

Recdit: A Text Editor With a History

Will Thimbleby
FIT Lab
Computer Science
Swansea University
Singleton Park
Swansea, SA2 8PP, UK
will@thimbleby.net

ABSTRACT

A novel text editor, Recdit, provides a complete character by character history of text documents is introduced and discussed. Recdit provides the ability to see an entire document's history and to "scrub" through it like a movie.

Recdit also provides highlighting and graphs that tracks and shows overviews of a document's edits and changes. The ability to do this provides many novel uses. These features are discussed and are useful for both single and multiple authors, both of which Recdit supports.

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

General Terms Design, Experimentation

KEYWORDS: Undo, history, versioning, track changes, collaborative writing

INTRODUCTION

Tracking the changes in text documents is an essential task [?], especially when multiple users are editing the same document. Recdit provides a character by character recording of the entire history of text documents. This history provides the ability to extensively track changes in text documents.

The primary contribution of this paper is the combination of a complete edit history, its visualisation and a controlling user interface. These provide novel ways of tracking document changes and interacting with them.

Some of the features of Recdit provide are:

- Multiple concurrent authors
- A complete edit history
- A slider user interface to control viewing the history
- Trails which highlight the last few edits
- Graphs providing an overview of the entire history
- Sideways text layout that provides more structure for the text

Recdit is a fully working text editor designed for Mac OS X. This paper was written in Recdit of which every edit can be reviewed. The application and paper are available to download from <http://will.thimbleby.net/truetext/>

INSPIRATION

This text editor was inspired primarily by seeing users enjoying the undo-clock in the pen-based interactive calculator presented in [?, ?].

The calculator instead of providing a discrete step-based undo similar to most modern user interfaces, provides an undo that is linear and smooth. The undo user interface of the calculator is presented to the user as an analogue clock that allows the user to manually set the time by rotating the clock's hands.

Users really enjoyed interacting with the undo-clock when using the calculator. Like most undo systems, users did not often use the undo feature, but they still liked to play with it and enjoyed the interaction.

This success led to the original design question for Recdit: "What would a text editor look like if it had a similar undo-clock?"

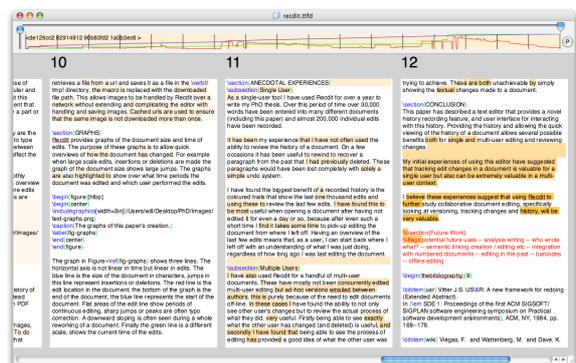


Figure 1: Recdit's user interface.

Time-machine computing [?] describes a whole system containing multiple applications which can be navigated in time, Recdit provides the same navigation in a more focused and refined user interface specifically for textual documents.

DESIGN

Implementation

To provide the similar ability to rewind the edits to a text document as the calculator provides for equations, Recdit records every single event as the user type. The recorded event data includes a timestamp, the current user, and the edit data itself. As a user types, each modification, the document records an event, every single character press is recorded as a individual event.

The edit history of a document is recorded in the saved files so that the history of a document is not lost between editing sessions.

When a document is opened in Recdit the document replays its creation from the start. This allows keyframes of the document's state to be recorded throughout the its creation. To jump to a location in history the document is "wound" forward from the nearest keyframe by replaying the edits made between the keyframe to the new location. The keyframes that are stored every couple hundred edits allow jumping to any location in the document's history to be very fast and allow scrubbing through the history to be immediate and interactive.

User Interface

A screenshot of editing this paper in Recdit is shown in Figure ???. The text is laid out sideways in individual pages with page numbers at the top. Some highlighting of this text is visible showing the last few edits made to this paper when the screenshot was taken. At the top of the window is a graph of the documents state throughout its creation and a slider that controls the currently displayed version of the document in history.

