Moving Markup: Repositioning Freeform Annotations

Gene Golovchinsky and Laurent Denoue
FX Palo Alto Laboratory, Inc.
3400 Hillview Ave., Bldg. 4
Palo Alto, CA 94304,USA
{gene, denoue}@fxpal.com

ABSTRACT
Freeform digital ink annotation allows readers to interact with documents in an intuitive and familiar manner. Such marks are easy to manage on static documents, and provide a familiar annotation experience. In this paper, we describe an implementation of a freeform annotation system that accommodates dynamic document layout. The algorithm preserves the correct position of annotations when documents are viewed with different fonts or font sizes, with different aspect ratios, or on different devices. We explore a range of heuristics and algorithms required to handle common types of annotation, and conclude with a discussion of possible extensions to handle special kinds of annotations and changes to documents.

KEYWORDS: freeform digital ink, annotation, repositioning annotations, dynamic document layout

INTRODUCTION
When people read for pleasure, they relax on the couch, lounge in bed, sprawl on the beach, or even stand in the train. Although work-related reading can take place in similar environments, chances are the reader has a pen or pencil in hand, and scribbles, highlights, underlines, comments, and otherwise engages with the reading material to understand it, to select important points, to re-read, to prepare for writing. [8] The work practice of annotation is common in the paper world, and in part accounts for the observation that people print documents they had retrieved online to read them [1].

When building earlier versions of XLibris [13], we treated our software as a substitute for a printer: a document was loaded into XLibris, paginated, and presented to the user. The pagination was fixed at import time, and the freeform digital ink marks made by the user were positioned relative to the page.

This was a useful simplification that allowed us to explore the richness of interaction possible through freeform digital ink [13,14]. In some cases, however, it is desirable to relax this assumption, to allow the pagination to change, but still preserve the ability to view and create annotations.

Even more so than textual annotations, freeform annotations rely on their context to communicate meaning: shifting an underline just a few pixels relative to the underlying text can render it confusing, meaningless, or even contradictory to the annotator's intent. The algorithms we present in this paper leverage existing strategies for associating annotations with documents (e.g., [3,11]), and extend them to accommodate the nuances of positioning freeform digital ink marks.

There are several reasons the layout of a document may change, as illustrated in Figure 1 and Figure 2. The user may change the font, the document may be viewed with a different aspect ratio or on a different device, it may be reformatted with a different style sheet, or the document itself may change. The practical importance of such transformations is illustrated by the recent announcement from Adobe that PDF, its traditionally page-oriented document format, would now be supported on Palm devices [2]. The general case of an edited document involves two steps: detecting differences between document versions, and repositioning annotations on the new version; formatting changes require only the second (repositioning) step. We chose to solve the repositioning problem first; detecting and representing changes between documents remains beyond the scope of our present work.

In the following sections, we start with a description of XLibris, our system for reading and annotating documents. We describe the challenges posed by freeform annotations and distinguish them from other annotations systems. We then present our algorithms to reposition freeform marks over changing document layouts. This includes attaching the annotations to a specific document fragment, storing the annotations relative to this fragment, and repositioning the annotations over the document when its layout has changed. We then discuss evaluation and redesign, and conclude with a discussion of future work.

FREEFORM ANNOTATIONS
For the purposes of the following discussion, we can distinguish three kinds of annotation: explicit selection, textual annotation, and freeform digital ink. Explicit selection may be implemented as a font or style change (e.g., underlining, highlighting) that uses conventional text formatting and layout conventions to render the annotation. Textual annotation associates a reader-created text chunk with a passage of an existing document. Finally, freeform digital ink allows the reader to mark anywhere on the document, does not constrain the shape of the marks, and does not impose any structure on them.

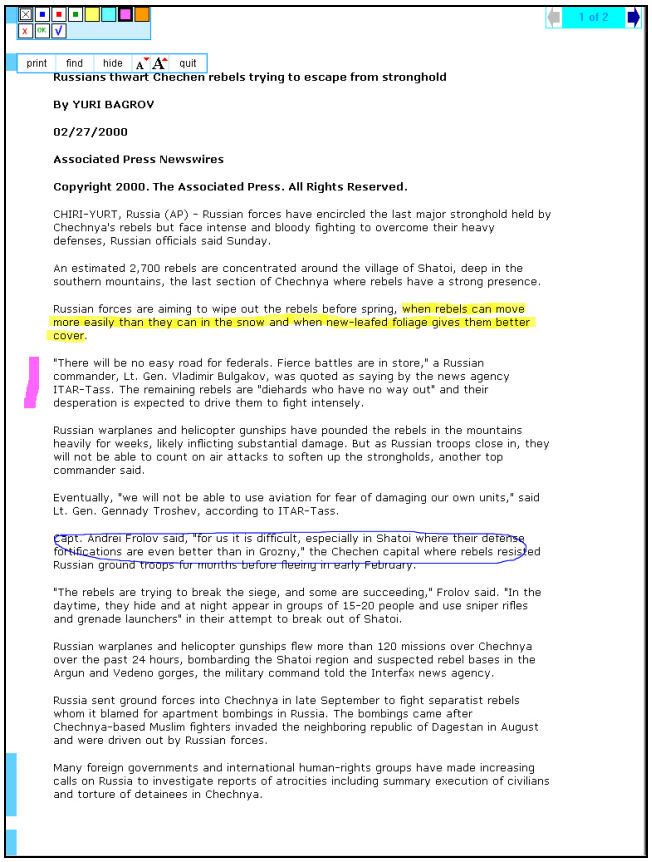


Figure 1. Annotated page in XLibris.

