napkin sketch visualizations

pen-centric improvisational visualizations

William O. Chao

December 2009

CPSC 533C final presentation

summary of this tool

- design paradigm (to find limitations/constraints)
  - “if a napkin could let me quickly visualize then what could I do on it, and how would I do it?”
- pen-centric (or touch-centric) front-end for protovis
- intended for light-weight visualization authoring
- intended for creating quick visualizations “on-the-fly”

why visualize on a napkin?

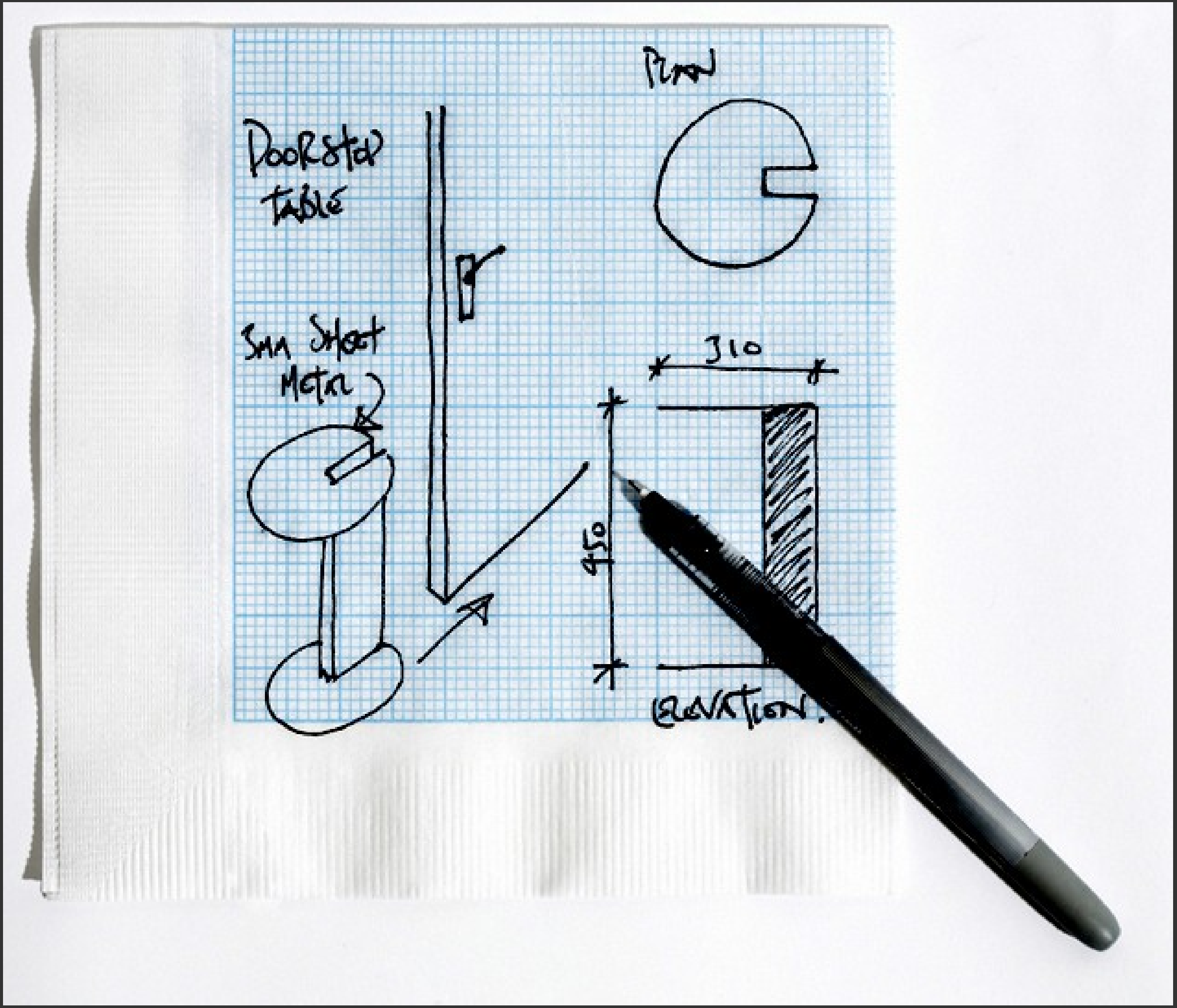


image source: http://designmuseumshop.com/media/item/14141/472/graphkin-doodle-634.jpg

motivation (covered previously)

- because doodling gets you thinking in a certain way
  - interaction feels fluid and free-form
  - it's quick, sloppy, and fun!
- to build on previous work supporting rapidly emerging technologies
  - e-paper napkins?
  - smart-boards, tables, phones?
- to see if it can even be done

for today's show...

