

PapierCraft: A System for Interactive Paper

Chunyuan Liao, François Guimbretière
Department of Computer Science
Human-Computer Interaction Lab
University of Maryland
College Park, MD 20742, U.S.A
{liaomay, francois} @cs.umd.edu

Ken Hinckley
Microsoft Research
One Microsoft Way
Redmond, WA 98052-6399, USA
kenh@microsoft.com

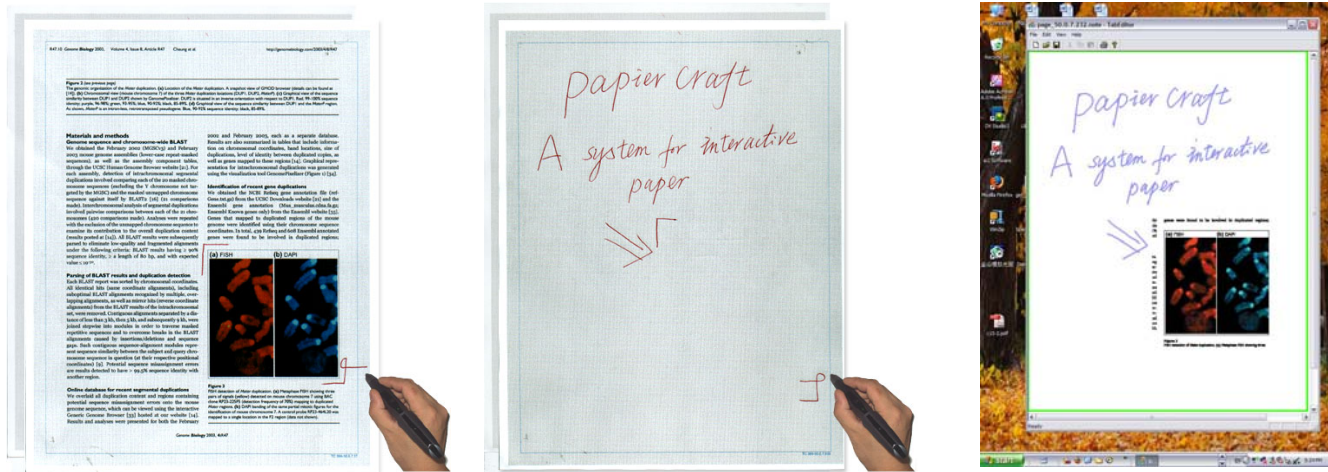


Figure 1. An example of interactions in PapierCraft: an image is copied from a printout on the left, then pasted to a note sheet on the right. Upon pen synchronization, the result is shown on our PapierCraft viewer.

ABSTRACT

The affordances of paper (e.g., ease of annotation and navigation) make it a fundamental tool for knowledge gathering and crystallization tasks. During such tasks, users create a rich web of annotation and cross references. Unfortunately, as paper is a static media, this web often gets trapped in the physical world. Some systems such as XLibris [33] address this problem by transferring this task in the digital realm where it is easy to capture all links created by the users. This approach is very powerful but suffers from the limitations of current tablet computers such as a limited screen space.

In this paper we propose a paper-based interface to support the knowledge gathering and crystallization process. Our system considers document printouts as *proxies* of digital documents stored on the user's computer. Users can draw command gestures on printouts to indicate operations such as copying a document area, pasting an area previously copied, or creating a link. Upon pen synchronization, our infra-

structure will execute these commands and present the result in our custom built viewer.

In this paper we present the design and implementation of the PapierCraft system and report our experience while building this system as well as early feedback from a small group of users.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

GENERAL TERMS: Management, Documentation, Design, Human Factors

KEYWORDS: Paper interfaces, pen gestures, multiple display surfaces, distributed systems

INTRODUCTION

In an age when users are besieged by personal computers, handheld devices, tablets, smart phones, and digital watches, knowledge workers still want paper, use paper, and strongly prefer paper for many tasks. Paper is inexpensive, comfortable to read, easy to annotate, light to carry, quick to access, and simple to use. In short, the strengths of paper are the weaknesses of the computer.

While mobile devices will improve in the future, paper offers many properties that are hard to beat. The key affordances of paper (e.g., ease of annotation, the option to flip

Copyright

quickly between several documents, and the ability to display large quantities of information all at once by physically spreading documents in space) are well adapted to typical knowledge gathering and crystallization tasks [25, 34]. During such *active reading* tasks, knowledge workers process and organize the information for later retrieval. For example, users may annotate a specific region of text with hand written notes or mark-up. The users may take notes reflecting their own understanding of the material on a separate pad, possibly including explicit references (like “See also figure 7 on page 23”). Users can literally cut and paste information between documents, attach post-it notes, or join two separate documents by placing them side by side. These notations and physical arrangements represent an implicit web of links between multiple documents, all of which is unfortunately trapped in the physical world. Thus, it is difficult for the user to later search, navigate, or build upon the work embodied by this network.

Two approaches have been proposed in the past to address this problem. One approach is to augment paper with the help of a nearby computer. The Digital Desk [37], for example, projects interaction feedback directly onto pieces of paper. Despite recent advances in projection/vision systems [19], this approach still requires hardware that is much more expensive and much less portable than plain paper. This is true even for systems which do not require direct projection, such as the A-book system [23] which still need a tablet connected to a nearby host computer.

Another approach is to fully transfer the knowledge gathering and crystallization process into the digital world. This is exemplified by the Xlibris [33] system described by Schillit as the “Active reading machine.” More recently, other systems such as Microsoft OneNote [27] have advanced this approach and made it easy to take notes and create collages from digital information sources. While moving into the digital world makes it easy to capture all user interactions, and to “link by inking” [28], it also presents a major drawback: current tablet computers only offer a limited display surface when compared to the typical surface space of a physical desk. Although this problem is mitigated by the use of windows virtualization, a dilemma remains: one can either look at small parts of many documents at once, or one must flip between multiple windows. While multi-display configurations or large high resolution displays, such as the Stanford mural [8], aim to address this problem, they also defeat the goal of portability.

