Poster: Rapid Pen-Centric Authoring of Improvisational Visualizations with NapkinVis

William O. Chao *
University of British Columbia

Tamara Munzner *
University of British Columbia

Michiel van de Panne *
University of British Columbia

ABSTRACT

We design and implement a web-based, pen-centric front end for the Protovis toolkit, allowing users to quickly create visualizations for improvisational purposes. The design of this system is constrained to the scope of visualizations that you would be able to sketch on a paper napkin. Even within this limited canvas size, we show that by creating visual interactions for authoring visualizations as a combination of separable marks, one can produce a wide variety of visualizations in a under a minute without needing to write a single line of code.

1 INTRODUCTION

This work is aimed at exploring the fast and code-less creation of visualizations for improvisational purposes, whether for quickly demoing data live in a presentation, visualizing as part of a collaborative conversation, informally toying around with visualization ideas to see how they would look, or even teaching information visualization principles in a classroom setting.

Anecdotally, many exceptional ideas have begun as simple sketches and doodles on paper napkins. Urban myth would say that this medium seems to help facilitate the creative process. Its free-form nature lets the user put down ideas in an unconstrained way, the size limitation keeps the amount of flowing thoughts in check, and in general the interface of paper napkins itself is very fast to use. With this in mind, many of our creative constraints came from asking the following question: "if you could somehow visualize something on a paper napkin sketch, how would you do so, and what could you visualize?"

2 DESIGN

NapkinVis is a sketch and gesture based program for quickly constructing visualizations, as shown in Figure 1. It incorporates ideas of sketching interactive user interfaces [3] and applies them to creating visualizations. The canvas size is similar to the size of a paper napkin, and the pen interaction is a simple on-(drag)-off, similar to many touch screens and ballpoint pens. Because this choice limits screen space and excludes distinguishing which mouse button is being held down, many WIMP-based GUI interactions become impractical. Interaction speed is important to keep a free-flowing feel, so another goal is that one should be able to construct visualizations in under a minute using this system.

NapkinVis is implemented as a web application and acts as a front end to the Protovis toolkit [1]. The user is provided with a sketch-based means of encoding data to single visualization marks such as wedges or bars, and by providing the user with a means to combine the marks to form more complex visualizations. This approach allows the user to access some of the flexibility of the toolkit without needing to be able to write code. It also allows the software to be used on any device that can allow Javascript-enhanced web pages.

*e-mail: {wochao,tmm,van}@cs.ubc.ca

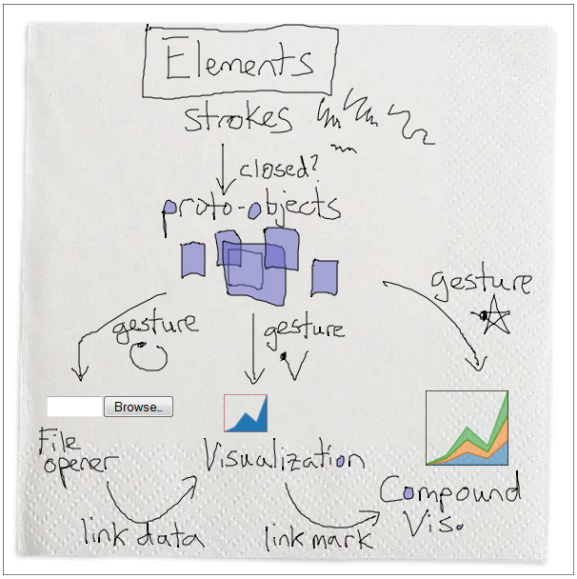

Figure 1: Elements of NapkinVis, demonstrated in the NapkinVis interface.

3 CREATING VISUALIZATIONS WITHOUT CODE

Figure 2 illustrates the key steps of creating visualizations with NapkinVis. We will use this as a starting point for our description and expand into more details in the following subsections.

3.1 Visualization as marks

The process of creating visualizations in NapkinVis builds on the Protovis toolkit from Bostock and Heer [1], where visualizations are specified in a bottom-up declarative language for composing marks with visual attributes. Although the idea of visualization specification through graphical marks has long been influential [4], previous toolkits [2] required significant programming effort to create even the simplest of visualizations.

3.2 Creating canvas actors

The basic workflow of NapkinVis begins with the user drawing ink strokes on a virtual canvas as they would when doodling or making quick sketches on a paper napkin. Further interactions can create actors, objects that can perform tasks, act as widgets, display things, and so on. The currently implemented set of actors allows the user to open files containing data to visualize, create single-mark visualizations, or create compound visualizations which allows marks to be combined into new visualizations.

Many of the interactions of NapkinVis are performed with actors. To create an actor, the user draws a closed box which transforms into a proto-object that can be further transformed into a variety of other actors via gestures. In addition to instantiation, this sketch-based interaction has the added effect of quickly encoding the position and size of visualizations.