If the user does not interact with the history features of the editor then the editor interacts identically to a simple text editor.

The main user interface for interacting with the history of the document is provided by the slider at the top of the document window. Using this slider it is possible to jump to any point in the document's history and to scrub back and forth to see the changes as they were made. Dragging the slider left moves the document back in history, up-to the beginning of the document, and dragging it right forward in history. This works like scrubbing through a movie. As the slider is moved the document is updated instantly, the state of the document and the slider are never inconsistent.

A slider is used instead of the undo-clock used by the calculator because it provides instant interaction for moving to specific points. Another benefit of a slider is that it can be overlaid on-top of graphs showing the state of the document at any time. A disadvantage of a slider is that as the document has a history in the thousands or tens of thousands edits the slider becomes more inaccurate, each pixel of the slider's position representing many edits.

Using the slider to scrub back and forth in the history of the document shows the edits that created the document. This can help a user to understand the process of the document's creation.

By replaying a document's creation when it is opened, the user sees the document recreated from the beginning character by character. This provides the user with a fast-forwarded reminder of how the current state of the document was reached.

MULTIPLE USERS

Group editing documents with multiple authors even with version control systems or concurrent group editors is not a simple task. Other than the technical problem of providing distributed access to the same document, one of the main problems of multi-user editing is keeping track of the changes made by other users [?]. Simple versioning is often used to provide a basic tracking of changes.

Recdit provides networked multi-user concurrent editing capabilities. A document can be served from a server to multiple individual users that connect to the document. This allows multiple users to concurrently edit the document at the same time.

By recording the history of a document Recdit creates the potential to review other user's changes. Allowing a user to potentially rescue paragraphs deleted by other users and to "see" their changes, to see how and what they wrote, corrected and deleted.

UNDO vs. HISTORY

Most undo systems create a tree of edits, usually each branch of the tree except the main current branch is lost. Several interesting undo frameworks have been proposed like US&R [?] and multi-user undo systems like [?]. Recdit is not focused on implementing an undo system. The history and undo are distinct, which makes both simpler to interact with.

The history records all the undo and redo changes, because the undo system is external to the history of a document. Moving back in time it is possible to see mistakes made and undone.

Unlike undo the history of a document is immutable, once an edit has been made it is recorded forever. Although it is possible to view the history of a document it is not possible edit the document in the past.

The history of the document can be represented as a sequence of states on a time based axis. The version of a document that existed at any point in time can be retrieved. No state the document was previously in is irretrievable, at no point is any data ever lost.

TRAILS

As the user types in Recdit a coloured trail is left behind. This appears as a light background colour behind the text, which can be seen in Figure ??. As the a continues to type this colour fades out over time until it disappears after one thousand edits. This means that at any point in the documents history the highlighted trail provides a overview of what the last one thousand edits are. Deletions are not highlighted because they do not leave behind any text.

A benefit of only colouring the last few edits is that the majority of the document looks normal, and the gradual fading of the highlighting provides a good overview of what edits

were made and when they were made.

Edits by different users are highlighted with different colours, so it is easy to see who wrote what in the last one thousand changes in a group authored document.

SIDEWAYS LAYOUT

Recdit lays the document out sideways, splitting the document into pages. The pages are laid out sideways in order to make best use of current widescreens which are becoming more popular and to provide additional structure to the text. This is to provide more structure to the document, such that it is easier to find where a part or page of the document is.

The pages the text is spilt into are sized so that they are a good size for reading and editing. Page-breaks between sections means that modifying one section will not affect the page layout of another section.

The document can also be zoomed in and out smoothly using another slider. This allows Recdit to provide a overview of the whole document, which is useful to show where edits are being made located in the document as changes are replayed. This is shown in Figure ??.



Figure 2: An overview of this entire paper, as a user can zoom it.

