Pen and Touch Gestural Environment for Document Editing on Interactive Tabletops

Fabrice Matulic, Moira C. Norrie
ETH Zurich, Institute of Information Systems
8092 Zurich, Switzerland
{matulic|norrie}@inf.ethz.ch

ABSTRACT
Combined pen and touch input is an interaction paradigm attracting increasing interest both in the research community and recently in industry. In this paper, we illustrate how pen and touch interaction techniques can be leveraged for editing and authoring of presentational documents on digital tabletops. Our system exploits the rich interactional vocabulary afforded by the simultaneous availability of the two modalities to provide gesture-driven document editing functionality as an expert alternative to widgets. For our bimanual gestures, we make use of non-dominant hand postures to set pen modes in which the dominant hand articulates a variety of transactions. We draw an analogy between such modifier postures and modifier keys on a keyboard to construct command shortcuts. Based on this model, we implement a number of common document editing operations, including several page and element manipulations, shape and text input with styling, clipart retrieval and insertion as well as undo/redo. The results of a lab study provide insights as to the strengths and limitations of our approach.

Author Keywords
Pen and touch interaction; digital tabletops; document engineering

ACM Classification Keywords
H.5.2.

General Terms
Human Factors; Design

INTRODUCTION
Within popular visions of the office of the future, the digital workdesk plays a central role, as an empowering tool for knowledge workers to carry out their tasks. Compared to traditional desktop computers, such types of devices support more direct interactions, thanks to which work can be performed more efficiently and intuitively, at least in theory. One can however observe that, even with the advent of touch devices, the vast majority of office work is still performed on PCs with mice and keyboards, especially so-called "productivity" tasks, i.e. tasks with high added value that involve a great deal of more or less sophisticated manipulations. Among those essential activities is all the work revolving around documents: the creation of presentations, editing and publishing of reports, memos etc. To support those needs in this device landscape increasingly influenced by (multi)touch, the latest office suites such as Microsoft Office provide some level of adaptation to those kinds of appliances. With the exception maybe of OneNote, however, the interfaces of the different programs still very much betray their WIMP (Windows, Icons, Menus, Pointers) origins. What is more, industrial designers of touch UIs overwhelmingly reason in terms of simple interactions with one or two finger touches (tap, pinch-spread, swipe etc.), which needlessly restricts the breadth of the expressional vocabulary potentially available to users on those platforms.

With the recent resurgence of the stylus pen as an additional modality that can be used not instead of, but in tandem with touch, the breadth of tabletops' interactional capabilities has further expanded, as illustrated by numerous research projects [4, 7, 8, 10-12, 16, 17]. Yet, despite this demonstrated potential and what seems like a natural fit between pen-operated digital tabletops and document-centric work, those types of scenarios have surprisingly received little attention so far from the HCI community.

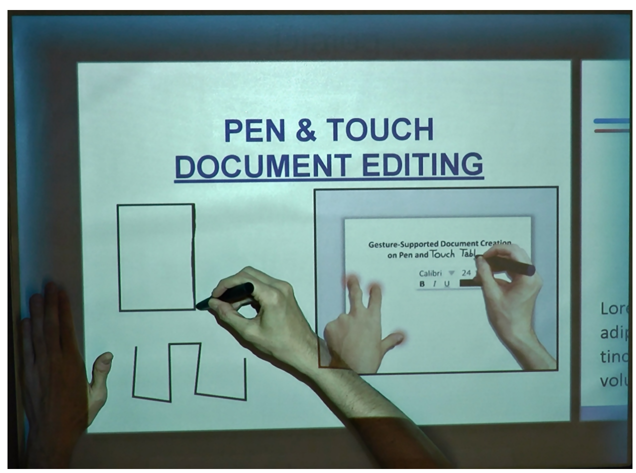


Figure 1. Our document-authoring environment on pen and touch interactive tabletop

In this paper, we propose to show how the pen and touch input paradigm can be utilised to support document-authoring tasks on an interactive tabletop, where we explore the feasibility of realising common document-editing opera-

ITS '13, October 06 - 09 2013, St Andrews, United Kingdom

ACM 978-1-4503-2271-3/13/10…$15.00.
http://dx.doi.org/10.1145/2512349.2512802

tions mostly via gestural shortcuts rather than WIMP-based transactions. Following Reality-Based Interaction [13] principles, we design a set of (uni- and bimodal) interactions, together forming a coherent and integrated document authoring environment. In particular, we systematise the use of the non-dominant hand to define different local and global pen modes through modifier postures, which we relate to modifier keys of a keyboard. We draw an analogy between our hybrid touch-modified pen interactions and keyboard shortcuts that can be used as expert commands instead, or in addition to traditional WIMP controls. Our contributions also include a few novel techniques for text and shape input and styling.

This paper is a continuation and extension of our work-in-progress introduced in [18].

RELATED WORK

Much of the theoretical background of prior work on pen and touch interaction originates from Guiard's kinematic chain (KC) model, which describes the roles of the dominant and non-dominant hands (respectively DH and NDH) in asymmetric bimanual motor actions [9]. According to this model, the NDH initiates and sets the frame of reference in which the DH operates. The expression of this principle in the pen and touch context is a division of labour, where the pen is used for inking and fine-precision operations (drawing, handwriting, lasso selections etc.), touch performs coarse manipulations (panning, zooming and other control gestures) and the combination of touch + pen yields new interactions [12]. As several authors have observed, however, in the digital world, it is not always wise to assign modalities to strictly distinct roles, since for certain types of actions such as tool selections, users expect to be able to use either [7, 12, 16, 24].

