CS-E4840
Information Visualization
Lecture 8a: Interaction

Tassu Takala <tapio.takala@aalto.fi>

25 March 2019

Recap

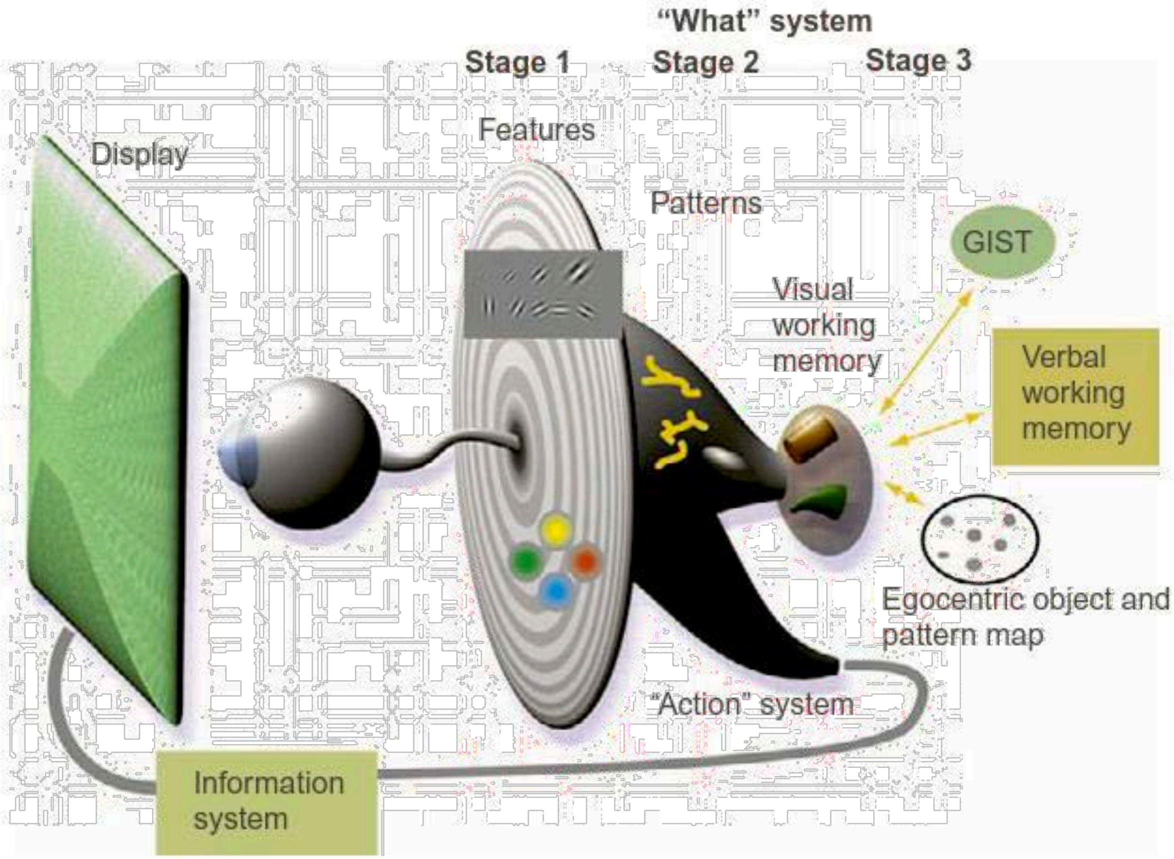
Visual patterns

Summary on glyph design

- Certain visual features “pop out” (pre-attentive features)
- Data variables should (usually) be mapped to pre-attentive features (they are processed fast)
- Restrictions (if you want pre-attentive design):
  - conjunction searches are usually not pre-attentive
  - one can effectively display only limited number of visual variables, with limited accuracy
  - integral visual dimensions interfere with each other: you should use separable dimensions instead

A model for perceptual processing

1. Parallel processing to extract low-level properties of the visual scene
   - rapid parallel processing
   - extraction of features, orientation, colour, texture and movement patterns
   - iconic store
   - bottom-up, data driven processing
2. **Pattern perception**
   - slow serial processing
   - involves both working memory and long-term memory
   - arbitrary symbols relevant
   - different pathways for object recognition and visually guided motion
3. Visual working memory



Ware 2013

Patterns in 2D data

- Exploratory visualization is based on finding patterns from data
- Oversimplification: the patterns are recognized between pre-attentive processing and higher level object perception
- Relevant questions:
  - How do we see groups?
  - How can 2D space be divided into perceptually distinct regions?
  - When are two patterns similar?
  - When do two different elements appear to be related?
- Patterns may be perceived even where there is only visual noise

Gestalt laws

- **Gestalt** is form in German
- The Gestalt School of Psychology (1912 onwards) investigated the way we perceive form
- They produced several Gestalt laws (laws of organisation) of pattern perception
- The Gestalt laws translate directly into design principles of visual displays
- Many of the rules seem obvious, but they are violated often

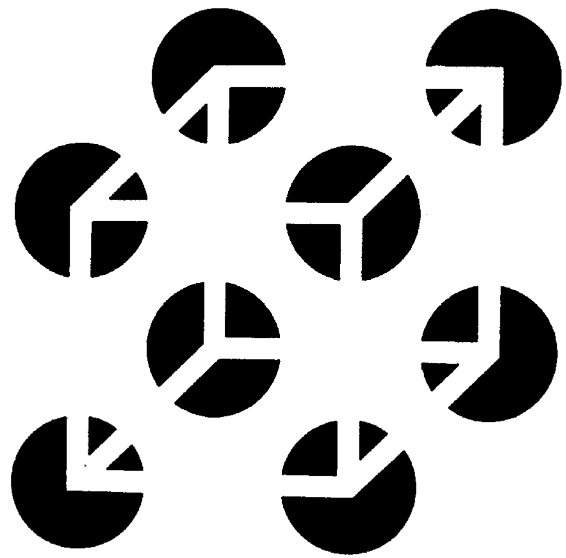


Figure 1. The subjective Necker cube. A phenomenally complete Necker cube can be seen overlying a white surface and eight black discs; so viewed, illusory contours corresponding to the bars of the cube can be seen extending between the discs. The illusory bars of the cube disappear when the discs are seen as 'holes' in an interposing surface, through which the corners of a partially occluded cube are viewed; curved subjective contours are then seen demarcating the interior edges of the 'holes'

Bradley and Petry 1977

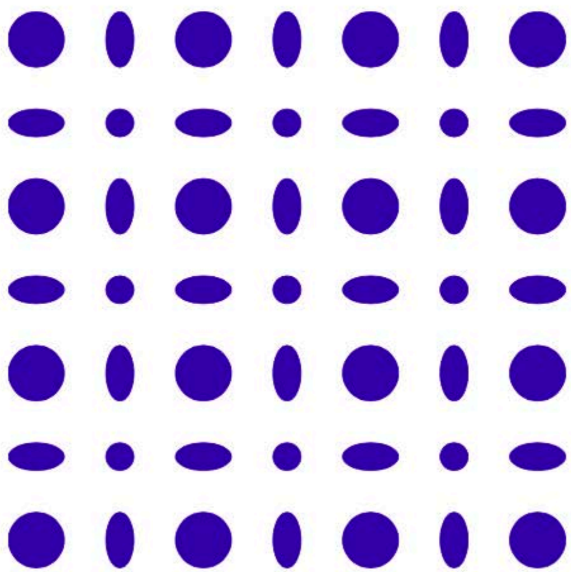
Gestalt laws

- Similarity
- Good continuation
- Proximity
- Symmetry
- Closure
- Relative size
- Common fate

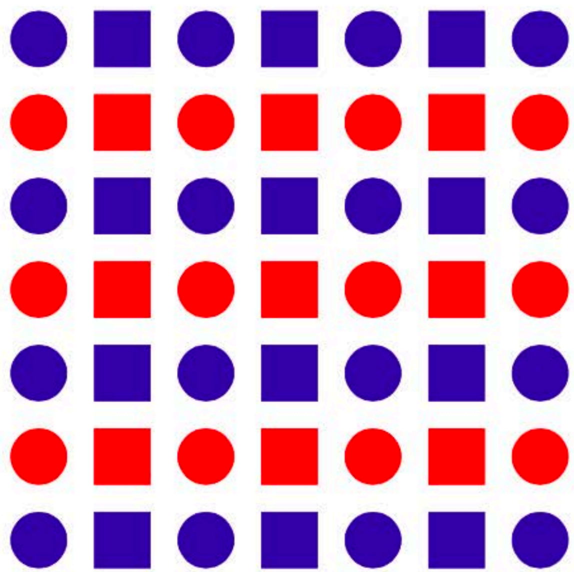
- some “new” motion-based Gestalt(-like) laws:
  - Patterns from motion
  - Animation and perception of shapes
  - Causality

Similarity

- Similar objects appear to be grouped together
- When designing a grid layout of a data set, code rows and/or columns using low-level visual channel properties, such as colour and texture



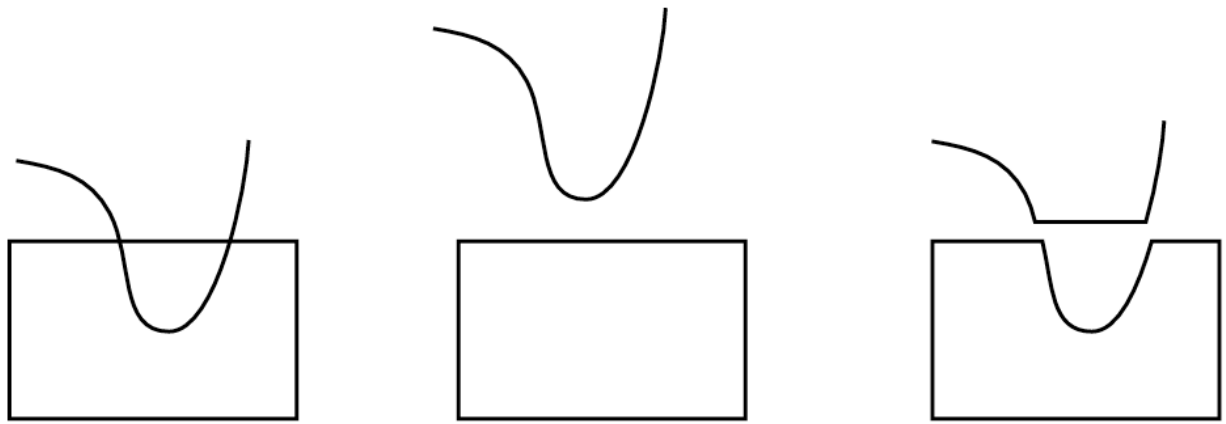
integral dimensions
emphasise overall pattern



separable dimensions
segment rows and columns

Good continuation

- Visual complete objects are more likely to be constructed from visual elements that are smooth and continuous, rather than ones that contain abrupt changes in direction
- In networks, lines connecting nodes should be smooth and continuous, so the nodes are easily identified

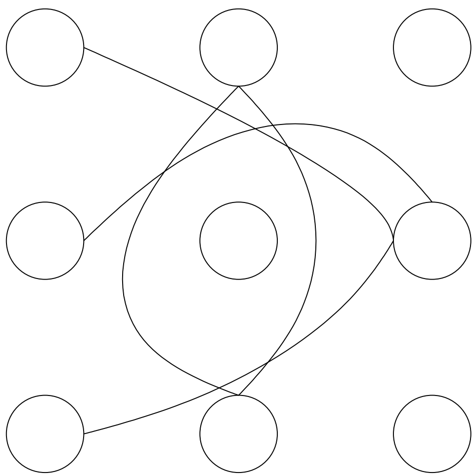


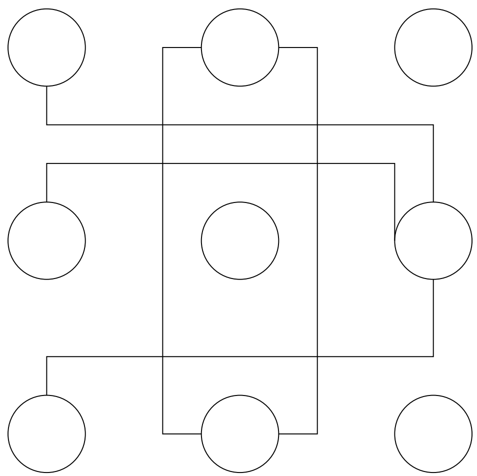
The pattern on the left is perceived as a curve overlapping a rectangle (centre) rather than 2 irregular shapes touching (right).