LATEX

Recdit does not support rich text. All edits and the history of the document are recorded as plain text. Recdit instead provides inbuilt \LaTeX syntax colouring and built in PDF generation. This provides any rich text support. Using \LaTeX means that the architecture of Recdit is simpler, because it only needs to work with plain text. It also means that the typesetting and layout capabilities can be very powerful.

To provide network based editing support for images, without including the images in the plain text, Recdit provides a `geturl{http://...}` macro that retrieves a file from a URL and saves it as a file in the `/tmp` directory. This allows images to be handled by Recdit over a network without extending by handling and recording images. (Cached URLs are used to ensure that the same image is not downloaded more than once.)

GRAPHS

Recdit provides graphs of the document size and time of edits. The purpose of these graphs is to allow quick overviews of how the document has changed. For example when large scale edits, insertions or deletions are made the graph of the document size shows large jumps. The graphs can be highlighted to show over what time periods the document was edited and which user performed the edits. Edit wear and read wear [?] introduced graphs that answer different questions based on edit location instead of time. Combinations of these visualisations could be interesting.

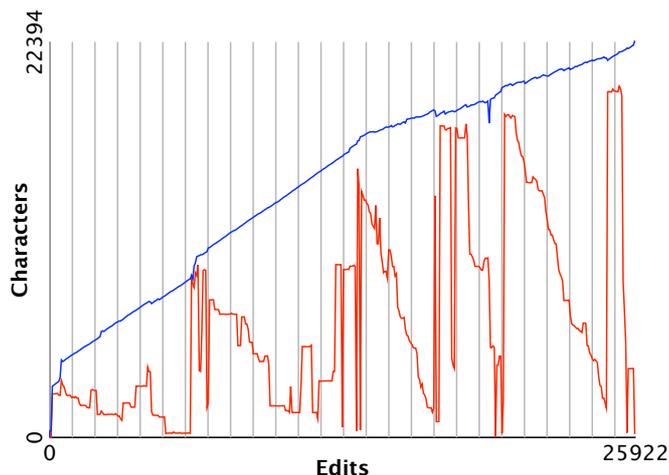


Figure 3: A graph of this paper's creation.

Figure ?? shows two graphs that are drawn of the editing history of this paper. The same graphs can be seen underneath the slider in Figure ??. The horizontal axis on this graph is not linear in time but linear in edits. The top line that travels from bottom left to top right is the size of the document in characters. A steady increase in this line represents typing, jumps in this line represent insertions or deletions, edits that create large changes in the character count. In Figure ?? this line tails off towards the end, as more of the edits made were correcting.

The lower line is the edit location in the document. The start of the document is represented by the top line, the end of the document is the x-axis. When this line is near the bottom of the graph the edits are being made at the end of the document. Flat areas in the edit line are periods of continuous typing, sharp spikes are often small corrections. A downward slope is often seen when a user is editing the document from start to end. Repeated edits like this create a sawtooth graph, like that in Figure ??.

ALTERNATIVES

There are tools that currently provide some capabilities for tracking changes in text documents. These tools provide simple functionality, they are not capable of Recdit's character history or able to "scrub" through history.

- Microsoft Word provides a feature called *track changes*. Additions are highlighted and deletions are scored out. *Track changes* can be tedious to use, it does not show the actual edits and it creates documents that when the changes are shown, are increasingly hard to read.
- Various web based editors, like *EditLive* and *Google Docs*, provide similar tracking changes functionality to Microsoft Word. These editors often provide good group authoring support.
- *diff* is a typical file comparison utility that is used to show the changes between two documents. *diff* is often used to provide change tracking when combined with version management systems, like *subversion*. The results of *diff* can be confusing when there are large edits or even simple

rearranging. Other similar tools are more capable [?].

- Wikis provide web based multi-user authored pages, some of these provide the ability to track the changes between versions. The history of these pages can contain thousands of edits and users and can provide interesting insights into the authoring of the pages [?].
- Ad hoc emails and conversation provide the majority of change tracking for most co-authored documents. The change information is usually passed between authors in an unstructured form, with multiple versions of the same file in different places.