The examples in the literature show that hybrid pen and touch actions can materialise in different ways, but the most common pattern is that of the NDH setting the operational context of the pen, in accordance with the KC model. For instance, a flat hand can constrain the pen to draw straight lines [4] or an object held down by a finger can be duplicated by dragging it off with the pen [7, 12]. A particularly powerful manifestation of combined pen and touch interaction is pen-mode switching driven by the NDH, that is, the function performed by the stylus is determined by a specific position or pose of the NDH on the touch screen. Those mode-setting poses can be maintained touch selections of items in a toolbar [17] or finger-based postures [10, 11, 16]. Requiring users to maintain a function state through continuous muscular tension (a particular instance of a quasimode [20]) as opposed to system-maintained modes activated by single taps or clicks has been shown to be an effective kinaesthetic awareness mechanism to keep a non-regular mode active [16, 22].

In all the bimanual gestural patterns used in the prototypes developed so far, one can observe that the atomic unimodal actions are simple (tap, hold, drag) and that both hands are never in motion simultaneously. This makes sense considering complex synchronised movements of the two hands likely require more demanding coordination effort (similar to patting one's head and rubbing one's stomach at the same time). While there are indeed constraints to human motor skills one should take into account when designing bimanual and bimodal gestures, we believe prior work has only scratched the potential of the expressional power that can be achieved with synergistic pen and touch interactions. In our prototype, we attempt to further push the envelope of hybrid interactions by making more systematic use of them in our document-editing context.

With regard to modern keyboardless interfaces for document creation, we are not aware of any significant achievement in the research community beyond scrapbooks, note-taking and basic editing applications. SketchPoint is a tool to quickly assemble informal presentations from freeform notes and sketches [14]. As the name reveals, the resulting presentations are only rough sketches and not electronic documents. In [27], Wu et al. describe a rudimentary publishing application, which was developed to illustrate their gesture design framework. Their application relies on various multitouch interactions as well as stylus input, but interestingly, inking is not the default mode and has to be explicitly activated with a two-finger NDH posture. Another document-creation program based on reuse of scanned document elements was developed for pen-operated copiers [15], but it does not support insertion of new user content. Also relevant in this category to some extent are annotating and active reading systems developed for interactive surfaces [21, 23, 24]. Those prototypes, however, only deal with editing of existing documents and do not consider the authoring of rich documents from scratch.

DESIGN APPROACH

Target Document Types

We approach the problem by first considering what types of documents might be more suitable for our pen and touch tabletop platform. Intuitively, we surmise that page-based documents with much graphical content or assembled from various elements are more fitting candidates than continuous, text-intensive documents, for which the PC and physical keyboards are arguably still the best tools. Hence, in this prototype effort, we gear the design of our interface and interactions primarily to documents of the former category, e.g. presentations, flyers, brochures and similar documents typically authored in presentation and publishing software. Narrowing the target scope to a particular kind of documents also allows the design approach to be more focused, so that the resulting interface does not become too complex and thus unwieldy, in attempts to support all the specifics of diverse types of data.

Editable Document Element Properties

A quality of advanced document authoring software is that almost every possible element property can be edited and changed. Text can thus be styled in a number of ways, from font type and size, appearance, alignment, foreground and background colours etc. For shape elements, in particular, users expect to be able to modify attributes such as stroke colour, width and style even after the shape has been drawn. Thus, it is not sufficient to simply detect shapes, strokes and their features (e.g., if the line is continuous or dashed) upon inking. The interface has to provide means to conveniently edit those properties after input, without requiring the element to be entirely redrawn.

Reality-Based Interaction

Jacob et al.'s Reality-Based Interaction (RBI) framework [13] provides a theoretical foundation for the design of post-WIMP interfaces, which we can apply to pen and touch tabletop applications. This model is especially relevant for a document-authoring system, as people can bring in their experience of engaging with physical documents in the real world. The challenge, therefore, is to try to leverage those skills, when possible, by designing interactions that are either directly imported from real pen-on-paper experiences or based on powerful analogies. Those RBIs, together as a cohesive toolset, should enable users to efficiently produce structured electronic documents similar to those authored using professional office software. Our design process therefore considers what types of functions can be achieved through direct gestural input and when helper tools are unavoidable (e.g. to select a colour or a particular font). We explain how we address those challenges following those principles in our system description below.

Modifier Postures

We define modifier postures as specific NDH poses, which change the operational context of the pen. A posture is a combination of fingers [10, 16] or any distinctive shape formed by the contact area of the hand on the surface that can be reliably identified [25]. Similar to modifier keys on a keyboard such as Control and Alt, NDH postures can be viewed as touch-based modifiers, which, jointly with pen strokes, can be used to form a variety of commands. We distinguish global postures that control the general pen mode and can be triggered anywhere on the surface, from local or location-constrained postures that apply to the specific area or component on which they are performed (e.g. finger hold + pen drag gesture [7, 12]). Note that in both cases the user can remain concentrated on the DH's task at all times, as global postures allow blind switching (since the position of the NDH does not matter) and for local postures, the NDH operates in the vicinity of the DH so the locus of attention is the same. In this sense, we differentiate ourselves from Hamilton et al. [10], who, even for global postures, require that they be executed on a special command panel at the bottom of the screen. Our method is more similar to that used in the third experiment of our previous pen and touch study [16], but instead of setting a single inking property (stroke style) we construe modifier postures as general controllers of different categories of non-inking pen functions.

Another important issue is maintaining awareness of the currently active mode via appropriate feedback mechanisms. In [16], we used a simple icon, which may be adequate for local postures, if shown in the area of action, but is easily overlooked in the case of global postures, where pen operations can occur anywhere on the screen. At the same time, one should be careful not to choose too large visual indicators, as this would be too disruptive for the user. Our solution, here, uses light gradient borders surrounding the workspace with distinct mode-characteristic shades and matching stroke colours for the pen. Such a visual cue is easily spotted by the peripheral vision, without interfering with the main interface and the task at hand.