Our PapierCraft system explores a third path that considers paper printouts as *proxies* of digital documents. In this approach, users use a gesture-based command system to interact directly on paper documents for which a digital version is stored in the system. Users can copy and paste information from one paper document to another paper (or electronic) document (see figure 1), create links between content found in two different paper documents, or juxtapose (“stitch”) two paper documents together by drawing a pen stroke across them. Upon synchronization of the user’s

digital pen with a host computer, PapierCraft executes all operations on the corresponding digital documents and presents the results in a digital document browser. All information gathered in the paper world becomes accessible to users, allowing them to easily explore the implicit links they created on paper. Thus, PapierCraft combines the advantages of paper with those of digital annotation systems like Xlibris and OneNote.

PapierCraft uses the capability of the Anoto digital pen [1] to track strokes made on paper printouts. The Anoto pen uses a camera embedded in the pen tip to image a subtle dot pattern printed (using an ordinary printer) in the background of a paper document. The dot pattern encodes the absolute position of the pen tip on the paper, as well as a unique page ID. PapierCraft goes beyond the form-fill approach supported by the Anoto infrastructure (or the PaperPDA [4, 10]) by supporting various command gestures that can act on a user-specified region of a printed document.

In this paper, we present the design and implementation of the PapierCraft system. At the core of the system is a new pen-based marking interface designed specifically for passive media such as paper. In its most basic form presented here, this interface does not require any *active* feedback beyond the ink laid on the paper during pen interactions. We show how this marking interface can be built on top of an infrastructure which maintains the correspondence between digital documents and their printout such as the Paper Augmented Digital Document infrastructure, PADD [7]. Based on this information, PapierCraft interprets gestures made on paper printouts when the digital pen is synchronized. Finally, because more and more paper and Tablet computer uses are interleaved, our system is not limited to paper-to-paper interaction but also supports paper-to-computer and computer-to-computer interaction with the same interaction syntax across situations.

RELATED WORK

We draw on three main areas of related work: paper-computer integration, marking-based interfaces, and distributed interaction systems.

Bridging the paper-computer gap

Many systems have been proposed to bridge the gap between paper and the digital world. These systems can be classified into four broad categories depending on the roles played by computer and paper while using the system.

First, fully digital systems attempt to eliminate the need for paper by mapping paper affordances onto digital media. Such systems include Xlibris, Dynamite [38], OneNote and Screen Crayons [26]. These systems support active reading activities using digital media. For example, they allow for annotation of digital sources and let users copy information from one document to another. These fully digital systems can leverage all affordances of digital media and can record all interactions. This makes it easy to “link by inking” [28]. However, because of limited screen real estate, these systems make it difficult to navigate through several documents

simultaneously.

Second, tightly coupled systems augment paper by interweaving interactions on paper and on a nearby computer. Systems in this category, such as the Digital Desk [37], Ariel [22], Video Mosaic [21] and EnhancedDesk [17], offer a large spectrum of interaction styles. This approach is powerful, since the computer can provide direct feedback during paper interactions. However, these systems require cumbersome technology such as digital projectors and do not match paper's high portability. Other systems such as Intelligent paper [6], PaperLink [3], Paper++ [24], and Books-with-Voices [15] provide more flexible input devices (e.g., light pen, PDA), but offer a simpler set of interactions (e.g., clicking on a link). PapierCraft, like the A-Book system, explores a different interaction model in which users can actively manipulate the content printed on paper. We believe that this approach is better adapted to active reading tasks. Importantly, PapierCraft extends upon the A-Book system by investigating how such interactions can be performed in a paper-only setting.

Third, several systems, including Xax [13], PaperPDA and the Anoto system, have investigated paper as a form filling medium. These systems offer an asymmetric view of the paper-computer relationship that is biased towards entry of highly structured information on paper: input must be on pre-printed areas and for each sheet, only a limited number of actions can be applied to the input. We believe that this approach is far too restrictive for active reading tasks, which by their very nature are complex, ill-defined tasks that take unexpected twists as the user works. In this regard, PapierCraft is probably more related to the Audio NoteBook [36], which helps people capture and organize free form notes.

The fourth and final strategy, known as cohabitation, places paper and computer on an equal footing as a way to edit Paper Augmented Digital Documents [7]. Existing implementations of PADD already allow paper-based annotation of digital documents. PapierCraft uses the PADD infrastructure to establish paper as a proxy for interaction with a digital document, but we extend previous work by bringing interactive capabilities to paper. In this respect, our approach builds on tangible interfaces such as NISMap [5] and Papier-Mâché [16]. Like PapierCraft, NISMap uses the Anoto pen as the underlying infrastructure, but it is focused on group-coordinated annotation of a shared map, while we focus on document content manipulation for personal knowledge work. Papier-Mâché is a toolkit, aimed at abstracting low level tangible input (e.g., symbols on paper), and facilitating application development. It does not include a paper-based command issuing mechanism.

Marking-based interfaces

Previous work has explored a number of marking-based interfaces. MATE [9] is the most relevant as it focuses on a proofreading application. However, MATE and most other marking interfaces are designed for computer displays where active feedback is readily available. In contrast,

PapierCraft does not offer active feedback (other than the physical ink left on the paper) while the user is inking. Our command system provides a more flexible command structure (as compared to PaperPDA for example) that brings some of the free-form capabilities of marking-based interfaces to paper documents.

Distributed Interaction

By using paper printouts as proxies, PapierCraft lets users issue commands that span several digital documents, which may be managed by different computers and databases. In this regard, our work builds on distributed interaction techniques such as Pick-and-drop [29], Augmented Surfaces [31], and particularly Stitching [11], which uses simple pen strokes to copy and paste information between computers or combine portable computer displays into a larger interactive surface. PapierCraft extends this interaction style to the paper world.

DESIGN GOALS

The Anoto digital pen system [1] combined with the Paper Augmented Digital Document system [7] makes it possible to track interactions performed on paper. With the PADD infrastructure, it is possible to capture every stroke added to the printout of a given document and to add those strokes to the original version of the document in the digital world. Many paper documents that modern knowledge workers read and annotate are printouts from digital sources. As a result, paper documents can be considered as proxies of digital documents that are accessible from a database.

This observation leads to the following question: is it possible to have users perform document manipulations supported by applications such as OneNote, Xlibris and ScreenCrayons in the paper world and apply these actions automatically to the corresponding digital document? While the PADD infrastructure provides a partial solution for simple annotations, our goal in PapierCraft is to explore more complex commands such as copying a piece of information from one document to another, or creating hyperlinks between parts of different documents.

