanded and collapsed. Note that widgets can also be categorized according to widget capabilities (Gajos & Weld, 2004), but this would consider a menu, tree-view, drop-down, and even a group of radio buttons equivalent, because all are capable of selecting a single choice given a set of possible options. However, in our categorization, these are different widgets because their functionality and visual appearance are different.

During our study, we expected participants to use 20 different widgets and five actions (Figure 5). The actions include three types of selection: marquee (enclosing two or more objects by defining a rectangle with a drag), cell (selecting an area of spreadsheet cells by dragging), and text (selecting text by dragging).

By analyzing our script, we calculated the expected minimum number of widget or action *occurrences* (Figure 5, column N) necessary to complete the scenario. If a complex widget is composed of other complex widgets (such as when a drop-down includes a scrollbar) we considered these to be nested but distinct occurrences of two different widgets. In the case of buttons, we did not create a distinct occurrence for a button used to open another widget such as a menu or drop-down, or a button that is inherently part of a widget, such as the pagination buttons on a scrollbar. The specific instance of each widget or action may vary by size, orientation, position, and magnitude of adjustment.

Note that the occurrence frequency distribution is not balanced and is dependent on the particular tasks in our script. This is a trade-off when adopting a realistic scenario such as in our study. However, we feel our tasks are representative of those

FIGURE 5. Ideal amount of widget and action usage in our study.

Widget/Action	N	I	Widget/Action	N	I	Widget/Action	N	I
button	52	*52*	up-down	8	*88*	window *handle*	4	*8*
drop-down	20	*40*	text select*	6	*6*	writing*	3	*3*
scrollbar	20	*20*	marquee select*	6	*6*	cell select*	2	*2*
menu	19	*36*	check-box	6	*6*	chart *object*	2	*3*
text-box *object*	18	*27*	slider	5	*9*	hyperlink	2	*2*
object handle	17	*17*	drawing*	5	*5*	color *choice*	1	*1*
tab	16	*16*	image *object*	5	*5*	splitter	1	*1*
context menu	16	*32*	tree-view	4	*7*			
file *object*	11	*11*	radio button	4	*4*			

Note. Actions are indicated with an asterisk. N = number of widget *occurrences*; I = number of *interactions* such as clicks and drags.

used in other common nontext entry office application scenarios and, compared to repetitive controlled experiments, are much more representative of real usage patterns.

The total number of widget or action occurrences conveys only part of their usage frequency. We also computed the expected ideal number of *interactions* (clicks, drags, right-clicks, etc.) for each occurrence (Figure 5, column I). Some widgets, such as a single-level context menu, always have a predictable number of interactions—right-click followed by a single-click—resulting in two interactions per occurrence. Other widgets such as an up-down widget can have wide variance due to their magnitude of adjustment. If an up-down has to be decremented from a value of 5 to a value of 4 using 0.1 steps, it requires 9 click interactions. Some widgets, such as the scrollbar, enable a task to be completed in different ways. For example, if a browser page must be scrolled to the bottom, the user may drag the scroll box to the bottom of the page, make many clicks in the paging region, or press and hold the scrollbar button (see Figure 28 for scrollbar parts). When calculating types of interactions for this type of widget, we selected the optimum strategy in terms of effort. For the ideal number of interactions we used the minimum. In other words, we use the most efficient interaction style without any errors. For example, our predictions may assume that a scrollbar can be operated with a single drag rather than a sequence of page-down taps for a long scrolling action. In practice, this ideal number of interactions may be difficult to achieve, but it does provide a theoretical bound on efficiency and enabled us to get a sense of the widget frequencies a priori. In total, we calculated ideal numbers of 253 widget occurrences and 407 interactions like clicking and dragging during the study (Figure 6).

4. ANALYSIS

We based our methodology on *Interaction Analysis* (Jordan & Henderson, 1995) and *Open Coding* (Strauss & Corbin, 1998). Interaction Analysis is "is an interdisciplinary method for the empirical investigation of the interaction of human beings

FIGURE 6. Ideal number of expected interactions, by interaction type.

Interaction	No.
Single-click	301
Double-click	11
Drag	79
Right-click	16
Total	407

with each other and with objects in their environment" (Jordan & Henderson, 1995, p. 39). It leverages video as the primary record of observation and uses this to perform much more detailed analyses of subtle observations than would be possible with traditional written field notes. However, we gathered an even richer and more quantitative collection of observational data, and focus on a much more constrained interaction context. Thus, we follow Scott (2005) and use the open coding approach rather than utilizing a more formal coding methodology with an interdisciplinary team of researchers. Open coding begins with a loosely defined set of codes in a first examination of the data logs, and then with subsequent iterations; the codes are refined and focused as trends emerge. In our case, we began coding general errors and refined this to ten specific types of errors, and then iterated again to code specific widgets and actions for more detailed analysis.

4.1. Custom Log Viewing and Analysis Software

Our user point-of-view video log, motion capture data, screen capture video, and pen event log were synchronized, segmented, and annotated using custom software we developed (Figure 7). Each data source is viewed in a separate player with the ability to pause, time scrub, frame advance, adjust playback speed, and so on. Our software is similar to the INTERACT commercial software package (www.mangold-international.com). However, by building our own system, we could include custom visualizations for the pen event log and motion capture (see also Figure 8), more accurately synchronize the logs using our "time mark" scheme (explained next), easily gather data points based on annotation, and write code to gather specific data for statistics. Although this tool was purpose built for this analysis, we have since revised it to be more general and used it for another video coding project (Vogel & Balakrishnan, 2010).

4.2. Synchronization

The user view video source was used as the master time log. To assist with synchronization, six visual "time markers" were created by the participant during the study session. Each time mark was created by lifting the pen high above the tablet and then swiftly bringing the pen down to press a large button in the pen event logging application. This created a time stamp in the pen event log and changed the button to red

FIGURE 7. Analysis tool: (a) screen capture player; (b) pen event player; (c) user view player; (d) synchronization markers and unified playback control; (e) motion capture player; (f) annotation detailed description; (g) annotation codes.

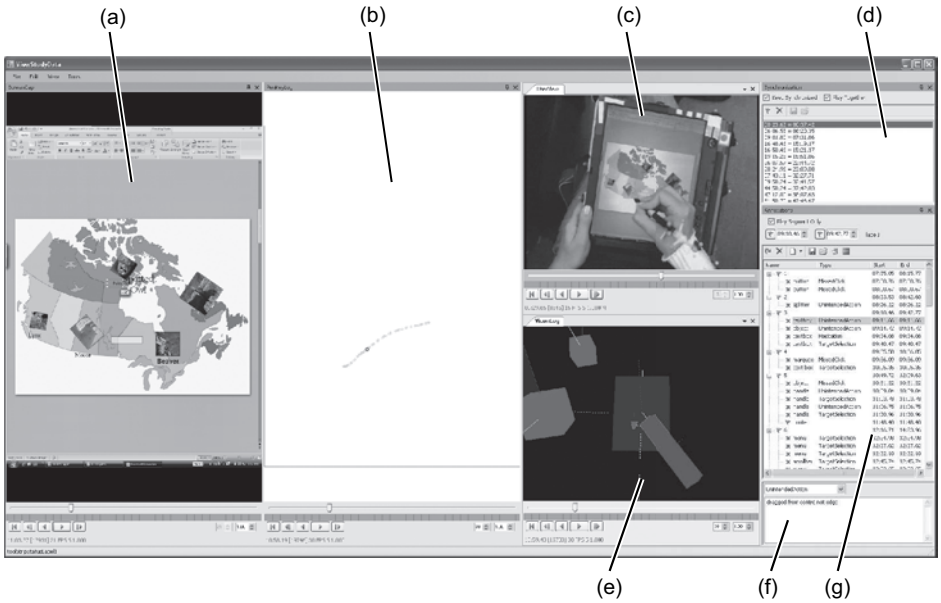
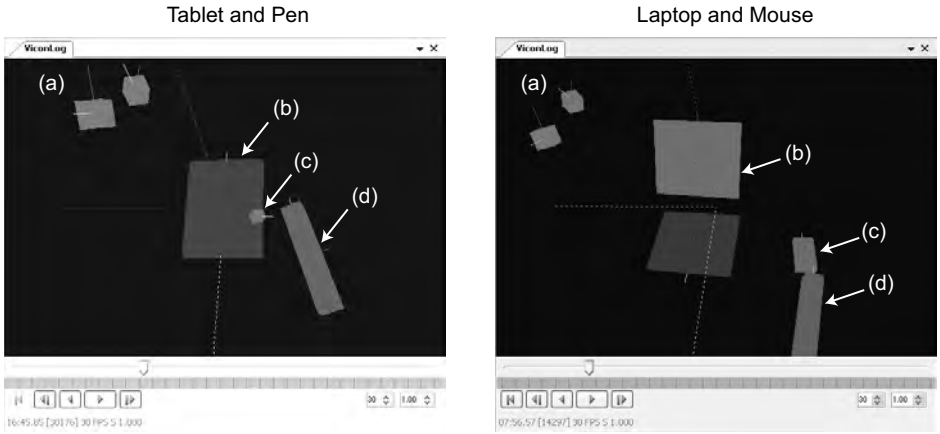


FIGURE 8. Motion capture player. *Note.* In addition to conventional playback controls, a 3D camera can be freely positioned for different views. In the views shown in the figure the camera is looking over the participant's shoulder, similar to the photo in Figure 1 right. Objects being tracked are: (a) head position; (b) tablet or laptop; (c) pen tip or mouse; (d) forearm.



for 1 s, which created a visual marker for the user view and screen capture logs. The distinctive pen motion functioned as a recognizable mark in the motion capture log.

4.3. Segmentation Into Task Interaction Sequences

Once synchronized, the logs were manually segmented into 47 sequences in which the participant was actively completing each task in our script. Because our study is more structured than typical interaction analysis, this is analogous to the first step of the open coding process where the data are segmented into a preliminary structure. A sequence began when the participant started to move toward the first widget or object, and the sequence ended when he or she completed the task. This removed times when the experimenter was introducing a task and providing initial instructions, the participant was commenting on a task after it was completed, or when stopping and restarting the motion tracking system. This reduced the total data log time for each participant to between 20 and 30 min.

4.4. Event Annotation and Coding

The coding of the 47 task sequences for each of the 16 participants was performed in stages with a progressive refinement of the codes based on an open coding approach (Strauss & Corbin, 1998) with two *raters*—two different people identified events and coded annotations.

First, a single rater annotated where some event of interest occurred, with an emphasis on errors. Next, these general annotations were split into three classes of codes, and one class, *interaction errors*, was further split into six specific types. A second rater was then trained using this set of codes. During the training process, the codes were further refined with the addition of a seventh type of interaction error and a fourth class (both of these were subsets of existing codes). Training also produced coding decision trees, which provided both raters with more specific guidelines for code classification and event time assignment (see next). The second rater used this final set of codes and classification guidelines to independently identify events and code them across all participants. The first rater also independently refined their codes across all participants as dictated by the final set of codes and guidelines.

There was a high level of agreement of codes for events found by both raters (Cohen's Kappa of 0.89) but also a high number of events identified by one rater but not the other. We considered an event to be found by both raters if each rater annotated events with times separated by less than 2 s. Both raters found 779 events, but Rater 1 and Rater 2 found 238 and 251 additional events, respectively. A random check of these additional events strongly indicated that these were valid events but had simply been missed by the other rater. Moreover, the codes for these missed events did not appear to have a strong bias; raters did not seem to miss a particular type of event. Thus, with the assumption that all identified events are valid, events found by both raters account for 61% of all events found. In a similar coding exercise,

Scholtz, Young, Drury, and Yanco (2004) also found that raters had difficulty finding events of interest (called "critical incidents" in their domain).

Given the high level of agreement between raters when both raters identified the same event, we felt justified to merge all events and codes from both raters. When there was disagreement (66 of 779 cases), we arbitrarily chose to use the code from Rater 1. We should note that Rater 1 was the primary investigator. To guard against any unintentional bias, we examined our results when Rater 2 was chosen: We found no significant changes in the quantitative results.

Annotation Events and Codes

Each annotation included the code type, synchronized log time for the event, the widget or object context if applicable, and additional description as necessary.

Code Types. We identified four classes of codes: *experiment instructions*, *application usability*, *visual search*, and *interaction errors*.

Events coded as experiment instructions, application usability, and visual search are general in nature and should not be specific to an input device. We felt these codes forced us to separate true interaction error codes from these other types of noninteraction errors:

- *Experiment Instructions*: performed the wrong task, adjusted wrong parameter (e.g., when asked to make photo 4-in. wide, the participant adjusted the height instead), or asked the experimenter for clarification
- *Application Usability*: application design led directly to an error (we identified specific cases as guidelines for raters; see below)
- *Visual Search*: performing a prolonged visual search for a known target

Because our focus is on pen based interaction on a Tablet PC versus the baseline mouse based interaction, we were most interested in *interaction errors,* which occurred when the participant had difficulty manipulating a widget or performing an action. We defined eight types of interaction error codes:

- *Target Selection*: could not click on intended target (e.g., clicking outside the scrollbar)
- *Missed Click*: making a click action, but not registering any type of click (e.g., tapping too lightly to register a click)
- *Wrong Click*: attempting one type of click action but a different one is recognized by the system (e.g., right-clicking instead of dragging a scrollbar, single-click instead of a double-click)
- *Unintended Action*: attempting one type of interaction but accidentally invoking a different one (e.g., attempted to open a file with a single-click when a double-click is required)
- *Inefficient Operation*: reaching the desired goal but without doing so in the most efficient manner (e.g., scrolling a large document with individual page-down clicks rather than dragging the scroll box; overshooting an up-down value and having to backtrack)

- *Repeated Invocation*: unnecessarily invoking the same action multiple times (e.g., pressing the Save button more than once just to be sure it registered)
- *Hesitation*: pausing before clicking or releasing (e.g., about to click on target, then stop to carefully position pen tip)
- *Other*: errors not described by the aforementioned codes

Event Times. The time to log for an event was almost always at the beginning. An ambiguous case occurs when the participant is dragging. We defined the time of the event to be when the participant set the error in motion. For example, when selecting text or multiple objects with a marquee, if the click down location constrained the selection such that an error was unavoidable, then the event time is logged at the down action. However, if the error occurs at the up action, such as a movement while releasing the pen tip from the display, then the event time is logged at the up action.

Coding Decision Trees. We developed two coding selection decision trees: One is used when a participant makes a noticeable pause between actions (Figure 9) and a second is used when a participant attempts an action (Figure 10). We defined "action" as an attempted click ("tap"); "right-click"; beginning or ending of a drag; or operating a physical key, wheel, or button. The definition of "noticeable" is somewhat subjective and required training—a rough guideline is to look for pauses more than 2 s that interrupt an otherwise fluid movement. With some practice, these noticeable pauses became more obvious.

Interactions of Interest

After a preliminary qualitative analysis of the task sequences and interaction errors, we identified specific interactions of interest and further segmented these for more detailed analysis. We selected widgets and actions that are used frequently (button), highly error prone (scrollbar), presented interesting movements (text selection), or highlighted differences between the pen and mouse (drawing, handwriting, and keyboard use). These are discussed in Section 6.

Other Annotations

We also transcribed relevant comments from the participant and noted sequences where problems were caused by occlusion or tooltips and other hover-triggered visual feedback.