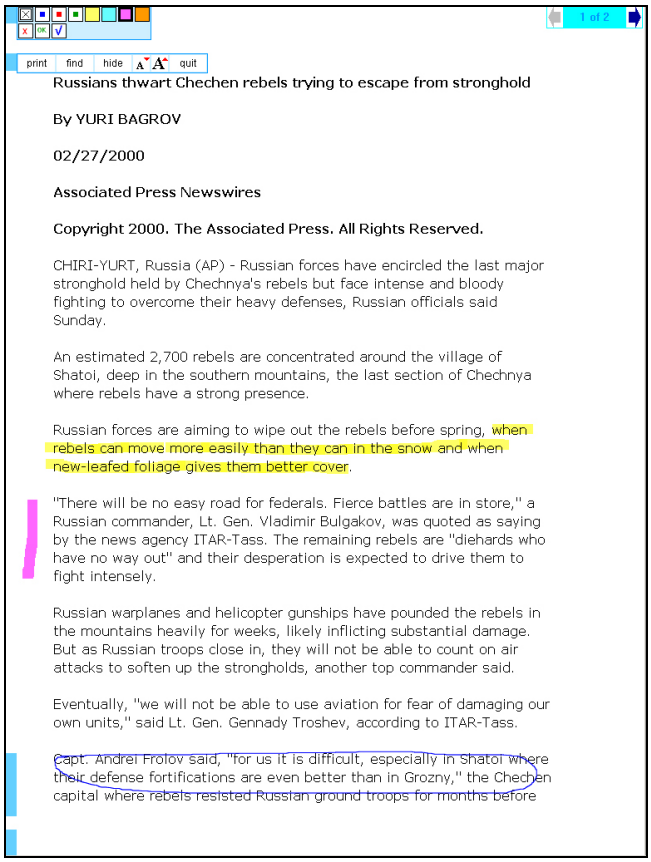


Figure 2. Same page, with a different font size, showing repositioned and resized annotations.

Thus, freeform annotations differ from annotations currently found in electronic documents on two aspects. First, freeform annotations are not *explicitly* attached to a text passage in a document. In conventional annotation systems, the user explicitly selects a text passage to which the subsequently-created annotation will be anchored. With freeform annotations, we first need to find the fragment which is referred to by the annotation.

Secondly, freeform annotations are drawn by the user, whereas explicit selections and anchors for textual annotations are constrained by document objects (e.g., text, images, etc.). When the document changes and words are repositioned in a page, the layout engine automatically reapplies the emphasis based on the new size and position of the text. But drawing freeform annotations when the position or size of the words have changed is not trivial: the original shape of the annotation should be preserved (because it carries meaning) while accommodating the new position, size, and pagination of the content.

A BRIEF OVERVIEW OF XLIBRIS

XLibris was designed to leverage existing practices of reading paper documents. Although it runs on any Win32 platform, XLibris is best used with a pen input device such as the Fujitsu 3400X tablet with a touch screen and stylus. The full screen is used to render a page of a document. Furthermore, documents are rendered in portrait rather than landscape mode, again to accommodate existing practice.

XLibris consists of several components: a parser and renderer for each document format, a DOM-inspired [16] framework for storing metadata and annotations, a set of interface modes (called views), and a widget hierarchy. Most of the interaction with documents in XLibris is centered on creating and manipulating annotations. Annotations are captured as stroke (polyline) information, with a few additional attributes such as the color, type of ink, time, and identity of the user. For each user, each document has an associated XML file that stores its annotations.

The separation of annotations from documents has several architectural advantages: it is possible to annotate documents that cannot (or must not) be modified, it is relatively compact, and supports sharing of annotations without requiring the retransmission of the documents themselves. Finally, our representation of annotations allows them to be transformed and repositioned as the underlying document changes. The rest of the paper is dedicated to the description of these algorithms and to the implications for the way XLibris can be used given this functionality.

REPOSITIONING ANNOTATIONS

The ability to reposition freeform annotations is important because it allows the user to change font size and family to accommodate changes in viewing preferences, because it

allows users with different preferences to share annotations, and because it allows the same document to be viewed on different devices (or in different-sized windows) with different screen sizes, resolutions, and fonts.

It is important to note that changes in window size or device screen size cannot always be accommodated by merely changing the scale factor of the display. A PDA does not use a proportionally smaller font size compared with a laptop; it displays less text. Subtler differences may occur even on the same device with different operating systems: for example, font metrics for Windows 98 may vary just enough from those of Windows NT to create different line breaks in some cases. Thus the ability to reposition annotations not only accommodates new work practices such as sharing of annotations, but also makes the system less brittle when confronted with small changes in layout.

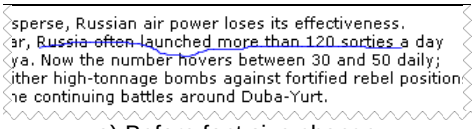
Repositioning annotations involves three steps. First, each annotation needs to be attached to a specific location in the document. This location should not depend on the document pagination. Instead, when the document is repaginated, the location is used to identify the new position of the annotation. Second, the size of the original annotation needs to be adjusted to the new layout (e.g., due to increased font size). Third, the original mark may need to be segmented into several annotations (e.g., due to changed in line and page breaks). We discuss these aspects in the following sections.