One of the most important problems to address in PapierCraft is the design of the command system. We set forth the following design goals:

Respect current paper based practice. This was the main goal of our design. Our system should modify the current patterns of paper usage as little as possible and support most of the users' existing reading behaviors. For example the system should not impose restrictions on the shape or location of marks used by users during their normal active reading process. We would also like to serve patterns of use such copying one part of a document into another, creating collages, or stitching several pieces of paper together to create a larger piece.

Simple and reliable command system. Since the ink laid on the paper is the only immediate feedback provided by the interface, the command system needs to be simple and re-

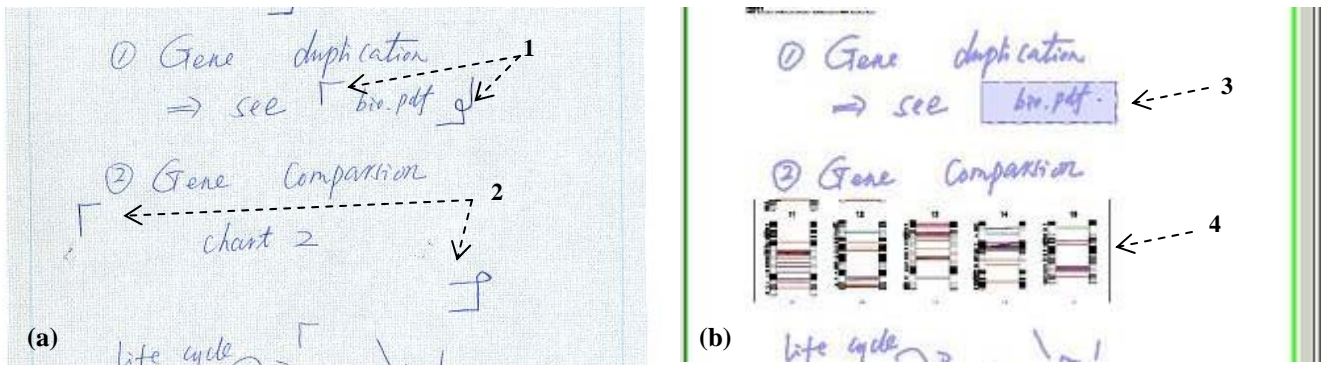


Figure 2 : Self-explanation of PapierCraft interface : (a) part of a paper note created with PapierCraft system, in which the gestures, labeled by “1” and “2”, and contextual annotations indicate operations, “create a hyperlink to bio.pdf” and “paste chart2 here” respectively . (b) part of the resultant digital document opened in the reader application, of which the area pointed by number 3 and 4 are execution result.

liable to limit the number of errors that the user may need to correct.

A human-readable command system. While the strokes used to describe the commands are intended for interpretation by a computer, it is also important that they can be understood by a human reader. For example, it should be clear for which scope a given command will be executed, or which area will be occupied by a pasted image. It should also be easy to distinguish strokes intended to be interpreted by the computer from other markings. The proofreading marks system [2] provides a good example, as its main purpose is to convey a set of instructions from one person to another. To meet this goal, proofreading marks were designed to be easy to read and unambiguous.

In the following sections we describe a command system that fulfills these requirements and illustrate how it can be implemented using the commercially available Anoto pen technology. With our system, one can copy/paste, establish hyperlinks, or create collages of digital documents using only a digital pen and paper. In fact, our system goes beyond the above requirements as it also allows for interactions between paper and digital media such as a tablet computer.

A PAPER INTERFACE

PapierCraft proposes an interface designed for use during paper-based active reading tasks. In such situations, users will manipulate multiple printed documents, probably spread out on their desk, and annotate them with a digital pen. Users might also use auxiliary paper surfaces such as a notepad or maybe some loose pieces of paper.

The specific digital pen technology is not important to our design as long as the system is able to distinguish between different physical pages, provide the local coordinates of each stroke made on a given page, and timestamp each stroke. Our current prototype uses the Anoto technology. We also assume that the document printouts are managed by a system such as Paper Augmented Digital Documents. Such a system maintains the correspondence between a given digital page and its paper version. In particular, given a stroke created on a paper printout, the system is able to

retrieve the original digital document that was used to generate the printout. It is also expected that while using the system, users will interleave “annotation” strokes and “command” strokes.

While designing PapierCraft we had to answer familiar questions such as how to distinguish “annotation” strokes from “command” strokes, how users can designate the scope of commands, and finally, how users can select a specific command, all within the constraints imposed by paper.

Ink and Gesture strokes

Many solutions have been proposed to distinguish between ink intended as content and ink intended to be interpreted by the system. Some systems propose a fully implicit approach [39] in which the computer automatically distinguishes ink strokes from gesture strokes. Other systems propose a fully explicit approach [12, 20], in which users indicate the type of the current stroke (e.g., by pressing a button). A mixed approach, where the computer and the user collaborate to resolved ambiguous input, is also possible [32].

Given the limited level of feedback provided by our system, we felt that a non-explicit approach would be problematic as users will have no way to know if the interpretation of a pen stroke was correct until the commands are executed sometime in the future. Mixed approaches are also problematic as they require immediate feedback during the disambiguation phase. As a result, our system requires a “gesture” button present either on the pen or in the environment (e.g., a foot pedal). A button integrated with the pen presents an obvious portability advantage, but because we were unable to modify our pen hardware, our prototype uses a foot pedal. Our system only requires a weak synchronization and a stroke is considered a gesture stroke as soon as the gesture button is pressed for some duration during that stroke.