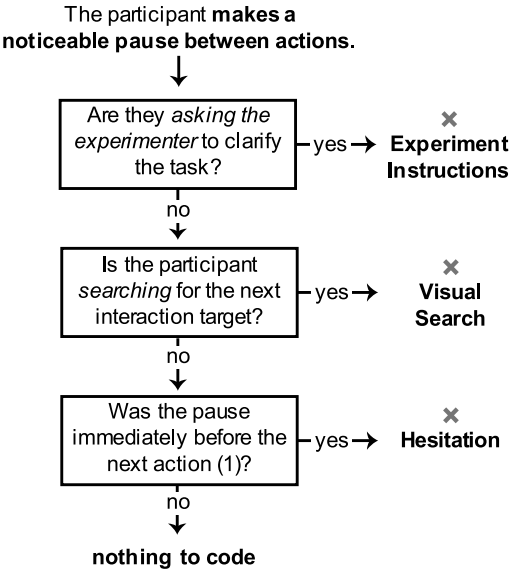
5. RESULTS

5.1. Time

To get a sense of performance differences between mouse and the three pen groups, we calculated group means for the total time used to complete the 40 constrained tasks because these tasks have a predictable sequence of steps and a clearly defined goal for task completion (Figure 11). The graph suggests slightly higher times

for the pen condition, decreasing with increased computer experience (a similar trend can be seen for the total time for all tasks, but with much higher variance). A one-way analysis of variance (ANOVA) found a significant main effect of group on time, $F(3, 12) = 7.445$, $p < .005$, with the total times for the *Pen3-ConventionalNovices* group significantly longer than *Mouse group* and *Pen1-TabletExperts* groups ($p < .02$, using the Bonferroni adjustment). No other significant differences were found between the other groups. Perhaps more interesting is the range of completion times for mouse and pen participants. The best and worst times for the Mouse group were 12.8 min (P1) and 19.0 min (P3), whereas the best and worst times across all pen groups were 16.1 min (P7-*Pen1-TabletExperts*) and 28.0 min (P13-*Pen3-ConventionalNovices*). The best time for a mouse participant is well below the best pen user time, even for expert Tablet PC users.

FIGURE 9. Coding decision when participant makes a noticeable pause. *Note.* See below for additional notes (numbered notes in parentheses).



(1) "just before": The participant has found the location for the action, but does not perform the interaction immediately.

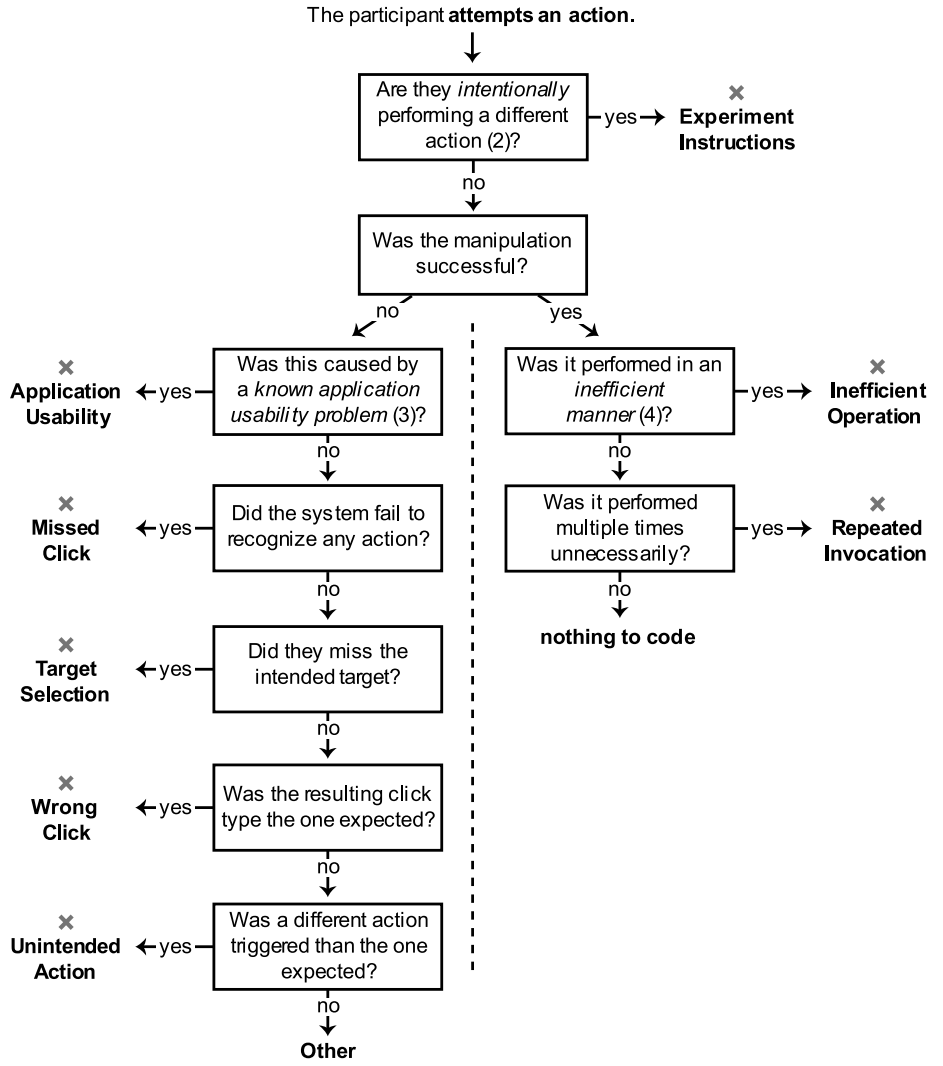
(2) "different action": wrong task, adjusting wrong parameter such as adjusting height instead of width, attempting to gain access to parameter in a different way than requested (e.g. using a context menu instead of a toolbar).

(3) Known usability problems were identified to remove errors that are application specific or exhibit obvious poor design. Many involved the *PowerPoint (PPT) textbox*:

attempting a move a *PPT textbox* by trying to drag the text directly (PPT requires the user to first select the text box, then drag it from the edge)

default insertion point in a *PPT textbox* selects single word only and subsequent formatting does not affect entire textbox (rather than selecting all text first)

FIGURE 10. Coding decision tree when participant attempts an action. *Note.* See below for additional notes (numbered notes in parentheses).



marquee selection misses the invisible *PPT textbox* bounding box

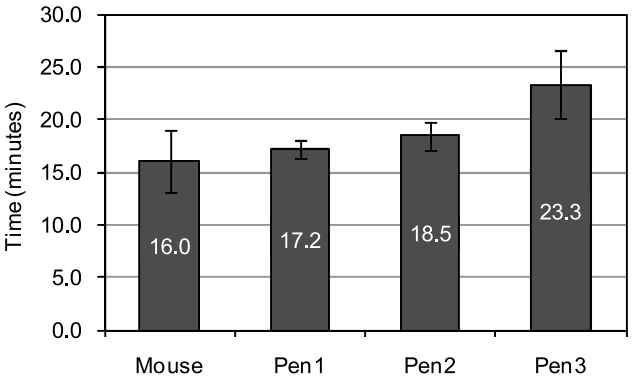
attempting to select text in a *PPT textbox* but the down action is off of the visible text which deselects the textbox and begins a selection marquee instead

changing a *checkbox* state by clicking on the label text (but the application only supports a direct click on a checkbox)

problem selecting an *Excel chart object* (when opening a context menu or moving a chart, the application often selects an inner "plot area" rather than entire chart).

(4) Inefficient Manner: reaching the desired goal, but doing so in a noticeably inefficient manner (e.g. scrolling a large document with individual page down clicks rather than dragging the scroll box; overshooting an up-down value by several steps and having to backtrack).

FIGURE 11. Mean time for all constrained tasks per group. ***Note.*** **Error bars in all graphs are 95% confidence interval; short pen participant group names are used in figures:** ***Pen1-TabletExperts*****,** ***Pen2-ConventionalExperts*****,** ***Pen3-ConventionalNovices*****.**



5.2. Errors

We annotated 1,276 errors across all 16 participants in all 47 tasks. This included noninteraction errors (*experiment instruction, visual search,* and *application usability*), which we briefly discuss before focusing on *interaction errors,* which are the most relevant.

Noninteraction Errors

We found 72 application usability errors, 151 experiment instruction errors, and 41 visual search errors overall. The mean number per group does not appear to form a pattern, with the possible exception of application usability appearing higher with *Pen3-ConventionalNovices* (Figure 12). However, no significant differences were found. The large variance for experiment instructions with the *Mouse* group was due to Participant 3, who often clarified task instructions (that person had 25 experiment instruction errors compared to the next highest participant in the study with 15, a *Pen2ConventionalExpert*).

The breakdown of specific application usability errors (which we identified during the coding process; see earlier) found a large proportion of errors when attempting to select text in the text box (49%) and marquee selection missing the "invisible" text box bounding box (17%).

Interaction Errors

Recall that interaction errors occur when a click or drag interaction is attempted. The mean number of interaction errors in each group suggests a pronounced difference between mouse and pen groups (Figure 13). A one-way ANOVA found a significant main effect of group on interaction errors, $F(3, 12) = 10.496$, $p = .001$. Post hoc analysis found the *Mouse* group lower than both *Pen2-ConventionalExperts* and *Pen3ConventionalNovices*, and *Pen1-TabletExperts* lower than *Pen3ConventionalNovices* (all ps $< .05$, using the Bonferroni adjustment).

FIGURE 12. Mean noninteraction errors per group.

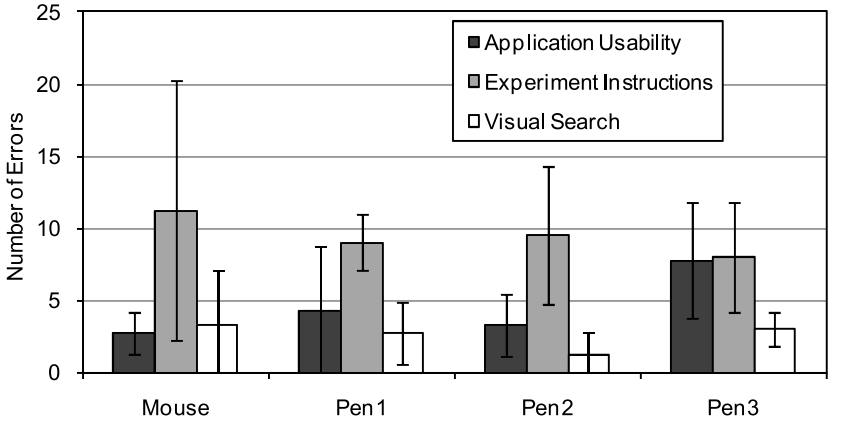
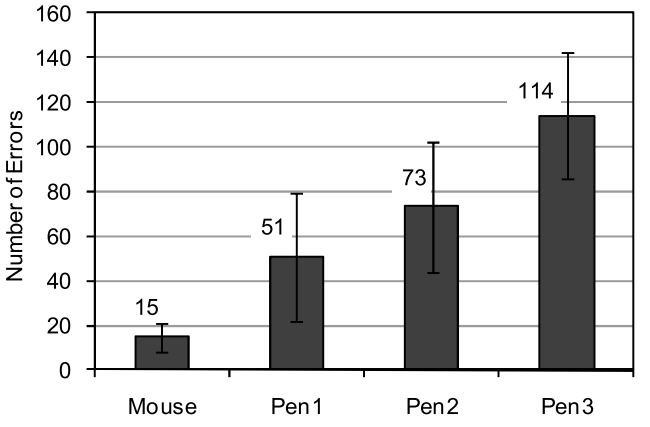


FIGURE 13. Mean interaction errors per group.

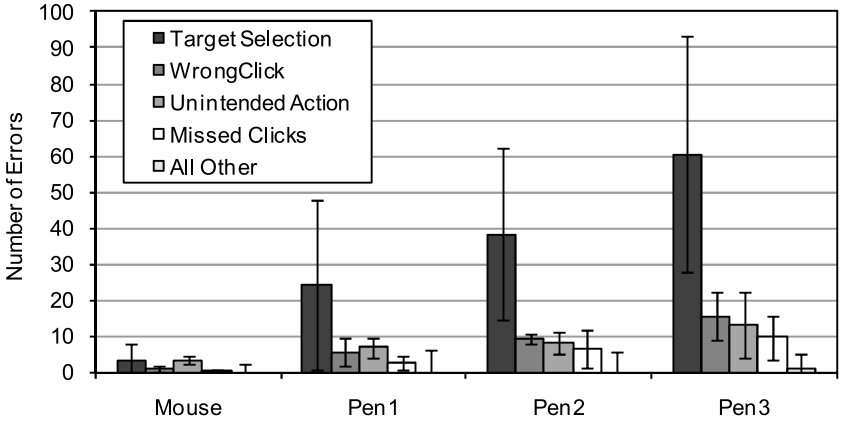


From a breakdown of interaction error type for each group (Figure 14), target selection appears to be most problematic with pen users, especially those with less experience, the *Pen3ConventionalNovices* group. For the mouse group, unintended actions and target selection were similar, but all errors were low. Wrong clicks, missed clicks, and unintended actions are roughly comparable across pen groups, with a slight increase with less experience. Not surprisingly, there were no missed click errors with the mouse group—a dedicated button makes this type of error unlikely. We analyze each interaction error type next, with an emphasis on target selection, wrong clicks, unintended actions, and missed clicks.

Target Selection Error Location

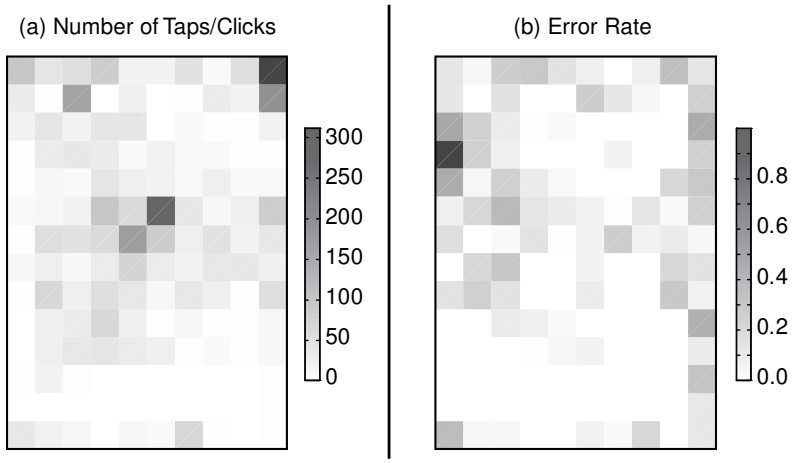
All Tablet PC participants had trouble selecting targets, with the number of errors increasing with lower computer experience. In reviewing the data logs, we

FIGURE 14. Mean interaction errors by error type. *Note.* All other errors include hesitation, inefficient operation, and repeated invocation.



noted that target selection errors may be related to location. A *heat map plot*—a 2D histogram using color intensity to represent quantity in discrete bins—compares the concentration of all taps and clicks for all tasks for all pen participants (Figure 15a) and the relative error rate computed by taking the ratio of number of errors to the number of taps/clicks per bin (Figure 15b). The concentration of taps/clicks is somewhat centralized (aside from the peak in the upper right where up-down widgets in Tasks 15, 21, 23, and 34 required many sequential clicks). The error rate has a different distribution, with higher concentrations near the mid-upper-left and along the right side of the display compared to all taps/clicks.

