

# Information Visualization

**Chris North**

Center for Human-Computer Interaction,  
Department of Computer Science  
Virginia Polytechnic Institute and State University  
Blacksburg, VA 24061 USA

## Table of Contents:

1	Introduction
1.1	Insight
1.2	Design
2	Visualization Pipeline
2.1	Visual Mapping
2.2	Visual Properties
3	Information Structure
3.1	Tabular Structure
3.2	Spatial and Temporal Structure
3.3	Tree and Network Structure
3.3.1	Trees
3.3.2	Networks
3.4	Text and Document Collection Structure
3.5	Combining Multiple Structures
4	Overview Strategies
4.1	Reducing Data Quantity
4.2	Miniaturizing Visual Glyphs
5	Navigation Strategies
5.1	Zoom+Pan
5.2	Overview+Detail
5.3	Focus+Context
6	Interaction Strategies
6.1	Selecting
6.2	Linking
6.3	Filtering
6.4	Rearranging and Remapping
7	The Future

## 1 Introduction

The information revolution is changing the way many people live and think. Vast quantities and diverse types of information are being generated, stored, and disseminated, raising serious issues about how to make such information usable. The need to understand and extract knowledge from stored information is becoming a ubiquitous task. As examples, in everyday life people

must sort through a variety of personal information such as email communications, schedules, news, finances, and computer directories. Students can access countless digital libraries of educational materials. Online shoppers must make decisions among dozens of alternative products, models, vendors, and prices. New disciplines such as bioinformatics are leading the revolution in information-intensive science, using high-throughput data collection technologies such as microarrays and online data repositories. Government intelligence analysts must sift through massive collections of information gathered on a daily basis from sensor networks and other sources.

*Information visualization* has evolved as an approach to make large quantities of complex information intelligible. An information visualization is a visual user interface to information, with the goal providing users with information *insight* [Spence, 2001]. The basic method is to generate interactive visual representations of the information that exploit the perceptual capabilities of the human visual system and the interactive capabilities of the cognitive problem-solving loop [Ware, 2004].

The goal of this chapter is to highlight the critical high-level design issues in the information visualization design process. Lower level details of visual display and human perception can be found elsewhere in this book. Other aspects of the design process that apply to the design of user interfaces in general, such as evaluation methods, are also covered in other chapters. While other major references focus on the ‘what’ and ‘why’ of information visualization [Card et al., 1999; Chen, 1999; Shneiderman & Plaisant, 2005; Spence, 2001; Ware, 2004; Wickens & Hollands, 2000], here we emphasize the ‘how’.

## 1.1 Insight

Human vision contains millions of photoreceptors and is capable of rapid parallel processing and pattern recognition [Ware, 2004]. The impressive bandwidth of vision as a mode of communication leads to the efficient transfer of data from digital storage to human mind. Yet, a more important benefit is the human ability to visually reason about the data and extract higher level knowledge, or insight, beyond simple data transfer [Card et al., 1999]. This enables users to infer mental models of the real phenomena represented by the data.

For example, Figure 1 demonstrates the mapping of a database of census demographics. From the visual representation, one can readily recognize the approximate proportional relationship between education and income, various outliers such as New York, NY, and predominance of large population counties in high income and education. These insights are not explicitly stored within the dataset, but inferred through visual pattern recognition. These insights are not so readily identifiable from the textual representation. Clearly, the design of the visual representation is important. A poorly designed visualization can hide insight, or even mislead with incorrect insight.

<<Figure 1 here>>

Figure 1 a & b:

	A	B	C	D	E	F	G	H
1	Name	State	Population	% College Grad	Income per Capita	Median Rent	Employees	Retail Sales
2	Aleutians East	AK	2,305	12.9	20,114	518	1,410	6,693
3	Aleutians West	AK	5,259	14.8	20,298	428	4,321	24,233
4	Anchorage	AK	251,335	26.9	26,619	528	93,037	2,612,640
5	Bethel	AK	15,525	13.1	15,493	460	2,269	59,687
6	Bristol Bay	AK	1,023	18.9	29,699	464	392	9,653
7	Dillingham	AK	4,360	15.3	22,616	518	1,060	23,077
8	Fairbanks North Star	AK	83,374	25.2	19,149	471	17,746	702,135
9	Haines	AK	2,181	17.6	26,413	405	371	15,154
10	Juneau	AK	29,378	30.7	26,066	587	8,369	289,620
11	Kenai Peninsula	AK	46,151	17.9	22,761	410	8,934	329,438
12	Ketchikan Gateway	AK	14,422	20.2	28,789	533	5,583	152,834
13	Kodiak Island	AK	14,087	21.5	20,889	609	4,224	84,179