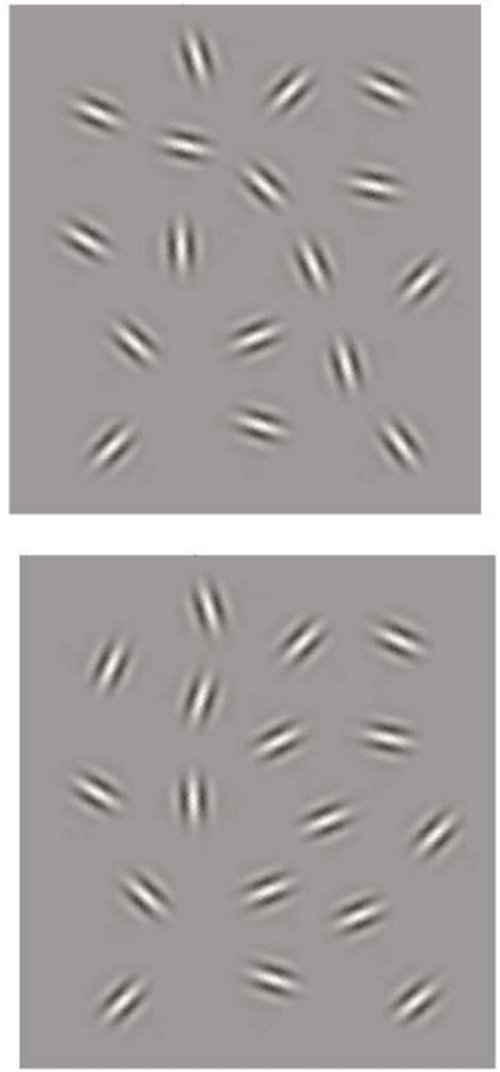
Ware 2013

Good continuation

- Connectedness is one of the most powerful grouping principles
- It is easier to perceive connections when contours run smoothly



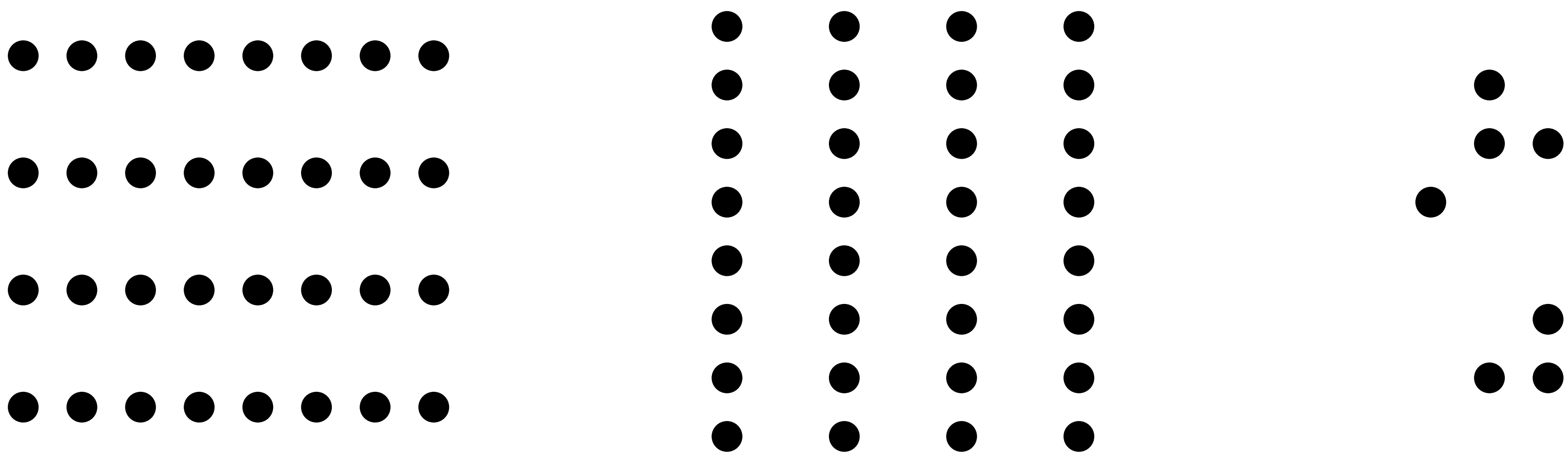




Ware 2013

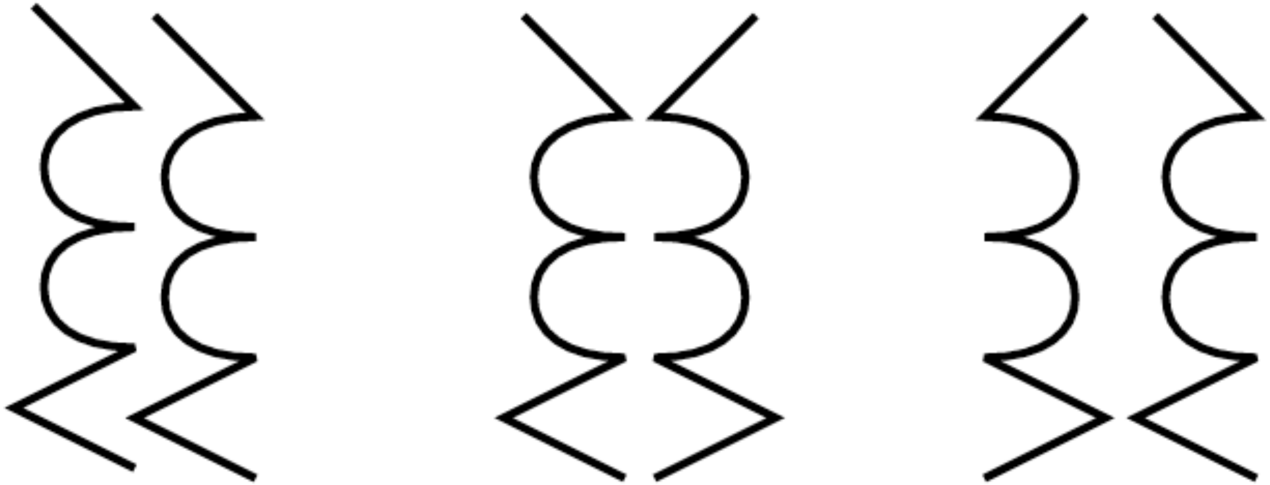
Proximity

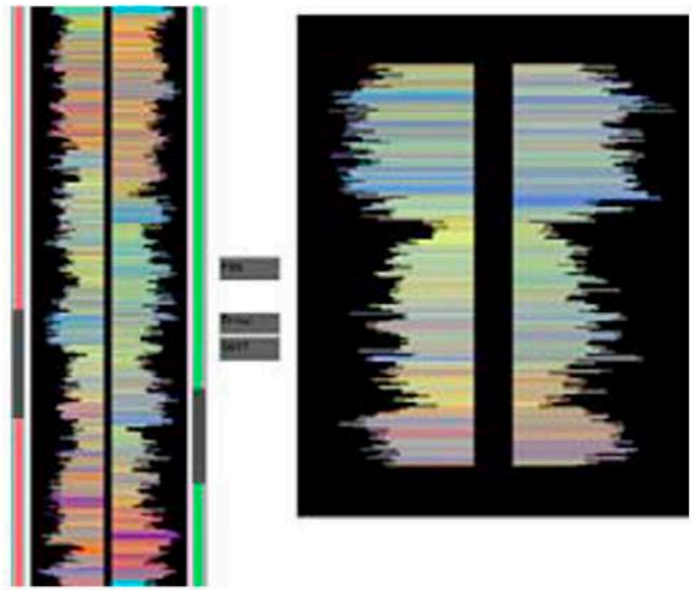
- Things that are near to each other appear to be grouped together
- Proximity is one of the most powerful gestalt laws
- Place the data elements into proximity to emphasise connections between them



Symmetry

- Symmetrically arranged pairs of lines are perceived together
- Use symmetry to make pattern comparisons easier
- Symmetrical relations should be arranged on horizontal or vertical axes (as symmetries are more easily perceived), unless a framing pattern is used

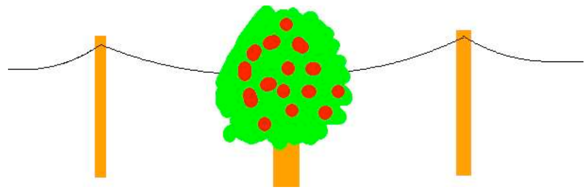


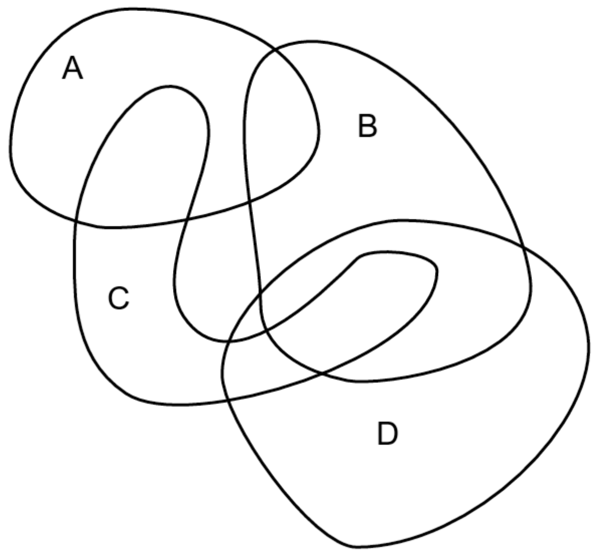


Ware 2013

Closure

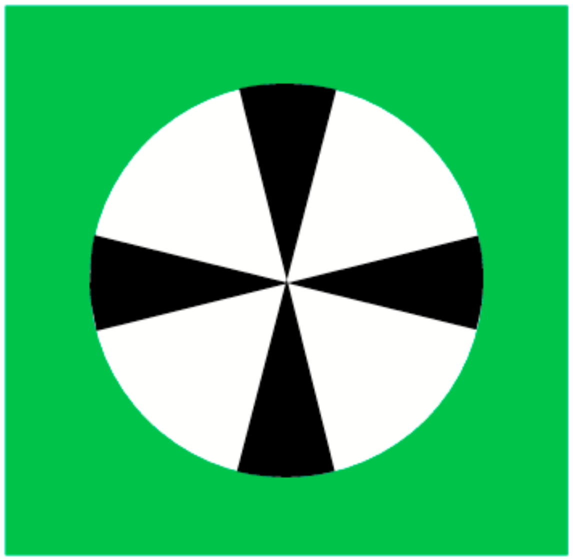
- A closed contour tends to be seen as an object
- There is a perceptual tendency to close contours that have gaps in them
- When a closed contour is seen, there is a very strong perceptual tendency of dividing space into a region enclosed by the contour (a common region) and a region outside the contour
- In window-based interface strong framing effects inhibit between window comparisons: related items should not be based in separate windows