Our system also needs to make a distinction between gesture strokes used to specify the scope of an operation and strokes used to select among several possible commands. To create such a distinction we adopted the Pigtail approach proposed by Hinckley et al. [12] in Scriboli. Like Scriboli, our system considers all strokes between the first gesture stroke and a

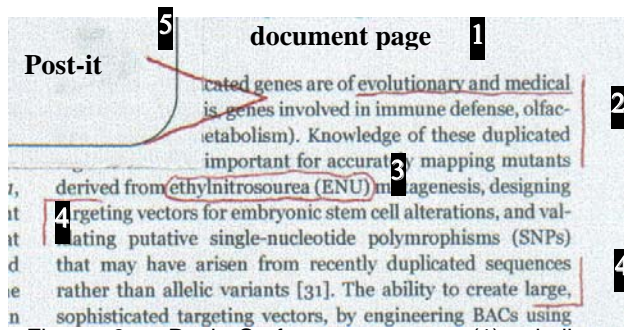


Figure 3 : PapierCraft scope types: (1)underline, (2)margin bar, (3)lasso, (4)cropping mark and (5)stitching mark which is drawn cross the document page and a Post-it note.

gesture stroke containing a pigtail as part of the scope selection, and all gesture strokes after the pigtail as part of the command selection. The pigtail notation is familiar to proofreaders and as shown in Hinckley et al. [12], it yields similar performance to presenting a “handle” (menu box) at the end of a stroke to specify part of the scope.

Specifying the scope of commands

The absence of real-time feedback had the greatest impact on scope selection in our interface. Consider the simple example of a cut and paste operation. When this operation is performed on a computer (e.g., using a system such as OneNote), one first selects the object to be copied, often using a marquee selection with the system providing instant feedback of the area selected. Once the selection is complete, one issues the copy command and moves to the paste location. As soon as the paste command is issued, visual feedback is provided immediately to show the result of this operation. Of course, on paper neither type of feedback can be provided. To address this problem, we ask users to draw the intended scope of all their commands. In the case of the copy operation, the scope identifies the region of the paper to be copied.

PapierCraft offers five types of scope selectors inspired by typical marks found on manuscripts (figure 3). One can select content by underlining a passage of text, creating a margin bar to select several lines at a time, or simply by lassoing an arbitrary area of a document. We also offer a special scope for the “Stitching mark.” Our stitching mark is a V reminiscent of the mark carpenters draw on two pieces of woods to remember their alignment. When drawn on top of two overlapping paper documents, this mark indicates that the paper sheets should be stitched together in the digital view. The stitching mark can also be used to “pin” a small piece of paper such as a Post-It note on top of a larger one.

For a paste operation, the scope indicates to the system the size and position of the information to be pasted. Before synchronization, the drawn scope serves as a placeholder reminding the user that some information will appear at that location upon synchronization. At synchronization time, the scope is used by PapierCraft to scale the pasted content so that it fits into the scope (Figure 1, middle). It is important

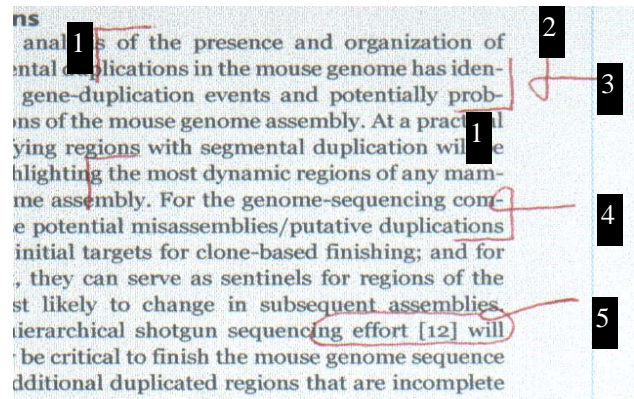


Figure 4. Structure of command “copying” : (1) parameter selector (2) delimiter (pigtail) (3) marking menu for “copy” (4) “short-cut” of the same command, (5) another shortcut of copying with lasso selection

to note that surrounding writings which are created during the active reading process, will help to remember the content that is pasted (see Figure 2 for more examples).

Simple operators such as copy and paste only require a simple scope. Other operators such as the “copy with keyword” command, which assigns a keyword to a specific area of a document, use a scope to select the area in question as well as an additional parameter to select the assigned keyword.

Selecting a command

For our command selection system, it was desirable to use a stroke based system which does not require any more feedback than the strokes drawn by the users. Such systems include marking menus [9], and Sensiva [35]. Given our choice of the pigtail as a scope-command separator, it was natural for us to pick the marking menu. Single level marking menus typically offer 8 different commands. We felt that it was also interesting to consider cases where more commands are needed. One option was to use a two-level hierarchy, but we felt that it might be problematic on paper. First, without any immediate feedback it would be difficult for users to discover and learn the different marks. Second, it might be even more difficult for users to remember the meaning of a mark drawn on paper after some time has passed. The simple mark hierarchical marking menus [12, 40] would only exacerbate this problem, because on paper, a human reader cannot discern the temporal order of the marks.

Instead, we decided to use a mixed approach. In PapierCraft the most frequent commands (copy, paste, hyperlink source, hyperlink target) can be accessed directly from the cardinal directions of our marking menu (East, West, North and South respectively). The full command set can be accessed by simply writing down an unambiguous prefix of the command name. For example, one can directly write the word “Paste” immediately after the pigtail to take precedence over the mark direction.

Writing a command name not only allows for a larger

command set, it also makes it easier for people to remember the command they wish to issue. As our system will also recognize unambiguous prefixes, command prefixes will naturally assume the status of shortcuts. This approach also makes it easy for people to read over the commands they issued on a piece of paper. While having to write a full command name has a cost, we believe that it might be well accepted by users as it fits naturally into the context of proofreading or annotating a document during active reading tasks. We also considered the possibility of character/word recognition errors, but given the small vocabulary in the present application we do not expect this to be a significant practical problem. A similar technique known as “mnemonic flicks” has recently been proposed, but not yet published, for a TabletPC-based marking interface [39]. Our work in PapierCraft shows how this technique is particularly well-suited to working on paper. We believe that users of such systems will come to expect a certain equivalence between paper and computer interfaces. Thus, for a consistent interaction experience, the same input should be recognized both on paper and in the fully digital situation.

Issuing a copy/paste in PapierCraft

We now review a simple copy/paste interaction in PapierCraft (Figure 1). To copy an image, the user first indicates the area of the document to be copied. To do so, she presses the gesture button (in our case a foot pedal), and draws cropping marks around the area of interest. Keeping the gesture button pressed down, she draws a pig tail followed by marking to the right (East). Note that the last stroke of the cropping mark, the pigtail, and the command mark can be issued as one continuous mark, making the interaction very fluid.

