Computer Science and Artificial Intelligence Laboratory
Technical Report

MIT-CSAIL-TR-2009-031 June 24, 2009

Interactive Visual Histories for Vector Graphics

Sara L. Su, Sylvain Paris, Frederick Aliaga, Craig Scull, Steve Johnson, and Fredo Durand

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

MIT Computer Science and Artificial Intelligence Laboratory Technical Report

Interactive Visual Histories for Vector Graphics

Sara L. Su *Massachusetts Institute of Technology*
Sylvain Paris *Adobe Systems*
Frederick Aliaga *Adobe Systems*
Craig Scull *Adobe Systems*
Steve Johnson *Adobe Systems*
Frédo Durand *Massachusetts Institute of Technology*

June 2009

ABSTRACT

Presentation and graphics software enables users to experiment with variations of illustrations. They can revisit recent editing operations using the ubiquitous undo command, but they are limited to sequential exploration. We propose a new interaction metaphor and visualization for operation history. While editing, a user can access a history mode in which actions are denoted by graphical depictions appearing on top of the document. Our work is inspired by the visual language of film storyboards and assembly instructions. Our storyboard provides an *interactive visual history*, summarizing the editing of a document or a selected object. Each view is composed of *action depictions* representing the user's editing actions and enables the user to consider the operation history in context rather than in a disconnected list view. This metaphor provides instant access to any past action and we demonstrate that this is an intuitive interface to a selective undo mechanism.

1 INTRODUCTION

Digital tools have introduced great flexibility to the illustrator's workflow. In addition to providing a rich toolbox of graphical elements, programs such as Adobe Illustrator, Corel Draw, and Microsoft PowerPoint facilitate exploration, trial-and-error editing, and refinement of designs. We aim to add to the flexibility of these interactions by leveraging the editing history of an illustration.

All modern text and graphics editors support a notion of history. The standard undo mechanism not only makes it easy to discard recent mistakes, it allows the user to compare the design before and after a modification. In many programs, this has been extended to storing the full history of actions, and users can roll back to arbitrary points in time. However, users are limited to sequential and causal exploration of this history. We argue that the creative process, which is often inherently nonlinear, should be supported by tools that are nonlinear. We present a new visualization that shows the actions in context and enables nonlinear exploration of history.

We show the illustration's history as a storyboard annotated with *action depictions*, graphical metaphors of editing actions such as fill color changes or spatial transformations (Figure 1). Our *graphical history mode* can be activated at any time during the editing process, and once activated, action depictions appear on top of the document. Users can view the history of the entire document or restrict it to a particular object or region.

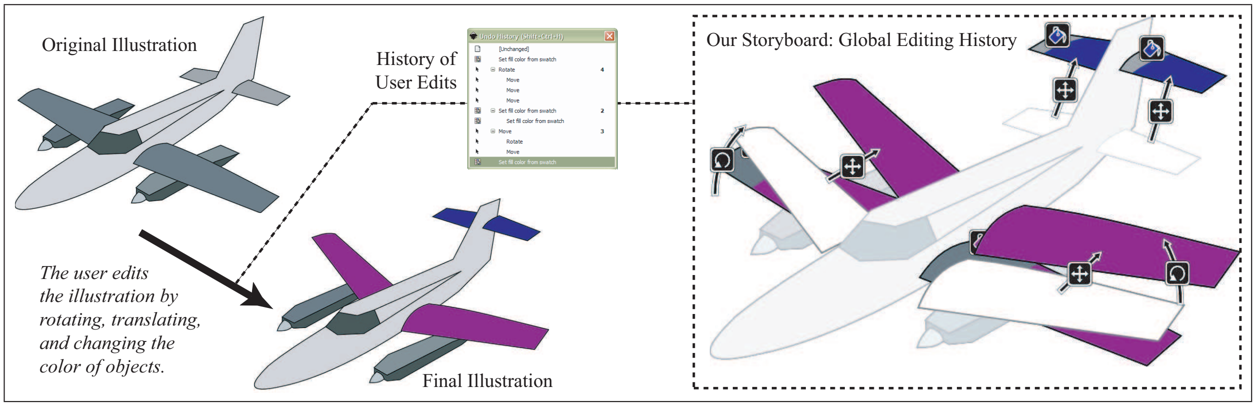


Figure 1: This automatically-generated visualization shows the editing history of an illustration. Arrows and icons depict spatial transforms and color change actions performed by the user. The user can click on any depiction to selectively undo the corresponding action.

1.1 Related Work

There has been much work on the history of web pages [19], graphic art [24], and datasets [14]. Our work follows most closely from the Chimera system [20, 21], which shows graphical history in a series of panels, each one containing the before and after of an action. Chimera depicts several "actions" in the sense that the object selection or caret placement is displayed at the same time as the illustration modification (translation, color change, etc.). The action is described by a text annotation. Our approach further develops the graphical aspect of the visualization, showing multiple actions at the same time and representing them by in-place graphical depictions rather than text annotations. We propose this fully graphical storyboard as an alternative, complementary visualization.