Relative size

- Smaller components of a pattern tend to be perceived as an object

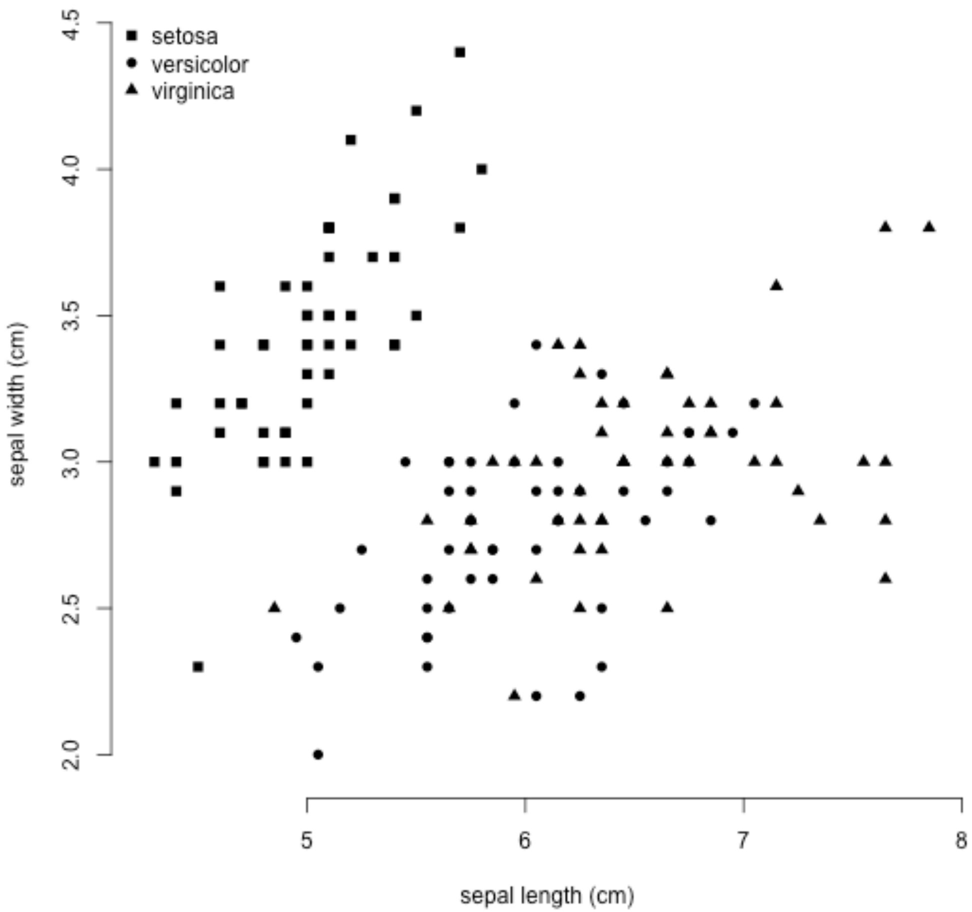


Rubin's reversible face-vase figure (multistability)

Ware 2013

Common fate

- Relative motion is an extremely efficient method of showing patterns from data
- Data points oscillate around center point
- Variables: frequency, phase, amplitude of motion
- Phase is the most effective variable



Animation and perception of shape

- Gestalt laws also work for animated images: structures and patterns are seen from partial data (as with static images)
- Mystery lights in the dark:

No delay

Causality

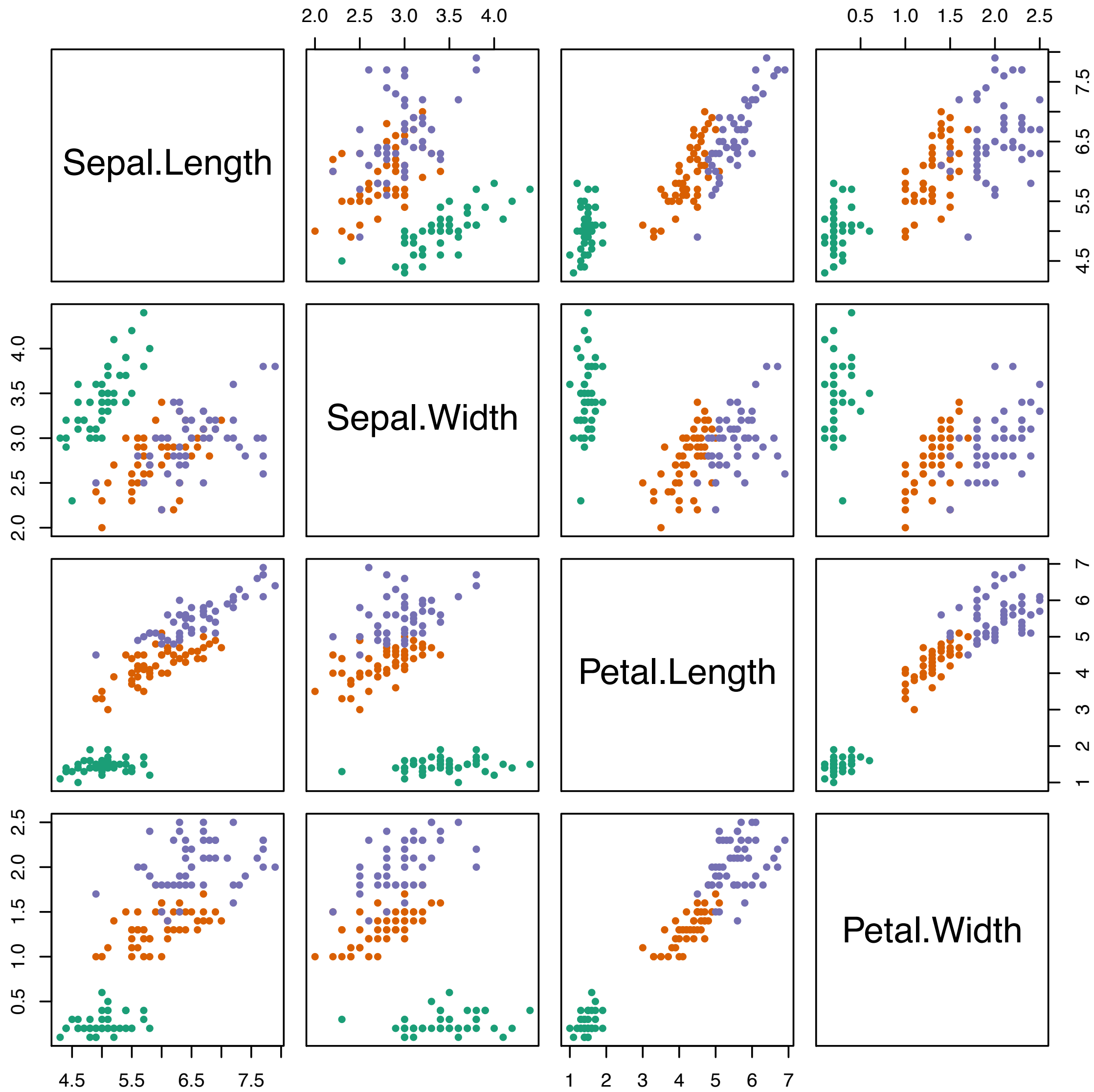
- *Launching:* an object is perceived to set another into motion
- Perception of launching requires precise timing (delays less than 0.07-0.16 s)
- Already infants can perceive causal relations, such as launching

Delay of 0.2 s

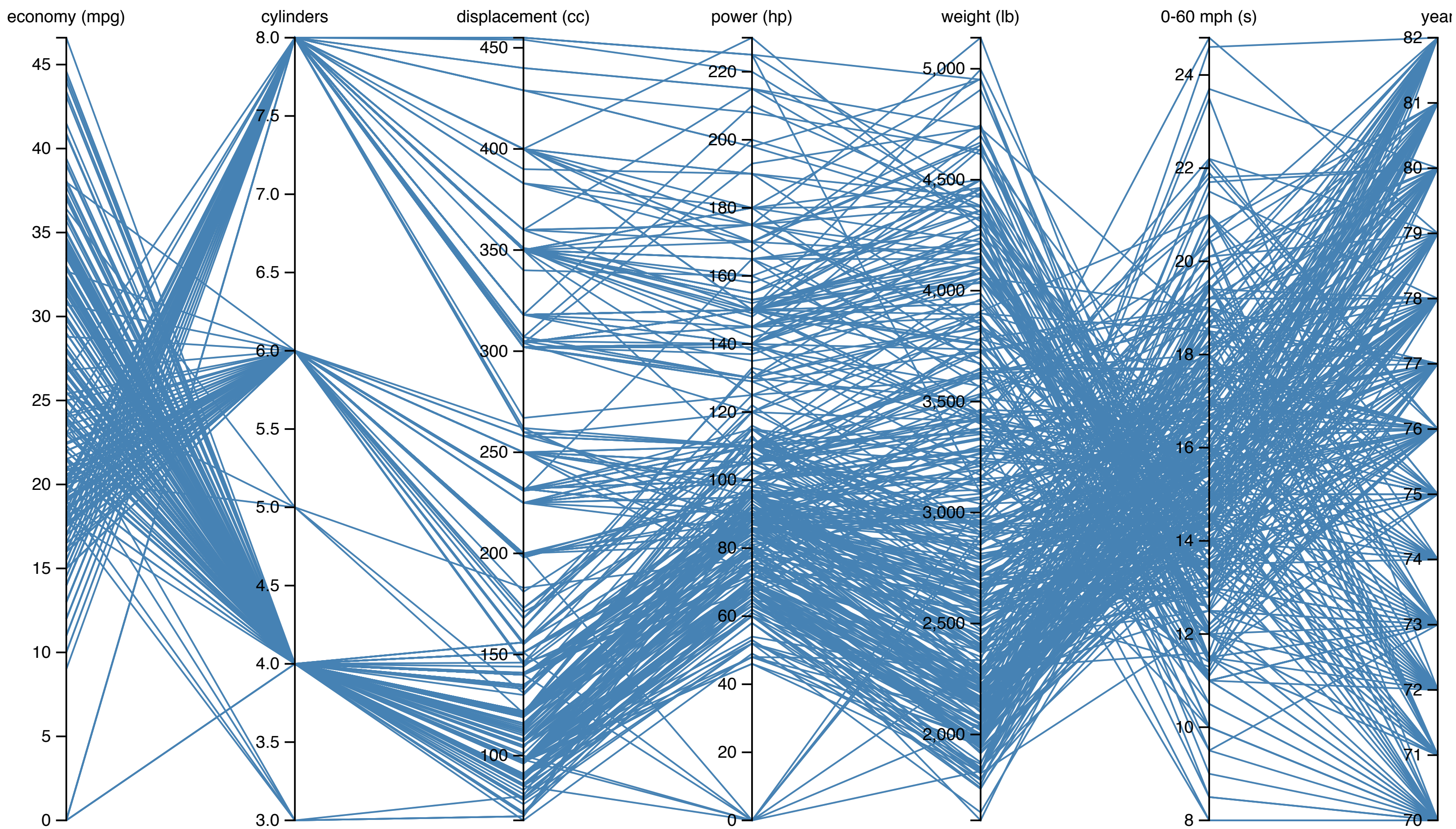
Sometimes it is difficult for you to guide your attention

Reading this text might be difficult because of the famous Finnish politician stealing your attention. Motion and especially *appearance* of a new object attracts attention. Human faces seem to be especially effective. This seems right and makes ecological sense. When early man was outside a cave, awareness of emerging objects in the periphery would have had clear survival value. Such movement may have signalled immediate and deadly danger.