To paste, the user would follow a similar pattern: First, she indicates the area where the paste will take place by drawing a crop mark and a pigtail. Then she selects the paste command by marking West. For this multiple-strokes command, the system is flexible with regard to the use of the gesture button. The user may decide to hold the gesture button down during the full paste operation or simply click the button each time she is drawing a “command” stroke. Upon synchronization of the pen, the copied image will appear properly scaled at the specified location in the document.

Command execution

As our system was designed to be used in contexts with no nearby computer and the pen itself cannot process the strokes in real time, the PapierCraft system must process the commands in batch mode at sync time. The execution process has three phases: (1) a stroke processing phase during which pen strokes are uploaded from the pen and processed locally; (2) an execution phase during which the different stroke sources are synchronized together and the resulting commands executed; and (3) a display phase during which client displays are notified of the possible modification to the documents they are managing. As in the Stitching system, these tasks are carried out by several processes at different hosts all federated through an event

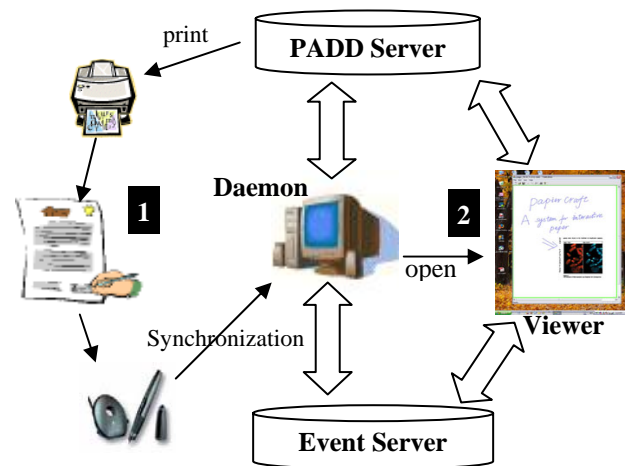


Figure 5: Media switching in PapierCraft system: (1) paper is used as a transient media for digital documents and the daemon works with servers on behalf of it (2) CraftViewer is the working media in digital world, sharing with the Daemon the similar procedures for stroking processing.

server that synchronizes the different event sources before executing the corresponding command.

Stroke processing

The PapierCraft daemon is invoked and receives all the strokes captured by the pen. After importing the strokes, the process first downloads the latest digital versions of the corresponding digital documents from a central database managed by the PADD server and contacts the event server to retrieve a record of the pedal states for the time when the strokes were drawn. This is required in our prototype because we use a separate foot pedal as a “gesture” button, but it would not be necessary if the button was integrated with the pen.

Upon receiving the information, the daemon can start processing strokes in their temporal order. It labels each stroke as a command or an annotation stroke. Annotation strokes are simply merged into the digital file. Command strokes are passed to a *gesture recognition engine* which recognizes commands from the stroke stream. Once a command is detected, a corresponding “Local Event” is sent to the event server. For example, in the case of copy, the local event will include a timestamp, the command type, and the following command-specific parameters: the selected image, text extracted from the digital file, and surrounding annotation strokes, if any, available at the same point in time.

One important thing is parsing the gesture strokes. Due to the availability of written command name, the number of strokes in a command can vary due. Furthermore while issuing several command in successions, the user might forgot to release the pressure from the gesture button. We use four rules to determine if a stroke, say *S*, is the last stroke of the current command: (1) the following stroke is spatially far away from *S*; (2) there is a long gap between their timestamps; (3) the user exists “gesture” mode after *S*; (4) *S* is

the last stroke drawn on this page. As long as any of the four criteria is satisfied, S is taken as the ending stroke and all strokes, if any, for command type will be submitted to the handwriting recognizer.

Event synchronization and command execution

Event synchronization is accomplished by the event server, which maintains a central Event Cache for Local Events coming from various devices/paper sheets during a session. Such a cache is necessary because these events may be submitted by the clients out of the order of the actual occurrences. Currently, we do not distinguish different pens. Instead, we temporally align all events in the cache and handle them as single thread. This enables interaction across different pens which might be convenient for a single user. It might certainly be a source of race condition in a group of users interacting with a given document. This single thread approach is a policy decision and other policies are also possible. For instance, each pen could have its private thread, allowing a separate “clipboard” for each pen.

It is important to note that associated events can be separated by events of other types. For instance, as in the case of a digital copy and paste sequence, one can first “Copy”, then create a hyperlink, and finally “Paste”. This is equivalent to copy/paste followed by linking. This feature reflects the common “clipboard” semantic which is so familiar to users.

Client notification

Upon execution of a command, the event server sends a “Global Event” notification to the client which is handling the corresponding page. For instance, the daemon may receive a “Paste data ready” event indicating that some data should be inserted into a region of a given page. If the notification is valid, the client will then send a request to the server and retrieve the pasted data. Finally, it updates the opened digital file and uploads it to the PADD server. If the daemon is not responsive, the Global Event will be held until another client opens the digital file, which will receive this notification and update as described above. Currently, we assume, at any given time, there is only one such client opening a digital file, so there will be no problem of version control.

After processing all strokes, the daemon opens the latest digital document in a viewer application, known as CraftViewer (Figure 1, right). Once this step is completed, the users can switch to the digital world to continue their work.

Dealing with errors

The lack of active feedback and the dependence on batch processing make dealing with errors a potentially challenging process. There are two potential sources of errors: user mishaps, which might occur when users change their mind about issuing a command; and system errors such as misrecognized gestures.