Recently, Nakamura and Igarashi [25] presented a system extending the visualization methods of Kurlander [21] and Su [27] to generic Java AWT/Swing applications. While they focus on visualizing GUI events, such as mouse movement, we focus on visualizing changes to the document and assisting the user's spatial memory of it.

We will discuss an application of the storyboard, using visual histories as an interface to non-sequential undo. The Photoshop history panel and SolidWorks object tree have shown the value of maintaining and editing visual histories, and there has been significant progress made in 3D modeling and CAD [12, 7, 8, 9, 26]. The object tree in CAD systems provides a convenient interface for selective undo and parametric changes but requires significant structuring of the document. We want to offer some of the flexibility of these systems at a lower overhead to the user. Unlike the typical CAD file, illustrations in presentation slides are often done by casual users who are unlikely to create any hierarchy.

1.2 Overview

Our storyboard is a visualization, as well as a new mode of interaction with a document's operation history. We describe how our storyboard interface enables non-sequential browsing and modification of a graphical content. We see the storyboard as complementary to existing interaction methods, and its visual histories are intuitive on the scale of the typical 2D vector illustration. In a usability study (Section 5), subjects easily understood how the storyboard relates to their actions and appreciated the new interaction modes. We target moderate complexity drawings on the scale of a typical clipart or PowerPoint illustration, which covers a large set of users. Comments from study participants also informed the design of extensions to the visualization (Section 6).

2 DESIGN GOALS

We have designed the storyboard annotations to be simple, with a common look and feel. The action depictions follow the visual language of assembly instructions [2], maps [3], comics, and film storyboards. Together, the depictions can be seen as the "assembly instructions" for a document. Principles from cognitive psychology [31, 29] inform our design strategy, summarized below.

Sequence of discrete steps. Humans naturally interpret and remember an event as a sequence of discrete steps [32], and it has been shown that diagrams reflecting this structure are easier to comprehend [2, 15]. Our approach naturally achieves such sequential structure since the history of a vector graphics documents is a list of discrete steps. We refer to these steps as *actions*.

In place, static visualization. Our interface allows the user to consider actions in spatial context. It has been shown that in-place visualization of user interface transitions [4] and variations [28] improve comprehension by exploiting spatial memory. Continuous changes over time, such as dragging an object to translate it, could be depicted with animated visualizations. However, because experiments on the effectiveness of animation in explanatory diagrams are inconclusive [30], we use static visualizations. Furthermore, one function of the storyboards is to enable users to select actions to undo, and a moving target could increase acquisition time.

Congruence, proximity, and comprehension. Most editing actions are visual in nature, and we rely on well-accepted schematic representations, such as an arrow for a translation [31]. We have designed these visual elements to have a consistent look, exploiting the drawing primitives of the editing software itself.

Pilot users commented that "detached" icons were difficult to understand. In our storyboard, an action depiction is directly in contact with the object upon which it acts, e.g. a change of color is represented by a paint bucket overlaid on the object, visually relating the action to the object. We ghost unmodified regions to draw focus to objects actively being edited.

Before and after. We enable the user to compare the before and after states of a drawing in order to facilitate the decision to undo it or not. For example, we show the locations of an object before and after a translation, with "before" states rendered semi-transparent and in-place to limit screen clutter.

Summarization. A single depiction limits the information provided to the user. Inspired by recent work on "reverse storyboarding" [10, 11], we summarize a segment of history with multiple depictions per view. Our interface shares some of the annotation elements, but while these storyboards visualize an existing video sequence as a static image, our goal is to visualize the steps of constructing an illustration. To alleviate the visual clutter that can result from storyboarding a complicated document history, we support a "magic lens" interface [6, 16] for viewing the history of a selected object. User evaluation confirms that restricting the scope of the visualization is the preferred mode of interaction.

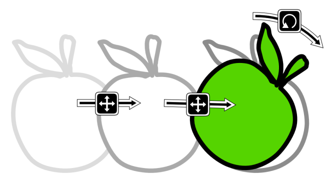
3 STORYBOARD VISUALIZATION

We provide a *graphical history mode* that can be activated at any time when editing an illustration document. Our algorithm takes the operation history and creates a storyboard by traversing it from present to past. For each type of action, we have designed an *action depiction*, a visual representation automatically generated based on the parameters of the action and the set of objects it acts on. For instance, a translation of a polygon is depicted with a straight arrow. The action depictions are overlaid on the illustration. Among the challenges we address are designing visual metaphors to facilitate user comprehension and laying them out to minimize clutter.