FIGURE 15. Pen participant *heat map* plots showing 2D concentration of (a) all taps/clicks; and (b) target selection error rate, the number of errors over taps/clicks. *Note.* A heat map is a 2D histogram using color intensity to represent quantity in discrete bins; in this case, each bin is 105 × 100 pixels.



There may also be an interaction with target size, which we could not control for. We could have carefully annotated actual target size based on the screen capture log, but this would have required considerable effort which did not seem to provide a suitably large gain in analysis. A better approach would be to conduct a controlled experiment in the future to validate this initial finding.

Wrong Click Errors

We observed three types of unintended actions which occurred with five or more Tablet PC participants (Figure 16). Participants had problems accidentally invoking a right-click when attempting to single-click or drag (Figure 16a, 16b). Right-clicking instead of clicking occurred most often with the up-down widget. In some cases participants held the increment and decrement buttons down expecting this to be an alternate way to adjust the value. Other common contexts were long button presses, and triggering in menus by dwelling on the top-level item to open nested items. Right-clicking instead of dragging occurred most often with scrollbars and sliders. With these widgets, pressing and holding the scrolling handle triggered a right-click if the drag motion did not begin before the right-click dwell period.

A similar problem occurred when a slow double-click was recognized as two single-clicks in file navigation (Figure 16c). This often put the file or folder object in rename mode, and subsequent actions could corrupt the text. It appears to be because of timing and location: If the two clicks were not performed quickly enough and near enough to each other, they were not registered as a double-click. Accot and Zhai (2002) note that the rapid successive clicking actions required by double-clicking can be difficult to perform while keeping a consistent cursor position, and our data support this intuition. This is a symptom of pen movement while tapping, which we explore in more detail below.

FIGURE 16. Wrong click errors.

Unintended Action Type	Tablet PC	Mouse	Frequent Pen Contexts
a) right-click instead of a single-click	32 occurrences 9 participants 0.9% rate[a]	none	up-down (15), drop-down (7), button (4), menu (2)
b) right-click instead of drag	26 occurrences 9 participants 2.7% rate[b]	none	slider (12), scrollbar (7)
c) click instead of double-click	24 occurrences 8 participants 18% rate[c]	1 occurrence 1 participant 2% rate[c]	file object (all)
d) click instead of right-click	14 occurrences 7 participants 7.3% rate[d]	none	context menu invocation (13)

[a]Occurrence rate calculated using 300 estimated single-clicks (Figure 6). [b]Occurrence rate calculated using 79 estimated drags (Figure 6). [c]Occurrence rate calculated using 11 estimated file operations (Figure 5). [d]Occurrence rate calculated using 16 estimated right-clicks (Figure 6).

Clicking instead of right-clicking was less frequent (Figure 16d), but when an error occurred, the results were often costly. For example, when invoking a context menu over a text selection, several participants clicked on a hyperlink by accident. This not only was disorienting but also required them to navigate back from the new page and select the text a second time before trying again. We also noted cases of dragging or right-dragging instead of clicking or right-clicking. Accidental right-dragging was also disorienting. The end of even a short right-drag on many objects opens a small context menu; because this occurred most often when participants were opening a different context menu, they easily become confused.

Unintended Action Errors

We observed three types of unintended actions that occurred with five or more Tablet PC participants (Figure 17). Five were caused by erroneous click or drag invocations, one occurred when completing a drag operation, and one was caused when navigating files folders. In fact, two of these unintended action errors occurred almost exclusively with file folder navigation. Another culprit was the right-click, which was a factor in four unintended error types.

There were three unintended errors with a wide distribution across pen participants. One common error was attempting to open a file or folder with a single-click instead of a double-click (Figure 17b). Ten of 12 pen participants attempted at least twice to open a file by single-clicking rather than using a conventional double-click. Two of these participants made this mistake four or more times. It seems that the affordance with the pen is to single-click objects, not double-click as one participant commented:

> I almost want this to be single-click which I would never use in normal windows, but with this thing it seems like it would make more sense to have a single-click. (P9-Pen1-TabletExperts 33:30)

Unintended movement when lifting the pen at the end of a drag action was another commonly occurring error (Figure 17d). This occurred most often when ma-

FIGURE 17. Unintended action errors.

Unintended Action Type	Tablet PC	Mouse	Frequent Pen Contexts
a) attempt to open file or folder with single-click instead of double-click	28 occurrences 10 participants 21% rate[a]	1 occurrences 1 participants 2% rate[a]	file folder navigation (all)
b) movement on drag release	27 occurrences 10 participants 2.8% rate[b]	1 occurrences 1 participants 0.3% rate[b]	dragging a drawing object's rotation or translation handle (22), or text selection (4)
c) dragging corner resize handle locks aspect ratio, unable to resize in one dimension only	9 occurrences 7 participants 0.9% rate[b]	1 occurrence 1 participant 0.3% rate[b]	PowerPoint image object (all)

[a]Occurrence rate calculated using 11 estimated file interactions (Figure 5). [b]Occurrence rate calculated using 79 estimated drags (Figure 6).

nipulating the rotation handle of a drawing object or when selecting text. Participants would position the rotation handle in the desired orientation, or drag the carat to select the desired text, but as they lifted the pen to end the drag, the handle or carat would unexpectedly shift.

A third unintended action error occurred when participants tried to resize a PowerPoint object in one dimension using the corner handle, but the adjustments with the corner handle automatically constrained the aspect ratio. This may have been better classified as an application usability problem.

There are two unintended actions with which we expected to find more problems: however, this did not turn out to be the case. We found only one occurrence of a premature end to a dragging action, which we expected to be more common because it requires maintaining a threshold pressure while moving (Ramos et al., 2004). Also, we intentionally did not disable the Tablet PC hardware buttons during the study, yet found only a single instance of an erroneous press.

Missed Click Errors

Due to how we classified missed click errors, 67 of 75 (91%) of these errors occurred when single-clicking (tapping with the pen). The chance of the system failing to recognize both clicks of an attempted double-click would result in a wrong click error instead.

All pen participants had at least one missed click, with four participants missing more than nine. The most common contexts were button (25%), menu (21%), and image (13%). The cause appears to be too little or too much force. Tapping too lightly is a symptom of a tentative tapping motion. We noted that when participants had trouble targeting small items (such as menu buttons, check boxes, and menu items) they sometimes hovered above the target to verify the cursor position, but the subsequent tap down motion was short, making it difficult to tap the display with enough force. Tapping too hard seemed to be a strategy used by some *Pen1-TabletExperts* participants as an (not always successful) error avoidance technique; see our discussion next.

Repeated Invocation, Hesitation, Inefficient Operation, and Other Errors

These errors had 184 occurrences in total, across all participants. There were only three occurrences of *other* errors, the remaining were repeated invocation, hesitation, and inefficient operation (Figure 18).

We noted 68 cases of obvious inefficient operation across all 12 pen participants (Figure 18a). More than half of these cases involved the scrollbar (31) or up-down (9). With the scrollbar, some participants chose to repeatedly click to page down for *very* long pages instead of dragging the scroll box. With the up-down, we noted several participants overshooting the desired value and having to backtrack. Of the 25 occurrences of inefficient operation with mouse participants, 14 involved the mouse scroll-wheel: Three of four mouse participants would scroll *very* long pages instead

FIGURE 18. Repeated invocation, hesitation, and inefficient operation errors.

Error Type	Tablet PC	Mouse	Frequent Pen Contexts
a) Inefficient Operation	68 occurrences 12 participants	25 occurrences 4 participants	scrollbar (31), up-down (4)
b) Hesitation	57 occurrences 11 participants	3 occurrences 3 participants	file folder navigation (all)
c) Repeated Invocation	27 occurrences 10 participants	*none*	PowerPoint image object (all)

of dragging the scroll box, resulting in a similar type of inefficient operation as pen participants repeatedly clicking page down.

We noted 57 cases of hesitation in all three pen groups but only three in the mouse group (Figure 18b). There seemed to be two main causes. One is due to the user visually confirming the position of the cursor, rather than trusting the pen tip, when selecting a small button or when opening a small drop-down target. The other is caused by many tooltips popping up and visually obscuring parts of the target. We discuss this type of "hover junk" in more detail next.

We identified 27 cases of repeated invocation across 10 pen participants (Figure 18c). Although a small number, it suggests some distrust when making selections. Eleven of these occurred when tapping objects to select them (images, charts, text boxes, and windows). Nine of these occurred when pressing a button or tab. Recall that a repeated invocation error is logged only if the first tap was successful. This is likely a symptom of subtle or hidden visual feedback, which we discuss next, and the fact that the extra cost of repeated invocation "just in case" is not that high (Sellen, Kurtenbach, & Buxton, 1992).

Error Recovery and Avoidance Techniques

We already discussed error avoidance through repeated invocation, but there were two other related observations with the experienced Tablet PC group. These participants seemed to recover from errors much more quickly, almost as though they expected a few errors to occur. For example, when they encountered a missed click error, there was no hesitation before repeating the selection action—one participant rapidly tapped a second time if the result did not instantly appear, which caused several repeated invocation errors. Sellen et al. (1992) also observed this type of behavior with experts using other devices. In contrast, participants in the *Pen3-ConventionalNovices* and *Pen2-ConventionalExperts* groups tended to pause for a moment if an error occurred; they seemed to be processing what had just happened before trying again.

A related strategy used by three *Pen1-TabletExperts* participants was to tap the pen very hard and very fast, almost as though they were using a very hard-leaded pencil on paper. When asked about this behavior, one participant commented that this helped avoid missed clicks (suggesting the participant may have felt that the digitizer was not very sensitive). However, we noted cases where a click was missed even with this

hard tap; in fact the speed and force seemed to be the cause. A better mental model for Tablet PC users is to think of the digitizer as being as sensitive as an ink pen on paper—this requires a medium speed (to allow the ink to momentarily flow) and a medium pressure (more than a felt tip marker, but less than very hard leaded pencil).

Interaction Error Context

By examining the most common widget or action contexts for interaction errors overall, we can get a sense of which are most problematic with pen input. Recall that because our study tasks follow a realistic scenario, the frequencies of interaction contexts are not balanced. For example, we expected 52 button interactions but only six text selection interactions (see Figure 5). Thus for relative comparison, a normalized error rate is appropriate.

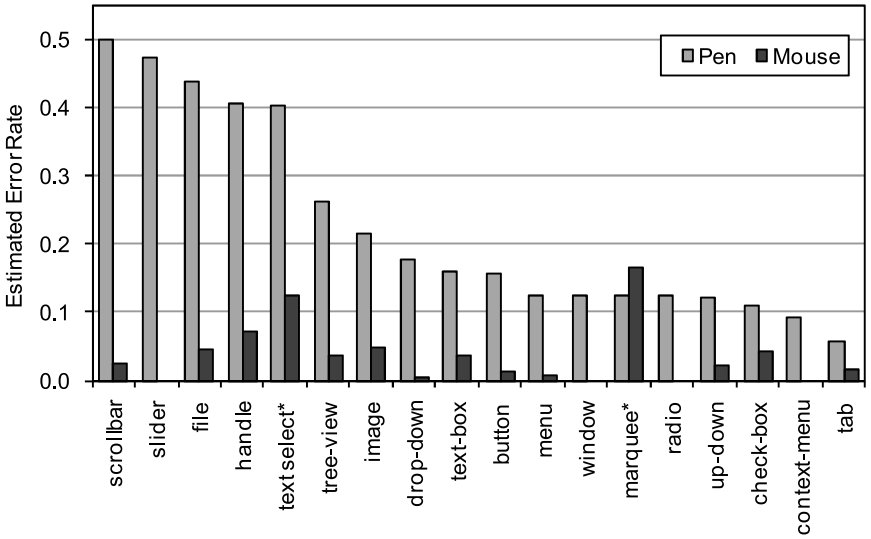
Ideally, we would normalize across the actual number of interactions per widget, but this has inherent difficulties: participants use different widget interaction strategies resulting in an unbalanced distribution, and we were not able to *automatically* log which widget was being used, much less know what the intended widget was (in the case of a target selection error). However, most interaction errors occur when an interaction is attempted, so we can normalize against the ideal number of interactions per widget and calculate an *estimated* interaction error rate. It is important to acknowledge that the ideal number of interactions is a minimum, reflecting an optimal widget manipulation strategy. Thus, although our estimated error rate may be useful for relative comparison between widgets, the actual rates may be somewhat exaggerated if a widget was used more times than expected or manipulated with an inefficient strategy.

If we accept this normalized error rate estimation, an ordered plot of widgets and actions reveals differences between mouse and pen (Figure 19). We include only widgets or actions with more than three expected interactions. The relative ordering is different between mouse and pen with the highest number of pen errors with the scrollbar and highest number of mouse errors with marquee selection.

The top error contexts for the pen range from complex widgets like sliders, tree-views, and drop-downs to simple ones like files, handles, and buttons. Note that three of five of the top contexts involve dragging exclusively: slider, handle, and text select.

The high error rate for the scrollbar is likely due to inefficient usage. Previously, we noted that many inefficient usage errors occurred when participants manipulated a scrollbar with a sequence of page-down taps rather than a single drag. Because in most cases our ideal interaction rate expected a single drag, this would create many errors relative to the normalizing factor. Later we examine the scrollbar in more detail and establish a more accurate error rate. We noted previously that the slider and handle both had small targets and the dragging action caused wrong click errors occasionally triggering right-clicks instead of drags. The high error rate for file objects is largely due to the high number of unintended action errors where participants attempted to open files with a single tap and a high number of wrong click errors when double taps where recognized as a single tap.

FIGURE 19. Estimated interaction error rate for widget and action contexts. *Note.* Error rate is calculated using the ideal number of interactions (see Figure 5). Only widgets or actions with more than 3 expected interactions are shown. Actions are denoted with an asterisk, all other contexts are widgets.



Note that although many errors with the text box were coded as application usability errors, it still produced a 16% estimated error rate. This is partly because even after participants avoided the application usability error of attempting to reposition a text box by dragging from the center, after they tapped it to select, the text box must be repositioned by dragging a very narrow 7 pixel (1.2 mm) border target: 39% of text box interaction errors were target selection.

Mouse error rates are consistently much lower, with the exception of the marquee. Several participants had a different mental model with marquee selection: They expected objects to be selected when they touch the marquee. This resulted in marquees missing desired objects for fear of touching an undesired object. There are also two mouse-specific action contexts not shown: scroll-wheel and keys. These contexts were not estimated by our script, so a normalized error rate cannot be computed. Mouse users often used the scroll-wheel to scroll very long documents, instead of simply dragging the scroll box, and often overshot the target position, both resulting in inefficient action errors. In addition, mouse users sometimes missed keys when using shortcuts.

5.3. Movements

In additional to errors, when reviewing the logs we could see differences in movement characteristics between mouse and pen. Using the pen seemed to require larger movements involving the arm, whereas the mouse had a more subtle movement style, often using only the wrist and fingers. Due to the number of target selection

and missed click errors with the pen, we also wished to investigate the movement characteristics of taps.

Overall Device Movement Amount

To measure movements, we first had to establish a threshold movement distance per data sample that would signal a moving or stationary frame. To find this threshold, we graphed Euclidian movement distance by time for different 10-s movement samples taken from four participants. Based on these graphs, we established a dead band cutoff at 0.25 mm of movement per frame (velocity of 7.5 mm/s).