Attaching Freeform Annotations to Document Content

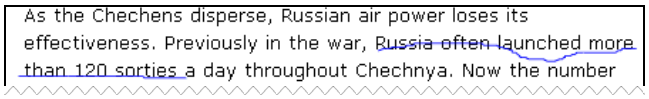
From the user's point of view, freeform annotations are attached to a particular representation of some content, like a line of text, a paragraph, etc. We call this content the anchor of the annotation. For some annotations, the anchor is very specific (e.g., highlighted words). For others, it is not (e.g., annotations in the upper and lower margins). Once an anchor is found, it needs to be represented and stored with the annotation. Several implementations are possible. When document content (as opposed to layout) does not change, we can safely use word positions to represent the anchor point (e.g., words 10 through 15 are highlighted). If content does change, document structure and content may be used to identify corresponding passages in the new document [11,3]. The implementation described in this paper does not consider changes to document structure, although we discuss this topic in future work.

In all cases, the anchor is a set of words, images, or other document objects referred to by the annotation. We have classified annotations into three categories, each of which is characterized by a set of heuristics used to find anchors.

Annotations attached to a line of text. In the first category, the annotations specifically refer to a series of words (and possibly small pictures such as mathematical formulae and icons). Examples include highlighted, underlined, struck through and circled phrases that all encompass a single line of text. In these cases, the anchor is identified as the set of words intersected by the freeform annotation.



a) Before font size change



b) After font size change

Figure 3. Finding the correct content even when the annotations are not perfectly aligned with the words.

Freeform annotations, by definition, do not perfectly intersect words. In Figure 3 for example, the annotation under the word "launched" intersects the words "number" and "hovers" on the next line. But because the annotation primarily intersects words from the first line (e.g., "Russia" to "sorties"), it will also be attached to the word "launched" rather than to the text below it.

After it has been created (i.e. when the user lifts the pen), an annotation from this category is split into multiple marks, each one being attached to a single word (or picture, if the annotation is drawn over a picture). The original annotation is split between two or more consecutive words or pictures as shown in Figure 4. Splitting annotations in this manner makes it possible to reposition them when text reflows (Figure 3b). Each derived annotation overlaps with its neighbors to avoid gaps between marks when they are repositioned in a new pagination. These individual marks are still linked, as discussed in the next section.

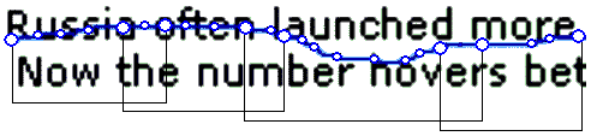


Figure 4. An underline annotation split into four segments based on the anchoring words.

Margin bars and circled passages. Annotations such as margin bars and circled passages do not refer to particular words, but rather to a word range (from several lines to a paragraph). In a new pagination, they will be stretched vertically as this word range is rendered.

Margin bars are vertical marks drawn in the margins of the page, where verticality means that the height of the bounding box of the annotation is 2.5 times bigger than its width. Circled passages are identified as annotations overlapping more than one line of text (as opposed to circled phrases that only overlap words on one line).

Examples of word ranges associated with these annotations are shown in Figure 5. Projecting the bounding box of the annotation horizontally, we find the first word in the page that intersects the bounding box of the annotation (i.e.

"[Correspondent]" for the margin bar and "[Murayev]" for the circled passage). Similarly, we find the last word in the page that intersects the projected bounding box (i.e. "it" for the margin bar and "inside" for the circled passage).

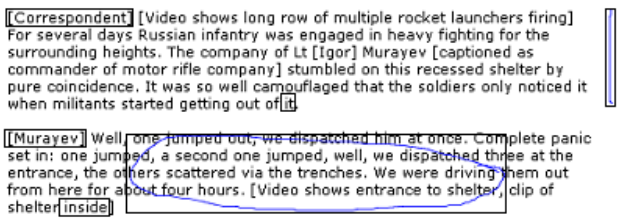


Figure 5. Computing the content associated to margin bars and large circled passages. Rectangular boxes have been added in this figure for emphasis; they do not appear on the screen.

Others. In the third category, we find marks that are not obviously attached to any content such as handwritten notes and symbols written in the margins of the document (Figure 6). Starting with annotations written in the left and right margins of the page, we apply the same method as for margin bars: the anchor is the first and last words on the lines whose bounding boxes intercept the vertical position of the annotation.

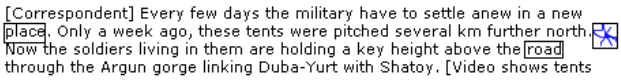


Figure 6. Other kinds of annotations, with boxed words and marks.

The only difference between marks in this category and margin bars is that these annotations will not be resized as the document layout changes: symbols and handwritten notes will remain the same, thus not altering their comprehensiveness. In some cases, however, they may need to be scaled down to fit on a reduced-size display. We investigate these cases in the future work section.

Annotations written in the upper (lower) margins of a page are associated with the first (last) line of text on the page. When the document is repaginated, these annotations are displayed in the upper (lower) margin in the new pagination on the page where the words appear.