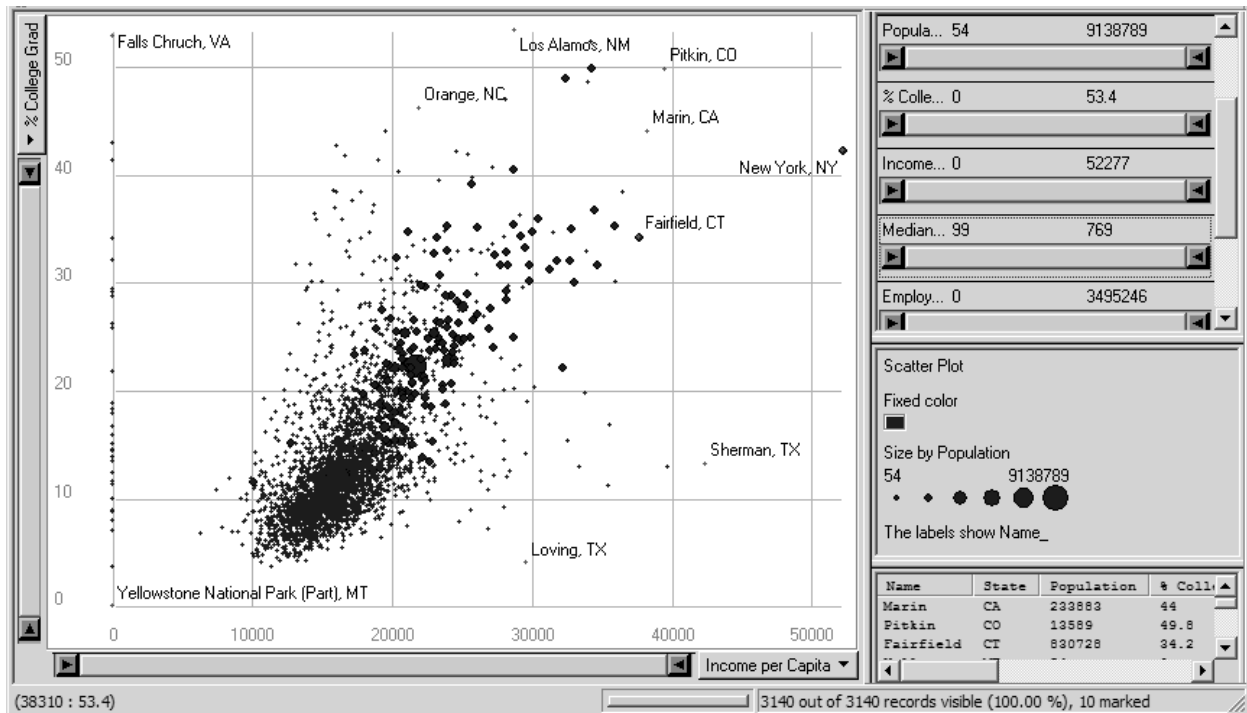


Figure 1: Census demographics dataset of 3140 U.S. counties shown in (a) spreadsheet form, and (b) scatterplot form using Spotfire® [Ahlberg & Wistrand, 1995]. The plot shows counties by ‘income per capita’ vs. ‘percentage of adult population that has college degree’, with dots sized by ‘population’, and labels for some outliers. The plot is interactive and reveals details of a county when selecting its dot, in the text sub-window at lower right. Dots can be filtered by other county attributes, such as ‘median rent’, using the dynamic query slider widgets on the right. (Courtesy Spotfire)

Visualization can enable a broad range of information insight, and several such insights are listed below (see also [Shneiderman & Plaisant, 2005; Wehrend & Lewis, 1990; Wickens & Hollands, 2000; Zhou & Feiner, 1998]). The first two are simplistic and can be readily supported by textual or query-based user interfaces such as spreadsheets or search forms, because they are precise and have solutions consisting of a single data entity. However, the later are more complex and are well supported by visualization. These involve open-ended questions with complex answers that require seeing the whole. A strength of visualization is the capacity for

discovery, the recognition of new insights unexpected by the users and potentially unforeseen by the visualization designers.

**Simple insights:**

- Summaries: minimum, maximum, average, percentages
- Find: known item search.

**Complex insights:**

- Patterns: distributions, trends, frequencies, structures
- Outliers: exceptions
- Relationships: correlations, multi-way interactions
- Tradeoffs: balance, combined minimum/maximum
- Comparisons: choices (1:1), context (1:M), sets (M:N)
- Clusters: groups, similarities
- Paths: distance, multiple connections, decompositions
- Anomalies: data errors.

## 1.2 Design

Like any user interface, effective information visualizations are difficult to design. Fundamentally, information visualizations make *abstract* information perceptible. Abstract information has no inherent perceptual form, as in the case of databases or computer directories. Hence, there are no natural constraints on the types of visual representations that creativity can produce for abstract information, and the possibilities are limitless. As a result, there is significant challenge, excitement, and opportunity in both creating novel visual representations as well as identifying the most effective representations amongst the endless possibilities. In contrast, *scientific visualization* [Rosenblum et al., 1994] typically emphasizes the visualization of data that represent physical 3D phenomena, which offer some natural constraints on visual representations and focuses the challenges on realism.

The two most challenging characteristics of information that make designing effective information visualizations difficult are:

- (1) **complexity**: supporting diverse abstract information that may have multiple interrelated data types and structures.
- (2) **scalability**: supporting very large quantities of information.

Because of these characteristics, visual representations alone are not sufficient and interactive techniques must also be designed. While the principles of static graphs and illustrations are fundamental to visualization design [Cleveland, 1993; Tufte, 2001; Wilkinson, 1999; Gillan et al., 1998], new human-computer interaction issues related to these two challenging characteristics become forefront in information visualization design.

As an overview, the visualization design process involves iterative requirements analysis, design, and evaluation (e.g. [Rosson & Carroll, 2001]). In the requirements analysis phase, it is important to identify the two primary inputs to design: the characteristics of the information to be visualized, and the desired types of insights the visualization should enable. Characteristics of the information include the data schema, underlying structures, and quantity. Since the number of data attributes and desired insights may be large, identifying a prioritization of attributes and insights will be helpful in balancing design tradeoffs. Other elements of requirements analysis

include broader user tasks, users' domain knowledge, data semantics, and computer system requirements. In the design phase, major design decisions (presented in this chapter) include the visual mapping of the information, the representation of information structures, visual overview strategies, navigation strategies, and interaction techniques.