3.1 Action Depictions

Our software prototype supports common actions on shapes, paths, and text. The actions we depict fall into the following categories: creations and deletions, spatial transforms, change of scale, fill and stroke attribute changes, and control point edits. Every action is responsible for its visualization. We have designed the action depictions with a common look and feel and display them in a layer on top of the illustration to help them stand out.

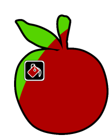
We depict **translation and rotation** using arrows that share a similar visual style. We use thin arrows to avoid hiding the main illustration. Each arrow carries an icon indicating the transformation type. This icon also provides a target easy to click. The arrows have a constant thickness to represent rigid transformations and a white outline to ensure visibility on any background.



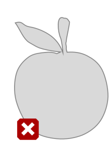
For translations, we use straight arrows between the object's old and new positions. The rotation arrow is similar; its length and orientation are the magnitude and direction of the transform. The arc shares the object's center of rotation and its radius is the larger dimension of the object. For very small changes, such as keyboard "nudges", we enforce a minimum arrow length.

The **resizing** depiction uses one to four axis-aligned arrows. To visually distinguish between scaling up and down and between scaling and other transforms, we use wide, tapered arrows: narrow-to-wide for up-scaling and wide-to-narrow for down-scaling. **Editing of control points**, often used for refinement involving a series of many, in-place adjustments, is simply shown with the before and after states.

We depict **changing fill and stroke style** by overlaying a partial representation of the previous state on top of the current one. In practice, to create this half-object, we first create a cutting path as the diagonal half of the "after" object's bounding box. We then take the Boolean difference of the object and the cutting triangle and overlay the result on the full "before" object. We complement the before/after overlay with icons, placed according to congruence rules, to provide an additional cue as to the type of change.



Unlike other actions, **creation and deletion** do not affect any transformation on the object's shape, position, or appearance style. Creation is easily identified due to the sequential appearance of the depictions, but for deletion we compensate by using an iconic cue. A gray, semi-transparent copy with an "X" icon represents a deleted object.



3.2 History in Context

Our storyboard provides an *interactive visual history* of a document, shown either as a summarizing *global storyboard* or a *local storyboard* for a selected object. From discussions with user experience designers, we understand that providing both modes of interaction provides the best support for common producer (editing) and consumer (viewing) tasks.

The storyboard can be activated at any time during the editing process. Action depictions appear on top of the document, and users can select them to undo the corresponding actions. This allows the user to consider the actions in spatial context rather than in a disconnected list view. This visualization provides a natural interface to a selective undo mechanism.

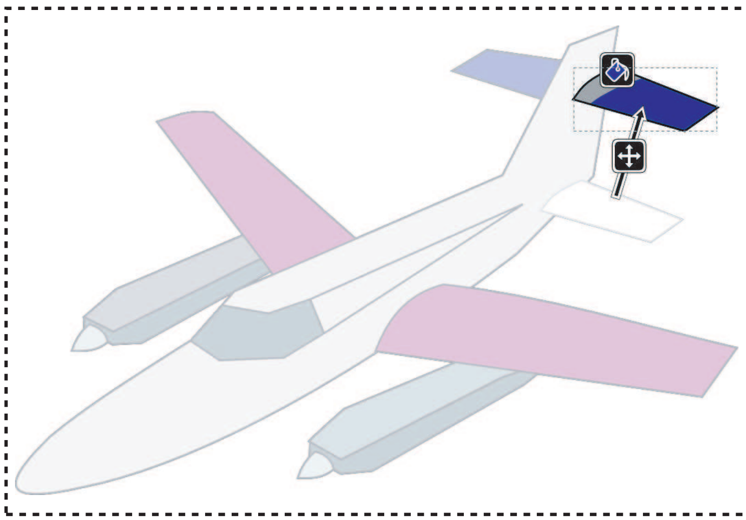


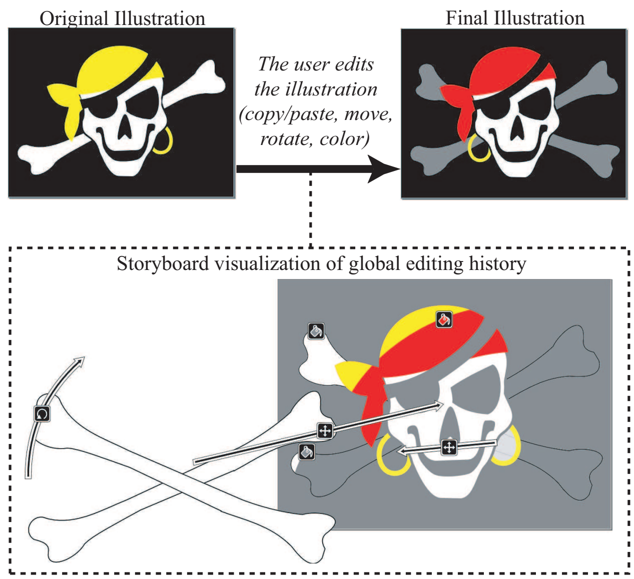
Figure 2: Per-object history. The user can restrict the history view (Figure 1) to a selected object.