Storing Annotations

Earlier versions of XLibris treated annotations as static entities: once created, they stayed put until the reader erased them. The fixed pagination and layout of documents made it possible to store annotations using page-based coordinates.

By allowing document layout to change, we can no longer treat the page as a persistent object. Increasing the font size may cause new pages to be created; decreasing it can reduce the number of pages. And even if a change in layout does not affect the number of pages, it can affect the ranges of words associated with each page.

Thus it is no longer possible to represent annotations using page-based coordinates. Whereas the original design stored stroke coordinates relative to the page, each annotation now saves the word range (in the document) with which it is associated, and the bounding rectangles of those words on the page at the time the mark is created. Coordinates are stored relative to the origin of the text bounding box.

In addition, a transient set of strokes is created for each pagination. Roughly speaking, transient marks are copies of persistent ones, but they differ from their persistent sources in three ways: first, they may be deleted without affecting the persistent marks from which they were derived. Second, their coordinates are transformed to account for the new location and size of the words to which they are anchored. Finally, transient strokes may be split across page or column boundaries, in which case a single persistent stroke is associated with two or more transient ones. While transformations are made to the transient strokes, the corresponding persistent objects are left unaltered, recording the layout (i.e., the sizes of the rectangles of words to which the marks are anchored) at the time of their creation. Thus sibling persistent strokes may represent annotations made on different devices or with different document layouts. Persistent strokes are never modified after they have been created to eliminate the propagation of round-off errors. Note that the transient – persistent distinction is made only when storing and repaginating the document; all other code treats the two interchangeably.

Multiple transient strokes created from a single persistent stroke are linked by the system and manipulated as a unit. If a transient annotation is erased by the user, all linked transient annotations and the original annotation, are erased. This linkage ensures predictable interaction: even if the system splits an annotation into several transient ones, the user does not perceive these fragmentations, and thus expects the annotations to behave as one.

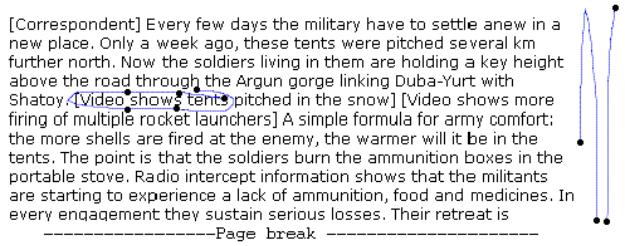


Figure 7. Examples of noncontiguous segments defining a transient annotation. The margin bar on the right has been split across a page break. Black dots are added in the figure only to illustrate segment ends.

When a persistent annotation is split into several transient ones, its polyline may be split into several noncontiguous segments assigned to each transient stroke. Splitting may occur when a stroke is created (for circled phrases, highlights and underlines), or when it spans columns or pages. In the example of Figure 7, the black dots on the circled phrase indicate the breaks in the original stroke

made when the stroke was split to anchor it to the words "video," "shows," and "tents." Note that "tents" receives two parts of the original stroke: portions from the beginning and end of the stroke that intersect its bounding rectangle.

Margin bars and circled passages are split when they fall across page breaks. Since we cannot anticipate where the break will be, we defer splitting the stroke until a new layout is computed. The polyline of the original stroke is split into two regions: one cropped by the bounding rectangle of the first page, and the other cropped by the bounding rectangle of the second page. As was the case with the word "tents" above, more than one part of the original stroke may be assigned to a particular transient stroke; margin bar in Figure 7, for example, is split into three parts: the first and the third are shown, and the second falls onto the next page (not shown).

Transforming Annotations

When a document is loaded, or when its pagination is invalidated by a layout change, the document is paginated and transient annotations derived from persistent ones are positioned to correspond to the new layout.

How annotations are transformed depends not only on the new layout of the document, but on the type of annotation as well. Annotations anchored to specific words or images are repositioned and resized relative to the new position and size of anchoring objects. Annotations anchored to passages are adjusted based on the new passage size. Finally, some annotations are repositioned without adjusting their size. We also need to handle two special cases: some annotations must be split across page boundaries (see Figure 9), and clipped regions must be adjusted to reflect the new pagination. In the following sections, we discuss these transformations in detail.

Stretching annotations. Annotations pertaining to the first category – highlights, underlines and circled phrases – are split at creation time into segments that correspond to words or images. To reposition the annotation, we get the new bounding box (of width W and height H) of the anchor object. We also retrieve the original bounding box stored when the annotation was created, of width *w* and height *h*. The vertices of the original annotation (*x*,*y*) are then transformed into new positions (X,Y) to accommodate the new bounding box:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} \frac{W}{w} & 0 \\ 0 & \frac{H}{h} \end{bmatrix}$$

No offset is required, as stroke coordinates are stored relative to the bounding box of the anchoring object. This is true even if the stroke falls outside of the bounding box of the anchoring word. (See, for example, the word "launched" in Figure 4).

For annotations identified as margin bars and circled passages, we retrieve the associated word range, and then locate the first and last word of this range in the new layout. In the example of Figure 8, these words are "jumped" and "hours." These two words do not necessarily appear at the start and end of a line in the new pagination, so we find the new first word and last word on the lines containing the stored words (they correspond to the words "[Murayev]" and "shelter" in Figure 8). The bounding box of this new word range (of size (W,H)) is computed. We also retrieve the old bounding box of the annotation and stretch the coordinates of the annotation as described above.