The evaluation phase must be continually considered during the design process [Plaisant, 2004]. A claims analysis identifies the positive and negative impacts of a visualization design's features on its insight capability, and seeks to overcome or balance these tradeoffs through iterative design [Rosson & Carroll, 2001]). Begin with analytic evaluations to determine if designs meet requirements, such as scalability to data quantity and appropriateness for producing desired insights. In later iterations, empirical evaluations involving users should be undertaken, such as the wizard of oz technique, usability testing, or controlled experiments [Chen & Yu, 2000; Tory & Möller, 2004a]. Desired insights identified in requirements analysis should be implemented as benchmark user tasks in the empirical evaluations. Alternatively, since benchmark tasks often overly constrain the testing to simplistic insights that discount the discovery aspect of visualization, the insight-based methodology [Saraiya et al., 2004] attempts to measure the insight generated by visualizations by using an open-ended experimental protocol without benchmark tasks. The following sections highlight the major design decisions in the information visualization design process.

## 2 Visualization Pipeline

The *visualization pipeline* is the computational process of converting information into a visual form that users can interact with [Card et al., 1999] (Figure 2). The first step is to transform raw information into a well-organized canonical data format. The resulting format typically consists of a dataset containing a set of data entities each of which has associated data attribute values. Various data processing steps can be used to manipulate the data as needed. Derived data, such as data mining or clustering results, can be very useful for assisting in insight generation [Fayyad et al., 2001]. The second step, the heart of the visualization process, is to map the dataset into visual form. The visual form contains visual glyphs that correspond to the dataset entities. The third step embeds this visual form into views, which display the visual form on screen and provide various view transformations such as navigation. The view is then presented to the user through the human visual system. Users interpret the view to (partially) reconstruct the underlying information. Finally, users can interact with any of the steps in the pipeline to alter the resulting visualization, and make further interpretations. This entire pipeline comprises an information visualization.

<<Figure 2 here>>

Figure 2:

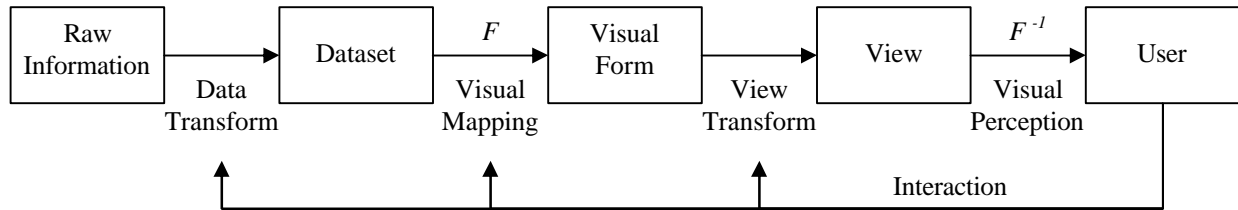


Figure 2: The visualization pipeline, converting information into interactive visual representations (adapted from [Card et al., 1999]).

## 2.1 Visual Mapping

The visual mapping at the second step is the heart of visualization, and must be designed carefully. The goal is to communicate information from computer to human. The medium of communication is a visual representation of the information. The dataset is computationally mapped into visual form by some function  $F$ , which takes the dataset as input and generates the visual representation as output. Then, when the visual representation is communicated to the users, they must cognitively reverse the visual mapping by inverting function  $F$  to decode the information from the visual representation. It is yet unclear how, when, and to what degree  $F^{-1}$  is cognitively applied in the perceptual process, and a variety of models exist [Ware, 2004]. While some cognitive reasoning operates on the visual representation itself, eventually meaning must be decoded. Nonetheless, this visual communication process implies four important characteristics of the visual mapping function  $F$ :

- **Computable:**  $F$  is a mathematical function that can be computed by some algorithm. Although there is significant room for creativity in the design of these functions, the execution of the functions must be algorithmic.
- **Invertible:** It must be possible to use  $F^{-1}$ , the inverse of mapping function  $F$ , to reconstruct the data from the visual representation to a desired degree of accuracy. If this is not possible, the visualization will be ambiguous, misleading, or not interpretable.
- **Communicable:**  $F$  (or preferably  $F^{-1}$ ) must be known by the user to decode the visual representation. It must be communicated with the visualization, or already known by the user through prior experience. In usability terms, this is a learnability issue.
- **Cognizable:**  $F^{-1}$  should minimize cognitive load for decoding the visual representation. This is a human perception and performance issue.

The visual mapping step is accomplished by two sub-steps (Figure 3) [Card et al., 1999]. First, each data entity is mapped into a visual *glyph*. The vocabulary of possible glyphs consists primarily of points (dots, simple shapes), lines (segments, curves, paths), regions (polygons, areas, volumes), and icons (symbols, pictures). Second, attribute values of each data entity are mapped onto *visual properties* of the entity's glyph. Common visual properties of glyphs include spatial position ( $x, y, z, \theta, \phi$ ), size (length, area, volume), color (gray scale, hue, intensity), orientation (angle, slope, unit vector), and shape. Other visual properties include texture, motion, blink, density, and transparency. For example, in Figure 1, the U.S. counties are mapped to circular points. Income and education levels of each county are mapped respectively

to the horizontal and vertical position of the point, and population value is mapped to the size of the point.

<<Figure 3 here>>  
Figure 3:


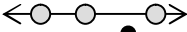







<u>Visual Glyphs</u>		<u>Visual Properties of Glyphs</u>	
Points		Position	
Lines		Length	
Regions		Color	
Icons		Orientation	
		Shape	

Figure 3: A vocabulary of glyphs and some visual properties of glyphs (adapted from [Card et al., 1999]).

2.2 Visual Properties

In general, data attributes should be prioritized according to the problem requirements and desired insights. The prioritization can then be applied to map the higher priority data attributes to the most effective visual properties. Spatial position properties are the most effective, and should be reserved to layout the dataset in the visual representation according most important data attributes.