Finally, because global postures, by definition, can be triggered everywhere, they can potentially collide with other unimodal gestures and basic multitouch interactions, especially one-finger panning and two-finger scaling. A simple solution to avoid such conflicts is to choose postures based on three or more fingers or large contact areas.

APPLICATION

Apparatus

Our enabling technology for our pen and touch tabletop system is based on the tried-and-true combination of a DiamondTouch screen [6] and Anoto technology[1] introduced by Brandl et al. [4] and used in our previous work [16, 17].

Workspace

The setting of our interactive workspace follows a desktop metaphor, where documents appear on a virtual work desk and are directly manipulated by users via multitouch and pen input (Figure 1). Rather than opting for a full 3D model of a desktop à la BumpTop [1], we opted for a design in between a close-to-real environment and traditional interfaces of office software, as our experience has shown that pushing the desktop or paper metaphor too far on a 2D surface can impact practicality and user acceptance [17].

The interface shows all open documents, by default a single page of each. Common drag, pinch/spread gestures to pan and zoom documents are available. Documents can be expanded and collapsed to respectively display and hide other pages using a single-finger shaking gesture (Figure 2). Expanded pages can be laid out horizontally or vertically depending on the direction of the shaking motion. When pages are collapsed, the document appears as a pile, on top of which the currently selected page appears. A double tap on a particular page triggers an instant fit-to-screen zoom to enable more precise content editing. Instead of repeated dragging to navigate between pages, lateral swipe gestures can be used to quickly flick from one page to another. Following our design objective to maintain a smooth experience without tedious context switches, document pages are always editable, regardless of zoom factor or state, i.e. users

[1] http://anoto.com

can input and manipulate content on pages whether documents are collapsed, expanded or magnified.

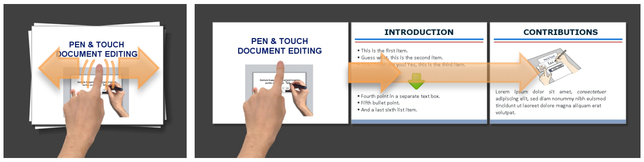


Figure 2. Finger shake gesture to expand and collapse documents.

We support various page manipulations to move and copy elements within and between documents: moving a page is achieved by placing three fingers on it (three fingers to distinguish the gesture from one-finger panning and two-finger scaling) and dragging it to a new location, inside an existing document or on the workspace to create a new document. Because the muscular tension imposed by a three-finger dragging operation is particularly demanding, we allow the gesture to be relaxed [27] so that the dragging motion can also be performed with a single finger after the move operation has been registered. Pages can be copied instead of moved by holding them with a finger (local posture) and dragging them away with the pen. This gesture is similar to those described in [7, 12] (and to a lesser extent, the cloning gesture of [8], which uses a second finger touch instead of the pen), but we additionally support moving and copying of page ranges, where the two hands determine the start and end pages of the range (Figure 3 left). Furthermore, if two fingers are used to pin down a page rather than one, a blank copy is created instead, as a convenience to add new pages with the same format (Figure 3 right). In the latter case, a quick pen swipe in the direction of where the page should be created is also sufficient to execute the operation.

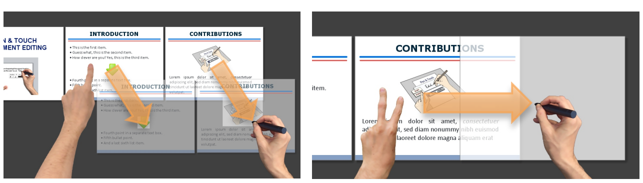


Figure 3. Copying a range of pages with one-finger hold + pen drag action (left) and creating a new blank page with two-finger hold + pen drag (right).

Document elements that make up the content of a page are independent components that can be moved within a page, between pages of the same document and between different documents. Resizing is performed in a standard manner, using control handles located at the edges of the element's bounding box.

Global Modes

For our document-editing purposes, we define five different global modes: two inking and three non-inking, which are listed in Table 1 and explained in detail in the next sections.

All postures are quasimodes [20], i.e. they are maintained throughout the pen action. In addition to providing kinaesthetic awareness, such a design has the advantage that an operation can be immediately cancelled during the pen movement without incurring any penalties, by simply lifting the NDH before the pen action has completed. From a performance point of view, using NDH postures to control how pen input will be interpreted means that only recognition engines that are necessary need be activated in the selected modes. Hence, handwriting recognition need only be active in text-entry mode, shape recognisers in shape-input mode and gesture detectors in command mode. Clearly separating those interactional categories increases responsiveness and makes sure there is no overlap between the different gestures, without compromising the range of available expressions.

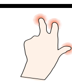
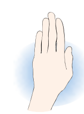
Inking Modes		Non-Inking Modes	
NDH Posture	Mode	NDH Posture	Mode
(hand re-leased)	Text		Command
	Shape		Selection
			Content Association

Table 1. Global Modes used in the editor

STROKE INPUT AND STYLING

As shown in Table 1, the two types of inked input for the stylus are text and shape. Both data types are subject to on-the-fly interpretation by dedicated recognition engines (VisionObject's MyScript[2]) in order to convert user strokes into properly formatted content, respectively typeset text and smooth vectorised shapes. The default input mode, i.e. when the pen marks a page without NDH-modifying, is text. For shape and freeform strokes, the user has to indicate the mode with an NDH posture. We chose a flat hand, which mimics the holding of a sheet of paper (as in Guiard's example [9]) while drawing with the DH. This pose has the further advantage of blocking all other touch input, thereby providing palm rejection while inking strokes.

