Spatial Layout

*Maneesh Agrawala*

**CS 294-10: Visualization**  
Fall 2008

---

**Assignment 3: Visualization Software**

Create an interactive visualization application – you choose data domain and visualization technique.

1. **Describe data and storyboard interface**  
due Oct 1 (before class)

2. **Implement interface and produce final writeup**  
due Oct 13 (before class)

3. **Submit the application and a final writeup on the wiki**

Can work alone or in pairs  
**Final write up due before class on Oct 13, 2008**
Cartographic Distortions

Cartograms: Distort areas

Scale area by data

[From Cartography, Dent]
Dorling cartogram

[Image: Dorling and Dorling-like Cartograms]

http://www.ncgia.ucsb.edu/projects/Cartogram_Central/types.html

Election 2004 map

[Image: Election 2004 map]

http://www-personal.umich.edu/~mejn/election/
Rectangular cartogram

American population [van Kreveld and Speckmann 04]

States as nodes in a graph

Graphical fisheye views of graphs [Sarkar & Brown 92]
Distorting distances

Scale distance by data (ticket price)

[From Cartography, Dent]

London underground

http://www.thetube.com/content/history/map.asp
Comparison to geographic map

Distorted

Undistorted

Summary

- Space is the most important visual variable
- Geometric properties of spatial transforms support geometric reasoning
- Show data with as much resolution as possible
- Use distortions to emphasize important information
Spatial Layout

Example: Timeline label layout
Problem

Input: Set of graphic elements (scene description)
Goal: Select visual attributes for elements
- Position
- Orientation
- Size
- Color
- ...

Topics

Direct rule-based methods
Constraint satisfaction
Optimization
Example-based methods
Direct Rule-Based Methods

Rule-based timeline labeling

- Alternate above/below line
- Center labels with respect to point on line
Rule-based timeline labeling

- Alternate above/below line
- Center labels with respect to point on line

Excentric labeling [Fekete & Plaisant 99]

http://www.cs.umd.edu/hcil/excentric/
Dynamic space management [Bell 00]

Manage *free space* on desktop to prevent window overlap

Video (0:46s)

---

Dynamic space management [Bell 00]

Goal: Place new elements to avoid overlap

- Elements are axis-aligned rectangles
- Keep track of largest empty space rectangles
Dynamic space management [Bell 00]

Goal: Place new elements to avoid overlap
- Elements are axis-aligned rectangles
- Keep track of largest empty space rectangles

Pros and cons

Pros
- Designed to run extremely quickly
- Simple layout algorithms are easy to code

Cons
- Complex layouts require large rule bases with lots of special cases
Linear Constraint Satisfaction

Network of layout constraints

TITLE ABOVE TEXT1
TITLE FULL PAGE WIDTH
TEXT1 LEFT OF PIC1
CAPTION1 BELOW PIC1
TEXT2 BELOW TEXT1

Constraints

NETWORK

Two possible layouts

[from Lok and Feiner 01]
Constraints as linear equations

Local propagation
- Set any variable
- Update other variables to maintain constraints

One-way
- Each constraint has 1 output variable
- Update output when any input changes

Multi-way
- Each constraint can be written so that any variable is output
- More complicated to maintain

One-way constraints

One-way constraints form a directed acyclic graph (DAG). Given the value for any variable we propagate it’s value locally through the graph updating the other variable.
Page layout example [Weitzman and Wittenburg 94]

Adaptive document layout [Jacobs 03]

Users authors templates which use one-way constraints to adapt to changes in page size
ADL template authoring [Jacobs 03]

Video

Pros and cons

Pros
- Often run fast (at least one-way constraints)
- Constraint solving systems are available online
- Can be easier to specify relative layout constraints than to code direct layout algorithm

Cons
- Easy to over-constrain the problem
- Constraint solving systems can only solve some types of layout problems
- Difficult to encode desired layout in terms of mathematical constraints
Optimization

Demo
Layout as optimization

Scene description
- **Geometry**: polygons, bounding boxes, lines, points, etc.
- **Layout parameters**: position, orientation, scale, color, etc.

Large design space of possible layouts

To use optimization we will specify …
- **Initialize/Perturb functions**: Form a layout
- **Penalty function**: Evaluate quality of layout
  - .. and find layout that minimizes penalty

Optimization algorithms

There are lots of them:
  - line search, Newton’s method, A*, tabu, gradient descent, conjugate gradient, linear programming, quadratic programming, simulated annealing, …

Differences
- Speed
- Memory
- Properties of the solution
- Requirements
## Simulated annealing

\[
currL \leftarrow \text{Initialize}() \quad \text{Form initial layout}
\]

\[
\text{while}(\neg \text{termination condition}) \quad \text{Perturb to form new layout}
\]

\[
newL \leftarrow \text{Perturb}(currL) \quad \text{Evaluate quality of layouts}
\]

\[
currE \leftarrow \text{Penalty}(currL) \quad \text{Always accept lower penalty}
\]

\[
newE \leftarrow \text{Penalty}(newL) \quad \text{Small probability of accepting higher penalty}
\]

\[
\text{if}((newE < currE) \text{ or } (\text{rand}[0,1] < e^{-\Delta E/T})) \quad \text{Small probability of accepting higher penalty}
\]

\[
\text{then } currL \leftarrow newL
\]

\[
\text{Decrease}(T)
\]

**Perturb:** Efficiently cover layout design space  
**Penalty:** Describes desirable/undesirable layout features

---

## Scene description

**Geometry**

- **Pie slices**
  - anchors for labels
- **Labels**
  - bounding boxes
Layout parameters

- Position \((x, y)\)
- Leader line
- Word wrap
- Color
- Alignment
- Orientation
- Scale
Many dimensions $\rightarrow$ large space

- Position \((x, y)\)
- Leader line
- Word wrap
- Color
- Alignment
- Orientation
- Scale

2D x 50 labels $\rightarrow$ 100D space
Penalties

Overlap & Distance
- Label – anchor slice
- Label – other slices
- Label – label

Leader lines
- Length
- Intersections

Word Wrap
Annealing minimizes sum of all penalties

Overlap: Label – Anchor Slice
Avoid partial overlap: No penalty if fully inside/outside
Overlap: Label – Anchor Slice

Penalize partial overlap by overlap amount

Distance: Label – Anchor Slice

Ensure label near center of edge of anchor slice
Distance: Label – Anchor Slice

Minimize distance \( d \)

Penalties

- Overlap & Distance
  - Label – anchor slice
  - Label – other slices
  - Label – label

- Leader lines
  - Length
  - Intersections

- Word Wrap

Annealing minimizes sum of all penalties
Demo

Pros and cons

Pros
- Much more flexible than linear constraint solving systems

Cons
- Can be relatively slow to converge
- Need to set penalty function parameters (weights)
- Difficult to encode desired layout in terms of mathematical penalty functions