The remaining visual properties, called *retinal properties* [Bertin, 1983], can be used next. The effectiveness of these properties are determined by many interdependent factors including preattentive processing [Healey et al., 1996], perceptual independence (separability) [Ware, 2004], data type (quantitative, ordinal, categorical) [Card et al., 1999], polarity (greater than, less than) [Ware, 2004], task [Carswell, 1992; Wickens & Hollands, 2000], and attention [Chewar et al., 2002]. Commonly accepted orderings of the effectiveness of these attributes are based on empirical evidence (e.g. [Cleveland & McGill, 1984; Nowell et al., 2002]), as well as experience [Bertin, 1983; Mackinlay, 1986]. The order shown in Figure 3 is intended for quantitative data. For categorical data, color and shape become more predominant. Some visualization design systems attempt to use such rules to automatically generate effective mappings (e.g. Apt [Mackinlay, 1986], Sage [Roth et al., 1994]).

Finally, for any remaining attributes, interaction techniques can be applied. In general, the direct visual mapping of information is the most effective for rapid insight, while interaction techniques require slower physical actions by the user to reveal insights. When mapping additional data attributes would overly clutter the visual representation and reduce comprehension of more

important attributes, interaction techniques can be used instead. Interaction techniques enable users to alter the visual mapping function or other stages of the visualization pipeline, based on additional attributes. By viewing the resulting changes in the visual representation, users can infer additional information about those attributes. For example, the *dynamic queries* technique provides interactive query widgets for other attributes, and can be used to dynamically filter the entity glyphs [Ahlberg & Wistrand, 1995] (Figure 1).

### 3 Information Structure

The visual mapping process provides an initial starting point for visualization design, but more advanced methods are needed as data complexity increases. Identifying underlying structures within the target information helps to further guide the design process. These structures provide high-level organization to a dataset, and often provide guidance for the design of appropriate visualizations. Since these structures are likely to be very important to users' mental models of the information, they are typically mapped to the spatial position attributes and form the primary layout of the visualization. In general, there are four common types of information structures (adapted from [Shneiderman, 1996; Card et al., 1999; Spence, 2001]), discussed in the following subsections. These are not strict or mutually exclusive classifications, but useful guidelines.

#### 3.1 Tabular Structure

Tables consist of rows (entities) and columns (attributes). This is often referred to as multi-dimensional or multivariate data, because each attribute defines a dimension of the data space within which each entity identifies a single point. Examples include databases and spreadsheet tables, such as the census data in Figure 1. Visualizations of tables that contain a small number of attributes can be designed relatively easily using the visual mapping process. However, such visualizations lack scalability to many attributes due to the limited number of non-conflicting visual properties to choose from. To address this problem, a variety of creative methods have been developed for tables of many attributes. Primarily, these involve the use of more complex glyphs and spatial layouts.

TableLens [Rao & Card, 1994] (Figure 4a) preserves the tabular spreadsheet visual representation, but converts cells to horizontal bar glyphs with cell values mapped to bar length. This exploits the length property, which is excellent for encoding quantitative data. Also, since the bars are very thin, many values can be packed onto the screen, providing an excellent overview of a large dataset. TableLens encodes each data entity (row) with multiple glyphs (bars), one glyph for each of the entity's attribute values (columns). Interactively selecting a set of rows will expand them to reveal the detailed data values in textual form. Users can vertically sort the table by any attribute. By spatially arranging the data according to one attribute, distributions and relationships to other attributes can be seen. However, it is perceptually difficult to relate two non-sorted attributes. Hence, users must interactively sort each attribute to explore all potential relationships. The *proximity compatibility principle* [Wickens & Hollands, 2000] predicts that representations that use a single glyph per data entity, such as in scatterplots (Figure 1), would be better than TableLens for recognizing relationships between attributes. But as previously indicated, such representations are more limited in scalability. Herein is the tradeoff: TableLens provides an overview of many attributes with reasonable capability for

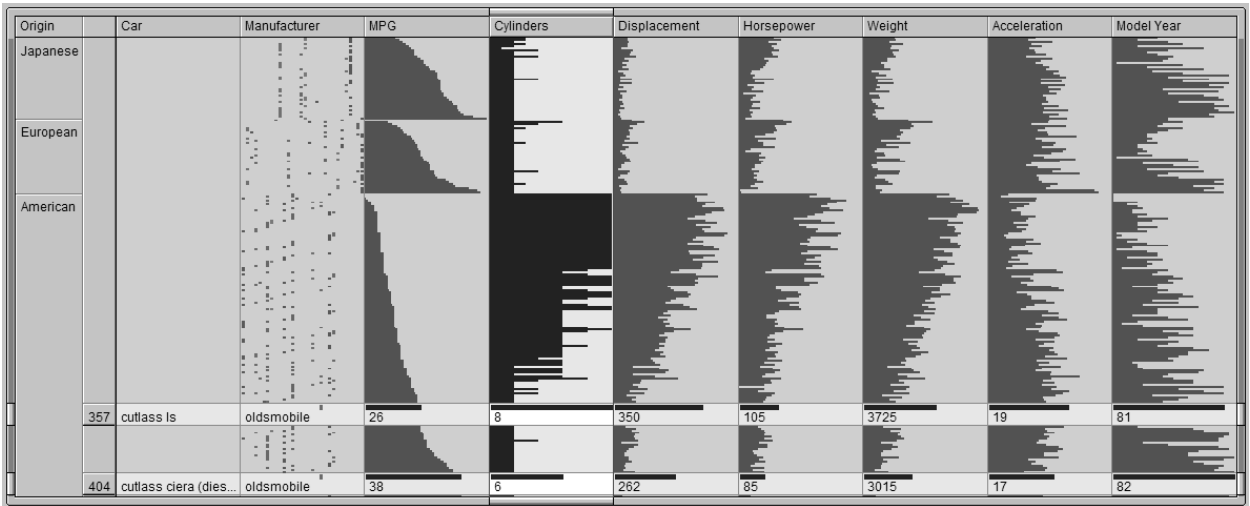


relationship insights. Whereas, scatterplots provide excellent insight on relationships, but only for the 2 attributes mapped to x and y (and potentially a small number of other attributes using color, size, etc.).