In the initial version of our prototype we did not require explicitly setting which type of content was being inserted, as we wanted users to be free to ink pages as they would a sheet of paper [18]. To determine how to convert pen data, we relied on an algorithm, which estimated the likelihood that input strokes were text or not. But early tests showed that in many cases the wrong data type was inferred (e.g. a letter 'O' instead of a circle or a line instead of a hyphen etc.) and so rather than risking causing user frustration we decided to strictly enforce the separation of the two kinds of content with different modes. We do include a level of automatic element segmentation in that if the user draws a

[2] http://www.visionobjects.com/en/myscript

grid-like sketch, a table element is created in our document model instead of a shape. Our current prototype does not feature table-specific interactions other than adding rows and columns through transversal lines and inserting text in cells, but this is an area we are interested in exploring more thoroughly in the future (see Future Work section below).

Text

As stated above, text is entered through normal handwriting and converted on-the-fly by the recogniser, where the resulting typeset text appears *in situ*, i.e. it replaces the user's handwritten content. We opted for this more direct input method instead of a separate writing pad, as such tools create a disconnection between the user's ink and the output, which we feel detracts from an experience close to pen-on-paper. For text styling, we propose a principle inspired by RBI that what can be reliably derived from pen input should be done so, while other aspects that are hard to accurately infer from hand-drawn content should be determined through explicit selections. Following that strategy, we tease apart properties such as text size, underlining etc. which can be directly obtained from ink data and characteristics such as font type and style that are difficult for users to imitate and so need to be specifically set. Thus, when text is entered in a blank area of a document page, the size of the font used for the recognised text is determined so as to roughly match that of the user's script. If, on the other hand, inked data is initiated close to or within another text element it is considered an edit operation and so the recognised result is added to the element and the font size and style of the nearest letter in this element is adopted, which is the usual convention in word processors.

Figure 4. Insertion of handwritten text within text element. Text after the insertion point is gradually shifted to create writing space.

Our text input scheme also includes a novel feature that typical handwriting-based techniques lack: the possibility to add and convert handwritten phrases directly in the middle of formatted text elements. For that, the user simply needs to start writing within the element for a gap to open at the insertion point. As the user writes their text, the right part of the edited element continuously shifts to create more space (Figure 4). If a portion of the text is highlighted the user-inserted content replaces the selection. This blended text-editing technique is an attempt to recreate standard text-insertion behaviour with handwriting.

Because it is not possible to insert isolated white spaces in formatted text elements using handwritten input, we include two convenience pen gestures in Command Mode for that purpose: a chevron stroke gesture (^) adds a space (a specialisation of the common text-insert mark used in proof-reading) and a ↲ gesture inserts a line break.

In Selection Mode, text highlighting with the pen is performed in a manner similar to most pointer-based interfaces, i.e. by dragging the pen across the desired portion of text. Quick selection shortcuts found in modern word processors to highlight entire words and paragraphs with double and triple taps, respectively, are also available. As with pages, we support cut/copy paste of text through single-finger touch dragging and NDH touch-hold + DH pen-dragging gestures. The selected text can then be touch-dragged within the element to be placed at another index. If the cut/copied text portion is dragged outside the bounds of the element and released on free page space, a new text element is created.

Our interface features a set of "chopping" gestures to perform alignment operations on text elements. Those shape-based interactions allow users to set the horizontal alignment of a text element based on how the gesture contact shape intersects the element's bounding box. Single chops close to the element's left or right borders accordingly align the text left or right, whereas a chop in the middle of the object centres the text. Finally, a double-chop with the two hands on both lateral edges of the element's bounding box causes the text to become justified (Figure 5). Those simple gestures have the advantage that they combine scope framing (target text range) and operation execution (alignment) in a short, single-action chunk [5], which differentiates them from fixed guides and rulers such as rails [25] and the composite and dragged gestures of [8].

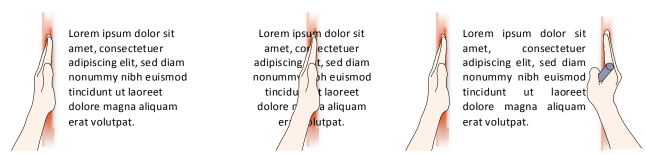


Figure 5. Chopping gestures for text alignment. Illustrated are left, centered and justified alignments.

Except for underlining, which is directly drawn under the desired text with the pen, styling options for stroke data are set via a pop-up toolbar summoned by a single-finger touch action, a short-press if initiated on empty space and a slightly longer press if activated inside a document (in order to distinguish it from move operations). The toolbar also automatically appears on a text-highlighting action just below the edited text element. The style properties that can be set in this manner are font family, colour and font style (bold, italic and normal). Here both touch and the pen can be used to select style options, the second alternative being perhaps the most effective when the pen is close to the widget (as in the text-highlighting case) or when several selections need to be made in succession. To maintain user awareness about the currently active font and style options at all time, the font family name styled according to the selected options is displayed semi-transparently at the top of the interface.

