11. Presentation Approaches II
Dealing with the presentation problem

Lecture „Informationsvisualisierung”
Prof. Dr. Andreas Butz, WS 2012/13
Concept and slides: Thorsten Büring,
3rd, revised edition
Outline

• Introduction focus&context
• Generalized fisheye view
• Graphical fisheye
  – Early examples
  – Graph fisheye
  – Multiple foci
  – Speed-Coupled Flattening
  – Symbolic Representation of Context
• Use-case: mobile devices
  – Visualizing out-of screen context
  – Designing mobile scatterplot displays
Focus+Context

- Recap presentation problem: information space is too large to be displayed on a single screen
- Approaches in previous lecture
  - Zoomable user interface: scale and translate a single view of the information space
  - Overview+detail: use multiple views with different scale / detail granularity
- Focus+Context (f+c) means a presentation technique where both focus and context information are integrated into a single view by employing distortion
  - Local detail for interaction
  - Context for orientation
- No need to zoom out to regain context as in ZUIs
- No need to switch and relate between multiple separate views as in overview+detail interfaces
- Focus+context is commonly known as fisheye views
Focus+Context Screens  [Baudisch 2001]

http://www.youtube.com/watch?v=yKe2ahApaz4
Generalized Fisheye Views

• Furnas 1986
• Idea: trade-off of detail with distance
• Naturally occurring, e.g.
  – Employees being asked about the management structure: they know local department heads, but only the Vice president of remote divisions
  – Regional newspaper contain local news stories and only more distant ones that are compensatingly of greater importance (e.g. war in a remote country)
• Formalization
  – Presentation problem: interface can only display \( n \) items of a structure that has a number of items > \( n \)
  – Degree-of-interest function: assign importance value to each item in structure - only display the \( n \) most important items
Degree-of-Interest

- $\text{DOI}_{\text{fisheye}}(x|y) = \text{API}(x) - D(x,y)$
  - $\text{DOI}_{\text{fisheye}}$: the users‘ degree of interest in point $x$, given the focus point $y$
  - $\text{API}(x)$: Global a priori importance of point $x$
  - $D(x,y)$: distance between $x$ and focus point $y$

- Can be applied to any structure for which the components can be defined

- Example: rooted tree structure of programming code

- Components definition
  - $D(x,y) = d_{\text{tree}}(x,y)$ = path length between node $x$ and node $y$ in the tree
  - $\text{API}(x) = -d_{\text{tree}}(x,\text{root})$ = distance of node $x$ from the root node (nodes closer to the root are generally more important than nodes further away)

- $\text{DOI}_{\text{fisheye(tree)}}(x|y) = \text{API}(x) - D(x,y) = -(d_{\text{tree}}(x,y) + d_{\text{tree}}(x,\text{root}))$
Fisheye Tree

(a) Distance from y:
\[ d_{\text{true}}(x,y) \]

(b) A Priori Importance in the tree:
\[ \text{Imp}(x) = -d_{\text{true}}(x,\text{root}) \]

(c) The Fisheye DOI:
\[ \text{DOI}_{\text{fisheye}}(x, y) = \text{API}(x) - D(x,y) \]
\[ = -(d_{\text{true}}(x,y) + d_{\text{true}}(x,\text{root})) \]

An arithmetically larger number means that the node is more interesting for interactions focused on y
Fisheye Tree

- To obtain fisheye views of different sizes, set a DOI threshold $k$ with $\text{DOI}(x) > k$

$k = -3$; direct ancestral lineage
(a) Zero-order tree fisheye:

$k = -5$; siblings are added
(b) First-order tree fisheye:

$k = -7$; cousins are added
(c) Second-order tree fisheye:
Fisheye Tree Applied

- Full view of the program
  - Box: lines in default view
  - Underlines: lines in fisheye view
Fisheye Tree Applied

• Working on line marked with „>>“

```c
#define DIG 40
#include <stdio.h>

int main()
{
    int c, i, x[DIG/4], t[DIG/4], k = DIG/4, noprint = 0;
    while((c = getchar()) != EOF){
        if(c == '0' && c <= '9'){
            ...16
        } else {
            switch(c){
                case '+':
                    ...27
                case '-':
                    ...38
                case 'e':
                    ...39
                    for(i=0; i<k; i++) t[i] = x[i];
                    break;
                case 'q':
                    exit(0);
                    break;
                default:
                    noprint = 1;
                    break;
            }
        }
        if(!noprint){
            for(i=k-1; t[i] <= 0 && i > 0; i--);
            printf("%d", t[i]);
        }
    }
}
```

Figure 3. Standard 'flat-window' view of a C program. Line numbers are in the left margin.

Figure 4. A fiseheye view of the C program. Line numbers are in the left margin. "..." indicates missing lines.
Graphical Fisheye Views

- Applied rather to layouts than to logical structure
- Furnas fisheye: items are either present in full detail or absent from the view
- Objective: continuous distortion of items and item representation
Bifocal Display

- Spence & Apperley 1982
- Office environment of the future
- Virtual workspace showing documents on a horizontal strip
- Centered detail region and two compressed context regions
- Scroll compressed documents in the detail region to decompress
- Distortion increases the amount of information that can be displayed
Perspective Wall

- Robertson et al. 1991
- Same approach as the bifocal lens but using perspective
- Detail information about objects recedes into the distance
Document Lens

• Robertson & MackInlay 1993

Distortion Approaches Used

• Overview of the different distortion techniques

Bifocal display

Perspective wall

Document lens

Magnification
Transfer function
Graph Fisheye

- Sarkar & Brown 1994
- Fisheye lens for viewing and browsing large graphs
- Present focus vertex in high detail but preserve context
- Recap node-link representation
  - Vertex (node)
  - Edges (links)
How did they do that...?