To analyze the scalability of TableLens, consider an approximate screen resolution of 1000x1000 pixels. If each bar glyph is only 1 pixel thick, 1000 data entities (rows) can be shown. Columns will need to be approximately 50-100 pixels wide to enable reasonable visual discretion of quantitative data such as percentages, leading to 10-20 attributes visible. Hence, this approach can display a tabular dataset containing 1000 data entities and 20 attributes. Much larger datasets can be explored in TableLens by using its aggregation and interactive navigation (e.g. scrolling) strategies, but only 1000x20 values can be simultaneously perceived at a time. For datasets larger than the screen size, TableLens aggregates adjacent rows by showing averages or minimum and maximum values.

The Cartesian coordinate system uses orthogonal axes to visually map 2 or 3 attributes of a tabular dataset to space. However, orthogonal axes fundamentally limit scalability of attributes. As an alternative, Parallel Coordinates [Inselberg, 1997] (Figure 4b) displays attribute axes as parallel vertical lines. Each data entity is mapped to a polyline that connects the entity's attribute values on each attribute axis. Hence, attributes are mapped to the vertical position of the respective vertices of the polyline. Users can recognize clusters of similarly entities and relationships between adjacent attributes. Patterns of crossing lines between adjacent axes indicate an inverse relationship between those two attributes, while non-crossing lines indicate a proportional relationship. To combat occlusion and clutter, interactively selecting entities highlights their polylines across all axes. A scalability analysis of Parallel Coordinates would give a result similar to that of TableLens. Other arrangements of axes include radial [Kandogan, 2000], or circumferential [Miller, 2004].

<<Figure 4 a & b here>>  
Figure 4 a & b:



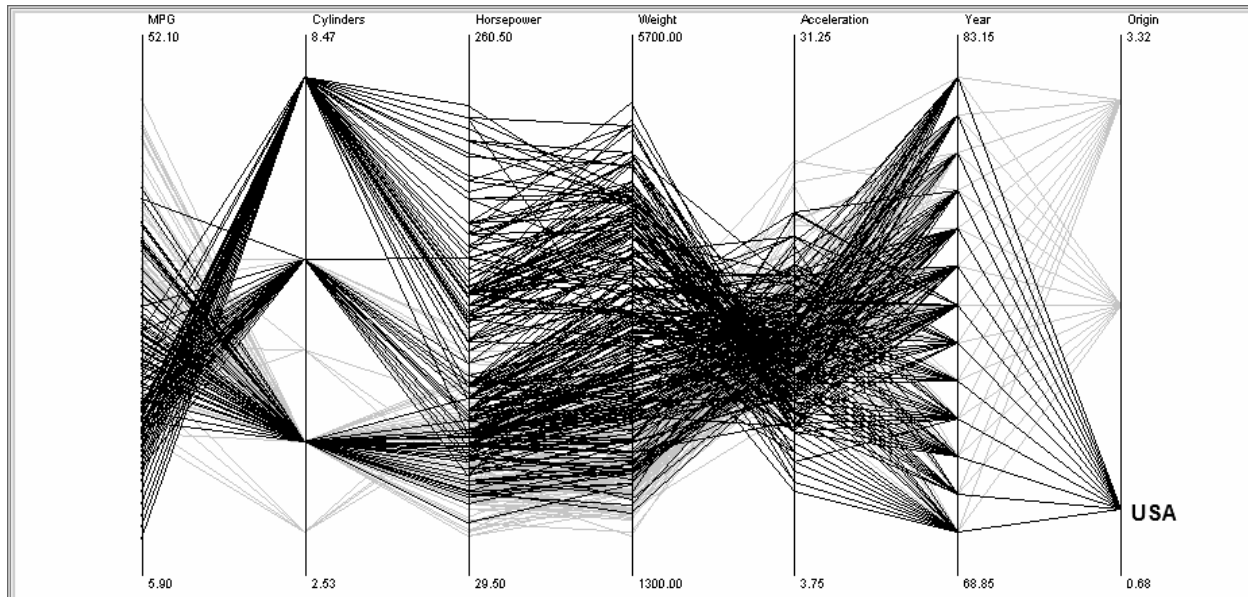


Figure 4: (a) InXight TableLens [Rao & Card, 1994] (Courtesy InXight), and (b) XmdvTool's [Ward, 1994] Parallel Coordinates [Inselberg, 1997] (Courtesy Matthew Ward). Both show the same dataset about automobiles. Clearly, USA dominates the market in large, powerful, gas-guzzling cars. Two outliers, cars with large engines yet high MPG, are highlighted in TableLens revealing their detailed values. TableLens sorts the data by one attribute, while Parallel Coordinates sorts all the attributes simultaneously.

Use of more complex iconic glyphs includes star plots [Chambers et al., 1983], which map data attributes to the length of the radial needles emanating from a star icon, and Chernoff faces [Chernoff, 1973], which attempts to exploit the human ability to rapidly recognize facial features and expressions. While these iconic methods do not scale up very well, they are useful for combining with other information structures (such as networks) because they leave the spatial position properties available for other uses [Ward, 2002].

The reverse approach is to simplify the glyphs and visual representations by splitting the attributes up into separate views. For example, four attributes can be displayed using the x and y axes of two scatterplots. An interactive technique called *brushing-and-linking* relates the two plots [Becker & Cleveland, 1987]. When users select glyphs in one plot (*brushing*), the corresponding glyphs for the same underlying data entities are highlighted in the other plot (*linking*). Scatterplot matrices take this to the extreme, showing plots for all combinations of attributes pairs [Cleveland, 1993]. This brushing-and-linking technique can also be used to view a single dataset in several different visual representations at the same time, such as plots and geographic maps, and to relate the different contexts [Roth et al., 1996; North et al., 2002] (see Section 6.2).