Small multiples (trellis)



Parallel coordinates



https://bl.ocks.org/jasondavies/1341281

Big data: too much for one view?

Dynamic visualization

- interactive navigation in information space
- data reduction techniques
  (clustering, dimensionality reduction)

Interactive visualisations

- Interactive visualisations can be characterised by *feedback loops*

- Three levels of feedback:
  1. visual-manual control loop (data manipulation)
  2. **view refinement and navigation control loop (exploration and navigation)** [discussed here]
  3. problem solving loop

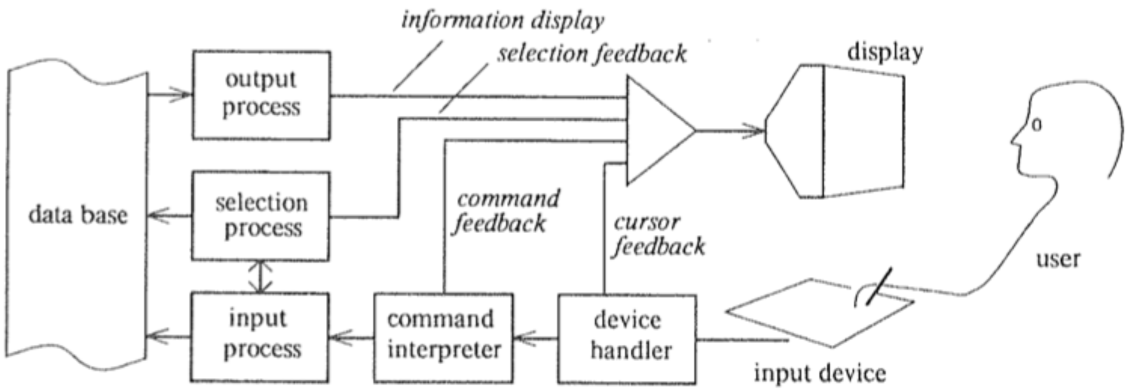


Figure 28-10 Expanded model of the interactive process showing feedback paths.

- Relevant time scales:
  1. ~0.1 s (psychological moment)
  2. ~1 s (unprepared response)
  3. ~10 s (unit task)

Way-finding in real spaces

- Seigel and White (1975):
  1. Key landmarks (e.g., post office, church) are learned with no spatial understanding (*declarative knowledge*)
  2. Procedural knowledge about routes from a location to another is learned, landmarks act as decision points (e.g., turn left at church; *procedural knowledge*)
  3. Cognitive map is formed (e.g., the church is about 1 kilometre north from train station; *cognitive spatial maps*)

- Cognitive maps form more rapidly if they have access to maps

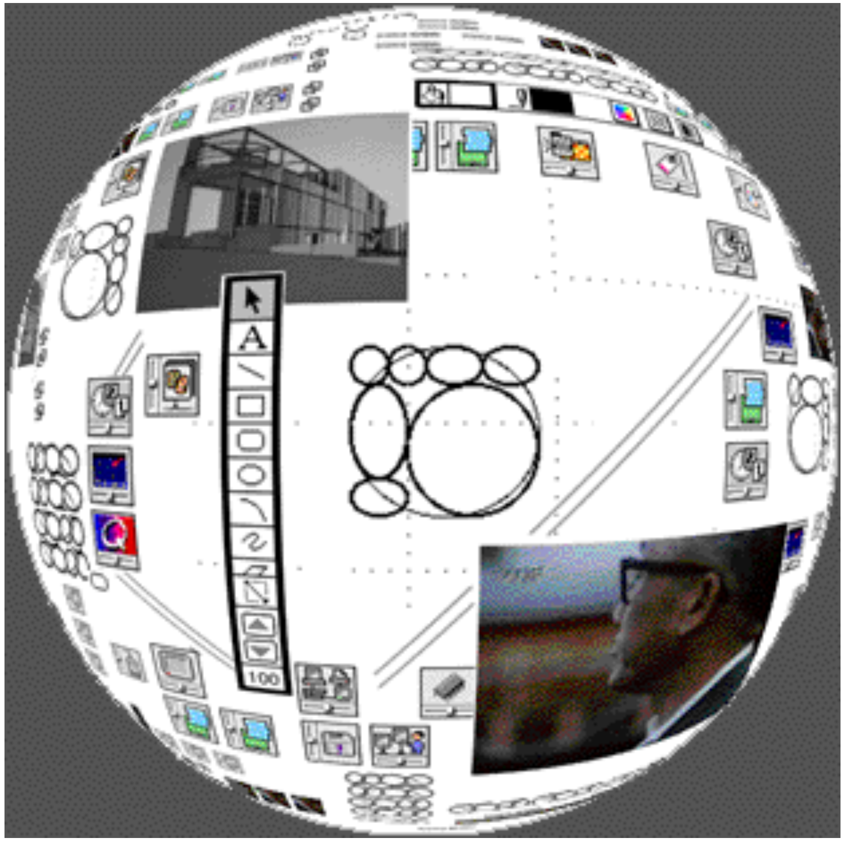
- Lessons to accelerate formation of cognitive maps: provide **distinctive landmarks** (focus) and **overview maps** (context)

Navigation + focus&context

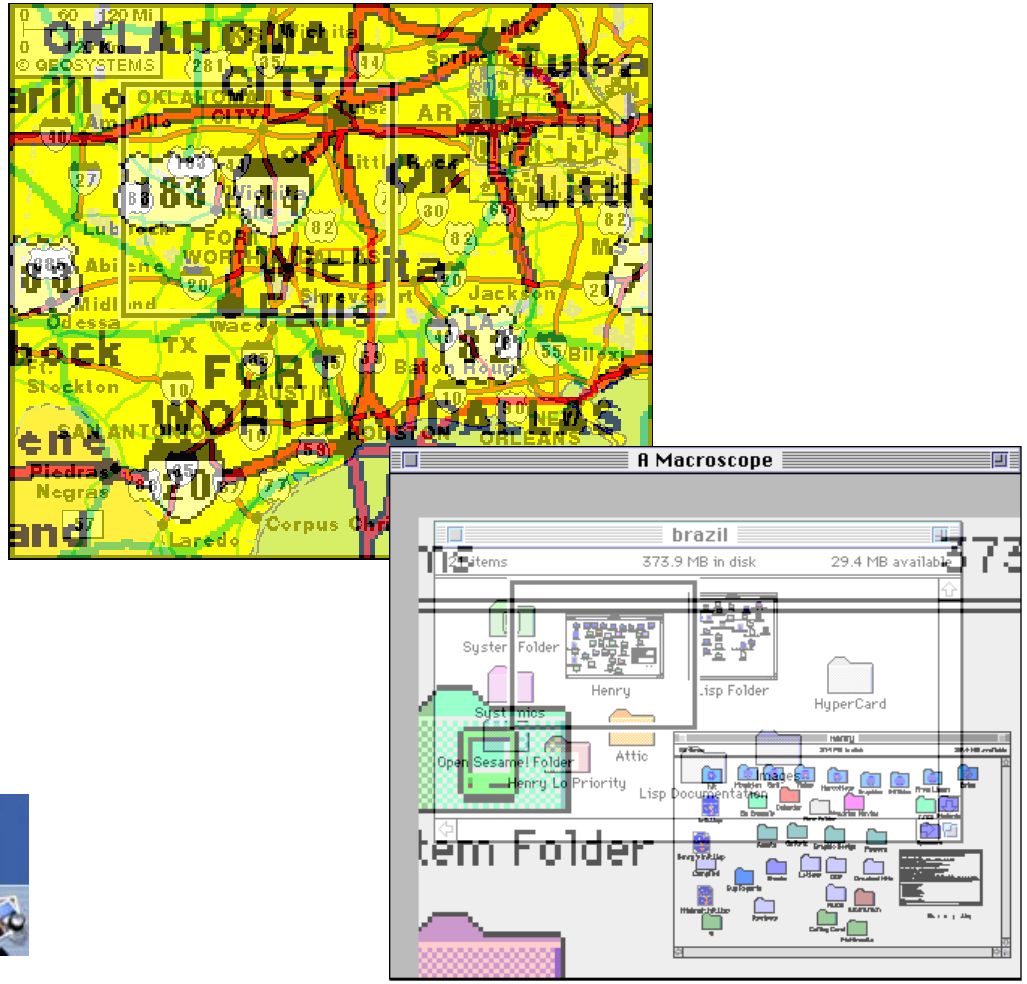
- **Focus+context problem**: how to find details from a larger context in information space. Or, how to *navigate efficiently* in abstract spaces.
- **Effective view navigation** (Furnas 1997): how to present information such that the traversal time from one node to another is minimised; and the network is navigable (all targets should have a good residue in each node)
- There are several visual techniques to help this (providing user overview, position and landmarks):
  - **Elision techniques.** Part of the structure are hidden until they are needed.
  - **Distortion techniques.** Magnify regions of interest, decrease space of irrelevant regions.
  - **Rapid zooming techniques.** User zooms in and out of regions of interest.
  - **Multiple windows.** Some windows show overview and others content.
  - **Micro-macro readings.** A good static visualisation supports focus+context.

Showing focus&context simultaneously in 2D space

Fisheye distortion



Simultaneous linear scales
(works well when animated – why?)