User mishaps

To address user mishaps, it is useful to look at the way users currently deal with errors during knowledge gathering and

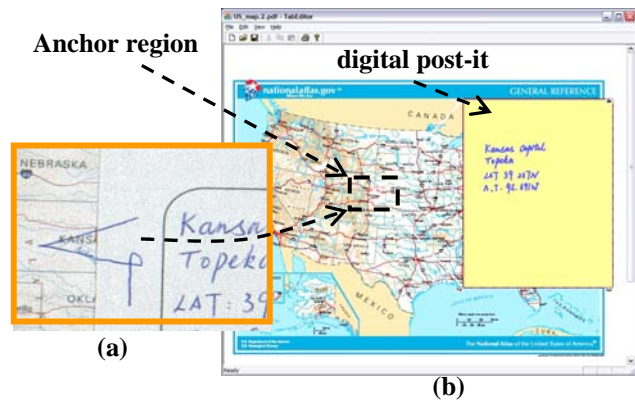


Figure 6. Use of “Stitching”: (a) a zoom-in view of a post-it note attached to a paper map by “stitching” (b) the resultant digital post-it in the viewer application

crystallization. According to Fawzia Khan [14], few people use an eraser to correct errors while taking notes. Instead users simply rewrite or leave errors as they are. Our system follows this practice by simply ignoring commands that cannot be parsed correctly. For example, one can cancel a copy command in progress by simply ignoring the current scope selection and reselecting a new scope before issuing the copy command itself.

Batch processing errors

Even though our system relies only on simple pattern matching, one has to expect that some errors will occur during the process. Furthermore, it is possible that users sometimes generate ill formed command syntax. To deal with such mistakes, the CraftViewer provides a context player to help users correct possible mistakes, similar to TMC [30] proposed by Rekimonto. After identifying the site of a possible mistake, the user selects the corresponding strokes for correction. The system then pops up a “Context Player”, and the user can employ it to replay the strokes occurring just before the problematic strokes. For example, if a pasted graph does not appear at the target location, one can select one of the “Paste” strokes and choose the menu “Show context.” A separate window will pop up, showing the strokes drawn just before or after the “paste” stroke along with the document pages containing them. The user can drag a slider to explore the stroke history and see the strokes and pages they were drawn on. Thus, it is possible to open the corresponding document and reissue the correct command from within the CraftViewer.

Implementation

PapierCraft is written in C++. The PADD server runs on Linux, and all other components run on Microsoft Windows. We also use Acrobat SDK, Microsoft Tablet PC Recognizer Pack, and Anoto SDK3.0.

USING THE PAPIERCRAFT SYSTEM

We now present an overview of how the features provided by our system support knowledge gathering and crystallization tasks.

Excerpting

Excerpting is a very common practice in active reading and can take many forms - from writing notes to gluing a photocopied graph into one's notes. As described above, PapierCraft can support this interaction in a natural way. The system always uses a slightly larger area than was selected by the user so that the surrounding text and annotations can be seen in context. The system also creates an implicit link between the pasted information and the original document to make it easy for the user to access the original document from her notes. Also, the system lets users display text present in the source scope, which can be used for search, copied to a text editor, etc.

Import information from legacy documents

As a bridge to legacy documents (documents without the Anoto pattern embedded in them), our system supports the use of tracing paper. We use a piece of Vellum on which we have printed our own Anoto pattern. Vellum is a translucent paper used by architects to trace details of a given plan. After tracing the content of interest on the Vellum, users can use these sketches as the source of a copy operation. This illustrates another important aspect of our system: transitivity. In PapierCraft, any annotation or pasted information can be used as a source for a new copy operation. For example, it is possible to create thumbnails of several pages of notes by simply copying each page on a single piece of paper. The resulting digital document will show all the strokes as well as the pasted information present in the original document. For example, this feature can be very helpful to create a storyboard.

Digital Collages

PapierCraft makes it easy to create digital collages. Collages are created by drawing a stitch mark (a wedge mark as shown in Figure 6) across the boundary of two documents. Thus, a side-by-side arrangement of paper documents can be stitched together. Upon synchronization, the PapierCraft viewer renders the digital versions of the documents side-by-side as well. Stitching can also be used to pin a smaller piece of paper (such as a Post-it note) onto a larger one (Figure 6). The corresponding digital version will include an "anchor region" around the position of the stitching mark vertex. Clicking on such an anchor inside CraftViewer allows users to review the Post-it in a separate window.

Creating and managing links

As shown above, implicit links are automatically created by PapierCraft during "Copy/Paste" and "Stitching". Users can also explicitly create a "hot spot": for example, one can select a region of a map with a cropping mark and draw the gesture "hyperlink start" followed by the gesture "hyperlink end" on another document. When the resulting digital file is opened in the viewer application, users can double-click the hotspot to open the associated document.

Tagging information

To facilitate information retrieval, people often tag mes-

sages or files. For example, they may assign certain keywords to a document or part of a document. PapierCraft supports this usage pattern by allowing two subsequent scoping operations during the copying command: One can first choose keywords in the text with an underline gesture and then select the context lines of text by drawing a margin bar. The copied paragraph will be indexed by the underlined keywords.

CraftViewer

The CraftViewer is the digital interface to the PapierCraft infrastructure (see Figure 5). It is an integrated document reader allowing one to review the work done in the paper world and make corrections if necessary. With CraftViewer, one can review all the information collected from various paper sheets, as well as the implicit connections among them. One can zoom in/zoom out, show/hide strokes based on their types, choose different views of pasted data (as text or as image), and follow hyperlinks. The system supports two types of links: implicit links that are created during copy/paste or stitching operations, and explicit links that are created when users issue the hyperlink command to link two documents together. This feature makes it easy for users to explore the web of links they created during the knowledge gathering and crystallization process.

Mixed media operations

Because paper and digital media coexist on one's desk, the CraftViewer implements a similar marking interface as that discussed above for paper. For example, a user can copy information from a paper document to a displayed digital document. As in the paper case, the scope drawn on the screen serves as a proxy for the pasted information. Upon synchronization of the digital pen, the view automatically updates using the information captured on paper. Thus, the CraftViewer plays both the role of a stroke processing client and a visualization client. For tablets equipped with a touch-sensitive screen, all interactions can be done with one pen, but for tablet using magnetic sensing technology, it is required to switch pens.

The long-term vision of PapierCraft is that users can again return to the physical world by printing out the latest documents or notes, and work on them further in a virtual cycle. However, printing out modified documents with the Anoto pattern is not yet implemented. By re-printing, a digital document could reside on various display surfaces at different stages in a long-term active reading task, and during this process interactions in physical world are seamlessly integrated with the digital world.

DISCUSSION

We presented PapierCraft, a paper-based document manipulation interface that supports tasks such as active reading and knowledge crystallization. Our system demonstrates how to design and implement a paper based interface that can accommodate the limited feedback provided by ink on paper. PapierCraft bridges the gap between the paper world and the digital world by letting people interact with printouts

as proxies of a digital document. This approach offers the information tracking capability of systems such as XLibris, while simultaneously offering users all of the affordances of paper. Here we reflect on our design decisions and discuss the lessons we learned.