### 3.2 Spatial and Temporal Structure

This structure has a strong one-, two-, or three-dimensional component in which navigation is likely to be required. 1D examples include timelines, music, video streams, lists, linear documents, and slide shows. 2D examples are road maps, satellite images, photographs, and blueprints. 3D examples are MRI and CT medical scans, CAD/CAM architectural plans, and virtual environments. Continuous functions, including those with domains greater than 3 dimensions, also fall in this category [Tory & Möller, 2004b]. These spatial and temporal structures are the most natural for mapping onto spatial displays.

For example, the Music Animation Machine [Malinowski, 2004] (Figure 5a) provides a simple timeline representation of music that scrolls as the music plays. Notes are represented as horizontal bars, with vertical position representing pitch, horizontal position representing timing, length indicating duration, and color indicating other attributes such as instrument, timbre, or hand (in the case of piano). Similarly, LifeLines [Plaisant et al., 1996] represents events in a person's medical history, but in a more compact form with zooming for navigation. For timelines that contain periodic cycles, such as calendars, visual spirals can be used to proximate the cycles while maintaining a continuous line [Carlis & Konstan, 1998]. Streaming video can be viewed as a 3D video cube (2D frames + 1D time) [Elliott & Davenport, 1994].

In 3D data spaces, the main challenge is viewing the interior of the 3D structure beyond the exterior surface when occlusion is problematic. Architectural walk-through applications (polygonal data) typically use first-person perspective projection, with 6 degree-of-freedom navigation for a life-like experience [Stoakley et al., 1995]. For medical imagery (volumetric data), strategies include slicing and transparency. For example, the Visible Human Explorer presents 2D slices that can be animated through the 3D body [North et al., 1996]. In 3D volume rendering, transparency can give users x-ray vision into the space by adjusting the opacity of different materials within the space through interactive control of the visual transfer function [Kniss et al., 2001].

Hyper-dimensional continuous spaces must somehow be reduced to 3 or fewer dimensions for display. Worlds within Worlds [Beshers & Feiner, 1993] (Figure 5b) displays subspaces of hyper-dimensional functions by nesting a 3D coordinate frame within another 3D coordinate frame. The location of the origin (0,0,0) of the inner frame within the outer frame determines the values of the outer frames dimensions used to generate the subspace for the inner frame. By interactively sliding the inner frame around the inside of the outer frame, the full space can be explored. Repeated nesting can enable greater numbers of dimensions. Other methods include hierarchical axes [Mihalisin et al., 1991], and slicing [van Wijk et al., 1993].

<<Figure 5: (a) (b) here >>

Figure 5 a & b:

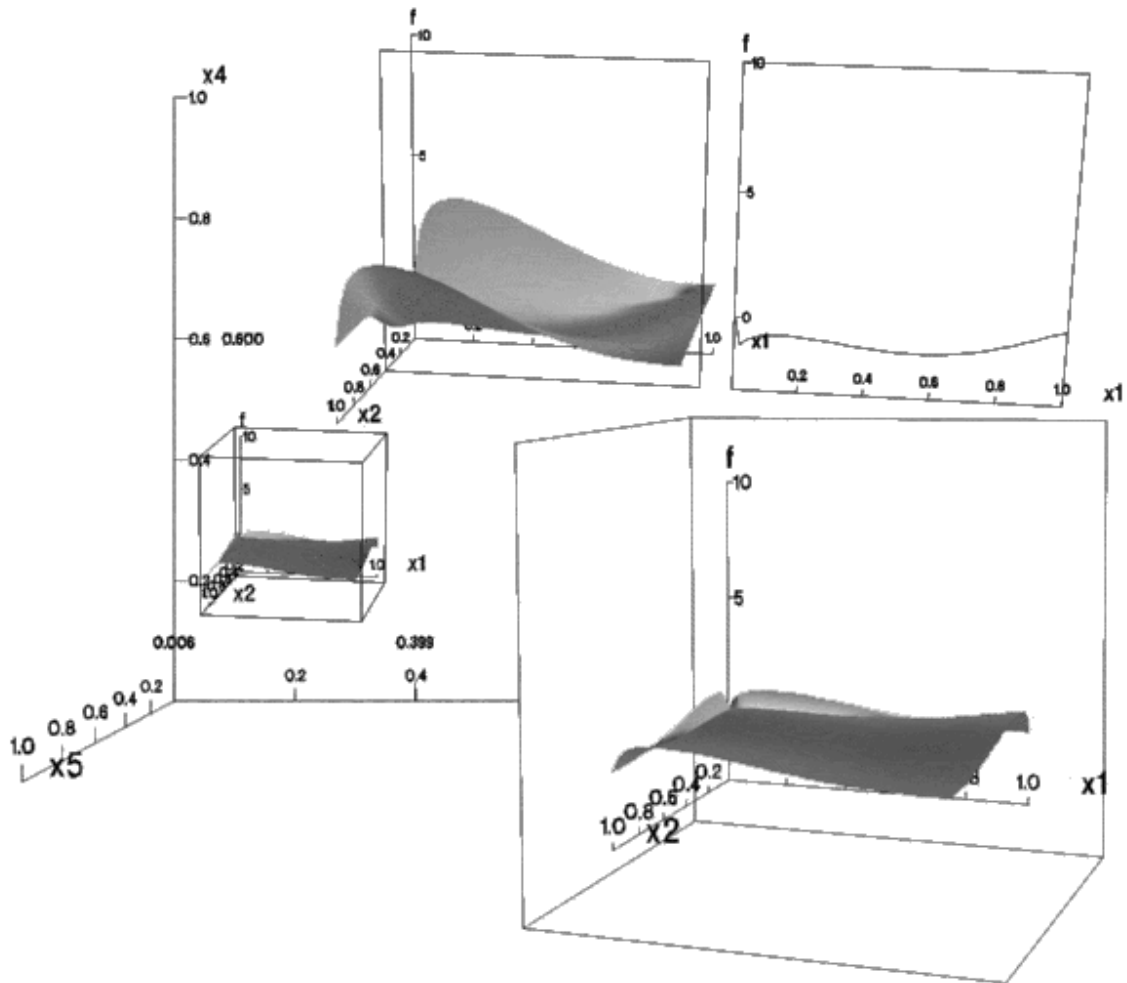


Figure 5: (a) Music Animation Machine [Malinowski, 2004] showing a portion of Bach's Brandenburg Concerto #4, 3rd movement (Courtesy Stephen Malinowski).  
 (b) Worlds within Worlds [Beshers & Feiner, 1993] nests inner coordinate frames within outer frames to decompose hyper-dimensional spaces (© 1993 IEEE).

### 3.3 Tree and Network Structure

This structure contains specific *connections* between individual entities. In graph theory terms, a network consists of a set of vertices (entities) connected by a set of edges (connections), which can be either directed or undirected. Like data entities, connections can also contain attributes. Examples include communications networks, literature citations, or web hyperlinks between pages. Tree structures are a special subset of networks, and are distinct and common enough in digital information to warrant separate treatment. Trees have a hierarchical structure that connects data entities by parent-child connections. To be a tree, each child entity should have only one parent. Examples include computer file directories, menu systems, organization charts, and taxonomies such as the Dewey decimal system. Other useful variants of tree structures exist, such as multi-trees [Furnas & Zacks, 1994] and polyarchies [Robertson et al., 2002]. New types of insights involve understanding the connection structure, such as the breadth or depth of the tree. The primary challenge for visualization is the spatial layout of the network or tree to reveal the structure of the connections. The secondary challenge is to also visualize data attributes of the entities and connections.

#### 3.3.1 Trees

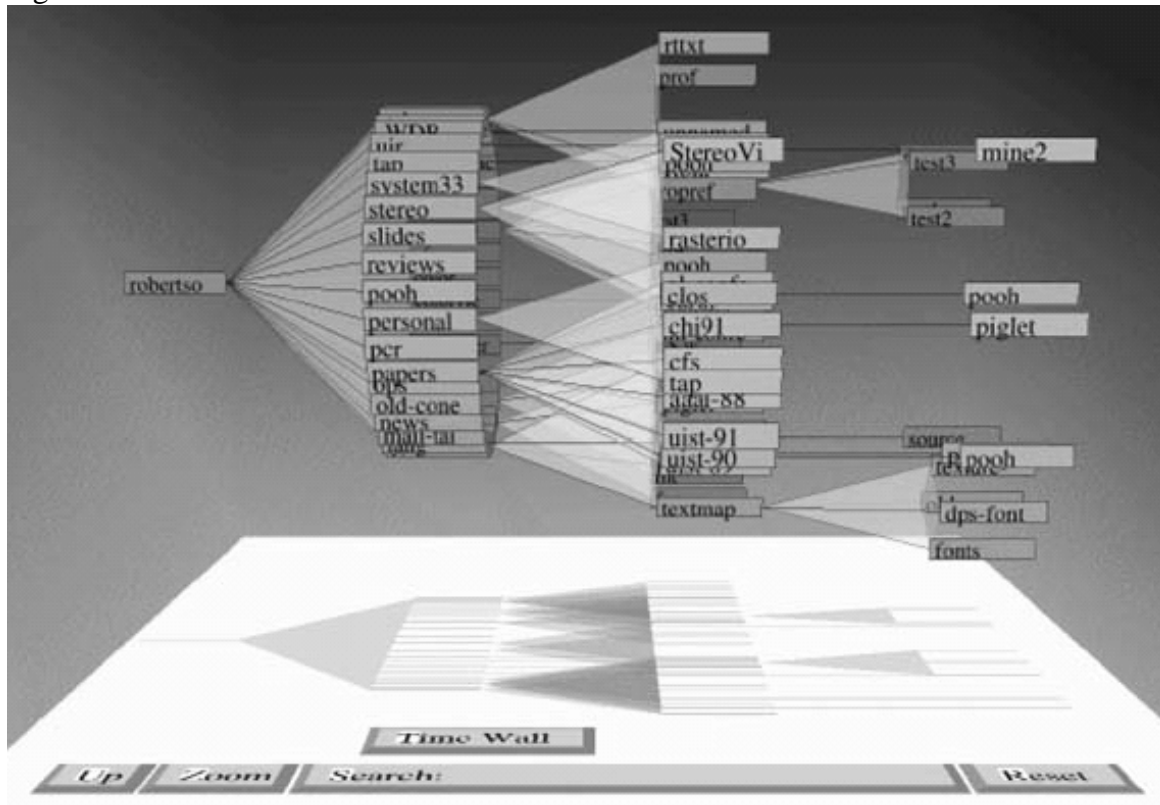
For tree structures, two primary approaches for visually representing parent-child connections exist: *link*, and *containment*. The *link* approach uses node-link diagrams. Entities are mapped to visual nodes, and connections are mapped to visual links between the nodes. Alternative spatial layouts of node-link diagrams include nested-indented (as in Windows Explorer or Mac Finder), top-down or left-right as in SpaceTree [Grosjean et al., 2002], radial as in Hyperbolic Tree [Lamping et al., 1995], or 3D ConeTrees that combine radial and left-right [Robertson et al., 1993] (Figure 6a). These systems emphasize the display of a single data attribute as a text label on the nodes. Node-link diagrams tend to be space consuming, due to the amount of white-space needed within each of these spatial layouts, making it difficult to get beyond 100 or 1000 nodes visible. Since large tree structures cannot be completely displayed on screen, each of these layouts requires interactive navigation. The focus+context technique (section 5) is a natural match for tree navigation, enabling users to drill-down within an individual branch of focus in the tree while maintaining context of the path to the root and siblings. Hyperbolic Trees shrink the size of nodes near the periphery to pack more nodes on the display. 3D approaches such as ConeTrees exploit the 3<sup>rd</sup> dimension for additional space to layout the tree, but due to occlusion it is unclear whether that extra space is useful. The most important factor in 3D designs is the interactive navigation [Wiss & Carr, 1999]. Simple 6 degree-of-freedom camera movement through a static 3D scene is clearly not effective in these structures. ConeTrees employs a much more efficient interaction technique, cascading rotation of the 3D cones to bring the desired child nodes to the front.