ANECDOTAL EXPERIENCES

Single User

As a single-user tool I have been using Recdit for over a year to write my PhD thesis. Over this period of time over 30,000 words have been entered into many different documents (including this paper) and almost 200,000 individual edits have been recorded.

It has been my experience that I have not often used the ability to review the history of a document. On a few occasions it has been useful to rewind to locate and recover a paragraph from the past that I had previously deleted. The process is to rewind the history until the section that was deleted is located, select and copy that section, then fast forward to the current state of the document and paste the copied section. These paragraphs would have been lost completely if I was only using a simple undo system.

The biggest benefit of a recorded history I have found is the coloured trails that show the last one thousand edits that make it easy to review the last few edits. I have found this to be most useful when opening a document after having not looked at the document for even a day or so, because after even such a short time I find it takes some time to pick-up the flow of editing the document from where I left off. Having an overview of the last few edits means that, as a user, I can start back where I left off with an understanding of what I was just doing, regardless of how long ago I was last editing the document.

Multiple Users

I have also used Recdit to author several multi-user documents. These were mostly not concurrently edited but edited using ad-hoc versions emailed between authors, this has been partially necessary because of the need to edit documents off-line. In these cases I have found the ability to not only see other user's changes but to review the actual process of what they did, very useful. Firstly being able to see exactly what the other user has changed (and deleted) is useful, and secondly I have found that being able to see the process of editing has provided a good idea of what the other user was trying to achieve. These are both impossible by only showing the textual changes made to a document.

CONCLUSION

This paper has described a text editor that provides a novel history recording feature and user interface for interacting with this history. The history combined with trails and graphs that provide overviews of the document history. These allowed several possible benefits both for single and multi-user

editing and reviewing changes.

My experiences of using this editor have suggested that tracking edit changes in a text document is both valuable for a single user and also extremely valuable in a multi-user context.

I believe these experiences suggest that using Recdit to further study collaborative document editing, specifically looking at versioning, tracking changes and history for different domains, will be very valuable.

AVAILABILITY

The Recdit application, this paper and a movie of its interaction are available at <http://will.thimbleby.net/truertext/>

ACKNOWLEDGEMENTS

Will Thimbleby is supported by a Swansea University PhD scholarship and by Microsoft Research Cambridge

REFERENCES

1. Chengzheng Sun. Undo as concurrent inverse in group editors In *ACM Trans. Comput.-Hum. Interact.*, Vol 9, Number 4 ACM, NY, (2002), pp. 309–361.
2. Christine M. Neuwirth C.M., Chandhok R., Kaufer D.S., Erion P., Morris J. and Miller D. Flexible Diff-ing in a collaborative writing system In *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*, ACM, NY, (1992), pp. 147–154.
3. Hee-Cheol Kim Eklundh, K.S. Collaboration between writer and reviewer through change representation tools In *System Sciences, 2002. HICSS*, IEEE, (2002), pp. 531–540.
4. Hill W. C. and Hollan J. D. and Wroblewski D. and McCandless T. Edit wear and read wear In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, (1992), pp. 3–9.
5. Rekimoto J. Time-machine computing: a time-centric approach for the information environment In *UIST '99: Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, (1999), pp. 45–54.
6. Thimbleby W. A Novel Pen-based Calculator and Its Evaluation In *Proceedings 3rd ACM Nordic Conference on Human-Computer Interaction*, ACM, NY, (2004), pp. 445–448.
7. Thimbleby W., Thimbleby H. A novel gesture-based calculator and its design principles In *Proceedings 19th. BCS HCI Conference*, Vol 2, BCS, (2005), pp. 27–32.
8. Viégas, F. and Wattenberg, M. and Dave, K. Studying cooperation and conflict between authors with history flow visualizations CHI, Vol. 6, No. 1. (2004), pp. 575–582.
9. Vitter J.S. US&R: A new framework for redoing (Extended Abstract). In *SDE 1: Proceedings of the first ACM SIGSOFT/SIGPLAN software engineering symposium on Practical software development environments*, ACM, NY, (1984), pp. 168–176.