With this threshold, we calculated the total amount of pen and mouse movement across all constrained tasks. Because participants completed the study in different times, we calculated a mean movement amount per minute for comparison and calculated the mean for each group (Figure 20). A one-way ANOVA found a significant main effect of group on device movement, $F(3, 12) = 19.488$, $p < .001$. The total movement for the mouse group was significantly shorter than all pen groups ($p < .001$, using the Bonferroni adjustment). We found that these statistics supported our observations: pen users moved more than 4.5 times farther than mouse users per minute.

Because we tracked the movements of the forearm in addition to the pen, we can examine the proportion of forearm movement to wrist and finger movements—with the assumption that if the pen or mouse moves without forearm movement, it must be the result of the wrist or fingers (Figure 21). We found that the pen had much greater proportion of combined wrist, finger, and forearm movements than the mouse (67–72% compared to 36%). A one-way ANOVA found a significant main effect of group on combined wrist, finger, and forearm movements, $F(3, 12) = 71.926$, $p < .001$, with less movement in mouse group compared to all pen groups ($p < .001$, using the Bonferroni adjustment).

This was most evident when pen participants selected items in the top or left side of the display. To do this they always had to move both their arm and hand, where

FIGURE 20. Average pen or mouse movement distance per minute for all constrained tasks.

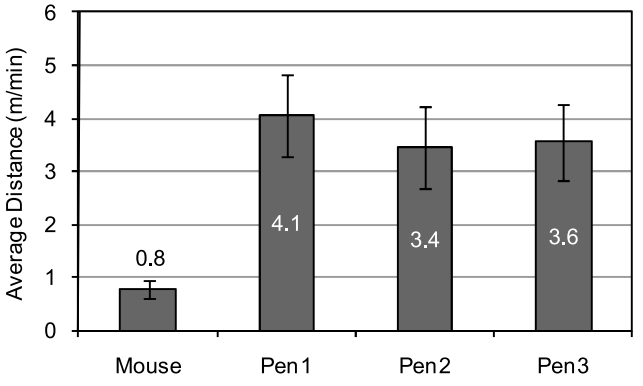
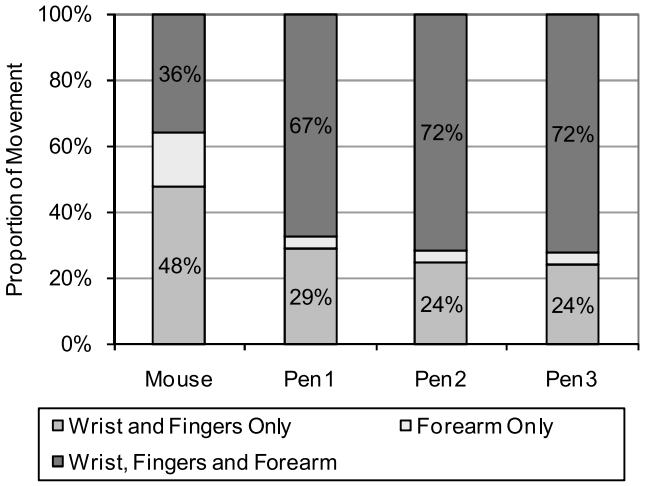


FIGURE 21. Proportion of movements greater than 0.25 mm per frame (velocity greater than 7.5 mm/s). ***Note.*** **Euclidean distance in 3D space.**



mouse users were often able to use only their wrist and fingers. The reason is partly because of pointer acceleration with the mouse, but also because mouse participants had a more central *home position,* which we discuss later in the Posture section. The larger *forearm only* portion with mouse group is likely attributed to the fact that unlike the pen, the mouse is not supported by the hand, and the forearm can more easily move independently.

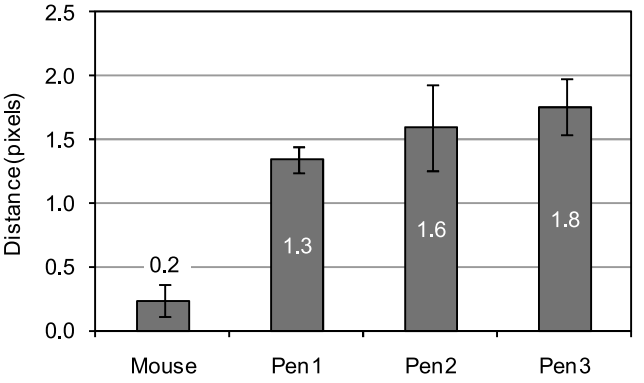
Pen Tapping Stability

When selecting small objects such as handles and buttons, most GUIs deem a selection successful only when both pen-down and pen-up events occur within the target region of the widget (Ahlstroem, Alexandrowicz, & Hitz, 2006). Our observations suggest that this is a problem when using the pen. To disambiguate single-clicks from drags, we included a pen-down and pen-up event pair only if the pen-up event occurred within 200 ms. Using this threshold, we found that the mean distance between a pen-down and pen-up event was 1.3 to 1.8 pixels. This is similar to results reported by Ren and Moriya (2000) and discussed by Zuberec (2000, Section 5.12). Although a relatively small movement, this can make a difference when selecting near the edge of objects. In contrast, the mean distance between mouse down and up events was only 0.2 pixels (see Figure 22).

We observed that after participants recognized that they were suffering from frequent selection errors, they began to visually confirm the physical pen position by watching the small dot cursor. One participant commented,

> It is a little finicky, eh. It's like I'm looking for the cursor instead of just assuming that I'm clicking on the right place, so it is a little bit slower than normal. (P9-Pen1-TabletExperts 27:39)

FIGURE 22. Mean Euclidian distance between down and up click (for down and up click pairs separated by less than 200 ms).



Participants also encountered problems with visual parallax, which made selecting targets difficult at the extreme top of the tablet, supporting the arguments of Ward and Phillips (1987) and Ramos et al. (2007):

> It seems that when I get to the a ... um ... portions like the menu, ... the dot doesn't seem to be in place of exactly where I'm pressing it. It seems to be rocking a little bit to the right where I need to be more precise. (P16-Pen3-ConventionalNovices 19:59)

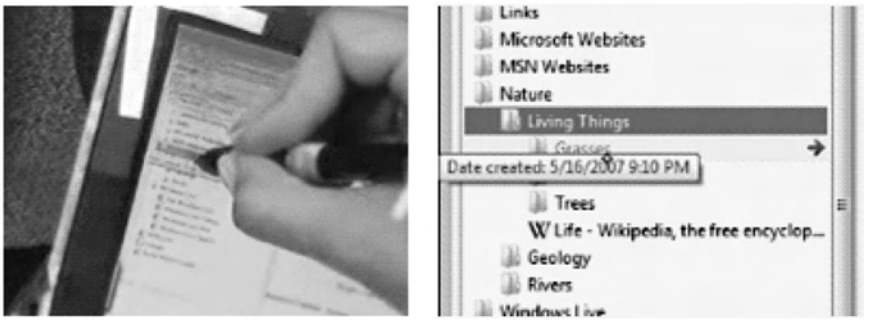
One of the purported global pen input improvements with Windows Vista is visual "ripple" feedback to confirm all single and double taps. However, not one participant commented on the ripple feedback or appeared to use it as a visual confirmation to identify missed clicks. In fact, this additional visual feedback seems to only add to the general "visual din" that seems more prevalent with the pen compared to mouse input.

Pen users seemed to be more adversely affected by tooltips and other information triggered by hover: 20% of pen hesitation errors occurred when tooltips were popping up near a desired target. When compensating for digitizer tracking errors by watching the cursor, participants tended to hover more often before making a selection, which in turn triggered tooltips more often. At times the area below the participant's pen tip seemed to become filled with this type of obtrusive hover visualization, which we nicknamed "hover junk" (Figure 23).

> Why doesn't this go away?" [referring to a tooltip blocking the favorites tree-view] (P13-Pen3-ConventionalNovices 33:05)

When there was a lot of tooltip hover visualizations, participants appeared to limit their target selection to the area outside the tooltip, which decreased the effective target size. We did not see this behavior with the mouse. The nature of direct interaction seems to change how people perceive the depth ordering of device pointer, target, and

FIGURE 23. Obtrusive tooltip hover visualizations: "hover junk", *P15-Pen3-Conventional-Novices*, task 11.



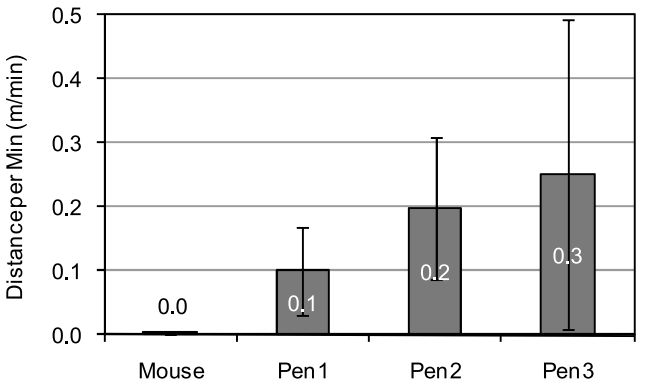
tooltips. With the pen, the stacking order from the topmost object down becomes: *pointer* → *tooltip* → *target*, compared to *tooltip* → *pointer* → *object* with mouse users. This is supported by comments from Hinckley et al. (2007) regarding the design of InkSeine. They tried using tooltips to explain gestures but found they were revealed too late, they can be mistaken for buttons, and the user's hand can occlude them altogether.

Tablet Movement

Unlike Fitzmaurice et al. (1999), whose participants used an indirect pen tablet on a desk, our participants held the Tablet PC on their lap. We expected that the lap would create a more malleable support surface—perhaps the legs would assist in orienting and tilting the tablet—so there may be more tablet movement.

Using the same 0.25 mm deadband threshold, we calculated the mean amount of device movement per minute (Figure 24). Not surprisingly, we saw almost no movement in the mouse condition in spite of their ability to adjust the display angle or move the entire laptop on the desk. The amount of tablet movement in the pen condition was also small compared to the amount of movement of the pen itself

FIGURE 24. Tablet or laptop movement per minute for all constrained tasks.



(Figure 20), less than 7%. There is no significant difference in pen groups due to high variance: some participants repositioned the tablet more than others.

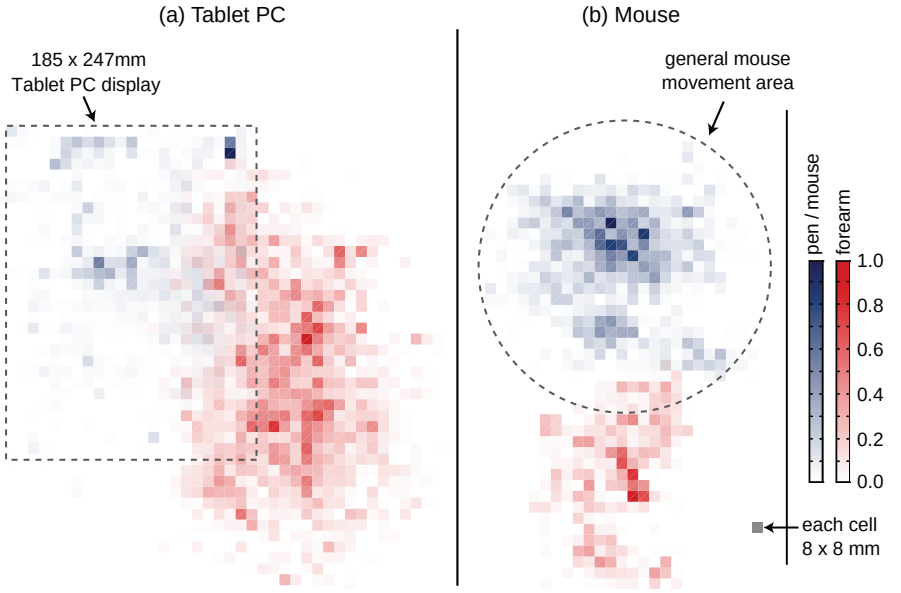
5.4. Posture

Related to movements are the types of postures that participants adopted to rest between remote target interactions and to avoid occlusion.

Home Position

We observed that at the beginning and end of a sequence of interactions, pen participants tended to rest their hand near the lower right of the tablet. A heat map plot of rest positions for forearm and pen illustrates this observation (Figure 25a). Pen rest positions approximate the distribution of clicks and taps (see also Figure 15). Forearm rest positions are concentrated near the bottom right of the tablet and do not follow the same distribution as the pen. In contrast, the mouse distributions are more compact with peaks near their center of mass suggesting a typical rest point in the centre of the interaction space (Figure 25b). The spread of movement follows from a greater overall movement with the pen (Figure 20).

FIGURE 25. Heat map plot of forearm and pen/mouse rest positions across all participants for (a) Tablet PC participants; (b) Mouse participants. *Note.* Rest positions are defined as movement less than 0.25 mm per frame (velocity less than 7.5 mm/s). Tablet positions are relative to the plane of the display, Mouse positions are relative to the plane of the laptop keyboard. The dashed areas represent the tablet display and general mouse movement area on the desk where all pen or mouse positions lie.



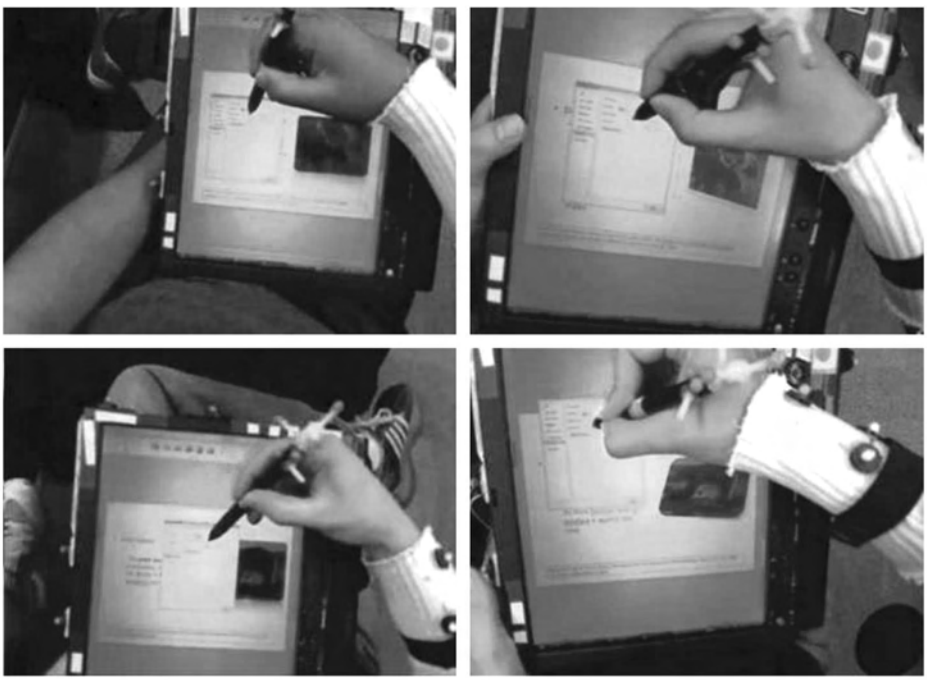
Occlusion Contortion

We observed all pen users adopting a "hook posture" during tasks in which they had to adjust a parameter while monitoring its effect on another object—for example, adjusting color values in task 17 or image brightness and contrast levels in task 25 (Figure 26). The shape of the hook almost always arched away from the participant, although Participant 12 began with an inverted hook and then switched to the typical style midway. We could also see this hook posture when participants were selecting text, although it was more subtle. This type of posture was also observed by Inkpen et al. (2006) when left-handed users were accessing a right-handed scrollbar.