3.3 Alternative Styles

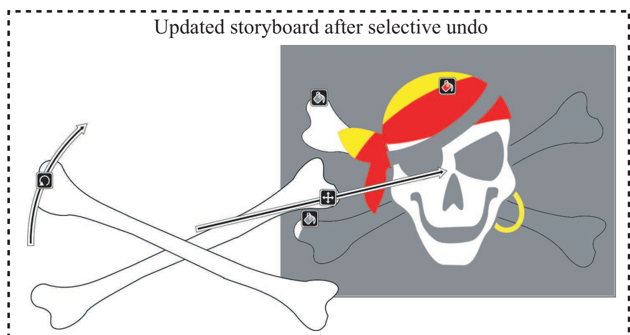
A possible extension is to offer several depiction sets so that users can select a style not conflicting with their own illustration. For instance, movie makers would appreciate thick 3D arrows typical from movie storyboards [13]. Figure 6 shows such a style, which we have considered in our preliminary studies. In this paper, we focus on the "thin arrow + icon" style since it minimizes clutter and fits well to most illustrations.

4 APPLICATION: NON-SEQUENTIAL UNDO

Sequential undo, found in all existing illustration programs, removes the latest action from the top of the undo stack. To return to a previous state, the user has no choice but to lose all actions between then and the current state. A number of techniques have been proposed for less destructive, *non-sequential undo* mechanisms [5, 24, 22, 18]. We show that a useful application of our storyboard is as a front-end to non-sequential undo. The user can click on the depiction for any previous action to index into the history. Any action, not only the most recent one, can be undone. We briefly describe our use of storyboards for non-sequential undo in vector graphics editing.



(a) The storyboard summarizes the user's edits between the two versions of the illustration, showing these edits in spatial context.



(b) This storyboard shows the state of the document after the user has selectively undone the translation of the earring.

Figure 3: The storyboard visualization provides a natural interface for non-sequential interaction, specifically selective undo. Note that the selective undo of the earring movement (an action occurring farther back in the history stack) does not affect any other object. In contrast, standard (sequential) undo would have required also undoing all subsequent actions.

4.1 Dependencies

We observe that many editing actions can be considered independently from each other. First, an action on one object can be safely undone without affecting actions on different objects. Second, we define the following *dependency classes*,

- Spatial transforms = {translation, rotation}
- Appearance changes = {fill style, stroke style}
- Shape modifications = {control point editing, resizing}

and argue that an action can be considered orthogonal to any action that does not belong to its class. We found that, in practice, these definitions are easy to understand. We keep as future work the study of alternative classes, e.g. setting fill and stroke independent.