Mac OSX Dock

Effective View Navigation in abstract information space

- Theoretical view by Furnas (1997) https://doi.org/10.1145/258549.258800

- The information landscape can be thought as a tree or network G

- Effective View Navigation in G, EVN(G): how to organise information with links so that we have

EVT efficient traversal

  - small views: number of outgoing links from a view (maximal *out-degree*, MOD) is small;

  - short paths: the expected cost of traversal (number of steps, defined by *network diameter*, DIA) is minimised;

and

VN view navigable

  - all targets have a good *residue* ('scent' of target) in each node, and *outlink-info* is small
    - requires good semantic classification of nodes

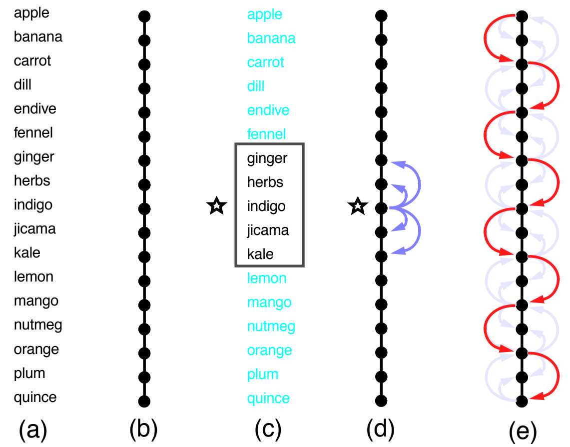


Figure 1. *(a) Schematic of an ordered list, (b) logical graph of the list, (c) local window view of the list, (d) associated part of viewing graph, showing that out degree is constant, (e) sequence of traversal steps showing the diameter of viewing graph is O(n).*

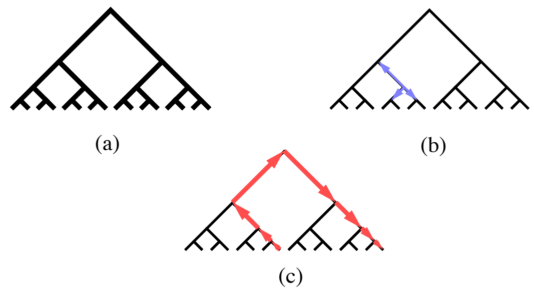


Figure 2. *An example of an Efficiently View Traversable Structure (a) logical graph of a balanced tree, (b) in gray, part of the viewing graph for giving local views of the tree showing the outdegree is constant, (c) a path showing the diameter to be O(log(n)).*

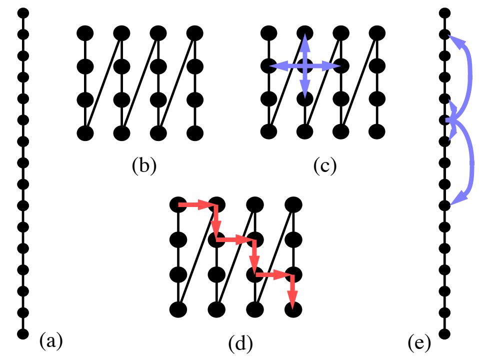


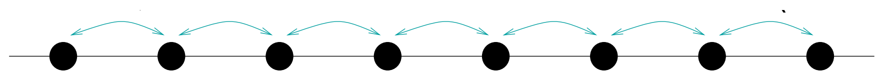
Figure 3. *Fixing the list viewer. (a) logical graph of the ordered list again, (b) the list is folded up in 2-D (c) part of the viewing graph showing the 2-D view-neighbors of Node6 in the list: out degree is O(1), (d) diameter of viewing graph is now reduced to O(sqrt(n)). (e) Unfolding the list, some view-neighbors of Node6 are far away, causing a decrease in diameter.*

Notes on Furnas' EVN paper

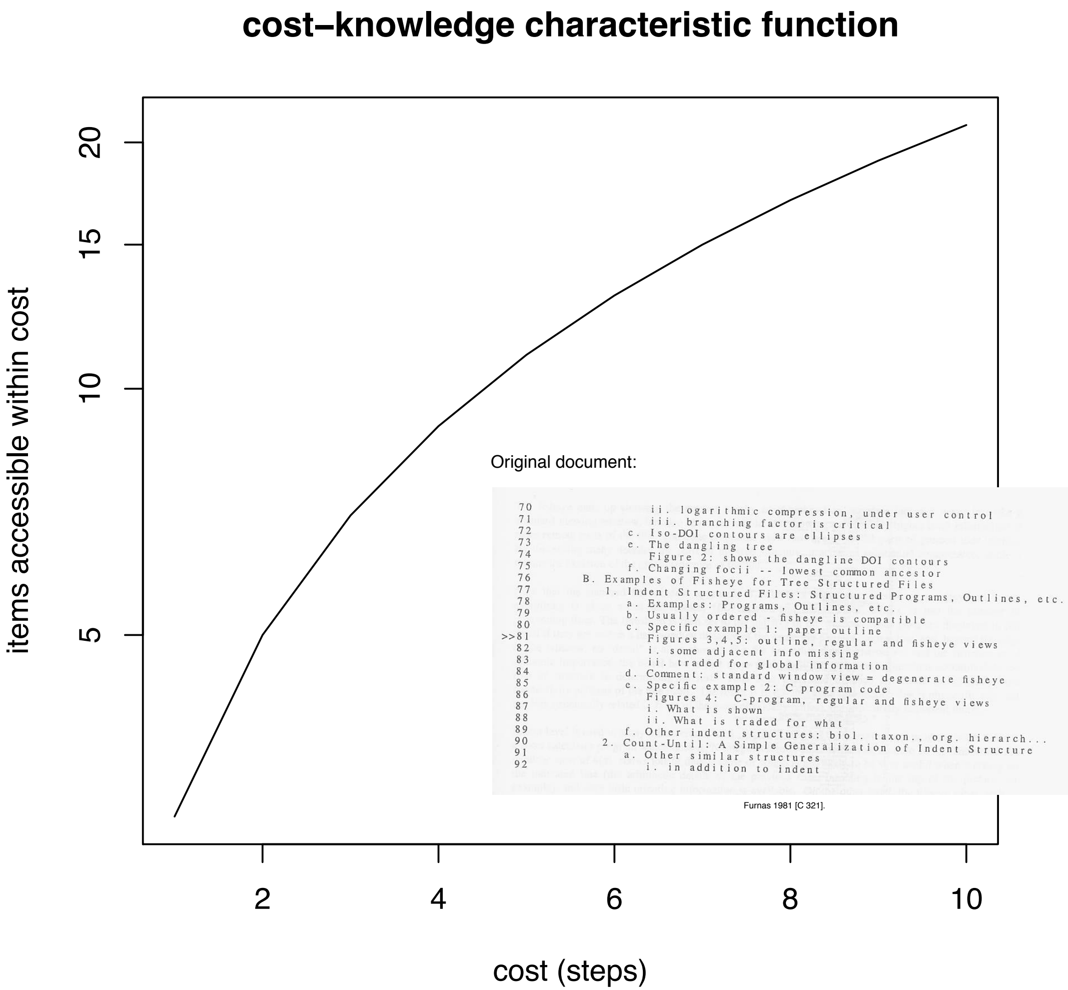
- Theoretical view ⇒ can be applied in very different cases
- Written in 1997, when WWW was relatively new
  - now search engines are often more effective than navigation with explicit links
  - further development: semantic web
  - (in both, search is based on auxiliary metadata)

- Example of EVN in the web: Wikipedia
  - organized (partly) with hierarchical categories
  - rich additional cross linking

Furnas' fisheye view



- Introduced by Furnas in 1981, using text as an example. The concept can (and has been) generalised also to other data structures.
- Accessing large structures (like a text document or program code) by scrolling is slow. For example, it takes on average $\mathcal{O}(N)$ steps to scroll from one line of text to another, where N is the number of items (lines of text)
- Cost-knowledge characteristic function (CKCF) is the number of items (lines of text) that user can access as a function of steps (or time)



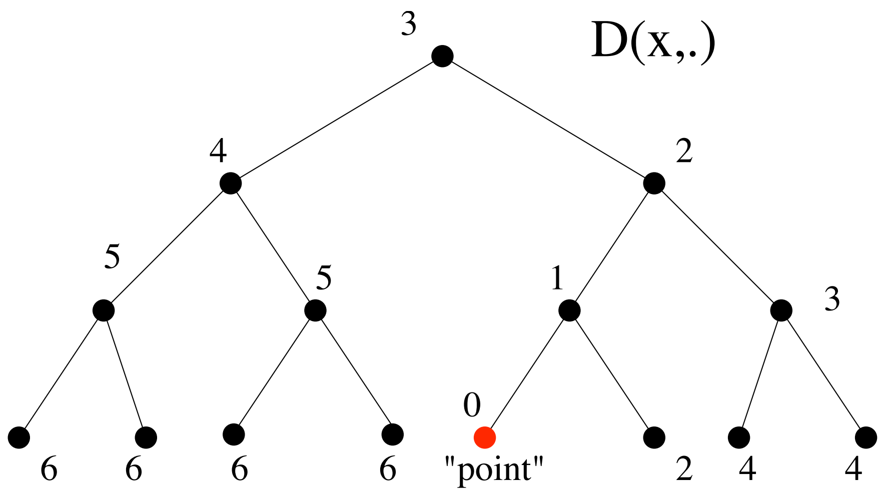
Original document:

```
 70          ii. logarithmic compression, under user control
 71          iii. branching factor is critical
 72        c. Iso-DOI contours are ellipses
 73        e. The dangling tree
 74           Figure 2: shows the dangline DOI contours
 75        f. Changing focii -- lowest common ancestor
 76    B. Examples of Fisheye for Tree Structured Files
 77      1. Indent Structured Files: Structured Programs, Outlines, etc.
 78        a. Examples: Programs, Outlines, etc.
 79        b. Usually ordered - fisheye is compatible
 80        c. Specific example 1: paper outline
>>81          Figures 3,4,5: outline, regular and fisheye views
 82           i. some adjacent info missing
 83           ii. traded for global information
 84        d. Comment: standard window view = degenerate fisheye
 85        e. Specific example 2: C program code
 86           Figures 4:  C-program, regular and fisheye views
 87           i. What is shown
 88           ii. What is traded for what
 89        f. Other indent structures: biol. taxon., org. hierarch...
 90      2. Count-Until: A Simple Generalization of Indent Structure
 91        a. Other similar structures
 92           i. in addition to indent
```

Furnas 1981 [C 321].