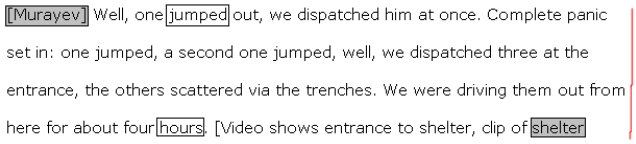


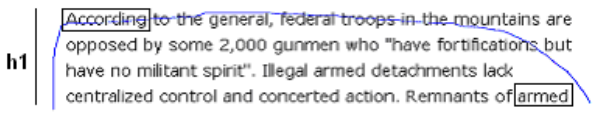
Figure 8. Finding the new bounding box for a margin bar (or circled passage).

Annotations identified as symbols or handwriting are not stretched. The origin of the repositioned stroke is scaled to position it relative to the associated words, but the size remains unchanged.

Splitting annotations across pages. Annotations that span multiple words (e.g., margin bars, circled passages) may need to be split across page or column boundaries if the anchor passage reflows across a page or column, as shown in Figure 9. Annotations such as highlights, underlines and circled phrases, on the other hand, are split at word boundaries at creation time. A persistent annotation needs to be split into several transient annotations when its anchor does not appear on one page.

To split and stretch a persistent annotation, we compute the height of the new bounding box of its anchor. In Figure 9, the anchor was the word range starting from "According" to "continued." In the new pagination, these words appear on two pages. We then compute the bounding boxes for each page, and record their heights (h1 and h2 in Figure 9). The persistent annotation is split into two parts, and scaled based on the vertical extents of the two segments relative to the vertical extent of the original passage.

Each vertex in the persistent stroke is examined and placed into one or the other transient stroke based on the rectangle into which it falls. The word range of each transient annotation is then set to the new word range (e.g. "According" to "armed" on the first page and "gangs" to "continued" on the second page).



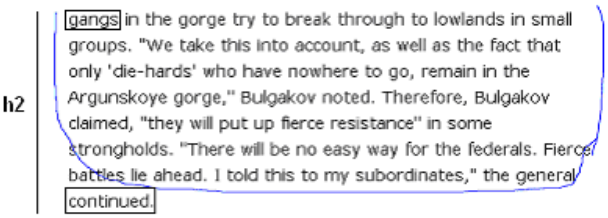


Figure 9. As the font size increases, the circled paragraph is split across two pages, and so is the blue mark.

Clippings. In XLibris, annotated segments of documents may be extracted and listed in a special view we call the annotation view. From this view, users can choose a particular annotated area, called a clipping, and paste it into a notebook page. Figure 10 shows three clippings, each centered on an annotated passage. A clipping is a dynamic view on a page, not a static bitmap. Users can resize the clipped rectangle by dragging corner handles, and they can add new annotations directly into the clipping. Annotations added in the document after the clipping has been pasted into a notebook page are reflected in the clipping. Annotations drawn on a clipping are visible in the document. Finally, erasing a mark in one view causes it to be removed in the other. See [10] for a more detailed discussion of clippings and the work practice that inspired them.

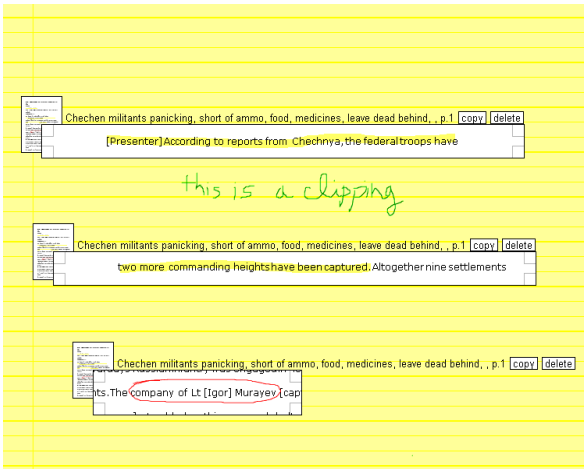


Figure 10. Clippings are pasted into the notebook view.

Each clipping object stores its bounding box and a reference to the corresponding text range. At runtime, this text range compared to the current pagination, and the page containing most of the span is associated with the clipping. The other remaining challenge with clippings is to adjust the text within the clipping given that the text referenced by the clipping originally will have shifted out of that region. We had considered several possibilities, including:

- Treat the clipping as a bitmap and leave it exactly as it was at creation time
- Resize the box to accommodate the new text layout
- Leave the box alone, and let the user adjust it
- Anchor the box on the first or last word of the range, but let the rest of the text flow out of the box.
- Keep the clipping box size the same, but center it on the rectangle corresponding to the new location of the previously-visible text.

We decided not to treat it as a bitmap in part because it would be difficult for the reader to anticipate the relationship between what was seen in the clipping and the current layout of the referred-to document, and in part because it would require us to store an ever-increasing number of bitmaps. We chose not to resize the box as that might affect the layout and organization of ideas on the notebook page, again leading to unpredictability from the user's perspective. Leaving the box alone might work for small changes in layout, but would make it difficult to recover the original passage if the layout change was too great.