- Focus: viewer’s point of interest
- Coordinates in the initial layout: regular geographic coordinates
- Coordinates in the fisheye view: fisheye coordinates
- Each vertex has
  - A normal position specified by geographic coordinates
  - Size (Length of the square-shaped bounding box)
  - A priori importance (API)
  - Edge
    - Straight line from one vertex to another OR
    - For bent edges: set of intermediate bend points
- Apart from the distortion, the systems calculates for each vertex:
  - Amount of detail (content) to be displayed
  - Visual weight: shall the vertex be displayed? - display threshold
Implementation

• Two step process
  – Apply geometric transformation to the normal view to reposition vertices and magnify / demagnify the bounding boxes
  – Use the API of vertices to determine their final size, detail, and visual weight

• Slides will only present the repositioning of vertices - for the remaining algorithm see the paper at ftp://ftp.cs.brown.edu/pub/techreports/93/cs93-40.pdf
Cartesian Transformation

- Compute the position of a point $P_{\text{norm}}$ from normal coordinates to fisheye coordinates

\[
P_{\text{feye}} = G \left( \frac{D_{\text{norm}}}{D_{\text{max}}} \right) D_{\text{max}} + P_{\text{focus}},
\]

\[
G \left( \frac{D_{\text{norm}}}{D_{\text{max}}} \right) D_{\text{max}} + P_{\text{focus}}
\]

- where

\[
G(x) = \frac{(d+1)x}{dx + 1}
\]

- $D_{\text{max}}$ : the horizontal / vertical distance between the boundary of the screen and the focus in normal coordinates
- $D_{\text{norm}}$ : horizontal / vertical distance between the point being transformed and the focus in normal coordinates
- $d$ : distortion factor, see graphs
Distortion Factor

- Example: distortion of a nearly symmetric graph
- Focus in the southeast
Polar Transformation

• With cartesian transformation all vertical and horizontal lines remain vertical and horizontal in the fisheye view
• Makes this approach well suited for abstract orthogonal layouts of information spaces (e.g. circuit design, UML diagrams, etc.)
• Problem: does not seem very natural
• Alternative approach: distorting the map onto a hemisphere using polar coordinates (origin = focus)
• Point with normal coordinates \((r_{\text{norm}}, \theta)\) is mapped to fisheye coordinates \((r_{\text{feye}}, \theta)\), where

\[
r_{\text{feye}} = r_{\text{max}} \frac{(d + 1) r_{\text{norm}}}{d r_{\text{norm}} + 1}
\]

• \(r_{\text{max}}\): maximum possible value of \(r\) in the same direction as \(\theta\)
• Note: \(\theta\) remains unchanged, origin of polar coordinates is the focus
• Distortion forms a pyramid lens
• Users know this effect from lenses and elastic materials in the real world, often find it fascinating
Cartesian vs Polar Transformation

Cartesian

Polar
More Fisheye Lenses

• Gutwin & Fedak 2004

Original pyramid lens (polar transformation, full screen)

Constrained hemispherical lens: constrain polar algorithm to a fixed radius

Constrained flat-hemispherical lens: insert a region of constant magnification
Discussion break

• What do you think of this? Ideas?
Victor Vasarely (1906-1997)
Multiple Foci

- Keahey & Robertson 1996
- Also multiple foci in a single domain are possible
- Interesting question: how to handle overlap?

Clipped  Weighted average  Composition transformation
Problem: Focus Targeting

• Gutwin 2002
• Move the fisheye lens to a target
• Problem: targets appear to move and thus are more difficult to hit directly (same effect as with a simple magnifying lens)
• Movement is in the opposite direction to the motion of the fisheye lens: focus target will move towards the approaching lens and vice versa
Focus Targeting

• Even worse: with the fisheye lens, targets move towards the focus more and more rapidly as the focus approaches them
• Depending on the distortion factor, the targets may move several times faster than the focus
• Leads to overshooting
• Approach to reduce problem: speed-coupled flattening
  – Detecting a target acquisition, the system automatically reduces the distortion
  – Distortion is automatically restored when the target action is completed
  – Algorithm is based on pointer velocity and acceleration thresholds
Speed-Coupled Flattening

• Found to significantly reduce targeting time and errors

Figure 4. Speed-coupled flattening. Top row shows the fisheye view and pointer path. Bottom row shows a stylized plot of pointer velocity and distortion level. The dotted line indicates the point in time that the corresponding screen was captured.

Gutwin 2002
Discussion: Mac OS X Dock
Symbolic Representation of Context

- F+c is limited to small zoom factors
- Allow for greater zoom factors by fusing graphical and symbolic content representations
- Example: Table lens (Rao & Card et al. 1994), (screenshot taken from inxight.com)
- Visualizes many more rows than a conventional spread sheet application
- Simple squishing of text rows would have rendered the content in the context unreadable
- Instead use small-size encodings of attribute values
Summary Focus+Context

• Advantages
  – Overview information is provided
  – No visual switching between separate views (compared to O+D)
  – Less display space is needed (compared to O+D)

• Potential problems
  – Performance is strongly task-dependent
  – Distortion has negative effect on the perception of proportions, angles, distances
  – Hampers precise targeting and the recall of spatial locations
  – Usually only suitable for small zoom factors: maximum of 5
    (Shneiderman & Plaisant 2005)
  – Can be inappropriate for visualizing maps (usually require high fidelity to the standard layout)
LaunchTile & AppLens

• ZUI and fisheye approach (Karlson et al. 2005)
Related Literature