Adopting such a hook posture can affect accuracy and increase fatigue. Because it forces the user to lift their wrist above the display, it reduces pen stability. One participant commented that they found it tiring to keep the image unobstructed when adjusting the brightness and contrast.

In Task 25, where the hook posture was most obvious, the participant had the option to move the image adjustment window below the image, which would have eliminated the need for contortion. However, we observed only one pen participant do this (P5-*Pen1-TabletExperts*), which suggests that the overhead in manually adjusting a dialog position to eliminate occlusion may be considered too high. Yet previous work found that adjusting the location and orientation of paper to reduce occlusion is instinctive (Fitzmaurice et al., 1999).

FIGURE 26. Examples of occlusion contortion: The "hook posture."



6. INTERACTIONS OF INTEREST

Based on our analysis of errors, movement, and posture we identified widgets and actions for more detailed analysis. We selected widgets and actions that are used frequently (button), highly error prone (scrollbar), presented interesting movements (text selection), or highlighted differences between the pen and mouse (drawing, handwriting, the Office MiniBar, and keyboard use).

6.1. Button

The single, isolated button is one of the simplest widgets and also the most ubiquitous. We expected participants to use a button (not including buttons that were part of other larger widgets) 52 times during our study, which constitutes 21% of all widget occurrences. Although simple, we found an interaction error rate of 16% for pen participants compared to less than 1.5% for mouse (Figure 5 using the expected number of button interactions as a normalizing factor). Fifty-five percent of these errors were target selection errors, 17% missed clicks, 11% hesitation, and 6% repeated invocation. We already discussed problems with target selection with pen taps, so in this section we concentrate on other errors and distinctive usage patterns.

Repeated invocation errors occurred when the user missed the visual feedback of the button press and pressed it a second time. This was most evident when the button was small and the resulting action delayed or subtle. There were three commonly occurring cases in our scenario: when opening an application using the quick launch bar in the lower left, pressing the *save file* button in the upper left, and closing an application by pressing on the "x" at the top right. Participants did not appear to see, or perhaps did not trust, the standard button invocation feedback or the new visual feedback used for taps in Windows Vista. Depending on the timing of the second press, the state of the application could simply ignore the second click, save the file a second time, or introduce more substantial errors like opening a second application.

Missing a click on a button could result in more severe mistakes. When saving a file, the confirmation of a successful save was conveyed by a message and progress bar located in the bottom right. Because the Save button is located at the top left, this meant that the participant's arm was almost always blocking this confirmation, making it easy to miss. We observed three participants who thought they saved their file when a missed click or target selection error prevented the save action.

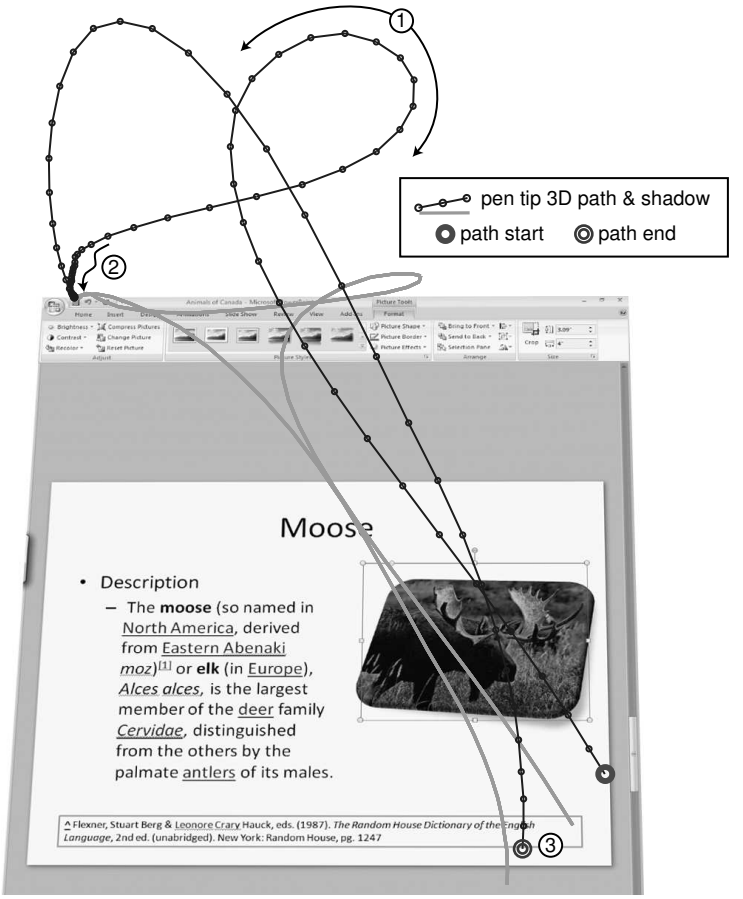
Sometimes the location of buttons (and other widgets) prevented participants from realizing the current application state did not match their understanding. For example, in Task 6, we saw four pen users go through the steps of selecting the font size and style for a text box, when in fact they did not have the text box correctly selected and missed applying these changes to some or all of the text. We did not see this with any of the mouse users. Because these controls are most often accessed in the menu at the top of the display, this is likely because the formatting preview is being occluded by the arm. After making this mistake several times, one participant asked,

> "How do I know if that's bold? Like I keep pressing the bold button." (P16-Pen3-ConventionalNovices 18:27)

Although the Bold button was visually indicating the bold state, they failed to realize the text they wished to make bold was not selected.

While reviewing the logs of button presses, we could occasionally see a distinctive motion that interrupted a smooth trajectory toward the target creating a hesitation error. As the user moved the pen toward the button, they would sometimes deviate away to reestablish visual confirmation of the location, and then complete the movement (Figure 27, Note ①). In some cases this was a subtle deviation, and other times it could be quite pronounced as the pen actually circled back before continuing. We saw this happen most often when the button was located in the upper left corner, and the deviation was most apparent with our novice group.

FIGURE 27. Pen tip trajectory when selecting the save button in the upper left corner. *Note.* ① movement deviation to reduce occlusion and sight target; ② corrective movements near target; ③ return to home position, (P15, *Pen3-ConventionalNovices*, task 26).



6.2. Scrollbar

In our study, we found that pen participants made an average of 10 errors while using a scrollbar. With 20 expected scrollbar interactions, our estimated scrollbar error rate is 50%. However, we suspected this error rate is inflated due to participants using scrollbars more often than expected (e.g., repeated invocation due to previous task errors), and participants using an inefficient and error-prone scrollbar usage strategy (e.g., multiple paging taps instead of a single drag interaction). For a more detailed examination, we coded scrolling interactions in Tasks 2, 6, 21, 23, 25, and 27. According to our script, we expected 15 scrolling interactions within these six tasks breaking down into four types: four when using drop-downs, four during web browsing, two during file browsing, and five while paging slides. We found an average of 14.8 scrolling interactions ($SD = 0.6$) which suggests that in these tasks, our estimated number of scrollbar interactions was reasonable.

All pen participants used different scrollbar interaction strategies, except for two *Pen2-ConventionalExperts* participants: One of these participants always clicked the *paging region*, and the other always dragged the *scroll box* (see Figure 28 for scrollbar parts). When participants changed their strategy, they often clicked in the paging region for short scrolls and dragged the scroll box for long scrolls, but this was not always the case. We observed only four cases where participants used the scrollbar button: One participant used it to increment down, and three participants held it down to scroll continuously as part a mixed strategy. Overall, we counted 91 occurrences of dragging and 54 occurrences of paging.

There were 17 occurrences of pen participants using a mixed scrolling strategy—where combinations of scrollbar manipulation techniques are used together for one scroll. Six participants used such a mixed strategy two or more times, and all did so exclusively for long scrolls in drop-downs or web browsing. Most often a scroll box drag was followed by one or more paging region clicks, or vice versa.

Regarding errors, we found two patterns. First, for scrollbar strategy, we found error rates of 16% for paging, 9% for dragging, and 44% for mixed strategies (rate of strategy occurrences with at least one error). A mixed strategy often caused multiple errors, with an average of 1.6 errors per occurrence. Participants often moved to a mixed strategy after an error occurred—for example, if repeated paging was inefficient or resulted in errors, they would switch to dragging. Pure paging and dragging strategies had 0.5 and 0.2 errors per occurrence. Many errors were target selection related, suggesting that the repetitive nature of clicking in the paging region creates more opportunities for error.

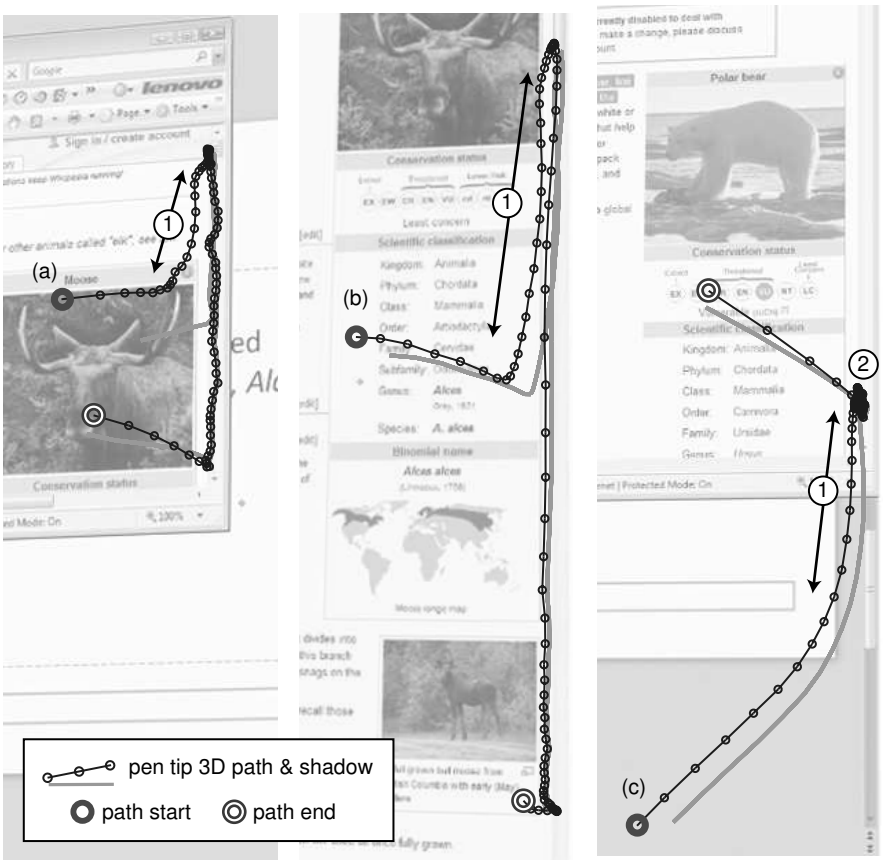
FIGURE 28. Scrollbar parts.

Second, regarding errors and location, we found that 77% of scrollbar errors occurred in the 100 right-most pixels of the display (i.e., when the scrollbar was located at the extreme right), but only 61 % of scrollbar interactions were in the 100 right-most pixels. Although not dramatic, this pattern is agreement with our observation of error locations (Figure 15).

We also noted a characteristic trajectory when acquiring a scrollbar with the pen, which we call the "ramp." When acquiring a scroll bar to the right of the hand, we observed several users moving down, to the right, and then up to acquire the scroll box (Figure 29). Based on the user view video log, we could see that much of the scrollbar was occluded and that this movement pattern was necessary to reveal the target before acquiring it.

The mouse scroll-wheel can act as a hardware alternative to the scrollbar widget, and we observed three of four mouse participants use the scroll-wheel for short scrolls.

FIGURE 29. Pen tip trajectories during scrollbar interaction. *Note.* (a) short drag scroll, centre of display, P16, *Pen3-ConventionalNovices*, task 21; (b) long drag scroll, edge of display, P7, *Pen1-TabletExperts*, task 21; (c) paging scroll, P9, *Pen2-ConventionalExperts*, task 23. In each case, ① denotes the characteristic "ramp" movement, ② denotes repetitive paging motion segment.



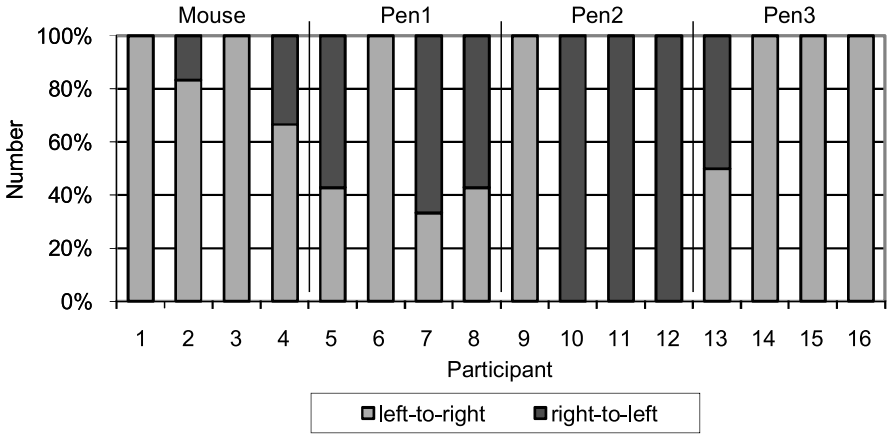
These three participants never clicked to page up or down during short scrolls. In fact, all mouse participants used the scroll-wheel at least once for longer scrolls, but we observed each of them also abandoning it at least once—and continuing the scroll interaction by dragging the scroll box. The scroll-wheel does not always appear to have an advantage over the scrollbar widget, corroborating evidence from Zhai, Smith, and Selker (1997). However, this may be due to the scrolling distance and standard scroll-wheel acceleration function (Hinckley, Cutrell, Bathiche, & Muss, 2002). In fact, all of our mouse participants encountered one or more errors with the scroll-wheel, but there were two mouse errors with the scrollbar widget.

6.3. Text Selection

There are six expected text selections in the script, in Tasks 12, 13, 21, and 23. Three involve selecting a sentence in a web page and three a bullet. We coded all text selections performed by participants in these tasks and found a mean number of 6.3 ($SD = 0.6$). The slightly higher number is due to errors requiring participants to repeat the selection action. We found high error rates for text selection with the pen. At 40%, this is three times the mouse error rate of 13%. Text selection errors were either target selection or an unintended action such as accidentally triggering a hyperlink. Most of these errors seem to be related to the style of movement.

An immediately obvious characteristic of text selection is the direction of movement, from right-to-left or left-to-right. Across pen participants, we found 43 left-to-right selections and 36 right-to-left (Figure 30). Given that our participants are right-handed, a right-to-left selection should in theory have an advantage, because the end target is not occluded by the hand. Instead, we found that all of our expert pen users performed left-to-right selection two or more times, with one expert participant (P6) only selecting left-to-right. Five pen participants exclusively performed left-to-right

FIGURE 30. Proportion of left-to-right and right-to-left text selection directions for each participant.



text selections, but three did exclusively use right-to-left selections. Surprisingly, the latter were all in the *Pen2-ConventionalExperts* group, not the *Pen1-TabletExperts* group as one might expect.