Finally, the decision was reduced to anchoring the clipping on some word in the clipped passage, or centering it on the new area. We chose the latter because words in this case were not always the primary reason for the existence of the clipping; in fact, in some cases, there were no words at all. Thus we decided to keep the clipping box size constant, and to center it on the bounding box of the passage. If no passage was available, the original position was retained. For example, in the first clipping shown in Figure 10, the word range is "[Presenter]" to "have". When the document layout changes, we find the position of this word range in the new pagination. The bounding box of the word range gives us a new rectangle. We center the old rectangle inside this new rectangle. This gives the new clipped rectangle, which is extracted and pasted on the notebook page. When the content associated with a clipping that is split across two or more pages, we display the page that includes the majority of the original annotated passage.

EVALUATION AND REDESIGN

We deployed the modified version of XLibris in support of a peer-review task. Two volunteer groups of authors engaged in writing journal articles nominated reviewers (three per group) to read and annotate the manuscripts. Reviewers' annotations were then merged and presented to the authors of the manuscripts. Although the goal of the study was to explore the potential for annotation sharing, as a side effect we observed how well our system handled real annotations.

Although much of the algorithm worked as expected, several failures were observed. The most significant problem involved repositioning printed (rather than handwritten) text. Some users reported that they found it easier to write in cursive, while others stuck to printing (making individual strokes for each letter and for parts of some letters). This not only created a large number of

stroke objects for each document, but also fooled the repositioning algorithm into misclassifying some strokes. Typical errors involved classifying vertical lines (e.g., the letter t or l) as margin bars, and horizontal marks as underlines. When this text was repositioned, parts of a single word could wind up in different places, an altogether unacceptable outcome.

Our solution is to group strokes as they are created into logical chunks such as words, double-underlines or margin bars, etc. This allows us to transform separately-created strokes as logical units, reducing the likelihood of unintentional fragmentation of annotations.

Ink grouping

Printed handwriting is characterized by rapid creation of spatially proximate strokes. Thus temporal and spatial proximity may be used to group individual strokes into compound objects [5]. Annotation marks differ from other freeform marks (e.g., notes on a blank page) because their meaning may also depend on their position relative to the document text (in addition to time, relative position and shape). Thus we used four sources of information to group strokes: time, location (with respect to other strokes), position with respect to the document, and shape. Of these, time deserves the most attention.

We categorized manually strokes (or groups of strokes, when appropriate) made by participants in our study as handwriting, circled passages, underlines, margin bars, and margin marks. We then analyzed the handwriting groups to derive a time threshold for grouping marks. We measured time differences between strokes that made up each word (end of one stroke to start of the next); the distribution is shown in Figure 11. The data exhibit a peak around 300 ms, followed by a gradual decline as the difference in time increases. The large spike at the end represents strokes that were added more than two seconds later; these strokes were typically added after the person had written elsewhere on the page.

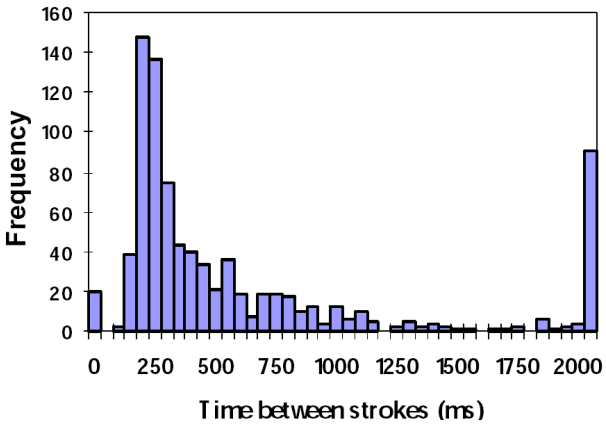


Figure 11. Distribution of time between strokes for a single word (data from two participants).

From this data alone, however, it is not clear how to proceed. We would like to be able to group strokes into word-level groups, and to distinguish among such groups and between handwriting and non-handwriting marks. But about 11% of strokes that belong to different words (or to other marks) occur within 500 ms (see Figure 12). The problem is more severe, however: it is likely that nearby words will be written sequentially; this should lead us to discount the right-most column in Figure 12, producing the distribution in Figure 13. These data suggest that time is not a good measure by which to discriminate between words; spatial location must be used for that purpose.

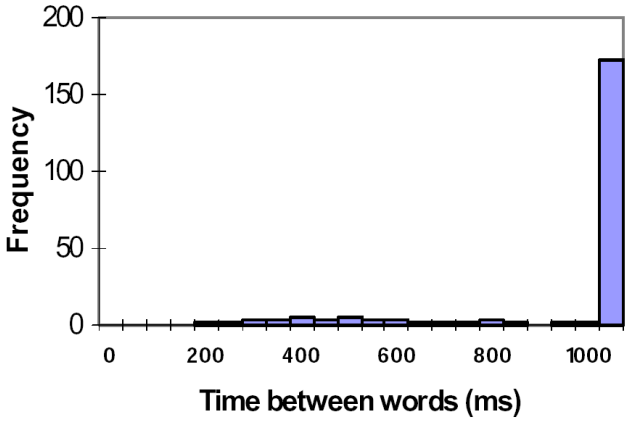


Figure 12. Short delays between words. Delays over 1 second represent 172 of 213 measurements.