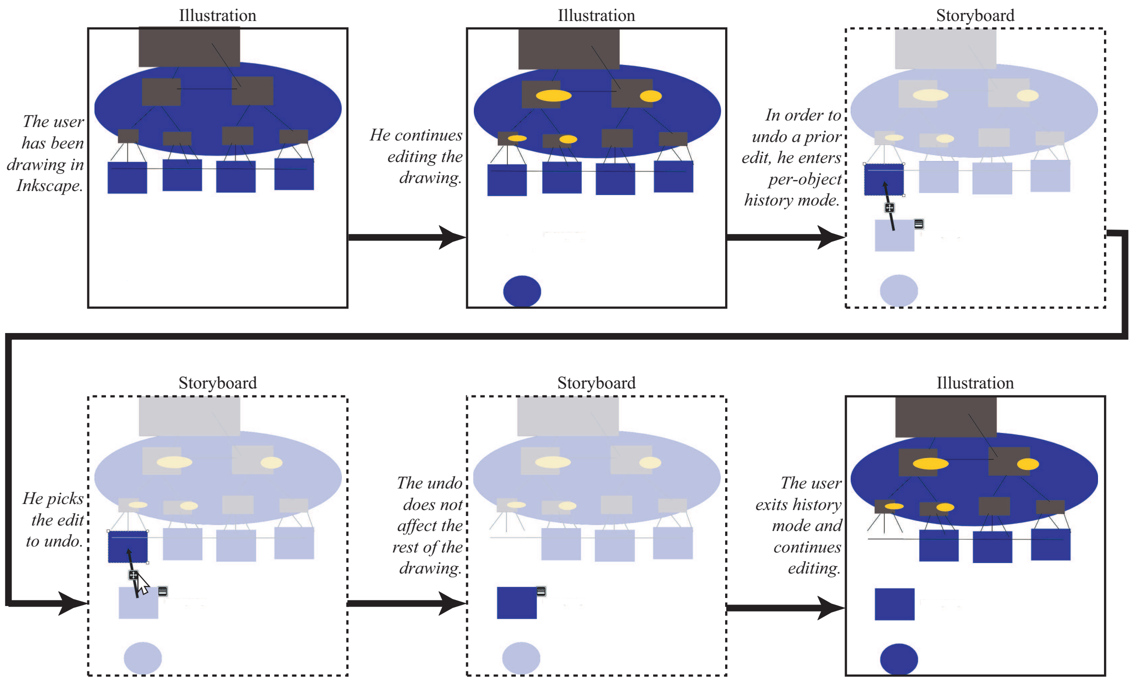


Figure 4: Participants in the user study were asked to recreate drawings from their personal experiences. Over the course of the editing sessions, they were asked to make modifications to their drawings in an effort to enourage the use of the visualization and undo.

4.2 Algorithm

Non-sequential undo is less destructive because, rather than canceling all actions after the selected one, it cancels only those acting on the same object that also belong to the same class. We execute a non-sequential undo command by issuing a number of queries to the undo stack. First, given the action depiction the user selected, we retrieve from the corresponding entry in the undo stack a list of objects affected by that action. Then, we query the stack for a list of actions subsequent to the undone actions that affected any of those objects. Among these actions, only those of the same class as the undone action are canceled. Figure 3 shows an example of the storyboard as an interface to selective undo.

5 USABILITY STUDY

We have implemented the storyboards and non-sequential undo as an extension to the vector graphics editor Inkscape [17]. Our implementation reuses Inkscape drawing primitives to display the storyboard and create action depictions. We conducted a concept evaluation by observing first-time users of our prototype.

User interaction with drawing tools is by nature complex. This suggests that using a standardized task for quantitative analysis would require a great simplification of this interaction. Rather than comparing measurements such as success rates or task completion times, we asked participants to recreate drawings from their typical use and performed a qualitative analysis by interviewing participants and extracting recurring themes.

5.1 Participants and Apparatus

Twelve participants (6 male, 6 female) aged 20-40 years were recruited for the study and received a gratuity for their time. Half were proficient computer users from a university and half were from the broader community. Some had familiarity with Illustrator; all were familiar with the drawing tools in PowerPoint. The university participants had prior experience creating figures for papers or presentations, and the participants from the community had a wide range of experience. None were graphic designers or creative professionals, and one was casually familiar with Inkscape.

Subjects used a computer running Windows XP, with a display at 1280×1024 pixel resolution. Our modified version of Inkscape, with history storyboard and non-sequential undo extensions, was used for all drawing tasks. We removed all non-essential widgets.

5.2 Design

First, participants were asked about their background: occupation, types of drawings normally made, and tools used. Second, they were given a short tutorial on Inkscape and the history features and then asked to create one of the drawings they had described earlier. During the drawing task, participants were encouraged to try out the new history functions. Finally, participants were interviewed about their experience to record subjective preferences.

5.3 Results

Participants were asked to recreate drawings typical for them, resulting in a diverse sample of illustrations. Some are shown in Figures 4 and 8. Despite the wide range of participant backgrounds and drawing experiences, themes emerged from the interviews.

Free experimentation (nonlinear working style). Participants said that, compared to the tools they normally use, the storyboard allowed them to more freely experiment and try out new ideas. Participants often made a series of precise alignments, pixel-level adjustments, and precise color changes which would have been tedious to recreate in a traditional history that undoes every change after the one selected.

Spatial memory cues. Some participants commented that visualization of the history helped them to recall previously made changes, which they found helpful to reconsider.

Persistent history. Some participants described past situations in which they would have wanted to revert to changes from prior working sessions. One participant said, “I have accidently made changes to a file before, saved it over top of the original and then had no way to retrace steps to the original version.”

Limitations. A common request was a shortcut or contextual menu for the undo functions. We had disabled these features for the study to avoid bias. Several participants reported being overloaded by the number of depictions in the global storyboard or felt that the history got in the way of the illustration; they would have liked a means to view their “pristine” drawing. We expected these comments as our approach deliberately adds visual depictions on top of the drawing. To address these reactions, we added a shortcut to swap between the history and “pristine” views, and developed the extension described in the following section.

6 EXTENSIONS

6.1 Multi-Frame Storyboards

A long illustration session can generate a dense storyboard when all actions are displayed at the same time. On the other hand, showing singular actions limits the information provided to the user. User feedback confirmed the need for balance in the visualization. We propose a hybrid storyboard that shows multiple actions per view or frame.