The insistence of a left-to-right motion in spite of occlusion is likely due to the reading direction in Western languages, which creates a habitual left-to-right selection direction. Indeed, we found that mouse participants most often used a left-to-right direction, with two participants doing this exclusively. However, even mouse users performed the occasional right-to-left selection suggesting that there are cases when this is more advantageous even in the absence of occlusion. One participant states,

> People write left to right, not right to left so my hand covers up where they're going. (P14-Pen3-ConventionalNovices 38:49)

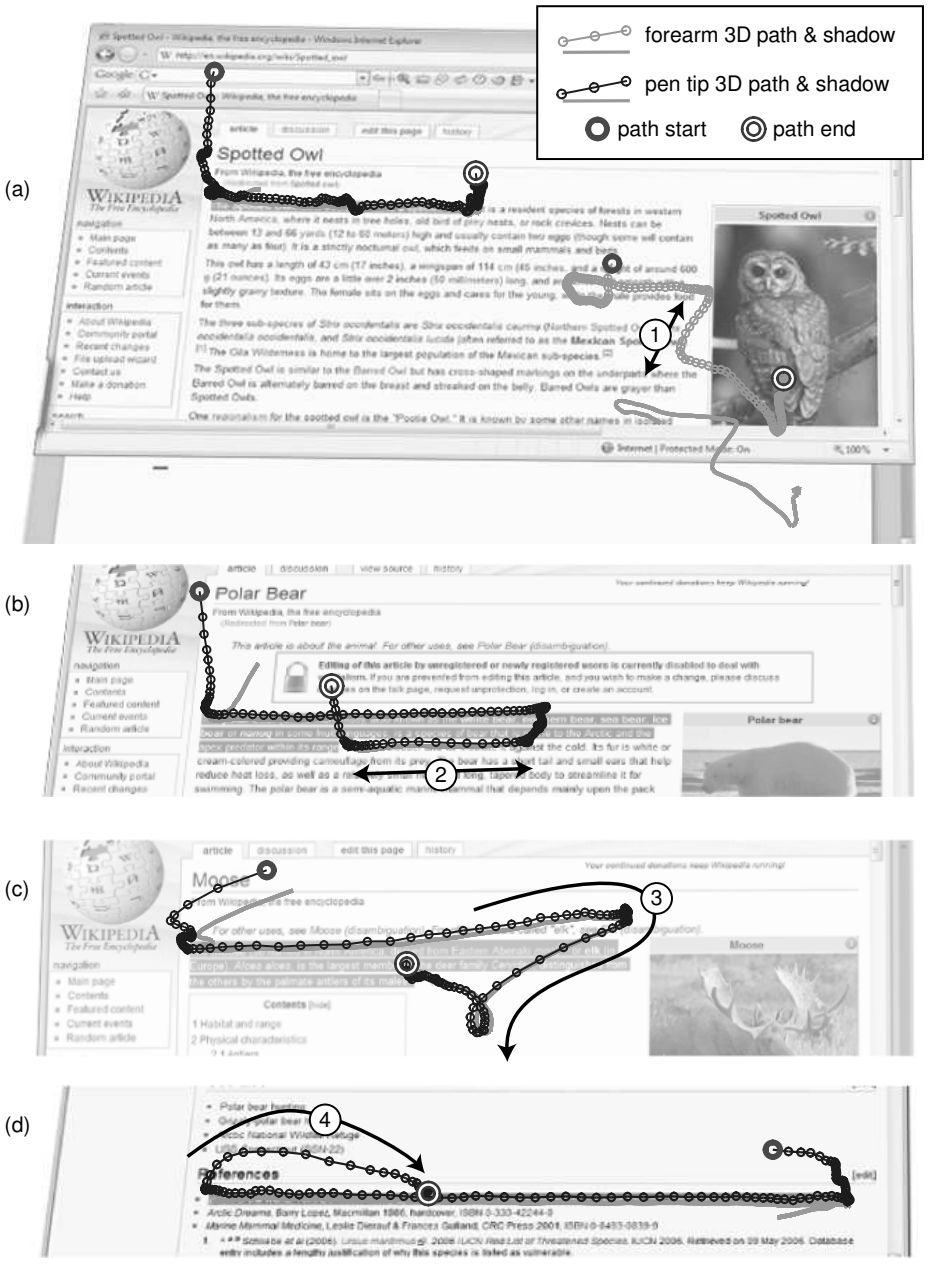
We observed three characteristic pen trajectory patterns which suggest problems with left-to-right selection and occlusion. We observed expert pen users intentionally moving the pen well beyond the bounds of the desired text during a left-to-right selection movement (Figure 31c). The most likely reason for this deviation is that it moved their hand out of the way so that they could see the end text location target. Another related movement is backtracking (Figure 31b), which more often occurred with novice participants. Here, the selection overshoots the end target and back tracks to the correct end target. This appears to be more by accident but may be the behavior that leads to the intentional deviation movement we saw with expert users. Another, sometimes more subtle, behavior is a "glimpse": a quick wrist roll downward to momentarily reveal the occluded area above (Figure 31a).

We also noted a characteristic trajectory when participants invoked the context menu for copying with a right-click. We observed many pen participants routinely introducing an extra movement to invoke it near the centre of the selection, rather than in the immediate area (Figure 31d). Because the right-click has to be invoked on the selection itself, this may be to minimize the possibility of missing the selection when opening the context menu. However, this extra movement was most often observed with right-to-left selection. This may be a symptom of participants needing to visually verify the selection before copying by moving their hand.

Common errors with text selection were small target selection errors such as missing the first character, clicking instead of dragging and triggering a hyperlink, or an unintended change of the selection when releasing. Although the first two are related to precision with the pen, the latter is a symptom of stability. As the pen is lifted from the display, a small movement causes the carat to shift slightly, which can be as subtle as dropping the final character, or if it moves down, it can select a whole additional line. We noticed this happening often when releasing the handle, another case of precise dragging. One participant commented,

> When I'm selecting text I'm accidentally going to the next line when I'm lifting up. (P7-Pen1-TabletExperts 16:40)

FIGURE 31. Pen tip (and selected wrist trajectories) during text selection: (a) left-to-right selection with forearm glimpse at ①, P15, *Pen3-ConventionalNovices*, task 12; (b) left-to-right selection with backtrack at ②, P9, *Pen2-ConventionalExperts*, task 23; (c) left-to-right selection with deviation at ③, P6, *Pen1-TabletExperts*, task 21; (d) right-to-left selection with central right-click invocation at ④, P11, *Pen2-ConventionalExperts*, task 23.



6.4. Writing and Drawing

Although we avoided formal text entry and handwriting recognition, we did include a small sample of freehand handwriting and drawing. In theory, these are tasks to which the pen is better suited. Tasks 39 and 41 asked participants to make ink annotations on an Excel chart (see Figure 3e). In Task 39 they traced one of the chart lines using the yellow highlighter as a very simple drawing exercise. In Task 41 they wrote "effects of fur trade" and drew two arrows pointing to low points on the highlighted line. In the poststudy interview, many pen participants said that drawing and writing were the easiest tasks. After finishing Tasks 39 and 41, one participant commented,

> You know this is the part that is so fun to work with, you know, using a tablet, but all the previous things are so painful to use. I mean, it's just like a mixture of things ... (P8-Pen1-TabletExperts 38:01)

Handwriting

We expected a large difference in the quality of mouse and pen writing, but aside from pen writing appearing smaller and smoother, this was not the case (Figure 32). We did see some indication of differences in mean times, with pen and mouse participants taking an average of 27.3 and 47.3 s, respectively ($SD = 3.5$ and 26.2). In terms of style, all mouse handwriting has a horizontal baseline, whereas four of the pen participants wrote on an angle. This supports Fitzmaurice et al.'s (1999) work on workspace orientation with pen input.

FIGURE 32. Handwriting examples for (a) mouse and (b) pen (approx. 70% actual size).

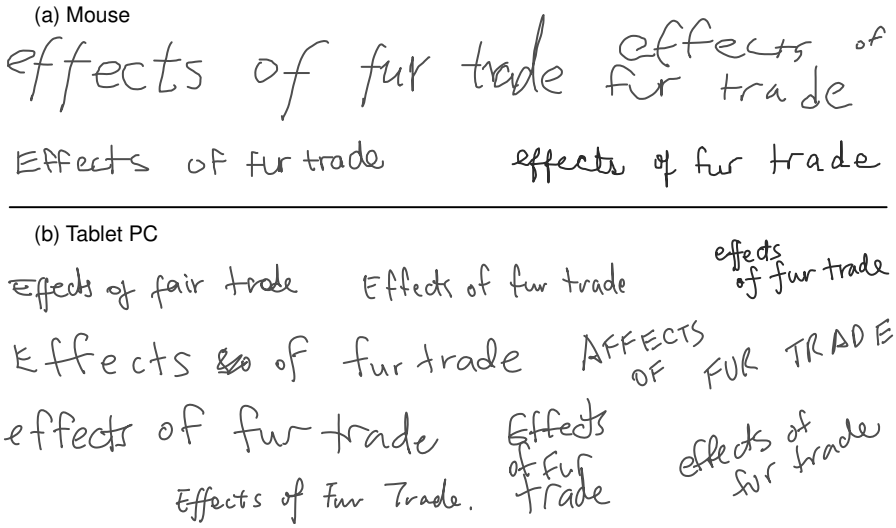
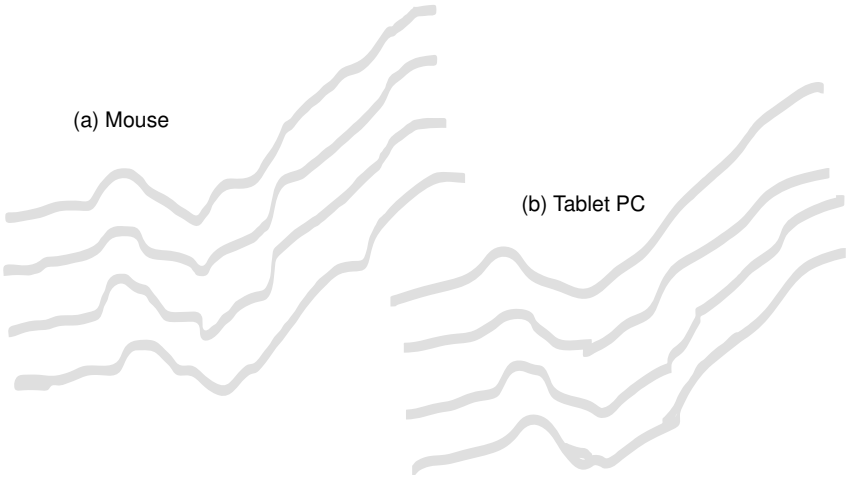


FIGURE 33. Tracing examples (approx. 70% actual size).



Tracing

When comparing participant's highlighter paths in Task 39, we could see little difference (Figure 33). Tracing appears slightly smoother but not necessarily more accurate. There also appears to be no noticeable difference in task time, with pen and mouse participants taking an average of 15.8 and 13.5 s, respectively ($SD = 6.4$ and 2). Half the mouse participants traced from right-to-left, as opposed to left-to-right. However, only three of 12 pen participants traced from right-to-left. As previously explained, with the pen, tracing right-to-left has distinct advantage for a right-handed person because it minimizes pen occlusion. Across all participants, all except one (a *Pen2ConventionalExperts* participant) traced the entire line with one drag motion.

6.5. Office MiniBar

PowerPoint has an alternate method of selecting frequently used formatting options, a floating tool palette called the MiniBar, which appears when selecting an object that can be formatted, like a text box (Harris, 2005). It is invoked by moving the pointer toward an initially "ghosted" version of the MiniBar; moving the pointer away makes it disappear. The behavior has some similarities to Forlines et al.'s (2006) Trailing Widget, except that the MiniBar remains at a fixed position.

In theory, the MiniBar should also be well suited to pen input because it eliminates reach. However, in practice it was difficult for some pen users to invoke. The more erratic movements of the pen often resulted in its almost immediate disappearance, preventing several participants from even noticing it and making it difficult for others to understand how to reliably invoke it. We observed one of our expert Tablet PC users *try* to use the MiniBar more than five times—the user finally aborted and returned to using the conventional application toolbar.

FIGURE 34. Occlusion of text-box preview when using the MiniBar floating palette (P12-*Pen2-ConventionalExperts*, task 6).

The other problem is that the location of the MiniBar is such that when using it, the object receiving the formatting is almost always occluded by the hand (Figure 34). We observed participants select multiple formatting options without realizing that the destination text was not selected properly; hand occlusion prevented them from noticing that the text formatting was not changing during the operation. A lesson here is that as widgets become more specialized they may not be suitable for all input devices, at least without some parameter tuning.

6.6. Keyboard Usage

Although we gave no direct instructions regarding keyboard usage for the mouse group, all participants automatically reached for the keyboard for common key shortcuts like Ctrl-Z, Ctrl-C, Ctrl-V, Ctrl-Tab, and often to enter numerical quantities. In Task 6, two mouse participants (P1 and P3) accelerated their drop-down selection by typing the first character. However, they each did this only a single time, in spite of this task requiring them to access the same choice, in the same drop-down, four times. Yet we saw that keyboard use can also lead to errors. For example, P1 accidentally hit a function key instead of closing a dialog with the Esc key; this abruptly opened a large help window and broke the rhythm of the task as they paused to understand what happened before closing it and continuing.

Three pen participants explicitly commented on the lack of accelerator keys when using the pen, with comments like

> "Where's CTRL-Z?" (while making key press action with left hand), then again later, "I can't tell you how much I wish I could use a keyboard …" (P9-Pen2-ConventionalExperts, 24:43 and 29:50)

However, not one pen participant commented on what the Tablet PC hardware keys were for, or if they could use them. Yet, we suspect they were conscious of their existence, as only one participant pressed one of these keys by accident.

7. DISCUSSION

The goal of our study was to observe direct pen input in a realistic GUI task involving representative widgets and actions. In the previous analysis we presented

findings for various aspects: time, errors, movements, posture, and visualization, as well as an examination of specific widgets and actions. Although we did not find a significant difference in overall completion times between mouse users and experienced Tablet PC users, this null-result does not mean that direct pen input is equivalent to mouse input. We found that pen participants made more errors, performed inefficient movements, and expressed frustration. For example, widget error rates had a different relative ordering between mouse and pen; the highest number of pen errors were with the scrollbar and highest number of mouse errors with text selection. The top error contexts for the pen range from complex widgets like scrollbars, drop-downs, and tree-views to simple ones like buttons and handles.

7.1. Overarching Problems With Direct Pen Input

When examined as a whole, our quantitative and qualitative observations reveal overarching problems with direct pen input: poor *precision* when pointing or tapping, problems caused by hand *occlusion*; instability and fatigue due to *ergonomics*; *cognitive differences* between pen and mouse usage; and frustration due to *limited input* capabilities. We believe these to be the primary causes of nontext errors and contribute to user frustration when using a pen with a conventional GUI.

Precision

Selecting objects by tapping the pen tip on the display serves the same purpose as pushing a button on a mouse, but the two actions are radically different. The most obvious difference is that this allows only one type of "click" unlike pressing different buttons on a mouse. To get around this issue, right-click and single-click are disambiguated with a time delay, overloading the tapping action to represent more than one action. Although participants did better than we expected, we found that the pen group were not always able to invoke a right-click reliably, and either unintentionally single-clicked, or simply missed the click. A related problem occurred with drag-able widgets like scrollbars and sliders: when performing a slow, precise drag, users could unintentionally invoke a right-click. We found these problems affected expert pen users as well as novices.