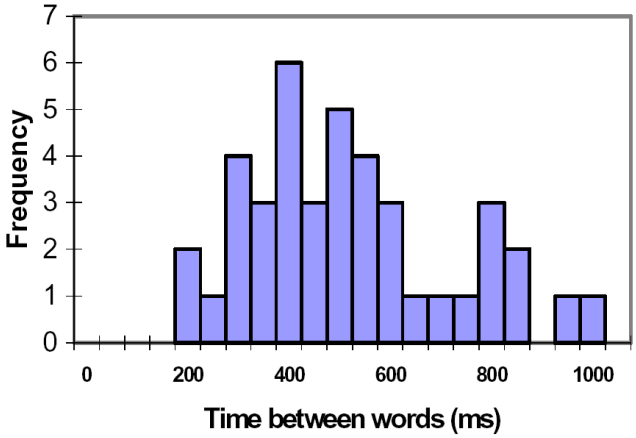


Figure 13. Distribution of short delays between words.

Merging strokes based on shape and time is a difficult problem: if you detect shape first (e.g., a margin bar) and then refuse to merge this mark with subsequently-created proximate marks that are not margin bars, you may fail to group the verticals of t's or l's with the rest of the word; if you merge first and classify later, you may fail to detect margin bars. Our algorithm prefers to keep strokes together, as a false grouping is less likely to be affected adversely by repositioning than merging that was not detected.

Thus our algorithm loops over all strokes on each page in the order they were created and performs the following tests to determine if consecutive strokes (s1 and s2) should be merged:

1. If s1 or s2 are underline, highlight, circled phrase, circled passage, or callout, they are not merged.

2. If s1 and s2 overlap but the diagonal of s2 is three times larger than that of s1, the strokes are not merged; a smaller stroke, however, could be merged into a larger one. This allows us to group marks such as dots on the letter i and cross bars on the letter t with the rest of the word, even if the marks are made much later.

3. If the two strokes are in the same margin (left or right) and overlap in their y coordinate, and neither is a margin bar, they are merged.

4. If two strokes occur within 500 ms of each other, they are merged.

5. If the two strokes overlap, they are merged.

The implication of steps 3, 4, and 5 is that words in the margin tend to be grouped together, whereas words written in the text tend to be grouped apart because they tend not to overlap spatially. This distinction allows handwritten words in the body of the document to ride along with the associated words as the document is reflowed.

Figure 14 shows two annotations on a document before the font size is changed. The boxes around the ink illustrate the system's grouping of strokes: the annotation on the top right is ungrouped; it consists of a margin bar and a number of handwritten strokes. The annotation on the bottom left consists of several grouped strokes. Figure 15 shows the same annotations repositioned after the font size has been increased. The ungrouped annotation (top right) fragments; the grouped one is preserved correctly.

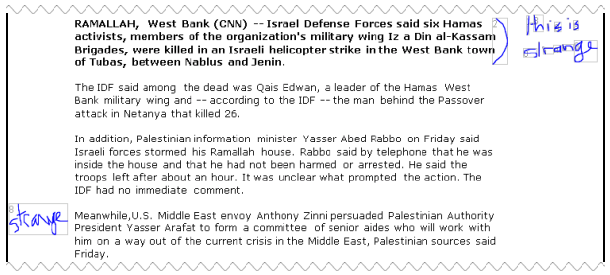


Figure 14. Annotations on a page before resizing.

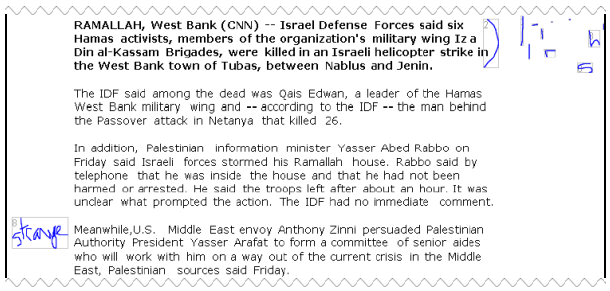


Figure 15. Annotations on a page after resizing. The top right annotation was repositioned without grouping strokes; the bottom left with grouping.

FUTURE WORK

This work is by no means complete. We've made some preliminary steps, but the system needs to be made more robust to handle the variety of annotations found in practice. Other issues include alternative approaches to ink grouping, and handwriting recognition. In this section we set out some of these issues, and suggest directions for further exploration.

Ink grouping

Our ink grouping algorithm is based on heuristics and can potentially become quite complicated. It may be possible to replace it with a machine learning approach that would categorize and group strokes. Again, the problem of sequence – should it categorize or group first – suggests that a multipass algorithm may be warranted. We may pursue this direction if our heuristic algorithm fails to account for a significant fraction of annotations.

Handwriting

It should be useful to develop heuristics about repositioning and rescaling handwritten text. It may be possible to identify the span of the document on which the written text comments, and to anchor the marks to that span. A focused experimental exploration of annotation practices (examining paper-based annotation, for example) should allow us to identify likely anchoring points.

Legibility may be preserved either by limiting scaling, or by incorporating zooming techniques similar to those described in the Fluid Documents work [4].

Finally, it may desirable under some circumstances to add handwriting recognition capability to the system. It is important to understand the role that recognized text will play in such interfaces, and not to sacrifice the perceptual advantages of freeform ink in pursuit of text. Nonetheless, value may be added by performing handwriting recognition on ink groups identified as handwriting candidates. Recognition should be performed offline so that it does not interfere with the process of creating the marks. Furthermore, processing the marks asynchronously should allow more powerful (and therefore more time-consuming) algorithms to be applied, thereby increasing the reliability of recognition. A separate interface for correcting recognized text should also be added.