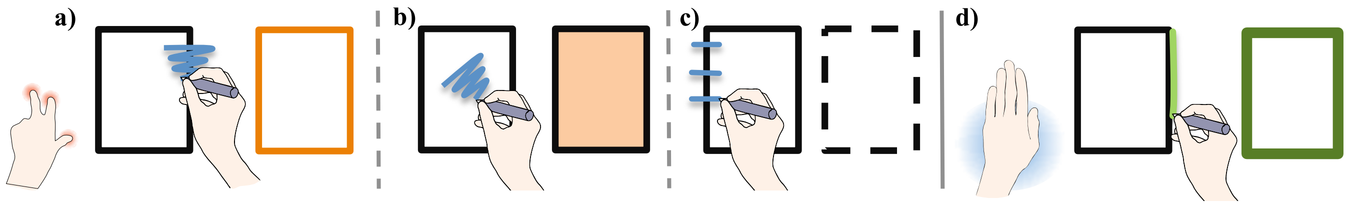


Figure 6. Shape styling gestures. In command mode: scratch gesture to change a) the colour of contours and b) inner areas; c) three short strokes toggle between dashed and continuous lines. d) In shape mode: line or curve tracing along shape contour to increase its width and blend colours (width is decreased if transparent colour is selected)

As an extension to its mode-setting role, we explore the possibility of allowing the maintained NDH posture to also control properties of the currently activated function through hand movements. We apply this technique for font style and size changes in Selection Mode. Hence, if text is highlighted, vertical dragging of the NDH modifies its font size, while in the horizontal direction, the font style can be changed via longitudinal toggle areas (Figure 7 left). This overloads functionality already provided through the popup toolbar but allows rapid phrasing of the most common text-styling operations through smooth sequences of NDH-DH chunks. If the whole text element is selected (or any type of element for that matter), dragging up or down changes the element's z-index to make it appear above or under overlapping objects (Figure 7 right). Here as well, we support gesture relaxation in order to allow single-finger dragging after the four-finger posture has been registered.

Figure 7. NDH vertical dragging in Selection Mode to change text size (left) and element Z-order (right)

As shown in the figure, in both cases an interactive overlay appears on the screen with visuals displaying how those properties are affected by directional dragging. We think this method of changing the font size via a continuous physical action is more intuitive for users compared to selecting a number from a drop-down menu, for which it is difficult to picture how it translates visually on the sheet (and even more so in vector-based editors). The WIMP equivalent of this technique would be a slider.

Shape and Freeform Input

In Shape Mode, the user can sketch freely on a page. The underlying shape beautifier attempts to detect geometrical shapes and if it recognises any, it smoothes the strokes accordingly. User strokes that "hug" the outline of existing elements are treated differently. For instance, if the user draws a rectangle that completely surrounds an element and that element has no visible contour, a border is created and attached to it. For shape elements and items with borders, curves drawn along their contours with a hard colour cause their thickness to increase and the border colour to be merged with the selected colour (Figure 6d), in analogy with physical drawing and painting effects. If, however, the transparent colour is active, the thickness is decreased, as if the shape outline was being trimmed by an eraser pen.

In Command Mode, the colours of shape elements can be changed using a scratch gesture, mimicking pencil-drawing techniques to add shading to line sketches. Both fill and contour colours can be modified in this way. Depending on whether the scratch gesture is executed inside the shape or on the contour, the fill or contour colour is replaced by the active colour (Figure 6a and b). If performed on free unoccupied space of a page, its background colour is changed.

Also in Command Mode, three short cutting strokes across the contour of a shape outline or element border toggle between dashed and continuous line styles (Figure 6c). This DH gesture is similar to that described in [7], but we additionally use the average distance between the intersecting strokes to determine the dash pattern.

Finally, in Selection Mode, the pen functions as an eyedropper tool to sample colours from objects.

GENERAL COMMANDS

Here we describe further operations executed by pen gestures in Command Mode:

New Page/Document Creation

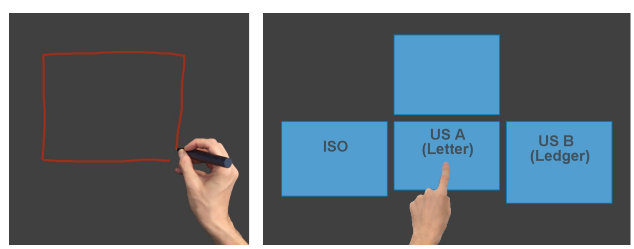


Figure 8. Drawing a rectangle to create a new page/document (left). A choice of standard page formats to select from appears (right).

Similar to [7]'s node-creation gesture, a rectangle can be drawn on the workspace to create new pages. Specifically, a new page is added to an existing expanded document if the rectangle is drawn in a location within, before or after the document. If the rectangle is drawn on empty space an entirely new document is created. When the user completes the gesture, an overlay showing different page formats appears with standard aspect ratios: ISO, US A and B (Figure 8). Depending on the length/width ratio of the rectangle, those page samples appear in portrait or landscape format. The user can then select one of the standard formats or the initially drawn rectangle.

Deletion

Line strokes delete elements (pages, document components, text) crossed by them. If the delete stroke is executed inside a text element, the start and end of the line determines the text to erase, unless a specific portion of the text is highlighted, in which case the strike-out gesture deletes the entire selection.

This gesture is distinguished from the contour dash toggle gesture, which is also based on line strokes, by its trigger area: for the former, the line needs to cross two distinct edges of its bounding box, whereas for the latter each line must intersect the same segment of the shape contour. As the bounding box has a minimum size requirement, it is always possible to clearly strike through two of its edges, even in the case where the shape element is a straight line.

Undo/Redo

Undo and redo are supported for all document operations through repeated anticlockwise and clockwise circling gestures (which other authors have used for scrolling [19]). Several operations can thus be performed in rapid succession without lifting the pen, with each completed circle triggering one undo or redo action. Those gestures inherently convey the idea of "rewinding" and "fast-forwarding" as if operating a dial to navigate within a timeline.

CONTENT REUSE

Content Registration and Insertion

Authoring office documents often involves reusing particular data elements in multiple compositions, for instance, addresses and company logos in letters, common decorations and symbols in presentations etc. To meet that requirement, we include a mode dedicated to gesture-based content insertion. This mode allows users to associate gestures with specific document elements and recall them by re-executing that same gesture.