- a quick demo (to introduce you to the system)
- some background information
- an in depth demonstration
- what I learned
- my wish-list
- concluding remarks

a very quick intro demo...

protovis in a single slide

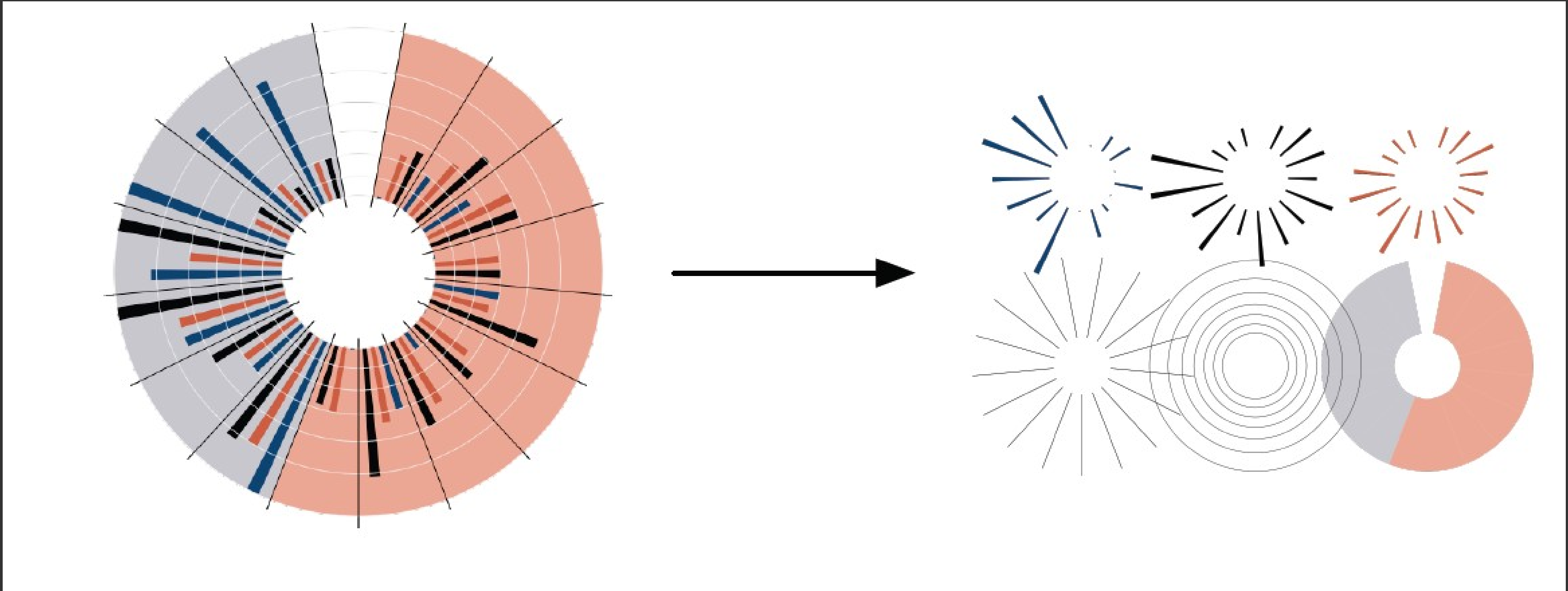
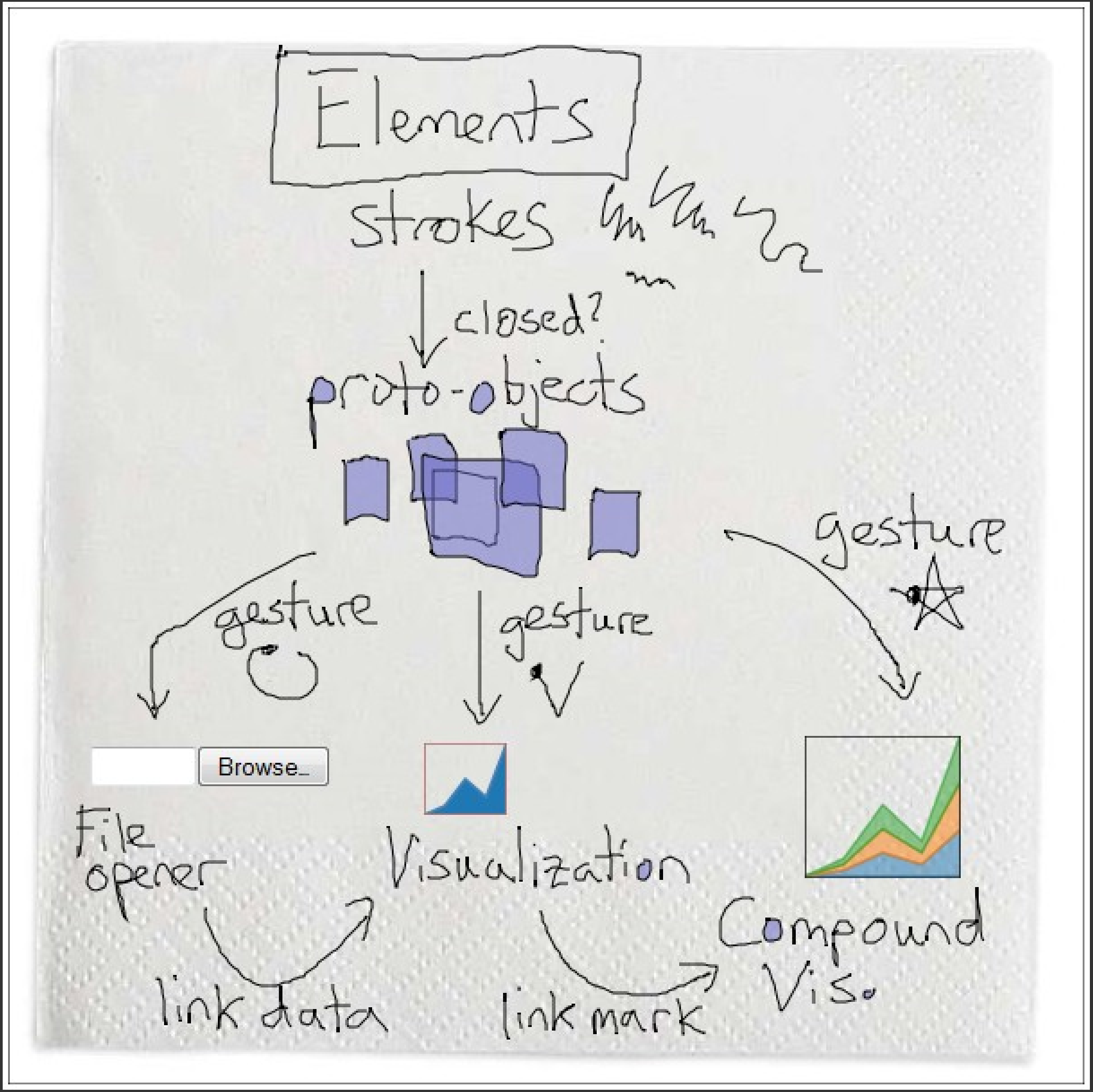