The second difference between mouse and pen selection may not be as immediately obvious: Tapping with the pen simultaneously specifies the selection action *and* position, unlike clicking with the mouse where the button press and mouse movement are designed such that they do not interfere with each other. The higher number of target selection errors with the pen compared to the mouse suggests that this extra coordination is a problem. Our findings also reveal subtle selection and pointing coordination issues: unintended action errors due to movement when releasing a drag-able widget, such as the handle, were nonexistent with the mouse but affected 10 of 12 pen participants; on average, the distance between pen-down and pen-up-down events were six to nine times greater than the mouse; and there were problems with pen double-clicks, either missed altogether, or interpreted as two single-clicks.

We also found problems with missed taps when the tapping motion was too hard or too soft. This could be an issue with hardware sensitivity, but given our other observations, it may also be a factor of the tapping motion. We found that some participants did not notice when they missed a click, leading to potentially serious errors. It seems that the tactile feedback from tapping the pen tip is not enough, especially when compared to the sensation of pressing and releasing the microswitch on a mouse button. Onscreen visual feedback for pen taps was introduced with Windows Vista, but no participants seemed to make use of this.

Surprisingly, we did not observe a large difference in the quality of pen writing and tracing compared to mouse input: Pen handwriting appeared smaller and smoother, and pen tracing appeared slightly smoother.

In the same way that hardware sensitivity is likely contributing to the number of missed clicks, other hardware problems such as lag and parallax (Ward & Phillips, 1987) also affect performance. When using a pen, any lag or parallax seems to have an amplified effect, as the visual feedback and input are coincident. When users become aware of these hardware problems, they begin to focus on the cursor, rather than trusting the physical position of the pen. This may reduce errors, but the time taken to process visual feedback will hurt performance.

Occlusion

Occlusion from the pen tip, hand, or forearm can make it difficult to locate a target, verify the success of an action, or monitor real-time feedback, which may lead to errors, inefficient movements, and fatigue.

We observed participants missing status updates and visual feedback because of occlusion. This can lead to serious errors, such as not saving a file, and frustrating moments when users assumed the system was in a certain state but it was not. For example, we saw more than one case of a wasted interaction in the top portion of the display to adjust formatting because the object to be formatted had been unintentionally deselected. This occurred in spite of the application previewing the formatting on the object itself: unfortunately, when the destination object is occluded and the user assumes the system is in a desired state (the object is selected), the previews do not help and the error is not prevented.

To reduce the effect of occlusion, we observed users adopting unnatural postures and making inefficient movements. For example, when adjusting a parameter that simultaneously requires visual feedback, we noticed participants changing the posture of their hand rather than adjusting the location of the parameter window. This "hook" posture did enable them to monitor an area of the display that would otherwise be occluded, but it also appears to be less stable and uncomfortable.

We also found that occlusion can lead to inefficient movements such as *glimpses*, *backtracks*, and *ramps*. Glimpses and backtracks tend to occur during dragging operations where the kinaesthetic quasi-mode (Sellen et al., 1992) enabled by the pen pressed against the display forces users to perform a visual search for occluded targets without lifting their hand. To work around this limitation, we observed

expert users intentionally deviating from the known selection area while drag selecting and novice users backtracking after they accidentally passed the intended target. Our tendency to drag and select from left-to-right, to match how text is read in Western languages, seems to make glimpse and backtrack movements more common.

The ramp is a characteristic movement that adjusts the movement path to reveal a greater portion of the intended target area. When the hand is in midmovement, it can occlude the general visual search area and require a deviation to visually scan a larger space. We observed ramp movements most often when locating the scrollbar widget on the extreme right, but also when moving to other targets, sometimes with helical paths, when the targets were located at the upper left.

Finally, pen users tend to adopt a consistent home position to provide an overview of the display when not currently interacting. Participants would return their hand to this position between tasks and even after command phrases, such as waiting for a dialog to appear after pressing a toolbar button. For right-handed pen users, the home position is near the lower right corner, just beyond the display.

Ergonomics

Although the display of a Tablet PC is relatively small, there still appear to be ergonomic issues when reaching targets near the top or at extreme edges. We found that pen movements covered a greater distance with more limb coordination compared to the mouse. Not only can this lead to more repetitive strain disorders and fatigue, but studies have shown that coordinated limb movement lead to decreased performance and accuracy (Balakrishnan & MacKenzie, 1997). In support of this, we found a different distribution of target selection error rate compared to the location of all taps/clicks: More errors seem to occur in the mid-upper-left and right-side. However, as we previously discussed, there may be an influence of target size that we did not control for.

Possible explanations for the extra distance covered by the pen compared to the mouse include making deviating movements to reduce occlusion, because the pen tip moves more frequently in three dimensions to produce taps, and to arc above the display when travelling between distant targets. However, other contributing factors are the unavailability of any control display gain manipulation with the pen and the tendency for pen users to return to a home position between tasks. By frequently returning to this home position, there are more round-trip movements compared to mouse participants who simply rest at the location of the last interaction. Although the home position allows an overview of the display due to occlusion, it may also serve to rest the arm muscles to avoid fatigue and eliminate spurious errors that could occur if a relaxed hand accidently rests the pen tip on the display.

Another issue with reach may be the size and weight of the tablet. Not surprisingly, we found that tablet users moved the display more than mouse users, but they only moved it less than 7% of the distance moved by the pen (in spite of the tablet resting on their lap, which we expected would make it easier to move and tilt

using the legs and body). Further support can be seen in the characteristic slant of some tablet participants' written text—these people elected to write in a direction that was most comfortable for their hand, regardless of the position of the tablet. This suggests that the pen is more often moved to the location on the display, rather than the nondominant hand bringing the display to the pen to set the context. Note that the latter has been shown to be a more natural movement with pen and paper or an indirect tablet on a desk (Fitzmaurice et al., 1999). Our speculation is that the problem is likely due to the size and weight of the device.

Cognitive Differences

Cognitive differences between the pen and mouse are difficult to measure, but our observations suggest some trends. Pen users prefer to single-click instead of double-click, even for objects that are conventionally activated by a double-click, such as file and folder objects, and hover visualizations appear more distracting. There seemed to be more tooltips appearing and disappearing with the pen group compared to the mouse group, an effect we refer to as "hover junk." Not only was this visually distracting, but pen participants also seemed to more be more consciously affected due to a possible difference in perceived depth ordering order of the pointer, tooltip, and target. These may reveal a difference in the conceptual model of the GUI when using a pen compared to a mouse.

Limited Input

It is perhaps obvious, but the lack of a keyboard appears to be a serious handicap for pen users. The main problem of entering text with only a pen has been an active research area for decades: refinements to handwriting recognition, gesture-based text input, and soft keyboards continue. However, even though text entry was not part of our task, several pen participants noted the lack of a keyboard and even mimed pressing common keyboard shortcuts like copy (Ctrl-C) and undo (Ctrl-Z). We observed mouse users instinctively reaching for the keyboard to access command shortcut keys, list accelerator keys, and enter quantitative values. Although all the tasks in our study could be completed without pressing a single key, this is not the way that users work with a GUI. Recent command-line-like trends in GUIs such as full text search and keyboard application launchers will further contribute to the problem.

7.2. Study Methodology

Our hybrid study methodology incorporates aspects of traditional controlled HCI experimental studies, usability studies, and qualitative research. Our motivation was to enable more diverse observations involving a variety of contexts and interactions—hopefully approaching how people might perform in real settings.

Degree of Realism

Almost any study methodology will have some effect on how participants perform. In our study we asked participants to complete a prepared set of tasks on a device we supplied, instrumented them with 3D tracking markers and a head-mounted camera, and ran the study in our laboratory. These steps were necessary to have some control over the types of interactions they performed and to provide us with rich logging data to analyze. It is important to note that our participants were not simply going about their daily tasks as they would in a pure field study. However, given that our emphasis is on lower level widget interactions, rather than application usability or larger working contexts, we feel that we achieved an appropriate degree of realism for our purposes.

Analysis Effort and Importance of Data Logs

Synchronizing, segmenting, and annotating the logs to get multifaceted qualitative and quantitative observations felt like an order-of-magnitude increase beyond conducting a usability study or a controlled experiment. Our custom-built software helped, but it did not replace long hours spent reviewing the actions of our participants. Qualitative observations from multiple rich observation logs are valuable but not easy to achieve.

Using two raters proved to be very important. Training a second coder forced us to iterate and formalize our coding decision process significantly. We feel this contributed greatly to a consistent assignment of codes to events and high level of agreement between raters. In addition, with two raters we were able to identify a greater number of events to annotate. Regardless of training and perseverance, raters will miss some events.

We found that each of the observational logging techniques gave us a different view of particular interaction and enabled a different aspect to analyze. We found the combination of the pen event log, screen capture video, and *head-mounted user view* invaluable for qualitative analysis. The pen event log and screen capture video are the easiest to instrument and have no impact on the participant. The head-mounted camera presents a mild intrusion, but observations regarding occlusion and missed clicks would have been very difficult to make without it. For quantitative analysis, we relied on the pen event log and the motion capture logs. Although the motion capture data enabled the analysis of participant movements, posture, and visualizing 3D pen trajectories, they required the most work to instrument, capture, and process.

8. CONCLUSION: IMPROVING DIRECT PEN INPUT WITH CONVENTIONAL GUIs

We have presented and discussed results from our study of direct pen interaction with realistic tasks and common software applications exercising a diverse collection

of widgets and user interactions. Our findings reveal five overarching issues when using direct pen input with a conventional GUI: lack of precision, hand occlusion, ergonomics when reaching, cognitive differences, and limited input. We feel that these issues can be addressed by improving hardware, base interaction, and widget behavior without sacrificing the consistency of current GUIs and applications.

Ideally, addressing the overarching issues previously identified should be done without radical changes to the fundamental behavior and layout of the conventional GUI and applications. This would enable a consistent user experience regardless of usage context—for example, when a Tablet PC user switches between slate mode with pen input and laptop mode with mouse input—and ease the burden on software developers in terms of design, development, testing, and support. Techniques that automatically generate user interface layouts specific to an input modality (Gajos & Weld, 2004) may ease the design and development burden in the future, but increased costs for testing and support will likely remain. With this in mind, we feel researchers and designers can make improvements at three levels: hardware, base interaction, and widget behavior.

Hardware improvements that reduce parallax and lag, increase input sensitivity, and reduce the weight of the tablet are ongoing and will likely improve. Other improvements that increase the input bandwidth of the pen, such as tilt and rotation sensing, may provide as of yet unknown utility—but past experience with adding buttons and wheels to pens has not been encouraging. More innovative pen form factors may provide new direction entirely, for example, a penlike device that operates more like a mouse as the situation requires. However, hardware improvements are likely to provide only part of the solution.

Base interaction improvements target basic input such as pointing, tapping, and dragging, as well as adding enhancements to address aspects such as occlusion, reach, and limited input. Conceptually these function like a pen-specific interaction layer that sits above the standard GUI. A technique can become active with pen input but without changing the underlying behavior of the GUI or altering the interface layout. Windows Vista contains examples of this strategy: the tap-and-hold for right-clicking, the ripple visualization for taps, and "pen flicks" for invoking common commands with gestures. However, we did not see any of our Tablet PC users taking advantage of ripple visualizations or pen flicks. Potential base-level improvements have been proposed by researchers, such as a relaxed crossing semantic for faster command composition (Dixon et al., 2008), Pointing Lenses for precision (Ramos et al., 2007), and enhancements such Hover Widgets for certain types of input (Grossman et al., 2006). Well-designed base interaction improvements have the capability to dramatically improve direct input overall.

The behavior of individual widgets can also be tailored for pen input, but this should be done without altering their initial size or appearance to maintain GUI consistency. Windows operating systems have contained a simple example of this for some time: There is an explicit option to cause menus to open to the left rather than right (to reduce occlusion with right-handed users). This is an example of how a widget's behavior can be altered without changing its default layout—the size and

appearance of an inactive menu remains unchanged—and could be improved further to occur automatically with pen input and detect handedness automatically (Hancock & Booth, 2004). For example, widget improvements proposed by researchers, such as the Zliding widget (Ramos & Balakrishnan, 2005), could be adapted to more closely resemble a standard slider widget before invocation. Others, such as the crossing based scrollbar in CrossY (Apitz & Guimbretière, 2004), may present too great of a departure from basic scrollbar appearance and behaviour. Hinckley et al.'s (2007) method of using a pen specific widget to remote control a conventional GUI widget may provide a useful strategy.

Although our study specifically targeted Tablet PC direct pen input with a conventional GUI, many of our results apply to other interaction paradigms and direct input devices. For example, in any type of direct manipulation (be it crossing, gestures, or other), if a target must be selected on the display, then many of the same issues with precision, reach, and occlusion remain. However, devices such as PDAs and smart phones typically have a GUI designed specifically for pen-based interaction, so targets may be larger to minimize precision errors and widgets placed to minimize the effect of occlusion (such as locating menu bars at the bottom of the display). Granted, techniques such as crossing may eliminate tapping errors, but because the input and display space are coincident, the collision of physical anatomy and visual space are inevitable creating occlusion problems.

In the case of touch screen input, the severity of these issues is likely to be more pronounced. Touch-enabled Tablet PCs are available, and vendor advertising touts this as a more natural mode of interaction. Of course, this is the same argument for pen interaction, and given the size of the finger compared to a pen, users will encounter the same types of problems when trying to operate the same GUI and the same applications. Enhancements such as the Touch Pointer (Microsoft, n.d.) introduced in Windows Vista and continued with Windows 7 may help with precision, but they could also be adding to occlusion problems. Multitouch input presents an even greater challenge for interacting with a conventional GUI. Certainly, as the device capabilities and characteristics begin to diverge from the mouse, the possibility that a satisfactory set of improvements can be found which bridge the gap becomes more doubtful. Windows 7 also supports *multitouch input* (Hoover, 2008)—if our findings regarding direct pen input have taught us anything, it is that touch and multitouch are unlikely to be relevant for typical users without an understanding of how touch and multitouch perform with conventional GUIs and conventional applications.

NOTES

Background. This article is based on the Ph.D. thesis of the first author.

Acknowledgments. We thank all our study participants and Matthew Cudmore for assistance during qualitative coding.

Authors' Present Addresses. Daniel Vogel, Department of Mathematics and Computer Science, Mount Allison University, 67 York Street, Sackville, New Brunswick, Canada E4L

1E6. E-mail: dvogel@mta.ca. Ravin Balakrishnan, Department of Computer Science, University of Toronto, 10 King's College Road, Room 3302, Toronto, Ontario, Canada M5S 3G4. E-mail: ravin@dgp.toronto.edu.

HCI Editorial Record. Received November 23, 2008. Revision received July 2, 2009. Accepted by Brad Myers. Final manuscript received November 24, 2009 — *Editor*

REFERENCES

Accot, J., & Zhai, S. (2002). More than dotting the i's—foundations for crossing-based interfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing our World, Changing Ourselves*. Minneapolis, MN: ACM.

Agarawala, A., & Balakrishnan, R. (2006). Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Montréal, Canada: ACM.