To register a gesture, users first select an element with a single finger of the NDH, lay down the four other fingers to activate the mode and enter a single-stroke pen gesture to complete the association (Figure 9 left) (in our implementation we use the $1 recogniser [26], but any other algorithm can, of course, be used instead). If a gesture is deemed similar to that associated with a previously registered element, the link is replaced and the gesture is assigned to the new element. To insert a "saved" element, users simply execute its trigger-gesture on a document page, where the size of the inserted element is computed based on the bounding box of the pen stroke (Figure 9 right).

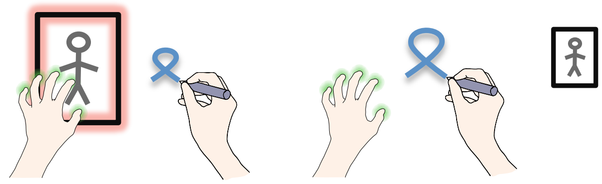


Figure 9. Content-gesture association (left) and recall/insertion (right) with a user-defined gesture.

Other than user-specified content gestures, the document editor can conceivably also include pre-defined system shortcuts provided by gesture libraries.

Clipart Retrieval with Query-by-Sketching

Another popular source of often used external resources when composing documents is a clipart database. For presentations, brochures and other illustrated documents, authors often seek to enrich their content with professionally designed graphics, art, symbolic drawings and illustrations. In addition to gesture-based content retrieval, we include a means to retrieve document elements via queries defined by sketches, which are sometimes more convenient to describe visual items with characteristic geometry (e.g. arrows, chart items, block shapes).

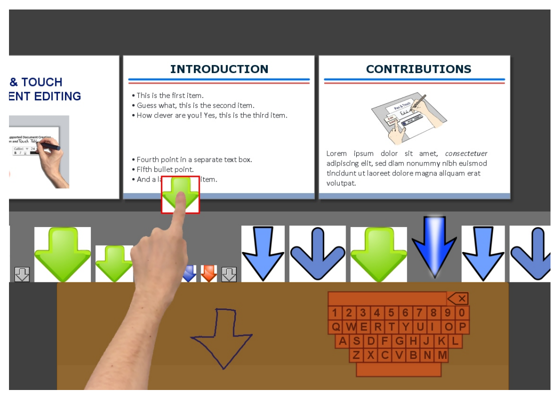


Figure 10. A user drags a selected clipart retrieved using a sketch-based query.

We implement this feature through a virtual "drawer" panel that can be pulled out from the bottom bezel of the interface. On this panel users can then draw sketches to perform queries using query-by-sketching [2] as well as enter search terms if keyword-matching is also desired. For instance, if a user would like to search for an arrow clipart in the database, they can sketch an arrow shape on the pad and additionally (or instead) enter "arrow" as a keyword to further specify the query (Figure 10). Keywords are input using a compact virtual keyboard, which is more straightforward for that purpose. We emphasise here that keywords are command parameters and not inked document content, which means they do not have to conform to the pen-on-paper metaphor and thus be handwritten.

Search results are displayed in a scrollable ribbon that appears above the query pad. Those results are updated after each pen stroke or key input. The user can then drag desired elements from the ribbon to insert them in opened documents on the workspace. We believe this method to retrieve and insert external content is relatively straightforward and efficient, but one could also consider alternatives, where query sketches are directly drawn at the desired insertion location on the target document page (as with our gesture-based content insertion technique) and results appear in place or next to the query.

LAB STUDY

Goals

To assess the pertinence of our approach, we conducted an informal lab study with 12 participants recruited among students and staff of our university (5 females and 7 males with one left-handed user). All our testers were familiar with touch interfaces thanks to smartphones or tablets they owned. They were however not used to operating a digital tabletop with simultaneous bimanual pen and touch input. Participants all knew how to use office software as part of their regular work, especially word processors and presentation programs, and so were able to formulate precise requirements and criticism regarding specific features of our system based on their experience.

Considering the breadth of interactions that needed to be demonstrated within the limited time of this first study, our interest was mainly to observe users and obtain qualitative feedback on the individual gestures after introducing them one after the other. When describing the rationale behind the NDH modifier postures, we explicitly mentioned the keyboard shortcut analogy, in order to find out to what extent users related the two concepts.

We did not expect participants to remember all of the techniques (much like people would most likely not memorise a plethora of keyboard shortcuts presented all at once) and hence we asked them to comment about the appropriateness of our gestures to perform a specific action, or informally "if they made sense" in the given context. After letting participants try the different interactions and hearing their remarks in a think aloud process, we allowed them to use the application freely and attempt to author their own documents using the different interactions they had been taught. We finally gathered general impressions about the application through brief interviews, where we also recorded suggestions about improvements that could be included in future iterations of the system.

A session with a participant lasted about an hour.

Results

We generally received very positive feedback about our system with many of our participants staying for extended periods of time to play around with our application. The overall feeling was that our gesture-function mappings were fitting and easy to execute. Popular features included our content retrieval techniques (both query-by-sketching and gesture-based insertion), the chopping interactions for text alignment and the circle-based undo/redo gesture. Regarding the parallel with keyboard shortcuts, four people commented that gestures were easier to remember since the mental association between a particular operation and a matching physical hand or pen action is a more effective mnemonic than a combination of keys. Those statements confirm previous results about the superior memorability of stroke-based vs. keyboard shortcuts [3].

Following is a summary of our main observations and participants' reactions with regard to some of the more salient features and aspects of our system.