Changing documents

When building this system, we assumed that a document's text would remain static. Changes in associated markup (new style sheets, for example) were allowed, but the words were not allowed to change. Attention has recently been given to the problem of repositioning annotations over a new version of a document. Several approaches have been suggested, such as using the document structure as well as content to anchor annotations in a document [11]. But positioning annotations robustly over changing document contents poses challenges other than technical. User expectations are particularly important and a recent user study has started to explore this problem [3].

Freeform digital ink annotations are particularly sensitive to the effectiveness of such algorithms. It is not sufficient to locate the same paragraph or chunk of text, if some of the words have changed. Open issues remain on how to handle changes to a passage that has been highlighted: should inserted words be highlighted as well, for example?

CONCLUSIONS

Our experience has shown that paper-like annotation practice can be an important interaction technique for managing analytic reading. Recording freeform annotations frees the reader to focus on the document; having a computer mediate the annotation process makes it possible to overcome some of the limitations of paper. We addressed some of these limitations – static content and poor support for re-reading and re-retrieval – in our earlier work. In this paper, we described a system that overcomes the fixed layout limitation typical of paper and of earlier freeform annotation systems (e.g., XLibris [13], Wang Freestyle [6], etc.). The system described here adjusts the position and size of freeform marks to reflect changes in document layout caused by user preferences, changes in devices, platform differences, etc. Ink grouping was discovered to be an important factor in improving the performance of the ink repositioning algorithm. This research increases the usability and usefulness of freeform annotations in real-world situations that involve a variety of systems and devices with different display characteristics.

ACKNOWLEDGMENTS

We thank Morgan Price and Bill Schilit for the original discussion of repositioning annotations, and Andreas Girgensohn and Matt Cooper for their comments on an earlier draft. We thank Jim Baker for supporting this research.

REFERENCES

1. Adler, A., Gujar, A., Harrison, B., O'Hara K. and Sellen, A. A diary study of work-related reading: design implications for digital reading devices, in *Proceedings of CHI98* (Los Angeles, CA, April 1998), ACM Press, 241-248.

2. Adobe press release. Available on the web at http://www.adobe.com/aboutadobe/pressroom/pressreleases/200104/20010410umbrella.html

3. Bernheim Brush, A.J., Bargeron, D., Gupta, A, Cadiz, J.J. Robust Annotation Positing in Digital Documents. In *Proceedings of CHI'2001 Human Factors in Computing Systems* (March 31–April 5, Seattle, Washington), 2001.

4. Chang, B-W, Mackinlay, J.D., Zellweger, P.T., and Igarashi, T. A negotiation architecture for fluid documents. In *Proceedings of UIST'98*, 1998, 123-132

5. Chiu, P. and Wilcox, L. *A Dynamic Grouping Technique for Ink and Audio Notes.* In *Proceedings of UIST'98 Annual symposium on User Interface Software and Technology*, ACM Press, San Francisco, California, 1998, pp. 195-202.

6. Levine, S.R. and S.F. Ehrlich, The Freestyle System: A Design Perspective. *Human-Machine Interactive Systems*, ed. A. Klinger. Plenum. pp. 3-21, 1991.

7. Magellan software. Available at htpp://www.bcl.com

8. Marshall, C.C. Toward an ecology of hypertext annotation, in *Proceedings of ACM Hypertext '98*, (Pittsburgh, PA, June 1998) ACM Press, 40-49.

9. Marshall, C.C., Price, M.N., Golovchinsky, G., Schilit B.N. *Collaborating over Portable Reading Appliances.* In *Personal Technologies*, Vol. 3, No. 1, 1999.

10. Marshall, C.C., Price, M.N., Golovchinsky, G., Schilit B.N. Designing e-books for legal research. In *Proceedings of ACM and IEEE Joint Conference on Digital Libraries* (Roanoke, VA, June 2001).

11. Phelps, T.A, Wilensky, R. Robust Intra-document Locations. In *Proceedings of WWW9 World Wide Web Conference* (Amsterdam, The Netherlands, May 2000).

12. Price. M.N., Golovchinsky, G. and Schilit, B.N. Linking By Inking: Trailblazing in a Paper-like Hypertext. In *Proc. Hypertext '98* (Pittsburgh, PA, June 1998), ACM Press, pp. 30-39.

13. Schilit, B.N., Golovchinsky, G., and Price, M.N. Beyond paper: supporting active reading with free form digital ink annotations, in *Proceedings of CHI98* (Los Angeles, CA, April 1998), ACM Press, 249-256.

14. Schilit, B.N., Price, M.N., Golovchinsky, G. Digital Library Information Appliances. In *Proc. Digital Libraries '98* (Pittsburgh, PA, June 1998) ACM Press, 217-226.

15. Raggett, D. *Clean up your Web pages with HTML TIDY*. (August 2000) Software available on the web at http://www.w3.org/People/Raggett/tidy/

16. W3C Recommendation. *Document Object Model Level 1 Specification.* Available on the web at http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/

17. W3C Working Draft. *XML Pointer Language (XPointer) Version 1.0*. Available on the web at http://www.w3.org/TR/2001/WD-xptr-20010108/