Ahlstroem, D., Alexandrowicz, R., & Hitz, M. (2006). Improving menu interaction: a comparison of standard, force enhanced and jumping menus. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Montréal, Canada: ACM.

Apitz, G., & Guimbretière, F. (2004). CrossY: A crossing-based drawing application. *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. Santa Fe, NM: ACM.

Autodesk. (2007). SketchBook Pro [Computer software]. Retrieved from http://www.autodesk.com/sketchbookpro

Balakrishnan, R., & MacKenzie, I. S. (1997). Performance differences in the fingers, wrist, and forearm in computer input control. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Atlanta, GA: ACM.

Bi, X., Moscovich, T., Ramos, G., Balakrishnan, R., & Hinckley, K. (2008). An exploration of pen rolling for pen-based interaction. *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*. Monterey, CA: ACM.

Bricklin, D. (2002, November 22). *About tablet computing old and new*. Retrieved from http://www.bricklin.com/tabletcomputing.htm

Briggs, R. O., Dennis, A. R., Beck, B. S., & Nunamaker, J. F. (1993). Whither the pen-based interface. *Journal of Management and. Information Systems*, *9*(3), 71–90.

Buxton, W. A. S. (1995). Chunking and phrasing and the design of human-computer dialogues. In *Human-Computer Interaction: Toward the Year 2000* (pp. 494–499). San Francisco, CA: Morgan Kaufmann.

Buxton, W. A. S., Fiume, E., Hill, R., Lee, A., & Woo, C. (1983). Continuous hand-gesture driven input. *Proceedings of Graphics Interface '83, 9th Conference of the Canadian Man-Computer Communications Society*. Edmonton, Alberta, Canada: A. K. Peters, Ltd.

Buxton, W. A. S., Sniderman, R., Reeves, W., Patel, S., & Baecker, R. (1979). The evolution of the SSSP score editing tools. *Computer Music Journal*, *3*(4), 14–25.

Cao, X., & Zhai, S. (2007). Modeling human performance of pen stroke gestures. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. San Jose, CA: ACM.

Chatty, S., & Lecoanet, P. (1996). Pen computing for air traffic control. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground*. Vancouver, Canada: ACM.

Chen, A. (2004, May 24). *Tablet PCs pass inspection.*Retrieved from http://www.eweek.com

Dixon, M., Guimbretière, F., & Chen, N. (2008). Maximizing efficiency in crossing-based dialog boxes. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Florence, Italy: ACM.

Fitzmaurice, G. W., Balakrishnan, R., Kurtenbach, G., & Buxton, W. A. S. (1999). An exploration into supporting artwork orientation in the user interface. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: The CHI Is the Limit*. Pittsburgh, PA: ACM.

Fitzmaurice, G., Khan, A., Pieké, R., Buxton, W. A. S., & Kurtenbach, G. (2003). Tracking menus. *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*. Vancouver, Canada: ACM.

Forlines, C., & Balakrishnan, R. (2008). Evaluating tactile feedback and direct vs. indirect stylus input in pointing and crossing selection tasks. *Proceeding of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems*. Florence, Italy: ACM.

Forlines, C., Vogel, D., & Balakrishnan, R. (2006). HybridPointing: Fluid switching between absolute and relative pointing with a direct input device. *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. Montreux, Switzerland: ACM.

Forsberg, A., Dieterich, M., & Zeleznik, R. (1998). The music notepad. *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*. San Francisco, CA: ACM.

Gajos, K., & Weld, D. S. (2004). SUPPLE: Automatically generating user interfaces. *Proceedings of the 9th International Conference on Intelligent User Interfaces*. Funchal, Portugal: ACM.

Gallenson, L. (1967). A graphic tablet display console for use under time-sharing. *Proceedings of the November 14-16, 1967, Fall Joint Computer Conference*. Anaheim, CA: ACM. doi:10.1145/1465611.1465703

Gross, M. D., & Do, E. Y. (1996). Ambiguous intentions: A paper-like interface for creative design. *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*. Seattle, WA: ACM.

Grossman, T., Hinckley, K., Baudisch, P., Agrawala, M., & Balakrishnan, R. (2006). Hover widgets: Using the tracking state to extend the capabilities of pen-operated devices. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Montréal, Canada: ACM.

Guimbretière, F., & Winograd, T. (2000). FlowMenu: Combining command, text, and data entry. *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*. San Diego, CA: ACM.

Gurley, B. M., & Woodward, C. E. (1959). Light-pen links computer to operator. *Electronics*, pp. 85–87.

Haider, E., Luczak, H., & Rohmert, W. (1982). Ergonomics investigations of work-places in a police command-control centre equipped with TV displays. *Applied Ergonomics*, *13*(3), 163–170.

Hancock, M. S., & Booth, K. S. (2004). Improving menu placement strategies for pen input. *Proceedings of Graphics Interface 2004*. London, Canada: Canadian Human-Computer Communications Society.

Harris, J. (2005, October 6). Saddle up to the minibar. *An Office User Interface Blog*. Retrieved from http://blogs.msdn.com/jensenh/archive/2005/10/06/477801.aspx

Hinckley, K., Baudisch, P., Ramos, G., & Guimbretière, F. (2005). Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Portland, OR: ACM.

Hinckley, K., Cutrell, E., Bathiche, S., & Muss, T. (2002). Quantitative analysis of scrolling techniques. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing our World, Changing Ourselves*. Minneapolis, MN: ACM.

Hinckley, K., Guimbretière, F., Baudisch, P., Sarin, R., Agrawala, M., & Cutrell, E. (2006). The springboard: Multiple modes in one spring-loaded control. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Montréal, Canada: ACM.

Hinckley, K., Zhao, S., Sarin, R., Baudisch, P., Cutrell, E., Shilman, M., & Tan, D. (2007). InkSeine: In Situ search for active note taking. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. San Jose, CA: ACM.

Hoover, N. (2008, May 28). Microsoft unveils multi-touch technology for Windows 7. *Information Week*. Retrieved from http://www.informationweek.com

Hourcade, J. P., & Berkel, T. R. (2006). Tap or touch?: Pen-based selection accuracy for the young and old. *CHI '06 Extended Abstracts on Human Factors in Computing Systems*. Montréal, Canada: ACM.

Inkpen, K., Dearman, D., Argue, R., Comeau, M., Fu, C., Kolli, S., Moses, J., . . . Wallace, J. (2006). Left-handed scrolling for pen-based devices. *International Journal of Human-Computer Interaction*, *21*, 91–108.

Jordan, B., & Henderson, A. (1995). Interaction analysis: Foundations and practice. *The Journal of the Learning Sciences*, *4*(1), 39–103.

Kurtenbach, G., & Buxton, W. A. S. (1991a). GEdit: A test bed for editing by contiguous gestures. *SIGCHI Bulletin*, *23*(2), 22–26.

Kurtenbach, G., & Buxton, W. A. S. (1991b). Issues in combining marking and direct manipulation techniques. *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology*. Hilton Head, SC: ACM.

Lank, E., & Saund, E. (2005). Sloppy selection: Providing an accurate interpretation of imprecise selection gestures. *Computers & Graphics*, *29*, 490–500.

Lee, J. C., Dietz, P. H., Leigh, D., Yerazunis, W. S., & Hudson, S. E. (2004). Haptic pen: A tactile feedback stylus for touch screens. *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. Santa Fe, NM: ACM.

Li, Y., Hinckley, K., Guan, Z., & Landay, J. A. (2005). Experimental analysis of mode switching techniques in pen-based user interfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Portland, OR: ACM.

Lin, J., Newman, M. W., Hong, J. I., & Landay, J. A. (2000). DENIM: Finding a tighter fit between tools and practice for Web site design. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. The Hague, the Netherlands: ACM.

Long, C. A. J., Landay, J. A., Rowe, L. A., & Michiels, J. (2000). Visual similarity of pen gestures. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. The Hague, the Netherlands: ACM.

MacKenzie, I. S., Sellen, A., & Buxton, W. A. S. (1991). A comparison of input devices in element pointing and dragging tasks. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Reaching Through Technology*. New Orleans, LA: ACM.

Markoff, J. (2000, November 9). Microsoft sees new software based on pens. *New York Times*. Retrieved from http://www.nytimes.com/2000/11/09/technology/09SOFT.html

Meyer, A. (1995). Pen computing: A technology overview and a vision. *SIGCHI Bulletin*, *27*(3), 46–90.

Microsoft. (2009, October 22). Windows 7 Product Guide. Retrieved from http://www.microsoft.com

Microsoft. (n.d.). What is the touch pointer? *Windows Vista Help*. Retrieved November 12, 2008, from http://windowshelp.microsoft.com

Mizobuchi, S., & Yasumura, M. (2004). Tapping vs. circling selections on pen-based devices: evidence for different performance-shaping factors. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Vienna, Austria: ACM.

Moran, T. P., Chiu, P., & Melle, W. V. (1997). Pen-based interaction techniques for organizing material on an electronic whiteboard. *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. Banff, Canada: ACM.

Moscovich, T., & Hughes, J. F. (2004). Navigating documents with the virtual scroll ring. *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. Santa Fe, NM: ACM.

Pogue, D. (2000, November 16). Microsoft sets sights on Tablet PC market. *New York Times*. Retrieved from http://www.nytimes.com/2000/11/16/technology/16GEE1.html

Ramos, G., & Balakrishnan, R. (2003). Fluid interaction techniques for the control and annotation of digital video. *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*. Vancouver, Canada: ACM.

Ramos, G., & Balakrishnan, R. (2005). Zliding: Fluid zooming and sliding for high precision parameter manipulation. *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*. Seattle, WA: ACM.

Ramos, G., & Balakrishnan, R. (2007). Pressure marks. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. San Jose, CA: ACM.

Ramos, G., Boulos, M., & Balakrishnan, R. (2004). Pressure widgets. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Vienna, Austria: ACM.

Ramos, G., Cockburn, A., Balakrishnan, R., & Beaudouin-Lafon, M. (2007). Pointing lenses: facilitating stylus input through visual- and motor-space magnification. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. San Jose, CA: ACM.

Ramos, G., Robertson, G., Czerwinski, M., Tan, D., Baudisch, P., Hinckley, K., & Agrawala, M. (2006). Tumble! Splat! Helping users access and manipulate occluded content in 2D drawings. *Proceedings of the Working Conference on Advanced Visual Interfaces*. Venezia, Italy: ACM.

Ranjan, A., Birnholtz, J. P., & Balakrishnan, R. (2006). An exploratory analysis of partner action and camera control in a video-mediated collaborative task. *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work*. Banff, Canada: ACM.

Ren, X., & Moriya, S. (2000). Improving selection performance on pen-based systems: a study of pen-based interaction for selection tasks. *ACM Transactions on Computer-Human Interaction*, *7*, 384–416.

Schilit, B. N., Golovchinsky, G., & Price, M. N. (1998). Beyond paper: supporting active reading with free form digital ink annotations. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Los Angeles, CA: ACM.

Scholtz, J., Young, J., Drury, J., & Yanco, H. (2004). Evaluation of human-robot interaction awareness in search and rescue. *Robotics and Automation, 2004* (Vol. 3, pp. 2327–2332). doi:10.1109/ROBOT.2004.1307409

Scott, S. D. (2005). *Territoriality in collaborative tabletop workspaces*. Unpublished doctoral dissertation. University of Calgary, Calgary, Canada.

Scott, S. D., Carpendale, S. M., & Inkpen, K. M. (2004). Territoriality in collaborative tabletop workspaces. *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*. Chicago, IL: ACM.

Sellen, A., Kurtenbach, G., & Buxton, W. A. S. (1992). The prevention of mode errors through sensory feedback. *Human-Computer Interaction*, *7*, 141–164.

Shao, J., Fiering, L., & Kort, T. (2007). *Dataquest insight: Tablet PCs are slowly gaining momentum* (No. G00147423). Retrieved from http://www.gartner.com

Sharmin, S., Evreinov, G., & Raisamo, R. (2005). Non-visual feedback cues for pen computing. *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005*. Washington, DC: IEEE Computer Society.

Shilman, M., Tan, D. S., & Simard, P. (2006). CueTIP: A mixed-initiative interface for correcting handwriting errors. *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. Montreux, Switzerland: ACM.

Smith, G., Schraefel, M. C., & Baudisch, P. (2005). Curve dial: Eyes-free parameter entry for GUIs. *CHI '05 Extended Abstracts on Human Factors in Computing Systems*. Portland, OR: ACM.

Sommerich, C. M., Ward, R., Sikdar, K., Payne, J., & Herman, L. (2007). A survey of high school students with ubiquitous access to tablet PCs. *Ergonomics*, *50*, 706–727.

Spooner, J. G., & Foley, M. J. (2005, August 30). Tablet PCs' future uncertain. *eWeek.com*. Retrieved from http://www.eweek.com

Stone, B., & Vance, A. (2009, October 4). Just a touch away, the elusive Tablet PC. *The New York Times*. Available from http://www.nytimes.com/2009/10/05/technology/05tablet.html

Strauss, A. L., & Corbin, J. M. (1998). *Basics of qualitative research: Techniques and procedures for developing*. Beverly Hills, CA: Sage.

Tian, F., Ao, X., Wang, H., Setlur, V., & Dai, G. (2007). The tilt cursor: Enhancing stimulus-response compatibility by providing 3D orientation cue of pen. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. San Jose, CA: ACM.

Tian, F., Xu, L., Wang, H., Zhang, X., Liu, Y., Setlur, V., & Dai, G. (2008). Tilt menu: Using the 3D orientation information of pen devices to extend the selection capability of pen-based user interfaces. *Proceedings of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems*. Florence, Italy: ACM.

Truong, K. N., & Abowd, G. D. (1999). StuPad: Integrating student notes with class lectures. *CHI '99 Extended Abstracts on Human Factors in Computing Systems*. Pittsburgh, PA: ACM.

Turner, S. A., Pérez-Quiñones, M. A., & Edwards, S. H. (2007). Effect of interface style in peer review comments for UML designs. *Journal of Computing Sciences in Colleges, 22*, 214–220.

Twining, P., Evans, D., Cook, D., Ralston, J., Selwood, I., Jones, A., . . ., Scanlon, E. (2005). *Tablet PCs in schools: Case study report*. Conventry, UK: Becta ITC Research. Retrieved from http://publications.becta.org.uk/display.cfm?resID=25914

Vogel, D., & Balakrishnan, R. (2010). Occlusion-aware interfaces. *Proceedings of the 28th International Conference on Human Factors in Computing Systems*. Atlanta, GA: ACM.

Ward, J., & Phillips, M. (1987). Digitizer technology: Performance characteristics and the effects on the user interface. *Computer Graphics and Applications, IEEE*, *7*(4), 31–44.

Whitefield, A. (1986). Human factors aspects of pointing as an input technique in interactive computer systems. *Applied Ergonomics*, *17*(2), 97–104.

Zhai, S., & Kristensson, P. (2003). Shorthand writing on stylus keyboard. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Ft. Lauderdale, FL: ACM.

Zhai, S., Smith, B. A., & Selker, T. (1997). Dual stream input for pointing and scrolling. *CHI '97 Extended Abstracts on Human Factors in Computing Systems: Looking to the Future*. Atlanta, GA: ACM.

Zuberec, S. (2000). The interaction design of Microsoft Windows CE. In E. Bergman (Ed.), *Information appliances and beyond* (p. 385). San Francisco, CA: Morgan Kaufmann.