Asymmetric Bimanual Interaction

A general observation that we made and that echoes the experience of other researchers [24, 28] is that users are heavily influenced by their interactional habits with devices they often use, especially with respect to unimanual/bimanual input. The vast majority of applications available for commercial touch devices are based either on single-handed interactions or symmetrical two-handed input, e.g. for typing and pinch/spread scaling. Hence, people are not used to asymmetric combinations involving two different modalities and are even sometimes wary of engaging with this new kind of interaction, even though productivity gains can be achieved. For instance, we occasionally saw users temporarily transferring the pen from their DH to their NDH in order to perform a unimanual touch interaction and returning the pen to their DH after they had finished. For most non-inking operations, however, our system forces people to use both hands simultaneously and while our study participants quickly came to grips with our modifier posture model, whose sequences of actions are consistent across all different modes, three of them were confused by locally modified pen interactions, especially those governing page-copying manipulations. They wondered why they had to bother placing several fingers on pages to move or duplicate them instead of simply selecting the elements first and then executing the desired operation, as determined by a global modifier posture or a button selection. One user deemed that because creating a new blank page was a more common action for him compared to copying the entire page with its content, the operation should have been assigned to the single-finger hold posture. This would however have broken the consistency with the duplicating interactions for other elements, which are based on the same gestural pattern.

Text Input and Styling

Participants did not experience major difficulties or express concerns about our text entry techniques, which we believe is due to the quality of our handwriting recogniser and the fact that most of our interactions mirror established behaviours of word processors and text-editing software. Even our somewhat unconventional method to insert handwritten content in the middle of typeset text did not cause particular inconveniences (our left-handed participant was particularly at ease, as in his case there was no occlusion of the shifting right portion of text), other than occasional misrecognitions of insertion points and text that was not properly appended to its neighbouring element (this happened to 3 users). We also registered incidental problems with underlining actions that were not perfectly performed below the text, resulting in hyphens being added in the text instead (3 occurrences as well).

As for the other text-styling options, users found it convenient to be able to summon the pop-up toolbar with touch and make individual selections with the pen. One person noted that setting several different options in a row by sliding an NDH finger on the styling categories to successively activate their submenus and then simply tap with the pen on the desired item was a quick and space-efficient method. Our NDH-triggered overlay elicited more mixed reactions, with four people explicitly criticising the cumbersomeness of simultaneously having two dragging directions to change two different types of parameters. This view was not shared by everybody, however, as two participants proved very dexterous at controlling text size and switching font styles with lateral taps and sliding actions.

Shape Input and Styling

Shape input also did not present any significant problems. Users were comfortable laying a flat NDH on the surface while sketching and drawing shapes. Switching modes to perform styling operations was also not an issue and the only mistakes we observed were two instances in which people tried to thicken/thin shape contours in Command Mode instead of doing so in Shape Mode.

Gesture Relaxation and Crosstalk

Our modifier posture model was designed to favour efficient mode switching with minimal mental and physical effort of the NDH and so we did not require the posture to be completely released in order to change to another mode. This meant that, for instance, a colour sampling action in Selection Mode could be immediately followed by a fill operation in Command Mode by simply releasing a finger. But while most participants appreciated this flexibility, four were confused by mode changes occurring when relaxing dragging gestures (i.e. when lifting fingers) initiated in Selection Mode. In particular, border colour changes associated with mode switches distracted those people while the styling overlay remained visible throughout the dragging actions. While we personally value mode-switching speed and the ability to perform smooth, effortless transitions, we recognise that this represents a potential conflict with relaxation policies and location-based gestures. In such cases more conservative gesture design approaches might be in order to appeal to a greater number of users.

Discussion

Our experiment reveals that many common document operations can be successfully executed using appropriate unimodal and bimodal gestures. The feedback we received from our study participants confirms that modifier postures associated with pen gestures are potent equivalents to keyboard shortcuts and that they are likely quicker to acquire than the latter [3]. In this iteration, gestures were directly shown to users to try out individually as a first test. We deliberately did not address the issue of discoverability or learnability that other authors have tackled and which we leave for future work. A possible idea would be to include demonstrative illustrations of available gestures for each mode within the gradient border instead of simply displaying a uniform shade. That way, users could simply cycle through the modes to find out which gestures exist.

Regardless of accessibility and ease of use, one must acknowledge that gestures require some initial effort that not all users are necessarily willing to invest. Our intention is not to show gesture-based interactions can entirely replace widgets and other traditional tools (especially for a feature-rich application such as a document editor), rather that they can be a useful additional asset for experts, much like keyboard shortcuts complement GUI controls of PC applications rather than replace them. Besides, as people are increasingly exposed to direct input interfaces (with growing use of simultaneous pen and touch) and those interfaces themselves continue to evolve, we believe users will become more familiar with hybrid interaction techniques, and within that context, bimodal gestural shortcuts stand a reasonable chance of gaining wider adoption. For that to happen, UI designers will have to create coherent and consistent gesture sets that neither overwhelm nor frustrate users. Compound interactions, in particular, should not consist of too many chunks resulting in exceedingly long phrases [5], especially if they require a great deal of bimanual coordination. A sequence "NDH-posture → DH/pen-gesture → bidirectional NDH-posture dragging" as in our text-styling technique is perhaps already too complex.

With regard to in situ handwritten text input and conversion, we think our technique naturally finds its place in a pen-based environment, where a smooth flow and seamless interleaving with other stylus interactions are desirable. While this would need to be formally confirmed by an appropriate test, we hypothesise that the throughput loss incurred by handwriting text compared to keyboard typing is compensated by productivity increases in other stylus manipulations, whose performance would otherwise be impacted by repeated context switches with the keyboard and it being "in the way". Our document-authoring scenario is one example, where we believe such an approach is justified (albeit, for particular categories of documents only) but there are doubtless many others.

CONCLUSION AND FUTURE WORK

In this paper, we presented a document-authoring environment for interactive tabletops, in which several common document-editing operations are supported by gestures instead of, or in addition to widgets. We showed how NDH postures can be systematically leveraged as effective modifiers to set different categories of pen functions and thereby function as hybrid shortcuts for a variety of document manipulations. A lab study confirmed the potential of this approach and provided insights about the practical challenges of its realisation.