Early user feedback

We conducted a small scale informal evaluation of our system, by asking 4 colleagues (not affiliated with this project) to use the system. For each participant, we conducted a hands-on demonstration of how to copy and paste. Next, we asked the participants to perform several copy and paste operations using different scope selection mechanisms. Some interactions were performed using the pen and pedal configuration described above, and some were done using two pens, one always in annotation mode, and one always in gesture mode. The latter pen used a red cartridge.

Overall the participants' reaction was positive. They believed that our gesture set would be easy to remember, especially if proper mnemonic cues were provided (e.g., pointing out that the paste mark looks like a P). The option of writing down the command name was very popular, and participant pointed out that it would be especially useful if the system was user configurable.

Participants were uncertain about the trade-off between using one pen (with a gesture button) or two pens - one for annotations and one for gestures. On the one hand, they liked the direct feedback provided by the use of two pens. Because the gestures were now shown in red, they were easy to identify. On the other hand, they also reported that this configuration was cumbersome as they had to switch pens all the time and, of course, carry two pens at all times. The general consensus was that as users become more familiar with the system, one pen would be their preferred option.

We also discussed with each participant about the minimal feedback provided by the interface. Two main points emerged. First, the level of feedback required would strongly depend on the reliability of the system. If the system had a very high rate of gesture recognition, some participants felt that the current level of feedback might be acceptable. In that respect, it is clear that our prototype needs to be improved. In particular, it might be very useful to use a trainable gesture recognizer. Second, two kinds of feedback would be very useful: 1) a gesture mode indicator; 2) a confirmation that strokes have been recognized. These requests were not surprising. Our original design called for a small LED to light the area around the pen tip for gesture mode, and haptic feedback upon gesture recognition. Unfortunately the current Anoto digital pen is not programmable, so we were unable to explore that aspect of the design. We hope that this situation will change rapidly as new products such as the Leapfrog [18] pentop computer become more common. Regarding the lack of content feedback during the paste operation, users pointed out that simply adding a note inside the paste area might be enough.

The next generation of digital pens

The PapierCraft system was designed to work at the current, somewhat limited level of pen technology: we assumed that the pen would be in a paper-only environment (assuming a pen with a gesture button) and could not provide any feedback. We now consider how advances in pen technology could influence our design.

Gesture button

Because we could not modify the commercially available pen directly, we had to rely on an external foot pedal to trigger the gesture mode. While this setting is perfectly adequate in an office environment where the foot pedal can be provided by the infrastructure, it creates difficulties in the informal context of a paper-only environment. We believe that this limitation can be easily addressed by a simple modification of the pen hardware. This would have the added advantage of simplifying the synchronization burden between the pen and the computer managing the foot pedal.

Wireless connectivity

Another limitation of current pen technology is the need for wired synchronization. Wireless synchronization can streamline users' interactions with PapierCraft. We envision a system where the pen automatically manages synchronization depending of the availability of wireless communication. If no link is available, the pen will simply store the strokes it has captured until a link for synchronization becomes available. If a link is available (e.g., when a mixed paper-computer interaction is underway), strokes will be streamed right away to provide immediate feedback on the computer in a way similar to the Stitching system.

Challenges in interactions between multiple devices

The activities on paper are unknown by the infrastructure until pen synchronization, while operations on a tablet computer can be reported to the event server in real-time. As the result, from the view point of the server, the events on paper may be delayed and the event stream may be not complete before synchronization. If the server failed to discern this situation and blindly executes the incomplete event stream, unexpected results might results. For example, imagine one copies a picture from a tablet PC to a piece of paper, and then copies another picture from the same paper to the same computer. The whole local event sequence is "Copy_{1, tablet}, Paste_{1, paper}, Copy_{2, paper}, Paste_{2, tablet}". If events "Paste_{1, paper}, Copy_{2, paper}" got delayed the event server simply run "Copy_{1, tablet}, Paste_{2, tablet}", an incorrect action. Similar problems also exist while using multiple pens.

In our prototype, we require a strict criterion that all events occurring earlier must be in the cache when events from a pen are processed. That means, for paper-compute interaction, the operations such as copy/link/stitch must be conducted in one direction: either from paper to computer or from paper to computer, but not both. While this is acceptable for a prototype (most of the time we are only using one pen), imposing such a constraint is unrealistic as a total order might not always be available. We are exploring possible

solutions to such a problem such as graceful rollback upon discovery of new strokes.

CONCLUSION

In this paper we presented PapierCraft, a system letting people interact with Paper Augmented Digital Documents by drawing gestures on paper printouts. Our system was designed to support active reading tasks such as copying and pasting information from one document to another, establishing links and stitching two documents. The Papier-Craft interface can easily be extended to other activities as well. PapierCraft demonstrates the feasibility of minimal feedback interfaces whose applications extend beyond the traditional paper medium to other medium such as whiteboard for example..

ACKNOWLEDGEMENTS

This work has been supported in part by Microsoft Research for the Microsoft Center for Interaction Design and Visualization at the University of Maryland.. We would like to acknowledge Jim Hollan for his ongoing support of the People Paper and Computer project and his repeated plea for a paper based interface. David Levin implemented the first version of the PADD infrastructure. Corinna Löckenhoff supported the production of this document in many ways. Lars Brorsson from Anoto provided us with an early version of the Anoto SDK V3.0 and Henrik Af Trampe, also from Anoto, provided valuable help in using the library.

