InkSeine: *In Situ* Search for Active Note Taking

Ken Hinckley[1], Shengdong Zhao[2], Raman Sarin[1], Patrick Baudisch[1], Ed Cutrell[1], Michael Shilman[3], Desney Tan[1]
Microsoft Research, One Microsoft Way, Redmond, WA 98052
[1]{kenh, ramans, baudisch, cutrell, desney}@microsoft.com,[2]sszhao@dgp.toronto.edu,[3]michael@shilman.net

ABSTRACT
Using a notebook to sketch designs, reflect on a topic, or capture and extend creative ideas are examples of *active note taking* tasks. Optimal experience for such tasks demands concentration without interruption. Yet active note taking may also require reference documents or emails from team members. InkSeine is a Tablet PC application that supports active note taking by coupling a pen-and-ink interface with an *in situ* search facility that flows directly from a user's ink notes (Fig. 1). InkSeine integrates four key concepts: it leverages preexisting ink to initiate a search; it provides tight coupling of search queries with application content; it persists search queries as first class objects that can be commingled with ink notes; and it enables a quick and flexible workflow where the user may freely interleave inking, searching, and gathering content. InkSeine offers these capabilities in an interface that is tailored to the unique demands of pen input, and that maintains the primacy of inking above all other tasks.

Author Keywords
Input, pen, tablet, gestures, handwriting, ink, search

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation: Input

INTRODUCTION
The convergence of pen computers, full text indexing of personal stores, and internet search engines offers a unique opportunity to design fluid interfaces for *active note taking*. Active note taking is the combination of pen-and-ink note taking with searching, linking, collecting, and sensemaking activities [26]. This is in contrast to "simple" note taking, which is characterized by moment-to-moment transcription. Active note taking is commonly performed by people engaged in challenging creative work such as scientific research and product design [4]. These persons often create informal preproduction artifacts on paper, in notebooks, or on whiteboards [20,22], sketching preliminary plans and manipulating their notes to find solutions to hard problems.

Kirsh observes that people tend to surround themselves with "task detritus" to help trigger extensions, variations, and associations on ideas [15]. For example, messy physical desks subtly structure information and remind users of work to do [14]. Virtual 3D desktops such as DataMountain [25], or the fluid pen-based interaction of BumpTop [1], enable users to create personally meaningful arrangements by piling together document thumbnails. Likewise, an unstructured notebook page imposes no formalism [20]; users can manipulate the spatial arrangement of information in order to assist sensemaking.

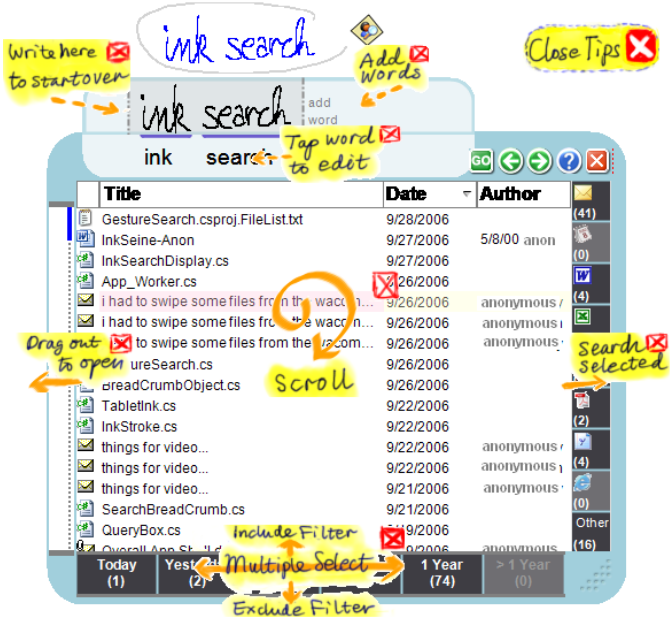

Fig. 1. The user lassos "ink search" to trigger search of his personal store. InkSeine recognizes the handwriting and lists result documents containing the words. InkSeine uses orange marks to illustrate sample gestures, with highlighted explanations.

Interfaces designed to support creative sensemaking tasks [26] should help users "stay in the flow" of focused attention [3,6]. We adopt this perspective for pen-enabled computing interfaces for active note taking. We describe InkSeine [1], a TabletPC application that offers rapid, minimally distracting interactions for users to seek, gather, and manipulate the "task detritus" of electronic work (links to documents, clippings from web pages, or key emails on a topic) across multiple notebook pages. Search offers a facile means to assemble such collages of notes, documents, and bitmaps (Fig. 2) while keeping the user engrossed in the inking experience as much as possible.

[1] The name "InkSeine" is a play on words. Fishing for information is a metaphor for search. To seine is to fish with a net; to fish for search results from ink is not insane, but rather ink-seine!

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CHI 2007, April 28–May 3, 2007, San Jose, California, USA.
Copyright 2007 ACM 978-1-59593-593-9/07/0004...$5.00.

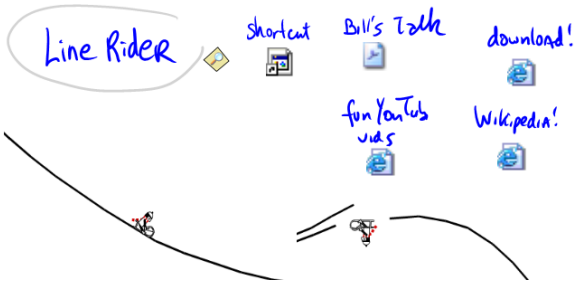

Fig. 2. An example InkSeine note. From a query on "Line Rider" the author has gathered together and annotated documents and web pages, as well as some screen clippings captured from the game.

InkSeine contributes the design and discussion of usability considerations for the tight integration of search with pen based active note taking. We call this tight integration of search with the user's task *in situ search*. In situ search is not new for textual queries [10,11,12,31], but we apply this strategy to ink queries and extend it by making queries persistent, first-class objects. We also discuss the merits of *in situ* search for enabling users to ink, search, and then gather content from search results back into their ink notes.