To create this multi-frame storyboard (Figure 6), we traverse the history stack from the most recent action. We use a greedy strategy to avoid visual clutter when generating the depictions. If adding the next depiction would fully obscure an existing one or overlap more than a set maximum number of existing ones, we create a new frame; otherwise we draw in the current one. In addition to the visibility rule, we set a maximum number of depictions per frame. The viewport and zoom level are set so that all depictions and the objects they affect are visible. In addition, we set minimum and maximum zooms as functions of the overall drawing size.

DRAWMULTIFRAMESTORYBOARD (HistoryStack H):

1. For each action a in H:
   (a) Create a new action depiction d_a.
   (b) i_a = number of intersections of d_a with depictions in the current frame.
   (c) If d_a fully obscures any existing depiction or if $i_a > maxIntersections$:
      i. Create a new frame.
      ii. Set the current frame to be the new frame.
   (d) Draw d_a in the current frame.
   (e) Adjust the zoom level.

Figure 5: Pseudo-code of the multi-frame storyboard creation process.

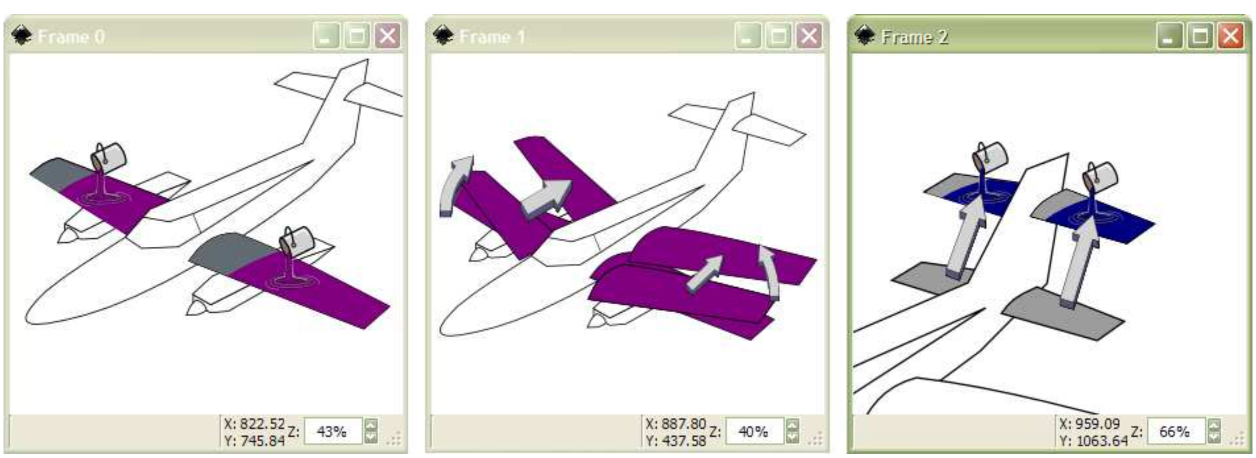


Figure 6: Multi-frame storyboard. This storyboard summarizes the editing history of a document, with time progressing from left to right. In addition, this storyboard shows an alternative depiction style we have considered.

6.2 Editing in Multiple-User Environments

The storyboard facilitates editing by multiple users in a manner similar to the “track changes” feature available in text editors such as Microsoft Word. Every editing action is tagged with the ID of the current user. In the storyboard, actions are color-coded by author, making visible the interactions between all the collaborators (Figure 7). This could, for example, assist multiple authors working on the same figure for a paper. Currently, we support asynchronous collaboration across several editing sessions. Synchronous editing raises new challenges for undo that are worth investigating, including user understanding of the undo mechanism and user intent [1].

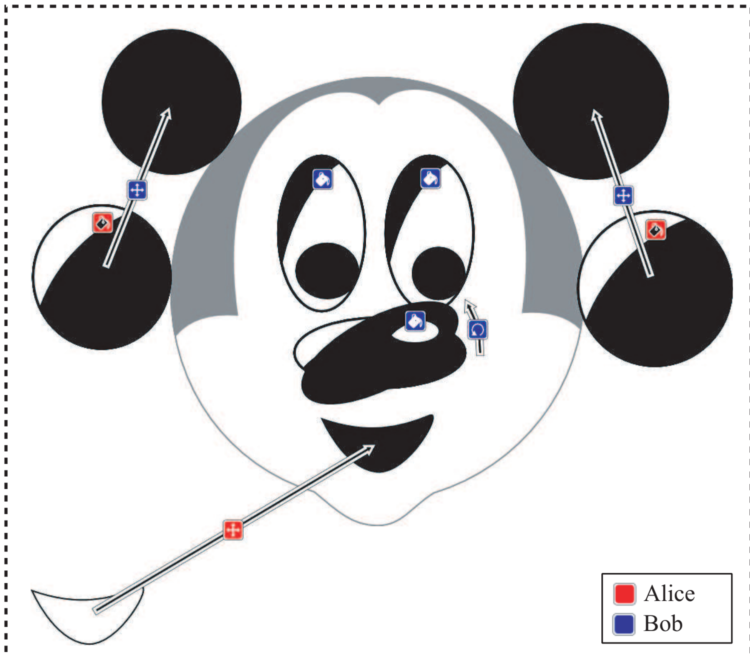


Figure 7: Collaborative editing scenario. Two users have edited this drawing, and the storyboard displays their actions color-coded in a “track changes” style.