REFERENCES

1. Anoto, Development Guide for Service Enabled by Anoto Functionality. 2002.
2. ANSI, *American national standard proof corrections*. 1981: American National Standards Institute.
3. Arai, T., D. Aust, and S.E. Hudson. PaperLink: a technique for hyperlinking from real paper to electronic content. *Proceedings of CHI'97*, pp. 327 - 334.
4. Avrahami, D., S.E. Hudson, T.P. Moran, and B.D. Williams. Guided gesture support in the paper PDA. *Proceedings of UIST'01*, pp. 197 - 198.
5. Cohen, P.R. and D.R. McGee, Tangible multimodal interfaces for safety-critical applications. *Commun. ACM*, 2004. **47**(1): p. 41 - 46.
6. Dymetman, M. and M. Copperman. Intelligent Paper. *Proceedings of EP'98*, pp. 392 - 406.
7. Guimbretiere, F. Paper Augmented Digital Documents. *Proceedings of UIST'03*, pp. 51 - 60.
8. Guimbretière, F., M. Stone, and T. Winograd. Fluid interaction with high-resolution wall-size displays. *Proceedings of UIST'01*, pp. 21 - 30.
9. Hardock, G., G. Kurtenbach, and W. Buxton. A marking based interface for collaborative writing. *Proceedings of UIST'93*, pp. 259-266.
10. Heiner, J.M., S.E. Hudson, and K. Tanaka. Linking and messaging from real paper in the Paper PDA. *Proceedings of UIST'99*, pp. 179 - 186.
11. Hinckley, K., G. Ramos, F. Guimbretiere, P. Baudisch, and M. Smith. Stitching: Pen Gestures that Span Multiple Displays. *Proceedings of AVI'04*, pp. 23 - 31.
12. Hinckley, K., P. Baudisch, G. Ramos, and F. Guimbretiere. Design and Analysis of Delimiters for Selection -Action Pen Gesture Phrases in Scriboli. *Proceedings of CHI'05*, pp. (in press).
13. Johnson, W., H. Jellinek, J. Leigh Klotz, R. Rao, and S.K. Card. Bridging the paper and electronic worlds: the paper user interface. *Proceedings of CHI'93*, pp. 507 - 512.
14. Khan, F., A Survey of Note-Taking Practices, Personal Systems Laboratory, HP Laboratories Bristol, 1993,
15. Klemmer, S.R., J. Graham, G.J. Wolff, and J.A. Landay. Books with voices: paper transcripts as a physical interface to oral histories. *Proceedings of CHI'03*, pp. 89 - 96.
16. Klemmer, S.R., J. Li, J. Lin, and J.A. Landay. Papier-Mâché: toolkit support for tangible input. *Proceedings of CHI'04*, pp. 399 - 406.
17. Koike, H., Y. Sato, Y. Kobayashi, H. Tobita, and M. Kobayashi. Interactive textbook and interactive Venn diagram: natural and intuitive interfaces on augmented desk system. *Proceedings of CHI'00*, pp. 121 - 128.
18. Leapfrog, (<http://www.leapfrog.com>). 2004.
19. Lee, J.C., P.H. Dietz, D. Maynes-Aminzade, R. Raskar, and S.E. Hudson. Automatic projector calibration with embedded light sensors. *Proceedings of UIST'04*, pp. 123 - 26.
20. Li, Y., K. Hinckley, Z. Guan, and J.A. Landay. Experimental Analysis of Mode Switching Techniques in Pen-based User Interfaces. *Proceedings of CHI'05*, pp. (in press).
21. Mackay, W. and D. Pagani. Video mosaic: laying out time in a physical space. *Proceedings of MM'94*, pp. 165 - 172.
22. Mackay, W.E., D.S. Pagani, L. Faber, B. Inwood, P. Launiainen, L. Brenta, and V. Pouzol. Ariel: augmenting paper engineering drawings. *Proceedings of CHI'95*, pp. 421 - 422.
23. Mackay, W.E., G. Pothier, C. Letondal, K. Bøegh, and H.E. Sørensen. The missing link: augmenting biology laboratory notebooks. *Proceedings of UIST'02*, pp. 41 - 50.
24. Norrie, M.C. and B. Signer. Switching Over to Paper: A New Web Channel. *Proceedings of Proceedings of the Fourth International Conference on Web Information Systems Engineering*, pp. 209.
25. O'Hara, K. and A. Sellen. A comparison of reading paper and on-line documents. *Proceedings of Proceedings of CHI'97*, pp. 335 - 342.
26. Olsen, D.R., T. Taufer, and J.A. Fails. ScreenCrayons: annotating anything. *Proceedings of UIST'04*, pp. 165 - 174.
27. OneNote, (<http://office.microsoft.com>).
28. Price, M.N., G. Golovchinsky, and B.N. Schilit. Linking by inking: trailblazing in a paper-like hypertext. *Proceedings of Conference on Hypertext and Hypermedia*, pp. 30 - 39.
29. Rekimoto, J. Pick-and-drop: a direct manipulation technique for multiple computer environments. *Proceedings of UIST '97*, pp. 31-9.
30. Rekimoto, J. Time-machine computing: a time-centric approach for the information environment. *Proceedings of UIST'99*, pp. 45-54.
31. Rekimoto, J. and M. Saitoh. Augmented surfaces: a spatially continuous work space for hybrid computing environments. *Proceedings of CHI'99*, pp. 378 - 385.
32. Saund, E. and E. Lank. Stylus input and editing without prior selection of mode. *Proceedings of UIST'03*, pp. 213 - 216.
33. Schilit, B.N., G. Golovchinsky, and M.N. Price. Beyond paper: supporting active reading with free form digital ink annotations. *Proceedings of CHI'98*, pp. 249 - 256.
34. Sellen, A.J. and R.H.R. Harper, *The Myth of the Paperless Office*. 1st ed. 2001: MIT press.
35. Sensiva, (<http://www.sensiva.com/>).
36. Stifelman, L., B. Arons, and C. Schmandt. The audio notebook: paper and pen interaction with structured speech. *Proceedings of CHI'01*, pp. 182 - 189.
37. Wellner, P., Interacting with paper on the DigitalDesk. *Communications of the ACM*, 1993. **36**(7): p. 87 - 96.
38. Wilcox, L.D., B.N. Schilit, and N. Sawhney. Dynamite: a dynamically organized ink and audio notebook. *Proceedings of CHI'97*, pp. 186 - 193.
39. Zeleznik, R., T. Miller, L. Holden, and J.J. LaViola, Fluid Inking: An Inclusive Approach to Integrating Inking and Gestures (under submission) Dept of CS, Brown University, 2005,
40. Zhao, S. and R. Balakrishnan. Simple vs. compound mark hierarchical marking menus. *Proceedings of UIST'04*, pp. 33 - 42.