High-Level Design Properties of *in situ* Search

Embedding search within the user's task context enables users to find content without switching to a "search application," which would derail users from the creative flow of active note taking, particularly on a tablet where screen real estate is limited and text entry is slow. *In situ* search also naturally affords several key design properties:

Leverages preexisting ink to initiate search. Users do not have to tediously transcribe search terms to a separate query box. Users can recycle the effort of writing notes by triggering a query from ink that already exists on the page. This reduces the cognitive barrier between creating ink on the page and creating queries based on that ink.

Promotes queries as first class objects that are commingled with ink notes. Users can quickly capture their intent to search by lassoing a phrase, but then defer opening the search results until later, when time and attention permit. The resulting queries are represented by a persistent "breadcrumb" (Fig. 3) that is saved in-place with the notes.

Fig. 3. Breadcrumb. **Left:** The user presses a button to enter gesture mode [18] and lassos some ink. The breadcrumb icon appears on pen-up. **Right:** The user can later stroke down to open Personal search results; on pen-up, the results appear (Fig. 1).

Interleaves inking, searching, and gathering. Because there is no artificial barrier between the ink notes and the search results, the user may freely move back and forth between the two. The compact search UI leaves room for taking notes and gathering documents or clippings. The user can trigger a search, browse the results, make new notes in the margin, start fresh searches for side-by-side comparison, or defer an ongoing search in favor of some other activity.

Tightly couples queries with application content. The breadcrumb allows queries to be commingled with ink, but the user can also drag search results directly into his notes (Fig. 4, step #4) to create ad-hoc arrangements of useful documents (Fig. 2). The user also may pull part of a document into his notes by taking a snapshot of it.

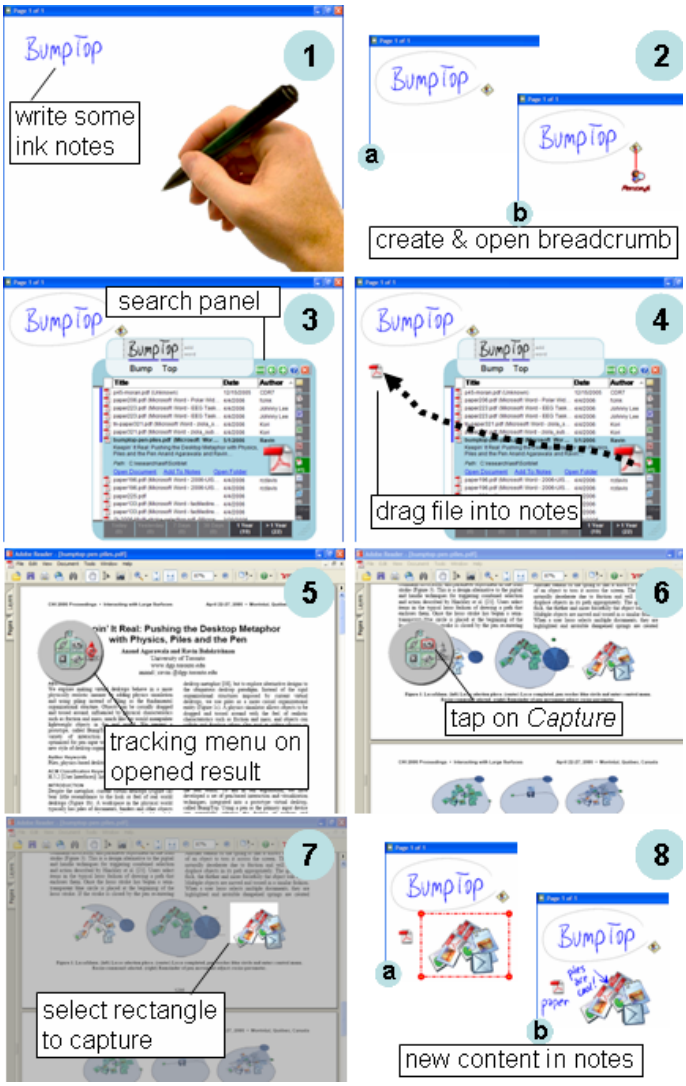

Fig. 4. InkSeine walkthrough. See text below.

Short Walkthrough of an InkSeine Usage Scenario

Fig. 4 shows an abbreviated walkthrough of InkSeine for a scenario where the user lassos ink notes to search for a document in his personal store, drags an icon representing the document into his notes, and then opens the document and takes a screen capture of a figure contained therein. The interaction techniques and rationale for the design will be explained in more detail in later sections.

In step **1**, the user inks "BumpTop" in his electronic notebook, and then decides he needs the PDF document for this CHI paper [1]. In **2a** the user lassos the ink to create a breadcrumb; in **2b** the user then opens the breadcrumb (see also Fig. 3). Since the breadcrumb is persistent, this could have been opened either immediately, or some time after it was created. Opening it brings up the results (step **3**) and the user taps the PDF filter to narrow down the results set and locate the desired document. At step **4** the user drags the PDF icon into his notebook for future reference. The user then decides to grab a piece of content from this PDF document. He employs the document's icon to open the PDF and a circular tracking menu appears (step **5**) with some pen-operated controls for manipulating the underlying document. He touches the pen down in the outer ring of the tracking menu and circles with the pen to scroll through the document. When he sees the desired content, he taps the *Capture* icon (step **6**) in the tracking menu. This places a transparent gray overlay on the screen and he sweeps out a rectangular selection region (step **7**). The user hits the *Close* icon in the tracking menu to close the PDF and return to InkSeine. The clipping appears in the user's notes (**8a**), and the user draws several ink annotations (**8b**) that refer to it.

RELATED WORK

The vision of creating a notebook with electronic ink has a long history going back to systems such as the Apple Newton and PenPoint [5]. The Tablet PC is the latest incarnation of this vision, yet it remains unclear if it can provide net value over paper notebooks, or whether augmenting paper with digital capabilities (e.g. [29]) might be the way to go. Tight integration of search may add significant value to electronic note-taking because it encourages a cycle of inking, searching, and gathering content into one's ink notes. This allows the creation of rich mixed-media notebooks that resemble the design collages and paper notebooks documented by Buxton [4].