Fig. 1. Decomposing a visualization into marks.

```
new pv.Panel().anchor("center").add(pv.Wedge)
        .data(myData).innerRadius(10).outerRadius(50).angle(function()
        ..... [lots of code] ....
        .render();
```

image source: Bostock and Heer (2009)

elements of this project



(screenshot)

elements of this project - summary

- strokes
- proto-objects
- files / external data
- single-mark visualizations
- compound visualizations

detailed demonstration...

strokes

- free-form inking
- rendered in several layers
  - speed considerations when refreshing
- in certain conditions, interpreted as special things such as
  - links
  - gestures

proto-objects*

- closed strokes
- can be transformed to other objects
  - by gestures

* *a tribute to Ron Rensink's "proto-object flux", a reference to "proto" in protovis, as well as labelling those unformed objects as prototypes*

file openers

- opens files
  - a work-around to accommodate some of the limitations I ran into in Javascript
- use to select data files
- selected files are loaded and automatically parsed
- data is then represented visually

data objects

- visually represents the different columns in the parsed data
- can see an overview of the data
- can link data dimensions to visualizations

single-mark visualizations

- use to select mark (bar, dot, area, wedge)
- can quickly see 'shape' of data
- will later be used to fine-tune more specific attributes

compound visualizations

- used to combine marks
- intended as a place to compare different dimensions of data
- partitioned for easy selection of mark anchors in scene graph
- direction of entry determines orientation of mark layout

what I learned

- learned Javascript, CSS, <canvas>, SVG, protovis, <div>, and so much more!
- discovered a feasible work-flow for quick pen-centric (and touch-centric) visualization authoring!
  - plan on continuing to see where the 'walls' are
- discovered that web programming should not be first choice for proof-of-concept programs
  - too many limitations and permissions issues

what I learned – an example

one reason for web programming limitations, a sample from an internet forum...

“I have a bunch of banners on my website, how do I make it seem like the user has clicked them all?”

-Anonymous web developer

known limitations (few of many!)