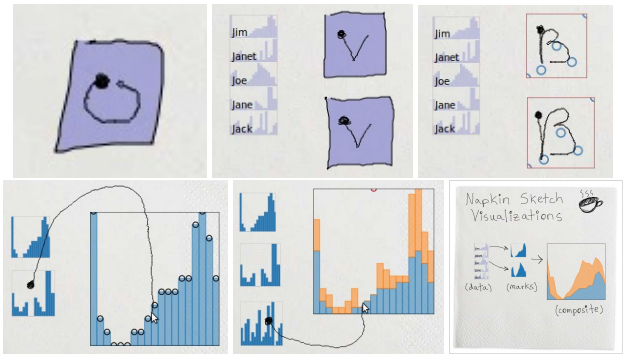

Figure 2: Key steps of creating visualizations with NapkinVis. **Top Left:** An 'o' gesture in the proto-object transforms it into a data actor to load data with. **Top Middle:** A 'v' gesture transforms a proto-object to a single mark visualization. **Top Right:** A 'b' gesture selects bars as the mark to use. **Bottom Left:** A single mark visualization is linked with the compound visualizations. Potential mark placement is indicated by circles. **Bottom Middle:** When linking from the bottom, marks are placed under already existing anchors. When approaching from the top, marks are stacked on top of other marks. Any existing mark can be selected for placement. **Bottom Right:** A sample napkin visualization.

3.3 Linking data to single marks

The first step in making a visualization after transforming a proto-object into a data actor is opening the file using the data actor. The user can choose a data file to import via a chooser, or type the file name into the actor if a keyboard is available. When imported, the data actor displays a small preview of the contained data that can be interacted with, acting like a scented widget [5]. These previews are used to link data to other actors.

Before a full visualization is made, the user must pick the marks to be used for encoding the imported data. In order to perform this encoding of data in NapkinVis, the user would create a single-mark visualization actor, and then use a gesture to specify the use of one of several marks: dots, wedges, bars, areas, or lines. After choosing a mark, the user would then link the data to the mark by drawing a link between the data and the mark.

For example, if the user wanted to visually encode some time-series data using an area mark, they would create a single mark visualization, draw the gesture 'a' in it to select the area mark, and then draw a link between the data and the visualization. After encoding the data to single marks, the user is then free to combine these marks into a compound visualization.

3.4 Combining marks to form a visualization

The final visualization is created using a compound visualization actor. After making several marks and linking data to those marks, the user can then add these marks to the compound visualization by drawing a link between the mark and the compound visualization. The direction of entry into the actor determines which Protovis anchor is chosen (top, bottom, left, or right), and the end point of the stroke within the compound visualization determines which group of marks the new marks will be positioned by.

For instance, to create a stacked bar graph, one would link several single-mark bar visualizations by drawing the links into the top of the compound visualization, and ending the drawn link when the appropriate anchor is highlighted, specifically the top of the previous group of marks.

Figure 3 shows several examples of compound visualizations made using NapkinVis.

3.5 Re-using visualizations

Once the visualization is constructed, new data can be linked into the existing visualization. This approach allows the user to re-use a novel type of constructed visualization. For instance, one could create a kind of stock market visualization that combines line marks and bar marks to view the daily average and daily fluctuations, then use this graph to visualize another day's data.

4 Discussion

Although limiting ourselves with design constraints similar to a paper napkin might at first seem impractical, fortunately lessons learned from these constraints can be applied to existing devices such as tablets, smart phones, interactive tables, and other emerging technologies. As a result we feel that these creative constraints served as good guidelines for interactions that can already prove to be useful today.

NapkinVis currently only supports tabular data stored in CSV format, and it is one of our goals to open up a wide variety of formats to the user in further versions of this work. In addition, because NapkinVis is intended as a tool to quickly visualize data on the fly, the created visualizations may not be as decorated or as polished as visualizations coded by hand. Future versions could incorporate standard visual features such as tick marks and labels, and provide more precise control of formatting such as scaling.

5 Conclusion

In this work we demonstrate the creation of a system to quickly author visualizations using only pen-based interactions. This system, NapkinVis, uses a workflow that requires the user to first author individual marks, link data to these marks, and then combine them in a compound visualization by using anchors from Protovis to help guide the placement of marks. As a result, users can quickly compose a wide variety of visualizations using this system, each in under one minute, and all without writing a single line of code.

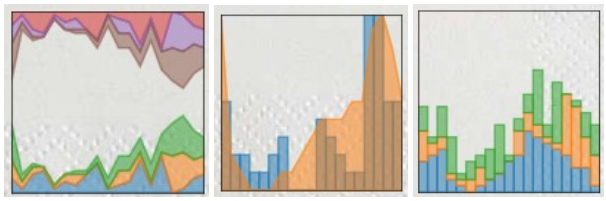

Figure 3: Examples of visualizations created using NapkinVis.

Acknowledgements

This work was supported in part by a grant from NSERC.

References

[1] M. Bostock and J. Heer. Protovis: A graphical toolkit for visualization. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2009)*, 15(6):1121–1128, Nov 2009.

[2] J. Heer, S. K. Card, and J. A. Landay. Prefuse: a toolkit for interactive information visualization. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 421–430, 2005.

[3] J. A. Landay and B. A. Myers. Interactive sketching for the early stages of user interface design. In *Proc. ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pages 43–50, 1995.

[4] L. Wilkinson. *The Grammar of Graphics (Statistics and Computing)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[5] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Trans. Visualization and Computer Graphics (Proc. InfoVis 2007)*, 13(6):1129–1136, Nov 2007.