XLibris [23,27] is a pen-based system for annotating electronic texts during *active reading*, which is the combination of reading with critical thinking and learning. XLibris offers implicit searches [11] using *text* that the user has underlined or otherwise marked. The system places the thumbnails of related documents found by these implicit searches in the margins. While implicit searches may sometimes yield serendipitous associations, they also could distract the user. For example, Bederson argues that optimal experience [6] for creative work requires interfaces that help users "stay in the flow" of focused attention [3]. To help achieve flow, InkSeine leaves the user in control by providing results only in response to lightweight, but explicit, gestural interactions.

SketchTrieve [12] is a graphic editor that allows users to construct and document a series of searches by using drag-and-drop to wire text queries to information retrieval services. Users can also drag search results into the workspace. SketchTrieve supports a retrieval-centered interaction model as advocated by Rao et al. in the InfoGrid system [24]. By contrast, InkSeine is not a workspace *for search*, but rather offers a freeform workspace with search as a secondary task *in support of* active note taking.

The NaviQue information environment [10] has a zoomable canvas where the user may type textual annotations at any location. By design NaviQue has no separate search window or query box; anything on the canvas can be selected and used as a query. InkSeine adopts this in-place strategy for triggering searches and explores how it can be realized for pen and ink interfaces.

InkSeine demonstrates the importance of an *in situ* strategy for ink search. The best prior example of *in situ* search that we are aware of is Yahoo! Messenger's LiveWords feature [31]. The user drags the mouse to select words in an Instant Messaging conversation, which causes a magnifying class icon to appear. When the user clicks on the icon, a new window appears with search results. This icon cannot be copied, pasted, or saved, and it ceases to exist as soon as user selects something else with the mouse. The key contribution of InkSeine's breadcrumb mechanism with respect to LiveWords is that the breadcrumb is a persistent, first-class representation of a query that is saved with the notes; the user can revisit it hours, days, or weeks later in its original surround of ink notes.

Some prior efforts have used tagging to retrieve ink notes. NotePals [8] is a note-sharing system that allows users to categorize notes in a hierarchy. Users can retrieve notes by category as well as author, subject, or date. Dynomite [30] uses explicit tagging of ink with properties and textual words to enable content indexing. InkSeine instead provides rich search and retrieval facilities, without the need for user tagging, by searching directly from ink.

We implemented InkSeine's search features using publicly available components of Phlat [7], a system that allows users to search for information from their personal store. Like Phlat, InkSeine leverages metadata such as document title, author, and date as memory landmarks that can help users to sort and filter result sets from their personal store.

EARLY-STAGE EVALUATION AND PAPER PROTOTYPE

Early in our design process, we wanted to convince ourselves whether or not users perceived value in an ink-based search facility. We also wanted to better understand what aspects of existing desktop search tools might fail to meet the needs of pen-based active note taking.

We recruited 12 computer science graduate students (9 male, 3 female, ages 25-35, all working at Microsoft as interns) to participate in user studies. Five participated in the early-stage evaluations reported in this section, while the other 7 participated in later rounds of iterative usability testing reported later in this paper. We feel that graduate students are representative of our intended user population because they are involved in challenging creative work.

In the first early-stage evaluation, the 5 users began by writing down some ink notes about their research using a

Tablet PC running Windows Journal. Once users had produced about one page of notes, we asked them to comment on the note taking process, and then asked if they would want to perform searches based on any of the words they had written. All users felt that searches would be useful when taking notes, and suggested scenarios such as searching to find background information, to recall project details, or to look up people from a meeting.

We next asked users to employ the pen to perform searches within the Phlat [7] desktop search tool (available at http://research.microsoft.com/adapt/phlat). Users found it tedious to enter a text query with the pen. Users commented that searching with Phlat requires an "extra step" of switching to the search application. They also disliked switching between the search application and the page of notes when copying information between the two.

One user wanted to highlight or lasso the search terms in the ink notes, and trigger the search from there. Another user commented that during note taking "attention is distracted" and she might not have time to examine the search results. She suggested she would want to save a query in context to keep a record of the thought to search. A similar idea was raised by a second user. These suggestions led to breadcrumbs (Fig. 3). Users liked the idea of dragging icons from the results into their notes, but two users commented that they often desired only a *piece* of information within a document. This led us to realize the need for opening result documents, scrolling through documents to find desired information, and then bringing that information back into the page of notes.

Finally, each user participated in a paper prototyping session where we provided cut-outs of interface elements (Fig. 5). Users chose individual cut-outs and placed them on the page of notes. Users valued seeing their notes side-by-side with search results, and thus avoided large results lists. Users felt it would be hard to tell the desired document from the title only, and wanted to see author, date, and a text snippet for the result they were considering.

Fig. 5. Physical set-up for paper prototyping.

These early explorations confirmed that existing desktop search tools do not meet the needs of users for ink-based search. Entering text with a pen is difficult, switching contexts is distracting, and gathering information from documents is tedious. User suggestions reinforced some of our early design ideas (lasso ink to trigger a search, keep the search results compact enough to view in the context of surrounding notes). But more importantly, they also led us to consider new aspects of the task workflow that our early designs had overlooked, such as "saving" queries in context for later viewing, opening and scrolling through result documents, and grabbing clippings from documents.

INKSEINE DESIGN GOALS

Given observations from our early-stage studies, we formulated a number of design goals for a search user experience that maintains flow during active note taking.

In situ search experience. Enable information needs to be met without departing from the pen-and-ink user experience. The design should not require the user to switch to a search application, shift attention to a separate query box, or force transcription or translation of ink to text.

Optimum workflow / maximum flexibility. To support the workflow, the interface should always allow the user to return to the primary active note taking task with at most a single pen tap. It should also maintain high flexibility to support workflow with a fluid evolution of steps in pursuit of search results and content, rather than a regimented schema to which the user must adhere. To allow users to "interrupt themselves" at any point, the design must allow the user to defer a search, and later rapidly resume any search from where it left off. Finally, the workflow should optimize high-value searches that users are most likely to pursue when engaged in active note taking.

Enable rich tradeoffs in the cost structure of sensemaking [26] so the user remains in full control of how much time and attention gets expended on the search task. InkSeine seeks to satisfy queries where, informally, the rough time cost of searching might range from:

- Record the thought to do a search (1 second)
- Trigger a search and see a results list (5 seconds)
- Open a familiar document from a small result set, drag out its icon, or capture a snapshot from the front page of the document (10-20 seconds)
- Scroll through a larger result set, apply filters, or revise the query (20s – 2 min)
- Inspect unfamiliar documents to assess if they meet information needs, or segue into "sideways searches" based on document title, author, etc. (several minutes).