- visualizations have value-scaling issues
- lack of labelling
- methodology might not be scalable to more than dozens of marks
- many anchors and attributes not considered
- very basic functionality such as “undo” left unimplemented
- filter and zooming non-existent in interface (and limited in libraries)

reason for intentional limitations

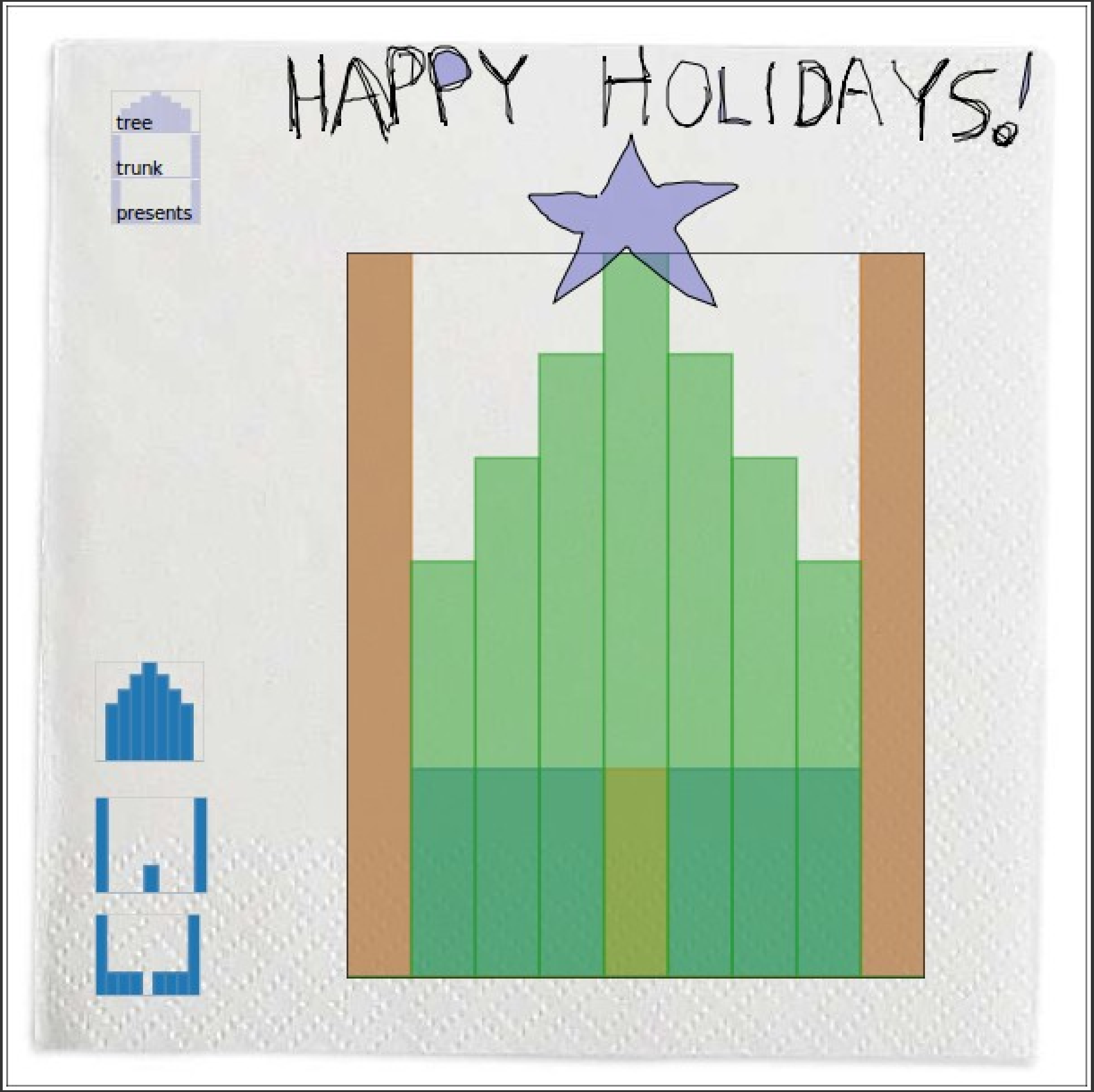
- time constraints
- triaging
    - find best "bang-for-the-buck": anchors
- scope of project
- *(not mouthwash)*
    - *(well, maybe a bit)*
        - *(actually, no.)*

my wish-list (given more time)

- explore other mark attributes
- explore more marks
- control data value scaling better
- utilize standard layouts
- use sloppy equations
- add data filtering
- write a paper worthy of Vis! (hopefully)

re-cap

- implemented a proof-of-concept, web-based, pen-centric visualization application
  - front-end for some of protovis's capabilities
- worked-out a work-flow to create a variety of visualizations in the process
  - fast enough for thinking on-the-fly (like while drawing on paper napkins)
- outlined possible future directions



thank you!

questions?