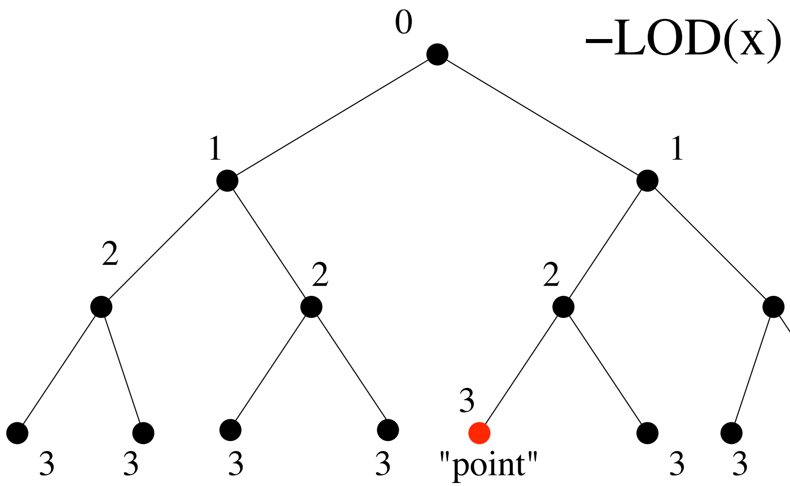
Furnas' fisheye view

- Notation:
  - . (*focal point*)
  - $D(.,x)$ (*distance from focus*), $D(.,.) = 0$.
  - Example: $D(.,x)$ is the number of links intervening on the path between two nodes



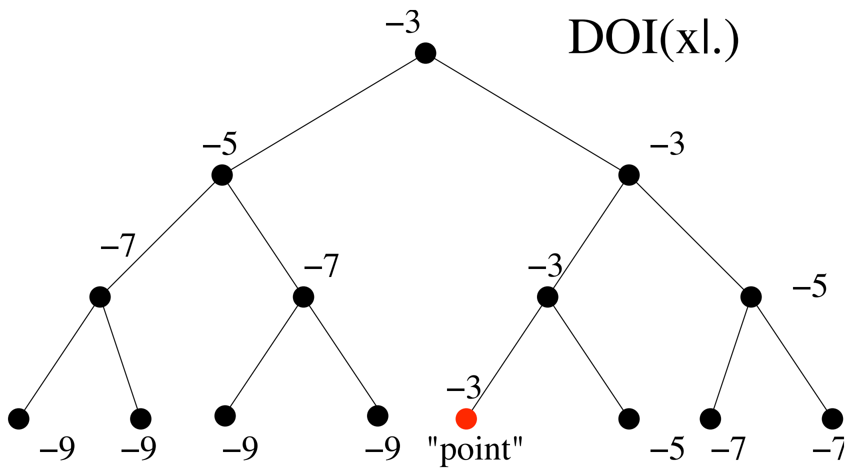
Furnas' fisheye view

- Notation (continued):
  - $LOD(x)$ (*level of detail*).
  - Example: $LOD(x) = -D(r,x)$, where r is the root of the tree



Furnas' fisheye view

- Notation (continued):
  - $DOI(x|.) = F(LOD(x), D(.,x))$ (*degree of interest*), where F is monotonously increasing in the first argument and decreasing in second.
  - Example: $DOI(x|.) = LOD(x) - D(.,x) = -D(.,x) - D(r,x)$



- Fisheye view: display x if and only if the degree of interest $DOI(x|.)$, is above some threshold k, $DOI(x|.) > k$.

Furnas' fisheye view

Original document:

```
  70             ii. logarithmic compression, under user control
  71             iii. branching factor is critical
  72           c. Iso-DOI contours are ellipses
  73           e. The dangling tree
  74             Figure 2: shows the dangline DOI contours
  75           f. Changing focii -- lowest common ancestor
  76       B. Examples of Fisheye for Tree Structured Files
  77         1. Indent Structured Files: Structured Programs, Outlines, etc.
  78           a. Examples: Programs, Outlines, etc.
  79           b. Usually ordered - fisheye is compatible
  80           c. Specific example 1: paper outline
>>81             Figures 3,4,5: outline, regular and fisheye views
  82             i. some adjacent info missing
  83             ii. traded for global information
  84           d. Comment: standard window view = degenerate fisheye
  85           e. Specific example 2: C program code
  86             Figures 4:  C-program, regular and fisheye views
  87             i. What is shown
  88             ii. What is traded for what
  89           f. Other indent structures: biol. taxon., org. hierarch...
  90         2. Count-Until: A Simple Generalization of Indent Structure
  91           a. Other similar structures
  92             i. in addition to indent
```

Furnas 1981 [C 321].

Furnas' fisheye view

"…" indicate missing lines, ">>" signals the current line:

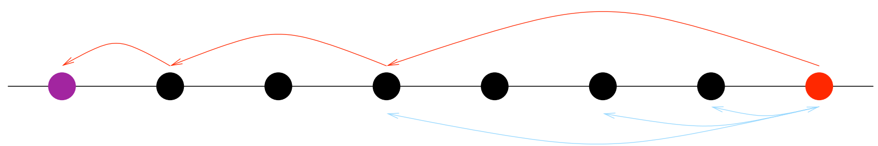
```
     1 The FISHEYE view: a new look at structured files
     2    I. ABSTRACT
     3    II. INTRODUCTION
 ...23    III. GENERAL FORMULATION
 ...51    IV. A FISHEYE DEFINED FOR TREE STRUCTURES
    52       A. The Underlying Fisheye Construction and its Properties
 ...76       B. Examples of Fisheye for Tree Structured Files
    77          1. Indent Structured Files: Structured Programs, Outlines, etc.
    78             a. Examples: Programs, Outlines, etc.
    79             b. Usually ordered - fisheye is compatible
    80             c. Specific example 1: paper outline
  >>81             Figure 3: outline, regular and fish views
    82                i. some adjacent info missing
    83                ii. traded for global information
    84             d. Comment: standard window view = degenerate fisheye
    85             e. Specific example 2: C program code
 ...89             f. Other indent structures: biol. taxon., org. hierarch...
    90          2. Count-Until: A Simple Generalization of Indent Structure
...100          3. Examples of the Tree Fisheye: Other Hierarchical Structures
...106     V. FISHEYE VIEWS FOR OTHER TYPES OF STRUCTURES
...117     VI. A FEW COMMENTS ON ALGORITHMS
...140     VII. OTHER ISSUES
...162     VIII. CONCLUDING REMARKS AND SUMMARY
```

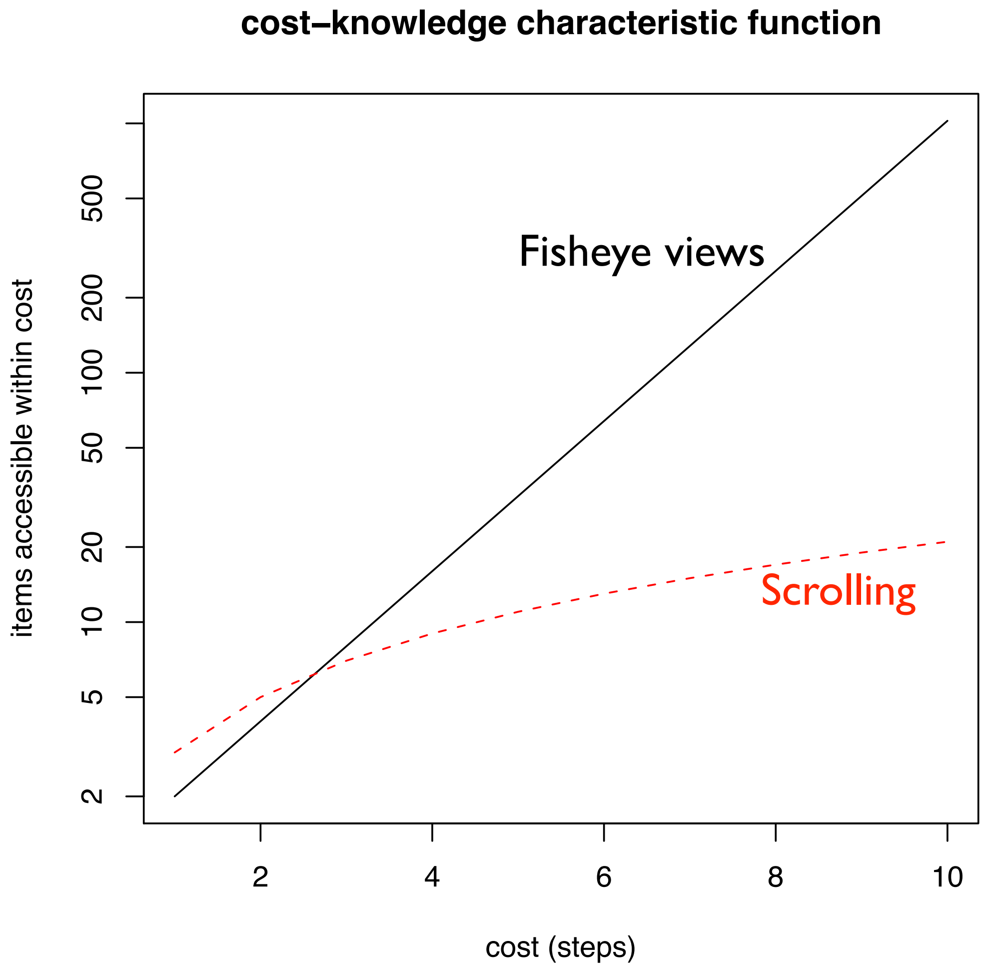
Furnas 1981 [C 321].

NB. Common technique (outline view) in current text editors

Furnas' fisheye view

- The Fisheye principle can be applied to all hierarchical (tree-like) data structures, if the Degree of Interest (DOI) function can be defined
- The expected cost of finding an arbitrary line (document) by traversing through Fisheye views is O(log N)
- One potential problem: the Fisheye view shows mbK nodes, where b is the branching factor of the tree, m is the height of the tree and K is the fisheye-order, typically adjustable by the user (in our example K=-1-D(.,r))

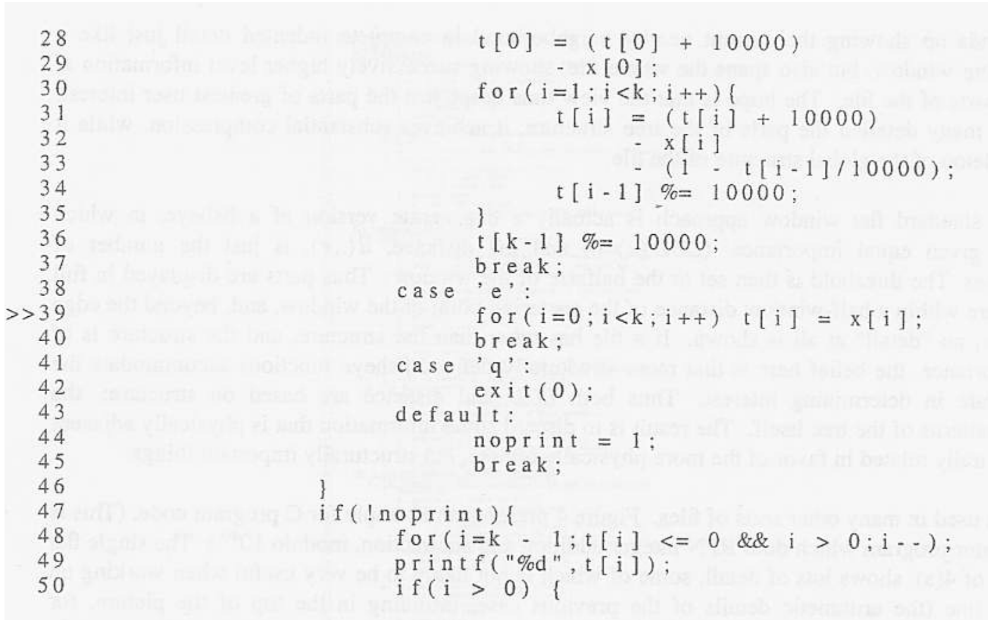




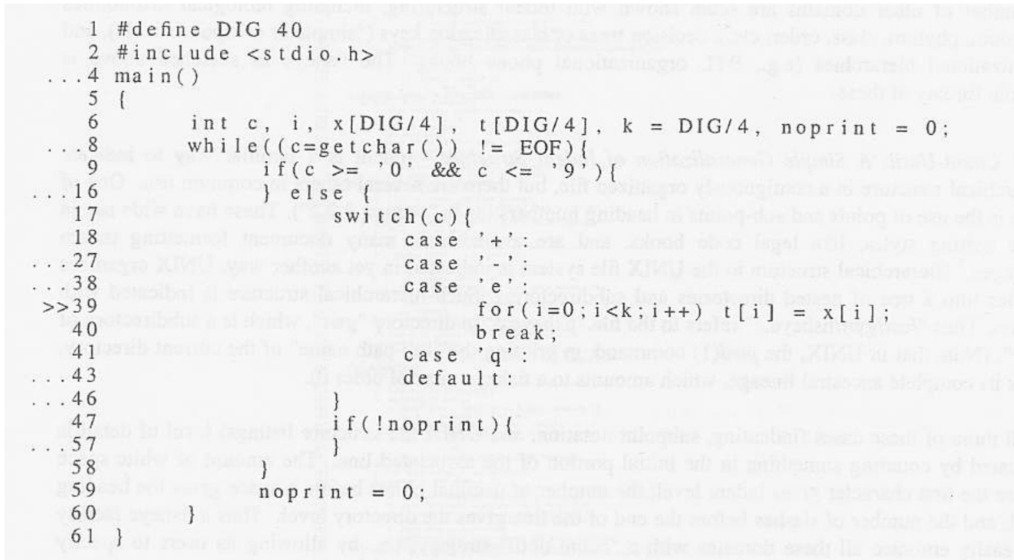
(Fisheye cost shown above is approximate; exact cost depends on the shape of the tree etc.)