Thus our design must support choices at the low end of this approximate scale of time costs while also offering enough depth to allow users to meet common information needs.

Gather content. Close the search loop by allowing users to gather beneficial "task detritus" that they find in their searches, and incorporate it directly into their ink notes.

Minimize search screen real estate. To keep search as a secondary task the result list must be significantly smaller than the tablet's screen. This keeps the margins usable for

inking, minimizes unnecessary hand and eye movement, and affords dragging content from the search into the notes. It also encourages "side-by-side-ability" [12] by making it possible to launch, view, and compare multiple searches.

Span application boundaries. Users may want information from a wide variety of source documents (such as email, ink notes, or spreadsheets). It is impractical for a lightweight search facility to present interactive content from any source, and may prevent users from leveraging specialized host application features. The design should enable access to information from a variety of sources, but provide seamless pen-operated mechanisms for common functionality such as opening documents, scrolling, or taking snapshots of any content that is visible on the screen.

Tailored to pen input. As with other pen-input interfaces, the design should minimize the modes and gestures to remember, favor rapid, predictable, habit-forming interaction mechanisms that minimize distraction and visual diversion, and employ eyes-free gestures where possible. The design should reveal itself so that users can discover the primary search features without prior instruction.

INKSEINE USER INTERFACE AND WORKFLOW

InkSeine's primary work surface is an electronic notebook that allows users to jot ink notes on a series of pages. Thus, all of the search facilities that are the focus of this paper are designed in support of the inking task itself.

Electronic Notebook Controls

By default, all pen strokes on the page are treated as ink. To interact with the notebook, for example to insert a new page, a user holds down a non-preferred hand button [18], or the stylus button, while drawing a gesture (Fig. 6). These gestures are based on prior work [1,13,16]. Note that the user does not have to hold down the gesture button when using the pen to interact with the breadcrumb (Fig. 3), icons (Fig. 6, center), or the search panel (Fig. 1).

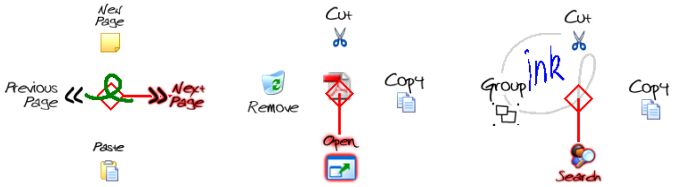

Fig. 6. Left: A pigtail gesture over empty space brings up a menu for the notebook. **Center:** A stroke starting on an icon opens a marking menu. **Right:** A pigtail at the end of a lasso activates a context menu.

Breadcrumbs

A search breadcrumb is a visible and persistent representation of a query that is attached to the ink that triggered the query (Fig. 3). To create a breadcrumb, the user draws an *incomplete lasso*, i.e. a lasso that does not intersect itself, around the desired ink. This creates a breadcrumb to be visited at a later time. Thus in about one second [13,19] and without interrupting the flow of a note taking task, the user can lasso some ink to specify a search. We believe this ability to capture the initial thought to perform a search with extremely low overhead, coupled with the ability to defer opening the search results until later, are important properties that make search a valuable adjunct to active note taking.

When the user does decide to revisit a breadcrumb, he can hover over the breadcrumb to see details (Fig. 7), or stroke down to open *personal search* results (Fig. 3). The user can also perform *web search* by drawing a stroke to the right from the breadcrumb. The user may fluidly flip a query back and forth between web and personal search.

Breadcrumbs are first-class objects that can be cut, copied, pasted, or selected and moved around. Unlike previous examples of search integrated with applications [31], breadcrumbs are persistent objects that remain with the user's notes. Breadcrumbs can thus serve as reminders to revisit previous queries, and implicitly record a history of queries in the context of the notes that led to the search, even if a note is reviewed long after its initial creation.

As stated above, an incomplete lasso creates the breadcrumb. To select ink *without* creating a breadcrumb, the user can draw a lasso that meets itself. To perform a command on the selection (such as cut, copy, group, etc.) the user draws a pigtail gesture [13] to bring up menu options (Fig. 6, right). InkSeine offers *Search* as one of the commands from the pigtail menu so that there is a shortcut to specify the query and open the search results in a single stroke. This serves task flows where the user immediately needs a search result before he can continue his task.

We use the incomplete lasso to serve the minimal "capture thought to search" workflow in order to reduce the time cost of this transaction, since drawing a lasso is faster than drawing a lasso plus a pigtail. However, the same basic interaction techniques give the user several options at the lowest rungs in our scale of time costs, thus serving our *Enable rich tradeoffs* and *Optimum workflow* design goals.

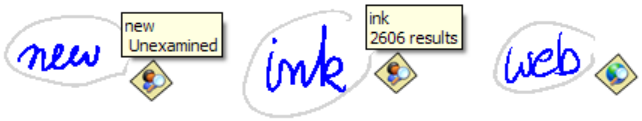

Fig. 7. Breadcrumbs provide information about the searches they represent. **Left**: A query the user has never opened is "unexamined." **Center**: A query that has been opened shows the number of results. **Right**: Web searches display a different breadcrumb.

Search Panel

When the user opens search results, a "search panel" appears. The search panel and most of the gestures it supports are shown in Fig. 1. Per our *Minimize search real estate* design goal, and our desire for an *In situ search experience* that allows simultaneous viewing of notes and search results, we kept the search panel as compact as possible. The search panel results list, widgets, and a few gestures allow the user to scroll through the results, apply filters, modify the query, and inspect result documents.

Multiple search panels can be opened simultaneously; each is an independent object with its own state, including query terms, filters, etc. Search panels persist their state when they are closed, and reinstate it when opened. This makes it easy for users to defer searches, and quickly pick up where they left off, thus supporting our *Optimum workflow / maximum flexibility* design goal.

Primary Components of the Search Panel

The *query area* (Fig. 8) at the top of the search panel shows the handwritten query, the recognized text, and action buttons. The main design element of the search panel is the results list. This list is surrounded on the bottom and right by timeline and file type filters that may be used to filter results. We describe each of these components below.