7 DISCUSSION

Our storyboard visualization can help users take fuller advantage of design on the computer by making the process more flexible and by helping them explore alternative versions of their designs. In particular, our user study has shown that people appreciate the additional freedom afforded by this interface to selective undo and that our storyboard aids their comprehension of the creation process as a whole, beyond the currently displayed illustration.

Storyboards encourage exploratory design by enabling the user to easily survey previous actions in spatial context and selectively undo them. An interesting direction for future work is to improve storyboards with design galleries [23] or other systems for visualizing variations to further facilitate prototyping. Another promising application is instruction. An expert’s storyboards could act as tutorials by revealing the editing process. While we have shown applications in vector graphics editing, we believe that these visualization techniques could be extended to aid prototyping, collaboration, and instruction in other domains.

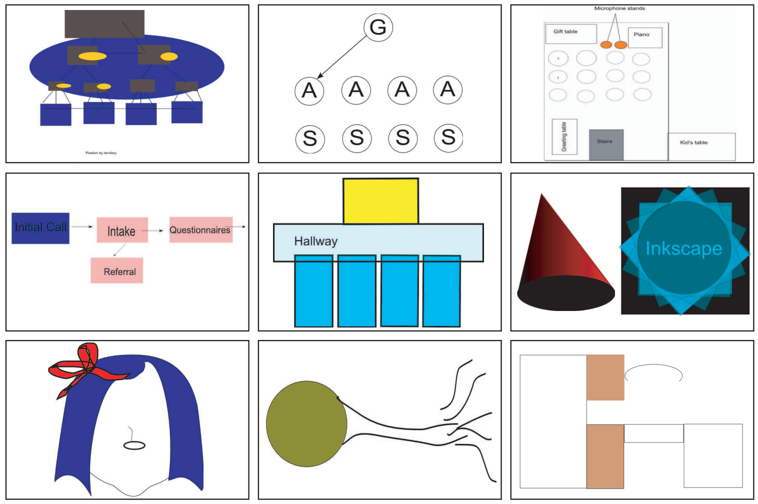


Figure 8: A sampling of vector drawings made by participants in our study. All were amateurs who had prior experience with Microsoft PowerPoint's drawing tools, and some were familiar with Adobe Illustrator. All were first-time users of our software, yet were able to create a wide range of drawings in a short period of time.

ACKNOWLEDGMENTS

The authors wish to thank the MIT Graphics Group, Steve Feiner, and Mira Dontcheva for discussion, the developers of Inkscape, and the maintainers of the Open Clip Art Library. Frédo Durand acknowledges a Microsoft Research New Faculty Fellowship, a Sloan Fellowship, and a generous gift from Adobe. This work was partially funded by the Singapore-MIT GAMBIT Game Lab.

REFERENCES

1. G. D. Abowd and A. J. Dix. Giving undo attention. *Interacting with Computers*, 4(3):317–342, 1992.
2. M. Agrawala, D. Phan, J. Heiser, J. Haymaker, J. Klingner, P. Hanrahan, and B. Tversky. Designing effective step-by-step assembly instructions. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 22(3):828–837, 2003.
3. M. Agrawala and C. Stolte. Rendering effective route maps: Improving usability through generalization. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 241–250, Aug. 2001.
4. P. Baudisch, D. Tan, M. Collomb, D. Robbins, K. Hinckley, M. Agrawala, S. Zhao, and G. Ramos. Phosphor: explaining transitions in the user interface using afterglow effects. In *UIST '06: Proceedings of the 19th ACM Symposium on User Interface Software and Technology*, pages 169–178, 2006.
5. T. Berlage. A selective undo mechanism for graphical user interfaces based on command objects. *ACM Transactions on Computer-Human Interaction*, 1(3):269–294, 1994.
6. E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: The see-through interface. *Computer Graphics (Proceedings of SIGGRAPH 93)*, 27:73–80, 1993.
7. M. Bussan and R. Hall. Abstraction, context and constraint. In *State of the Art in Computer Graphics*. Springer-Verlag, 1993.
8. V. A. Cicirello and W. C. Regli. Resolving non-uniqueness in design feature histories. In *SMA '99: Proceedings of the fifth ACM symposium on Solid modeling and applications*, pages 76–84, 1999.
9. V. A. Cicirello and W. C. Regli. Managing digital libraries for computer-aided design. *Computer Aided Design*, 32(2):119–132, 2000.
10. R. D. Dony, J. W. Mateer, J. A. Robinson, and M. G. Day. Iconic versus naturalistic motion cues in automated reverse storyboarding. In *Proceedings of 2nd IEEE European Conference on Visual Media Production*, pages 17–25, 2005.
11. D. B. Goldman, B. Curless, D. Salesin, and S. M. Seitz. Schematic storyboarding for video visualization and