Furnas' fisheye view

- Fisheye view satisfies the requirements of *effective view navigation* (Furnas 1997), resulting to good cost-knowledge characteristic function:
  - *effective view traversable*:
    - reasonable number of choices at each step
    - path from line *x* to line *y* is short, *O(log N)*
  - *navigability:*
    - every view (screenshot)provides information (*residue*) that helps user to find the shortest path to line *x*

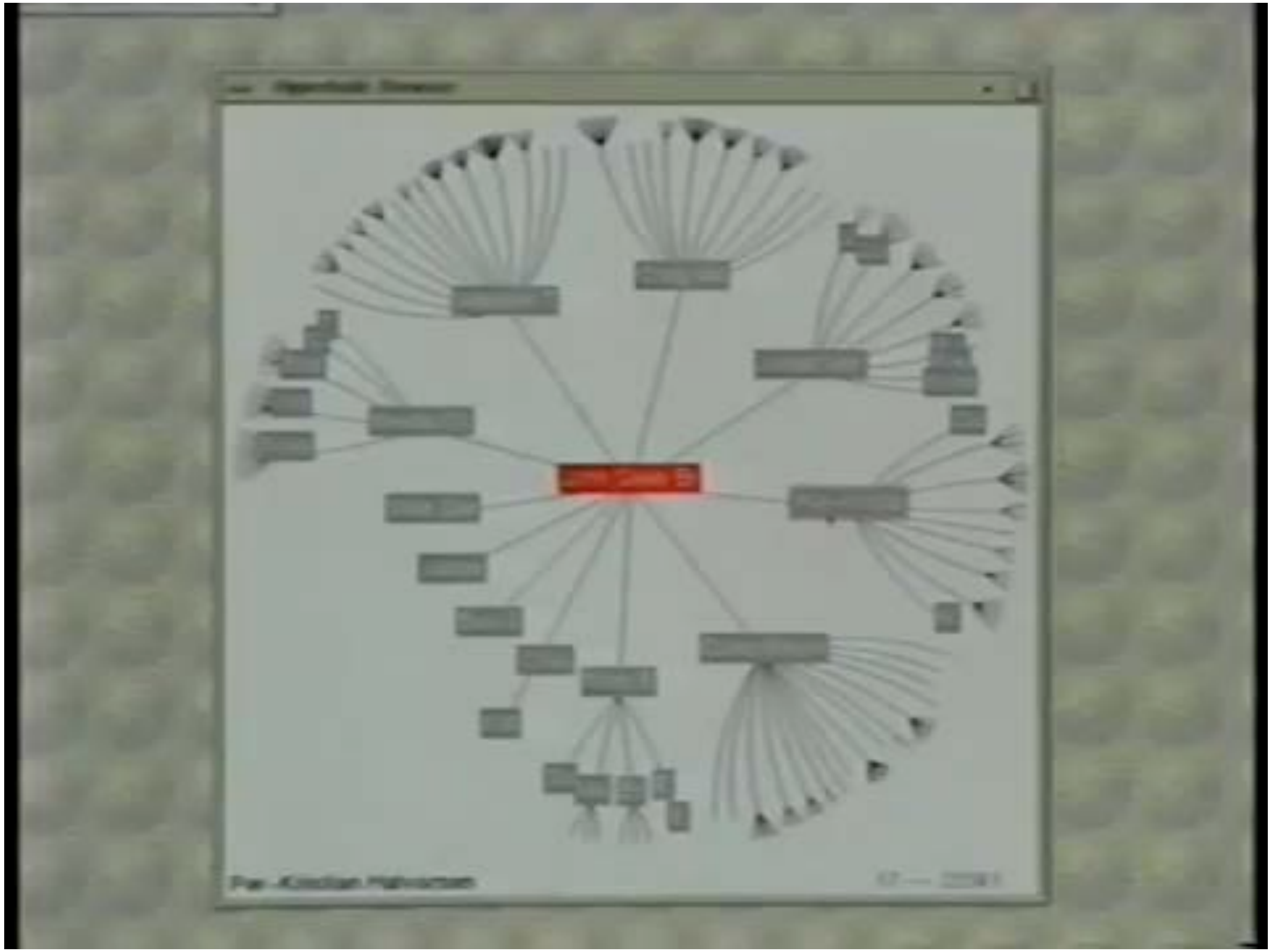


```
  28                  t[0] = (t[0] + 10000)
  29                      - x[0];
  30                  for(i=1;i<k;i++){
  31                      t[i] = (t[i] + 10000)
  32                              x[i]
  33                          - (1 - t[i-1]/10000);
  34                      t[i-1] %= 10000;
  35                  }
  36                  t[k-1] %= 10000;
  37                  break;
  38              case 'e':
>>39                  for(i=0;i<k;i++) t[i] = x[i];
  40                  break;
  41              case 'q':
  42                  exit(0);
  43              default:
  44                  noprint = 1;
  45                  break;
  46              }
  47          if(!noprint){
  48              for(i=k - 1;t[i] <= 0 && i > 0;i--);
  49              printf("%d",t[i]);
  50              if(i > 0) {
```



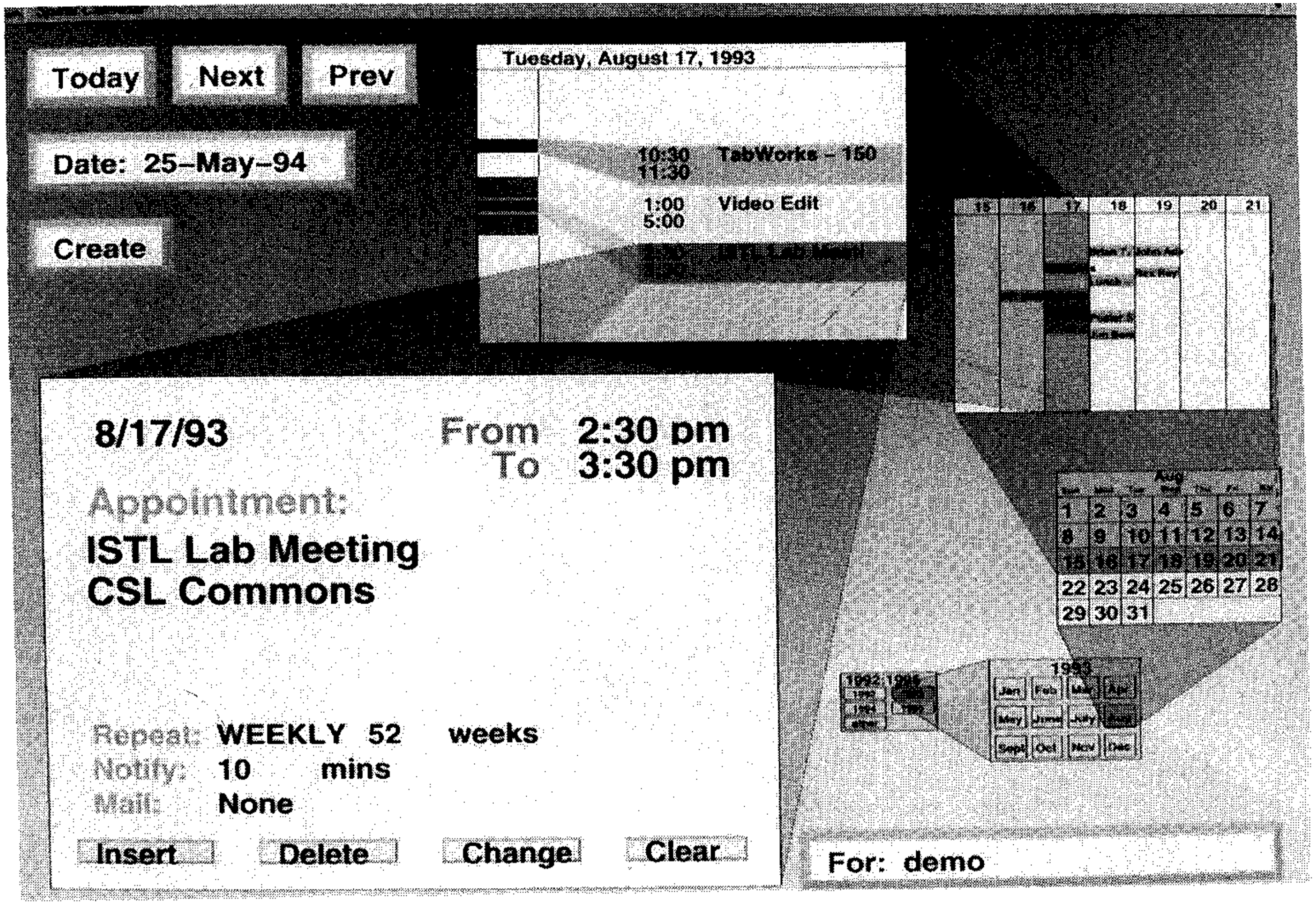
```
     1 #define DIG 40
     2 #include <stdio.h>
  ...4 main()
     5 {
     6         int c, i, x[DIG/4], t[DIG/4], k = DIG/4, noprint = 0;
  ...8         while((c=getchar()) != EOF){
     9             if(c >= '0' && c <= '9'){
 ...16             } else {
    17                 switch(c){
    18                 case '+':
 ...27                 case '-':
 ...38                 case 'e':
 >>39                     for(i=0;i<k;i++) t[i] = x[i];
    40                     break;
    41                 case 'q':
 ...43                 default:
 ...46                 }
    47                 if(!noprint){
 ...57                 }
    58             }
    59             noprint = 0;
    60         }
    61 }
```