Query Area and Action Buttons

The query is initially seeded with the lasso-selected ink; the user *does not* have to transcribe any ink. The ink is scaled to fit within the query area.

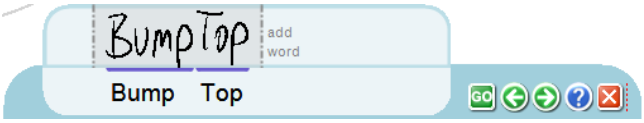

Fig. 8. Query area and action buttons.

InkSeine immediately starts a search based on the assumption that the ink has been recognized correctly. In the great majority of queries, recognition succeeds and the user can ignore the query area altogether, other than to perhaps verify that the handwriting recognition is correct in cases where the expected results do not appear. We strip non-alphanumeric characters from the text before issuing the query, as ink notes often contain extraneous punctuation the user does not intend as part of the query itself.

The query area unifies handwriting recognition correction and query refinement and gives users flexibility in fixing "errors," whether they result from messy handwriting or ill-conceived queries. For correction of the handwriting recognition, we use the CueTIP recognition and correction user interface [28]. The textual recognition of the handwriting appears immediately below the corresponding ink words. The user can correct segmentation errors by stroking on the purple segmentation bars. Tapping on the recognized text enables character-by-character editing. Typically, this is only required for out-of-vocabulary words. InkSeine uses the indexer to append all words in the user's document corpus to the list of words accepted by the recognizer, so this is rarely necessary.

InkSeine augments CueTIP [28] with features for query revision. The user can add terms to the query by writing in the blank space to the right of the initial ink used to trigger the search ("add word" appears when the pen hovers over this area). The user can then tap the *Go* button, or InkSeine automatically refreshes the query results two seconds after the pen is lifted. Unlike incremental search when typing text strings [3], "instant" update for search during handwriting is challenging because accurate handwriting recognition results cannot be obtained until the user has finished writing an entire word. Thus, in this context it is better to wait until the user has finished writing before reissuing the query. The user can also delete words from their ink query by scratching them out.

The user may start an entirely new query by simply writing over the existing ink starting at the far left. If the handwriting recognition is incorrect, it is often easier to rewrite the words than it is to perform detailed repairs to the recognition. This is also useful if the user realizes that the query itself needs to be scrapped to find a desired result.

The design does not let users add new ink on top of the existing ink, nor does it reformat selected ink that appears rotated or on multiple lines. Well understood algorithms exist for linearizing words in a lasso selection (for example, Windows Journal's "Convert handwriting to text" feature).

The action buttons include the *Go* button mentioned above. The forward and back arrows allow for undo and redo. The question mark brings up help (Fig. 1, discussed below). Close dismisses the search panel, but persists its current state so that it may be recalled using the breadcrumb.

Results List

The results list displays a one-line summary of each result with a file type icon, document title, date the document was created or revised, and the author of the document. By default, all searches initially sort the results by date since time has been shown to be a powerful filter for memory of events [7]. As with standard list box interfaces, the user can tap on the heading for any column to sort the results.

The Focus Result

The user taps on a result to make it the focus. The focus result displays extra information including a text snippet of the first two lines of the document, contextual hyperlinks that offer actions for that document, and a thumbnail of the document (Fig. 9). Per our *Minimize search screen real estate* design goal, there is only enough space to provide these details and extra options for one result.

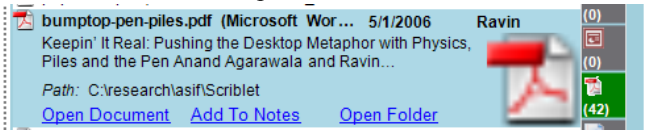

Fig. 9. Focus result with extra details and options.

A user can gather result documents by dragging the document icon out of the search panel and directly into the surrounding note page. Icons that have been dragged into the notes may be used to quickly re-access the document (Fig. 6, center). Clicking the Open Document link opens the document using its host application, and also brings up a semi-transparent tracking menu [9] (see Fig. 10) to support common pen-based cross-application interactions per our *Span application boundaries* design goal. The Add To Notes link pastes the document's icon into the upper-left corner of the page of notes. The Open Folder link opens a file folder that shows all the files co-located with the focus.

Results List Gestures

The primary gestures supported by the search panel appear in Fig. 1 and can be seen in our accompanying video figure. In accordance with our *Tailored to pen input* design goal, we provide a scroll ring [21] for scrolling through the results list. Making a circling motion in the results list invokes the scroll ring. Large, fast circling movements result in a fast scrolling motions, while small, controlled circling movements result in slow scrolling. The scroll ring maps circular pen motions to linear scrolling by fitting a circle to the last few samples of pen motion. Since the "circle" has no fixed center [21], it provides eyes-free scrolling that allows the user to focus on the search results.

We extend the scroll ring to support horizontal as well as vertical dragging by using a lazy evaluation strategy in which the scroll ring is treated as the "dominant" gesture. For example, the user can drag vertically to quickly scrub the focus (Fig. 9) across a series of results in the list. But if the user starts circling instead, our algorithm reverts to interpreting the input as a scroll ring. Once the scroll ring interpretation is attached to a pen stroke, the system stays in this mode until pen-up.

Making a horizontal stroke to the left is a quick gesture to immediately open any result document. This is equivalent to first tapping on a result (to make it the focus result) and then activating the Open Document hyperlink. The latter path is more obvious for beginners, but is slower, particularly if a user is certain that a result is the desired one just from seeing the one-line summary information.

Making a horizontal stroke to the right is used to launch a "sideways search" based on the title, date, or author of an individual document. For example, the user can drag right from the author field to find more results from the same author. Recognition of the horizontal left/right strokes does not occur until the pen-up event occurs; if the user just happens to invoke the scroll ring via an initial movement near the edges of the list view that is nearly horizontal, it will not accidentally invoke an action.

File Type and Timeline Filters

File type filters form a column along the right edge of the search panel (Fig. 1). The user may tap or cross any filter to activate it. Tapping on the PDF file type filter, for example, filters the result set so that only PDF files are shown. If a document type does not appear in the results list, the corresponding file type filter icon is shown grayed-out so that the user can quickly see which file types do or do not occur in the results. To filter by multiple file types, the user can cross several in a single stroke [2], for example by crossing the Word, Excel, and PDF filters to focus the results on common document formats. The user may avoid a filter by sliding the pen around it. To exclude unwanted file types from the result set, e.g. emails and calendar items, the user can cross the filter icons in succession, and then exit the filter strip to the right, away from the results list. This removes emails and calendar items from the results.