editing. *ACM Transactions on Graphics*, 25(3):862–871, 2006.

12. R. Hall, M. Bussan, P. Georgiades, and D. P. Greenberg. A testbed for architectural modeling. In *Proceedings of Eurographics '91*, pages 47–58, 1991.
13. J. Hart. *The Art of the Storyboard: Storyboarding for Film, TV, and Animation*. Focal Press, 1999.
14. J. Heer, J. Mackinlay, C. Stolte, and M. Agrawala. Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1189–1196, Nov.-Dec. 2008.
15. J. Heiser, D. Phan, M. Agrawala, B. Tversky, and P. Hanrahan. Identification and validation of cognitive design principles for automated generation of assembly instructions. In *AVI '04: Proceedings of the Workshop on Advanced Visual Interfaces*, pages 311–319, 2004.
16. S. E. Hudson, R. Rodenstein, and I. Smith. Debugging lenses: a new class of transparent tools for user interface debugging. In *UIST '97: Proceedings of the 10th ACM Symposium on User Interface Software and Technology*, pages 179–187, 1997.
17. Inkscape. `www.inkscape.org`, 2008.
18. Y. Kawasaki and T. Igarashi. Regional undo for spreadsheets. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, 2004.
19. S. R. Klemmer, M. Thomsen, E. Phelps-Goodman, R. Lee, and J. A. Landay. Where do web sites come from? capturing and interacting with design history. In *Proceedings of CHI 2002: ACM Conference on Human Factors in Computing Systems*.
20. D. Kurlander and S. Feiner. A history-based macro by example system. In *Proceedings of the 5th Annual ACM Symposium on User Interface Software and Technology*, pages 99–106, 1992.
21. D. J. Kurlander. *Graphical Editing by Example*. PhD thesis, Columbia University, 1993.
22. R. Li and D. Li. A regional undo mechanism for text editing. In *Proceedings of the Workshop on Collaborative Editing Systems*, 2003.
23. J. Marks, B. Andalman, P. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: a general approach to setting parameters for computer graphics and animation. In *Proceedings of SIGGRAPH'97*, pages 389–400, 1997.
24. C. Meng, M. Yasue, A. Imamiya, and X. Mao. Visualizing histories for selective undo and redo. In *Proceedings Third Asian Pacific Conference on Computer and Human Interaction*, pages 459–464, 1998.
25. T. Nakamura and T. Igarashi. An application-independent system for visualizing user operation history. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 23–32, 2008.
26. R. Schmidt, T. Isenberg, P. Jepp, K. Singh, and B. Wyvill. Sketching, scaffolding, and inking: a visual history for interactive 3D modeling. *Proceedings of the 5th International Symposium on Non-Photorealistic Animation and Rendering*, pages 23–32, 2007.
27. S. L. Su. Visualizing, editing, and inferring structure in 2D graphics. In *Adjunct Proceedings of UIST 2007 (Doctoral Symposium)*, pages 29–32, 2007.
28. M. Terry, E. D. Mynatt, K. Nakakoji, and Y. Yamamoto. Variation in element and action: supporting simultaneous development of alternative solutions. In *CHI '04: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 711–718, 2004.
29. B. Tversky, M. Agrawala, J. Heiser, P. Lee, P. Hanrahan, D. Phan, C. Stolte, and M.-P. Daniel. *Applied Spatial Cognition From Research to Cognitive Technology*, chapter Cognitive Design Principles for Automated Generation of Visualizations. Lawrence Erlbaum, 2006.
30. B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: can it facilitate? *Int. J. Hum.-Comput. Stud.*, 57(4):247–262, 2002.
31. B. Tversky, J. Zacks, P. Lee, and J. Heiser. Lines, blobs, crosses, and arrows: Diagrammatic communication with schematic figures. In M. Anderson, P. Cheng, and V. Haarslev, editors, *Theory and Application of Diagrams*, pages 221–230. Springer, 2000.
32. J. Zacks, B. Tversky, and G. Iyer. Perceiving, remembering and communicating structure in events. *Journal of Experimental Psychology: General*, 136:29–58, 2001.