Hyperbolic tree browser



Lamping et al. CHI 1995. https://doi.org/10.1145/223904.223956

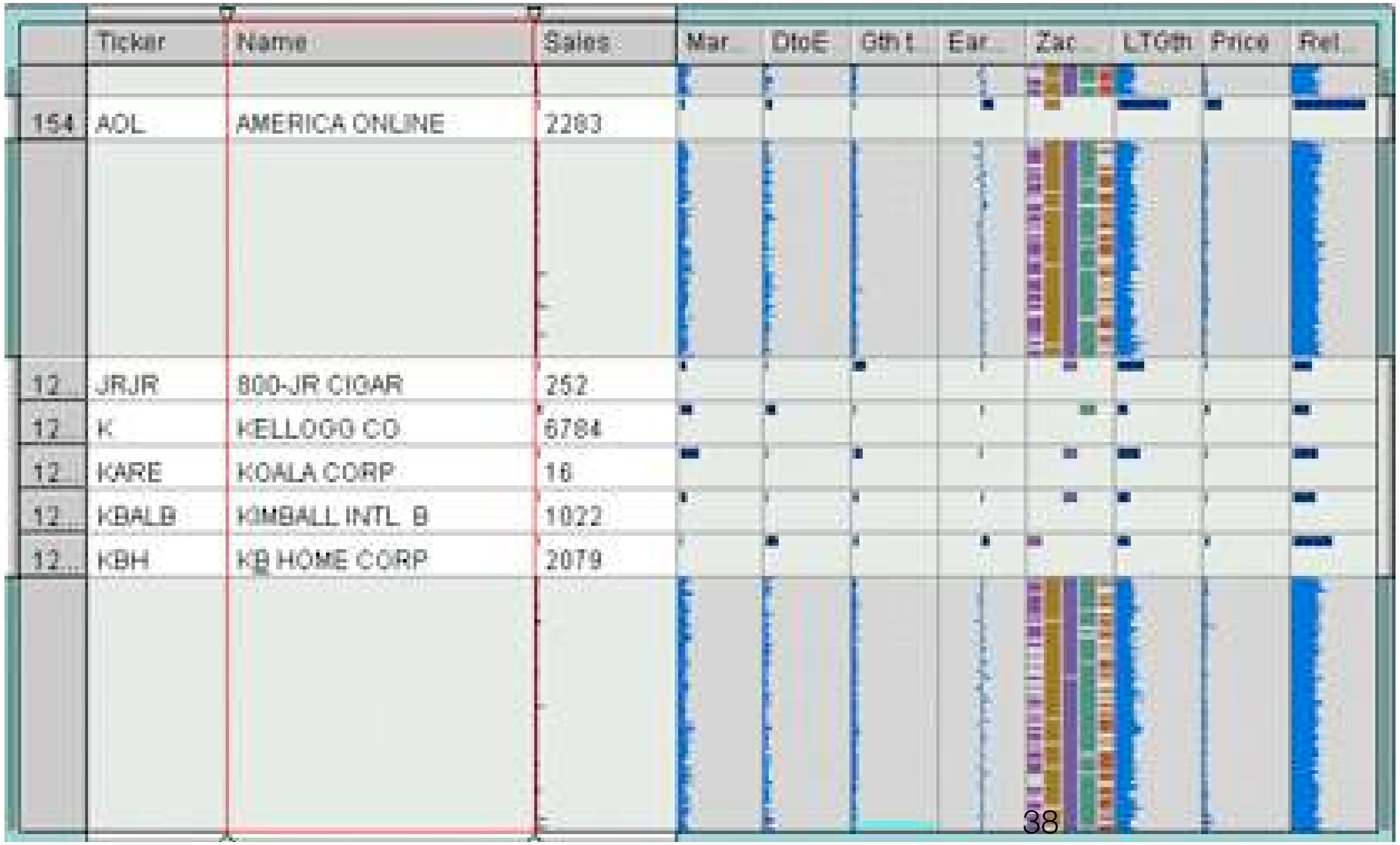
Spiral calendar



Mackinlay et al. 1995. https://doi.org/10.1145/192426.192470

Table lens (distortion technique)

- Table lens is a visualization tool for searching patterns and outliers in multivariate datasets (https://doi.org/10.1145/948449.948460)
- Time-cost function for different tasks (e.g., “find shape of the Nth column in the table lens”) can be calculated and verified experimentally (see the article)
- Demo at https://mitweb.itn.liu.se/geovis/eXplorer/world/



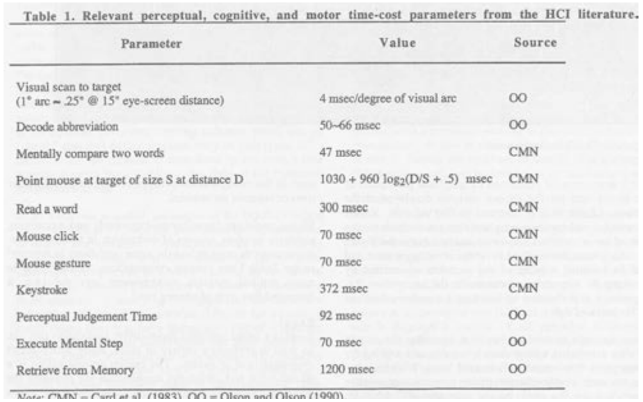


Table 1. Relevant perceptual, cognitive, and motor time-cost parameters from the HCI literature.

Parameter	Value	Source
Visual scan to target (1° arc = .25" @ 15" eye-screen distance)	4 msec/degree of visual arc	OO
Decode abbreviation	50–66 msec	OO
Mentally compare two words	47 msec	CMN
Point mouse at target of size S at distance D	1030 + 960 $\log_2$(D/S + .5) msec	CMN
Read a word	300 msec	CMN
Mouse click	70 msec	CMN
Mouse gesture	70 msec	CMN
Keystroke	372 msec	CMN
Perceptual Judgement Time	92 msec	OO
Execute Mental Step	70 msec	OO
Retrieve from Memory	1200 msec	OO

Note: CMN = Card et al. (1983), OO = Olson and Olson (1990)

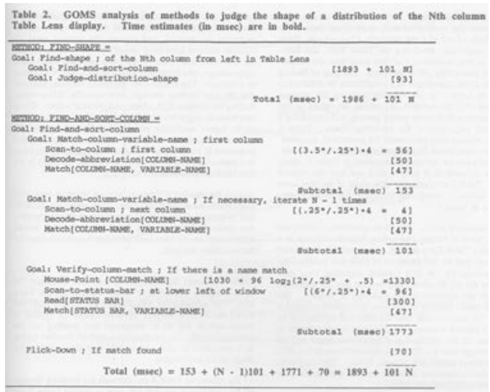
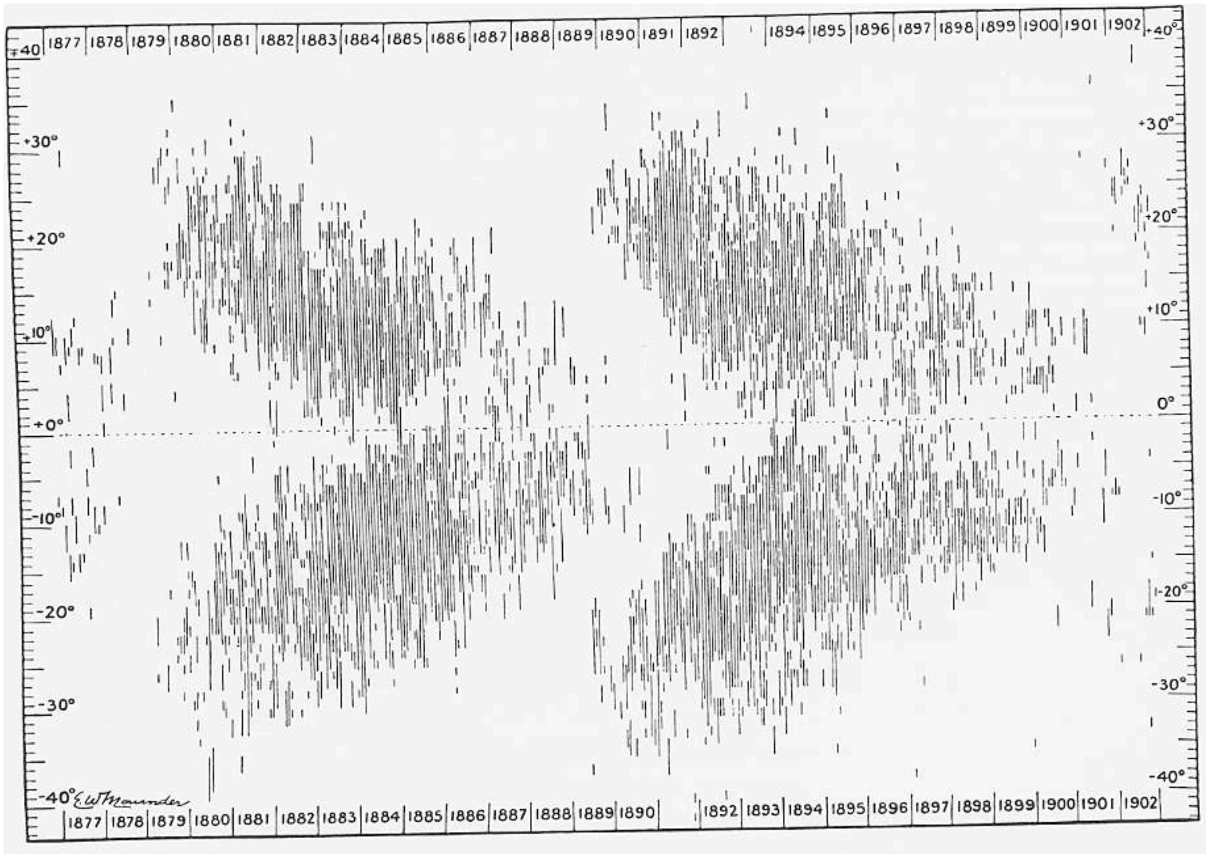


Table 2. GOMS analysis of methods to judge the shape of a distribution of the Nth column Table Lens display. Time estimates (in msec) are in bold.

```
METHOD: FIND-SHAPE =
Goal: Find-shape ; of the Nth column from left in Table Lens
   Goal: Find-and-sort-column                              [1893 + 101 N]
   Goal: Judge-distribution-shape                                    [93]
                                                 Total (msec) = 1986 + 101 N

METHOD: FIND-AND-SORT-COLUMN =
Goal: Find-and-sort-column
   Goal: Match-column-variable-name ; first column
      Scan-to-column ; first column                   [(3.5"/.25")*4 = 56]
      Decode-abbreviation[COLUMN-NAME]                               [50]
      Match[COLUMN-NAME, VARIABLE-NAME]                              [47]
                                                      Subtotal (msec) 153
   Goal: Match-column-variable-name ; If necessary, iterate N - 1 times
      Scan-to-column ; next column                    [(.25"/.25")*4 =  4]
      Decode-abbreviation[COLUMN-NAME]                               [50]
      Match[COLUMN-NAME, VARIABLE-NAME]                              [47]
                                                      Subtotal (msec) 101

   Goal: Verify-column-match ; If there is a name match
      Mouse-Point [COLUMN-NAME]     [1030 + 96 log2(2"/.25" + .5) =1330]
      Scan-to-status-bar ; at lower left of window     [(6"/.25")*4 = 96]
      Read[STATUS BAR]                                              [300]
      Match[STATUS BAR, VARIABLE-NAME]                               [47]
                                                     Subtotal (msec) 1773

   Flick-Down ; If match found                                       [70]

        Total (msec) = 153 + (N - 1)101 + 1771 + 70 = 1893 + 101 N
```

Micro-macro reading

- Focus+context in static visualisations



E. W. Maunder, 1904 [EI 22].