Timeline filters are located at the bottom of the search panel (Fig. 1). These enable the user to filter the results by date. The timeline includes corresponding gestures to narrow results to a particular range of time (e.g. crossing *Today*, *Yesterday*, and *7 Days* in succession to focus the results list on recent results). These quick filtering gestures help users narrow down lengthy result lists to a manageable size.

The timeline appears below the results list since the user typically wants to glance at the results before deciding if it is necessary to filter them further. We avoided placing both types of filters on the same edge of the search panel because we did not want to imply any ordering of which filter the user should apply first, and because users in the paper prototyping sessions seemed to prefer this separation. The timeline and file type filters use metadata that is only available from personal information searches, so the filters are inactive for Web searches.

Document Icons

Document icons provide shortcuts to result documents that can be included in notes, annotated, copied and pasted, and organized into sets of useful documents for subsequent access. These icons display the corresponding document title when the user hovers over the icon. We considered permanent labels, but design mock-ups suggested these would quickly clutter the screen. Nonetheless, this might be useful as a user-selectable option. A pen stroke on an icon activates a marking menu with options to Open, Remove, Cut, or Copy the document (Fig. 6, center). *Open* makes it extremely fast to revisit previously discovered result documents that have been dragged onto a page of notes.

Tracking Menu

We extend tracking menus to afford pen interaction with legacy mouse-and-keyboard applications [9]. Our tracking menu currently supports scrolling documents with a scroll ring gesture and a capture tool for grabbing clippings from a document. Our walkthrough (Fig. 4) shows full-screen views of the tracking menu in action, while the figures below provide detail views (Figs. 10-12).

A pen-down in the outer ring of the tracking menu invokes the scroll ring [21] on the underlying application window. Once the user acquires the outer ring, the natural circling movement allows the user to focus visual attention on the content scrolling by. This removes the need to interact with scroll bars, which can be difficult to use with a pen.

Fig. 10. Tracking menu for document functions. **Left**: inactive state when pen is not over the menu. **Right**: active state with pen over the outer ring ("scroll").

If the user finds some desired content while scrolling through the document, he can tap on the *Capture* button in the tracking menu to trigger a screen capture mode similar

to that offered by the Microsoft OneNote *Insert Screen Clipping* feature. We apply a transparent gray mask to the entire screen. The user can then sweep out a rectangle to indicate the desired content (Fig. 11). When the user lifts the pen after sweeping out the capture region, the gray mask disappears and the tracking menu reappears. The user may employ the tracking menu to further scroll through the document or take more clippings.

We do not offer the *Capture* command directly from the InkSeine search panel because the user does not know if the desired content has been found until they have opened a result document and scrolled to the appropriate place. Only at that point in the search workflow is the user ready to decide that a clipping is warranted. Our tracking menu also includes buttons for *Add Link* and *horizontal scrolling*, but we have not yet implemented these functions.

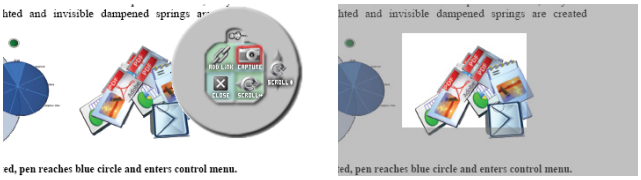

Fig. 11. Using the tracking menu to capture piece of a document. **Left**: The user taps on the Capture button. **Right**: A gray mask is applied to the entire screen and the user sweeps out the desired clipping.

Like the tracking menus described in [9], we enable the user to pin the tracking menu in a fixed position so that it does not follow the pen. This enables the user to interact directly with the host application if desired, simply by putting pen to screen anywhere beyond the outer limit of the tracking menu. By default, InkSeine initially pins the tracking menu in the upper right corner of the screen.

Returning to InkSeine

When the user is finished with a document, tapping the tracking menu's *Close* button dismisses both tracking menu and document, and returns the window focus to InkSeine.

The last clipping is automatically selected to allow quick movement and resizing, or the user can immediately start writing to annotate the new content (Fig. 12).

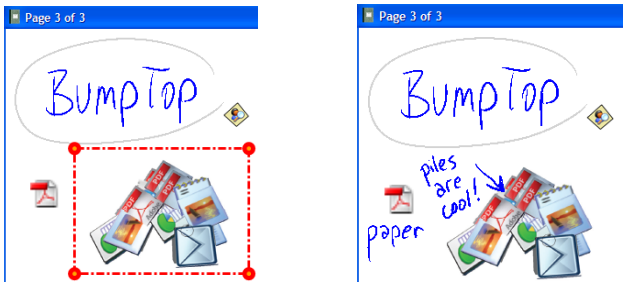

Fig. 12. Left: The clipping is placed in the notes and the user can move, resize, or annotate it. **Right:** The user can immediately annotate the gathered content.

Highlighter Hints for Self-Revelation of Gestures

Gesture interfaces, crossing widgets, and circular scrolling are not familiar metaphors to most users. We have observed that users' prior experience with point-and-click interfaces leads to tapping gestures as being the most obvious thing for new users to try with pen interfaces. As a result, during early tests of search panel prototypes, users were not able to discover any of the stroking or crossing gestures, or the scroll ring, on their own. Even if users were told about the gestures, they would often struggle with them since they had no clear mental model of what it was that they were supposed to do. We first tried tooltips that appeared on pen hover, but these were confusing since they did not appear until the user had already committed to set pen to screen. Furthermore, the user's hand would sometimes block these hover tooltips from view, and users sometimes thought that they were pop-up buttons that they should tap on.

Hence we designed gestural highlighter hints, which appear by default when the user first opens the search panel (Fig. 1). The highlighter hints are stylistically distinct from the underlying search results as well as the user's own ink content. The hand-printed labels for functions are surrounded by a bright yellow glow, and the gestures appear as if drawn with a thick orange chisel marker.