From there on there are many avenues that can be pursued, considering the amount of functionality typically available in professional publishing software. We are particularly interested in looking next into table manipulation and templates, which we think could also greatly benefit from a multimodal gesture approach. Tables are essential components of office documents such as spreadsheets and forms and we feel there is much to explore in that area. As for document templates, we believe they can function as constraint structures that can help solve some of the ambiguities, which our application tries to resolve through algorithmic inference. For example, a constraint-defining template would allow text input in a title box to be automatically converted to the correct font and size.

In terms of evaluation, task-based experiments will be required to determine if users are able to effectively learn and utilise the different gestures to create or edit documents on their own. When our system has achieved sufficient robustness, it will be interesting to see how it could potentially be deployed in a real-world context, i.e. in a real office.

We hope our work will eventually help inform the design of future document-centric applications for pen and touch interactive tabletops.

REFERENCES

1. Agarawala, A. and Balakrishnan, R. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proc. CHI 2011*, 1283-1292.
2. Al Kabary, I. and Schuldt, H. SKETCHify - an Adaptive Prominent Edge Detection Algorithm for Optimized Query-by-Sketch Image Retrieval. In *Proc. AMR 2012*.
3. Appert, C. and Zhai, S. Using strokes as command shortcuts: cognitive benefits and toolkit support. In *Proc. CHI 2009*, 2289-2298.
4. Brandl, P., Forlines, C., Wigdor, D., Haller, M. and Shen, C. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proc. AVI 2008*, 154-161.
5. Buxton, W. Chunking and Phrasing and the Design of Human-Computer Dialogues. In *Proc. IFIP Congress 1986*, 475-480.
6. Dietz, P. and Leigh, D. DiamondTouch: a multi-user touch technology. In *Proc. UIST 2001*, 219-226.
7. Frisch, M., Heydekorn, J. and Dachselt, R. Diagram editing on interactive displays using multi-touch and pen gestures. In *Proc. Diagrams 2010*, 182-196.
8. Frisch, M., Langner, R. and Dachselt, R. Neat: a set of flexible tools and gestures for layout tasks on interactive displays. In *Proc. ITS 2011*, 1-10.
9. Guiard, Y. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of Motor Behavior*, *19*, 4 (1987), 486-517.
10. Hamilton, W., Kerne, A. and Robbins, T. High-performance pen + touch modality interactions: a real-time strategy game eSports context. In *Proc. UIST 2012*, 309-318.
11. Hinckley, K., Bi, X., Pahud, M. and Buxton, B. Informal information gathering techniques for active reading. In *Proc. CHI 2012*, 1893-1896.
12. Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H. and Buxton, B. Pen + touch = new tools. In *Proc. UIST 2010*, 27-36.
13. Jacob, R. J. K., Girouard, A., Hirshfield, L. M., Horn, M. S., Shaer, O., Solovey, E. T. and Zigelbaum, J. Reality-based interaction: a framework for post-WIMP interfaces. In *Proc. CHI 2008*, 201-210.
14. Li, Y., Landay, J. A., Guan, Z., Ren, X. and Dai, G. Sketching informal presentations. In *Proc. ICMI 2003*, 234-241.
15. Matulic, F. SmartPublisher: document creation on pen-based systems via document element reuse. In *Proc. DocEng 2006*, 182-183.
16. Matulic, F. and Norrie, M. Empirical evaluation of uni- and bimodal pen and touch interaction properties on digital tabletops. In *Proc. ITS 2012*, 143-152.
17. Matulic, F. and Norrie, M. C. Supporting active reading on pen and touch-operated tabletops. In *Proc. AVI 2012*, 612-619.
18. Matulic, F., Norrie, M. C., Kabary, I. A. and Schuldt, H. Gesture-supported document creation on pen and touch tabletops. In *Proc. CHI EA 2013*, 1191-1196.
19. Moscovich, T. and Hughes, J. F. Navigating documents with the virtual scroll ring. In *Proc. UIST 2004*, 57-60.
20. Raskin, J. *The humane interface: new directions for designing interactive systems*. Addison-Wesley, 2000.
21. Schilit, B. N., Golovchinsky, G. and Price, M. N. Beyond paper: supporting active reading with free form digital ink annotations. In *Proc. CHI 1998*, 249-256.
22. Sellen, A. J., Kurtenbach, G. P. and Buxton, W. A. S. The prevention of mode errors through sensory feedback. *Human-Computer Interaction*, *7*, 2 (1992), 141-164.
23. Tashman, C. S. and Edwards, W. K. LiquidText: a flexible, multitouch environment to support active reading. In *Proc. CHI 2011*, 3285-3294.
24. Walny, J., Lee, B., Johns, P., Riche, N. H. and Carpendale, S. Understanding Pen and Touch Interaction for Data Exploration on Interactive Whiteboards. In *TVCG*, *18*, 12 (2012), 2779-2788.
25. Wigdor, D., Benko, H., Pella, J., Lombardo, J. and Williams, S. Rock & rails: extending multi-touch interactions with shape gestures to enable precise spatial manipulations. In *Proc. CHI 2011*, 1581-1590.
26. Wobbrock, J. O., Wilson, A. D. and Li, Y. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proc. UIST 2007*, 159-168.
27. Wu, M., Chia, S., Ryall, K., Forlines, C. and Balakrishnan, R. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In *Proc. TABLETOP 2006*, 185-192.
28. Zeleznik, R., Bragdon, A., Adeputra, F. and Ko, H.-S. Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving. In *Proc. UIST 2010*, 17-26.