These highlighter hints allow the user to see the full set of actions available, with his hand pulled away from the screen so that nothing is occluded. The orange highlighter strokes also clearly indicate the direction and shape of the gestures to be drawn. The user can easily dismiss the highlighter hints by tapping on *Close Tips*, which causes them to animate into the question-mark icon (Fig. 8). This also reveals to users how to recall them, and thus allows a user to incrementally explore the different gestures at the point when he is receptive and ready to act on the hints.

ITERATIVE USABILITY TESTS

In addition to the 5 users who participated in our formative evaluation sessions, we conducted two additional rounds of iterative usability evaluations. Four users participated in the first round, which was conducted as soon as we had enough of the interactive behaviors working to obtain useful feedback from users. In this first round, we had not yet developed the highlighter hints mechanism. Three users participated in a second round of testing with a substantially improved design, including highlighter hints.

First Round Observations

Primary Usability Problems and Resulting Improvements

In our first round, we observed that discovery of gestures was a big problem. Self-revelation of pen gestures is a longstanding problem in the field [1,13,17] that is not unique to our prototype. While all users did discover "single" filters, which are activated by tapping on them, none of the four users were able to discover the scroll ring on their own. Users also failed to discover stroking across multiple filters and horizontal dragging from the results list.

The hover-based tooltips that we attempted in this first round of testing were completely unsuccessful, even after the experimenter explained their purpose. One user even thought they were pop-up menus and tried to tap on them.

This user suggested having a question-mark appear, which he could tap to see a "tutorial" of the gestures available. This indirectly led to our idea for highlighter hints.

One user commented that "Once I get the hang of it [the scroll ring], it goes pretty good." However, we did observe that our initial combination of the scroll ring with the linear dragging motions was too sensitive to momentary deviations in user input. This caused accidental triggering and confusion. Hence, we made the circling motion the "dominant" gesture. Once enough of a motion has been observed to determine that it is circular, the scroll ring behavior "locks in" and remains in effect until the pen is lifted. This increased the stability of the scroll ring while also keeping the linear dragging gestures available.

We also observed that users became annoyed if the search panel covered up part or all of the line containing the ink they had lassoed, or if it did not leave a comfortable margin to drag document icons out. These comments further underscore users' desire to preserve the context of their surrounding notes. Based on these comments, we changed the search panel to appear just below the search lasso (whenever possible), with a minimum ~1cm margin of white space surrounding the search panel on all sides. This guarantees enough space to drag out icons in any direction.

One user really liked the "add word" feature to extend the query just by writing new words. But the start over area was less successful. Some users suggested having a "clear" button instead. To support this desire without adding any more clutter, we now allow the user to tap in the "start over" area as alternative way to clear out the existing query. The entire query area then becomes an "add word" field.

While users encountered few recognition errors, one user attempted to enter an out-of-vocabulary word that appeared in one of her documents, and had a frustrating experience of character-by-character editing of the recognition result. We now augment the recognizer's word list with the corpus of words from the system's underlying full text indexer.

Additional First Round User Comments

Once users had learned the basic gestures, all four of them found the functionality of the system very useful. For example, one user commented that "I wish I could have a tablet, with this tool installed." Two of the users commented that they really liked how the search task is broken into multiple stages. One user explained that "During brainstorming I don't have time to perform searches right away, it will break my train of thought, so it's nice to annotate them and go back to it later."

The search result gathering functionality was also popular with users. Two users discovered the ability to drag documents out of the search panel on their own, and liked this way of easily dragging icons for documents into their notes. Two users liked the ability to capture a piece from a document and then very quickly return to the notes.

Two users particularly liked the ability to open multiple search panels or perform a "search within a search" by writing down notes about search items, and then using those notes to trigger new searches with the previous search panel still visible on the screen.

Second Round Observations

Our second round prototype included the highlighter hints (Fig. 1). The resulting improvement in discoverability of the primary gestures was dramatic. Without any help from the experimenter, all three users who participated in the second round discovered the scroll ring, stroking across multiple filters, excluding items from the search results, and dragging left to open a document. Thus, highlighter hints appear to be a significant contribution to the repertoire of techniques that pen interaction designers may employ to aid self-revelation of pen gestures [1,13,17].

Furthermore, the highlighter hints seem robust to different user behaviors. One user, who found the hints in the way and closed them, remembered that they had animated into the question mark icon. He brought the tips back when he realized that he had forgotten how to scroll, and quickly learned the scroll ring using the hint. Our improvements to the scroll ring itself also paid off; all subjects mentioned the scroll ring as one of their favorite features.

We did not provide hints for several functions in this prototype and it is telling that two out of the three users did not discover the ability to drag out document icons. Likewise, two of the users did not initially notice the tracking menu when they opened a document. We are now incorporating highlighter hints for these functions.

Again, the second round test users liked the system and found it very useful. Users liked the ability to lasso select words to indicate search terms, liked the ability to quickly filter results, and one user particularly liked the "expert" pigtail gesture to lasso text and bring up the search panel in one stroke. However, a remaining problem was that the breadcrumb currently only appears on non-self-intersecting lassos, which was not obvious to users. The lasso and pigtail gestures themselves could probably benefit from highlighter hints in the main application, but these selection techniques build on previous research [1,13,16] and improving them are not the current focus of InkSeine.

Summary

Our formative evaluations and usability evaluations suggest InkSeine's design goals are on the right track. Users value having search results available in the context of their surrounding notes, appreciate the ability to break searches into multiple episodes so as to maintain attention on the primary note taking task, and like being able to gather the results of their searching efforts into their notes. Users can fluidly transition between searching, jotting notes, and triggering "searches within searches." These design attributes would be difficult to realize with traditional fixed query boxes or search applications that exist separately from the ink content itself.

DISCUSSION

InkSeine provides a quick way to collect and annotate content from multiple documents. While our current features for gathering content provide ways to *drag out* information from the search panel, there are opportunities to further leverage the value of *in situ* search by allowing the user to *pull in* material for searches from the notes. For example, two users in our usability tests wanted to drag additional search terms from the notes into the query area, or drag search terms between multiple search panels.

Since InkSeine's personal information search uses the underlying indexer to retrieve information, all searches are based on word matches, and not measures of relevance. This model makes sense when searching for personal information that the user has seen before [7]. However, systems such as NaviQue [10] and XLibris [11] show how vector-based searches can also yield value by allowing any object, or set of objects, which might represent a large collection of query terms, to be selected as the seed for a search for related items. It would be interesting to extend InkSeine to support vector-based matches if the user lassos long passages of ink, or multiple document icons.

It is illustrative to contrast *in situ* search with the dominant model of search offered by commercial applications. Most applications funnel all queries through a single type-in box, often in a separate search application. This divorces queries from the task context that sparked them, and precludes any use of spatial layout as informal notation [12], since all queries start at the same location. We considered the idea of lassoing ink and sending queries to a fixed search box at the top of the screen. But this design does not afford saving queries in context: each query sent to the box would replace the previous one, and there would be nothing in the user's notes to remind them to return to the searches that had been sent away. This would prevent InkSeine from offering effective low-cost options per our *Enable rich tradeoffs* design goal. It would also make switching between searches, or deferring searches and later picking them up where they had been left off, much more difficult, which also would run counter to our *Optimum workflow / maximum flexibility* design goals.

CONCLUSION

InkSeine's *in situ* ink search strategy helps to reduce the cognitive barrier between having the thought to do a search while inking, to actually capturing that thought, and potentially acting on it at a later time. We believe this "slippery slope towards search" increases the value of searching in the context of active note taking. This approach also maintains search's role as a supporting task that is subservient to note taking itself.

REFERENCES

1. Agarawala, A., Balakrishnan, R. *Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen*. CHI'06, 1283-92.
2. Apitz, G., Guimbretiere, F. *CrossY: A crossing based drawing application*. UIST 2004, 3-12.
3. Bederson, B., *Interfaces for Staying in the Flow.* Ubiquity, 2004. **5**(27).
4. Buxton, B., *Designing User Experiences: Getting the Design Right and the Right Design*. 2007: In press.
5. Carr, R., Shafer, D., *The Power of PenPoint*. 1991: Addison-Wesley.
6. Csikszentmihalyi, M., *Flow: The Psychology of Optimal Experience*. 1991: HarperCollins.
7. Cutrell, E., Robbins, D., Dumais, S., Sarin, R. *Fast, Flexible Filtering with Phlat: Personal Search and Organization Made Easy*. CHI 2006, 261-270.
8. Davis, R., et al. *NotePals: lightweight note sharing by the group, for the group.* CHI '99, 338-345.
9. Fitzmaurice, G., Khan, A., Pieke, R., Buxton, B., Kurtenbach, G. *Tracking Menus*. UIST 2003, 71-79.
10. Furnas, G., Rauch, S. *Considerations for Information Environments and the NaviQue Workspace*. Digital Libraries 1998, 79-88.
11. Golovchinsky, G., Price, M., Schilit, B. *From Reading to Retrieval: Freeform Ink Annotations as Queries*. SIGIR'99, 19-25.
12. Hendry, D., Harper, D., *An Informal Information-Seeking Environment.* J. American Society of Information Science, 1997. **48**(11): p. 1036-1048.
13. Hinckley, K., Baudisch, P., Ramos, G., Guimbretiere, F. *Design and Analysis of Delimiters for Selection-Action Pen Gesture Phrases in Scriboli*. CHI 2005, 451-460.
14. Jones, W. P., Dumais, S., *The Spatial Metaphor for User Interfaces: Experimental Tests of Reference by Location versus Name.* ACM Transactions on Office Information Systems, 1986. **4**(1): p. 42-63.
15. Kirsch, D., *The intelligent use of space.* Artificial Intelligence, 1995. **73**: p. 31-68.
16. Kurtenbach, G., Buxton, W. *Issues in Combining Marking and Direct Manipulation Techniques*. UIST'91, 137-144.
17. Kurtenbach, G., Moran, T., Buxton, W. *Contextual Animation of Gestural Commands*. Proc. Graphics Interface'94, 83-90.
18. Li, Y., Hinckley, K., Guan, Z., Landay, J. A. *Experimental Analysis of Mode Switching Techniques in Pen-based User Interfaces*. CHI 2005, 461-470.
19. Mizobuchi, S., Yasumura, M. *Tapping vs. Circling Selections on Pen-based Devices: Evidence for Different Performance-Shaping Factors*. CHI 2004, 607-614.
20. Moran, T., Chiu, P., van Melle, W. *Pen-Based Interaction Techniques for Organizing Material on an Electronic Whiteboard*. UIST'97, 45-54.
21. Moscovich, T., Hughes, J. F. *Navigating Documents with the Virtual Scroll Ring*. UIST 2004, 57-60.
22. Mynatt, E. D., Igarashi, T., Edwards, W. K., LaMarca, A. *Flatland: New Dimensions in Office Whiteboards*. CHI'99, 346-353.
23. Price, M. N., Golovchinsky, G., Schilit, B. N. *Linking by inking: trailblazing in a paper-like hypertext*. HYPERTEXT '98, 30-39.
24. Rao, R., Card, S. K., Jellinek, H. D., Mackinlay, J. D., Robertson, G. G. *The Information Grid: A Framework for Information Retrieval and Retrieval-centered Applications*. UIST'92, 23-32.
25. Robertson, G., Czerwinski, M., Larson, K., Robbins, D., Thiel, D., van Dantzich, M. *Data Mountain: Using Spatial Memory for Document Management*. UIST'98, 153-162.
26. Russell, D., Stefik, M., Pirolli, P., Card, S. *The Cost Structure of Sensemaking*. ACM INTERCHI'93, 269-276.
27. Schilit, B. N., Golovchinsky, G., Price, M. N. *Beyond paper: supporting active reading with free form digital ink annotations*. CHI'98, 249-256.
28. Shilman, M., Tan, D., Simard, P. *CueTIP: A Mixed-Initiative Interface for Correcting Handwriting Errors*. UIST 2006, 323-332.
29. Stifelman, L., Arons, B., Schmandt, C. *The audio notebook: paper and pen interaction with structured speech*. CHI '01, 182-189.
30. Wilcox, L. D., Schilit, B. N., Sawhney, N. *Dynomite: a dynamically organized ink and audio notebook*. CHI'97, 186-193.
31. Yahoo! Messenger, http://messenger.yahoo.com/feat_search.php.