The *containment* approach for tree layout is exemplified by Treemaps [Johnson & Shneiderman, 1991] (Figure 6b). Child nodes, represented as rectangles, are visually contained within their parent nodes as in Venn diagrams. Treemaps are space-filling to maximize the use of every available pixel, and easily scale to 10,000 entities. Data attributes are mapped to retinal properties of the node rectangles such as size and color. Hence, Treemaps emphasize the visualization of non-textual attributes. In dense Treemaps, there is not enough space left for textual node labels. Nodes can be arranged within their parent node according to a variety of

algorithms. The original Treemap used a slice-and-dice algorithm. It was simple, but tended to generate rectangles with many different aspect ratios, some square and some long narrow, that are difficult to visually compare. Newer algorithms generate squarified treemaps [Bederson et al., 2002]. SunBurst [Stasko et al., 2000] offers a radial version of the containment approach based on the stacked pie chart. In comparison to Treemaps, SunBurst can improve learnability for novices, but reduces scalability because the number of leaf nodes is limited by 1 dimensional circumferential space rather than the full 2 dimensional area available to Treemaps.

<< Figure 6: (a) and (b) here >>

Figure 6 a & b:



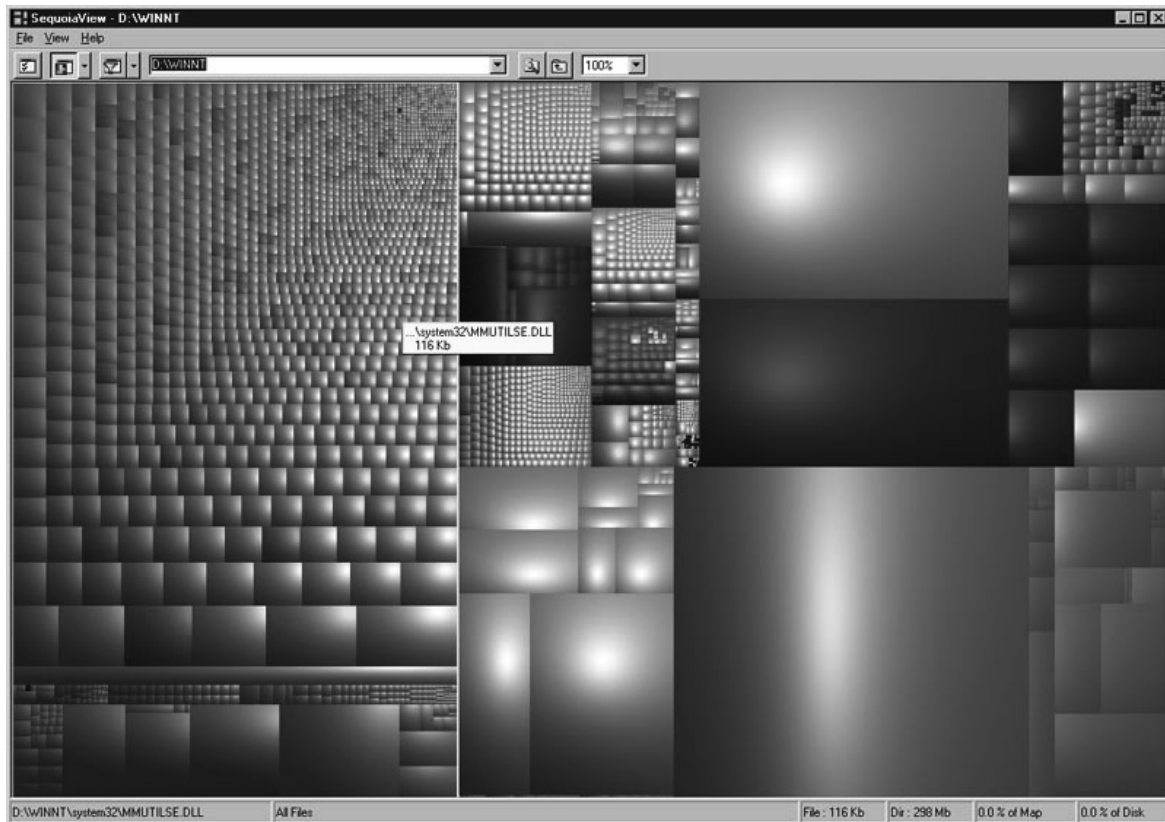


Figure 6: (a) ConeTrees [Robertson et al., 1993] (© 1993 ACM, Inc. Included here by permission), and (b) SequoiaView [van Wijk et al., 1999] (Courtesy Jarke van Wijk) Treemap showing directory structures on a computer system. ConeTrees emphasizes the tree structure and node labels, while Treemap emphasizes node attributes such as file size and type.

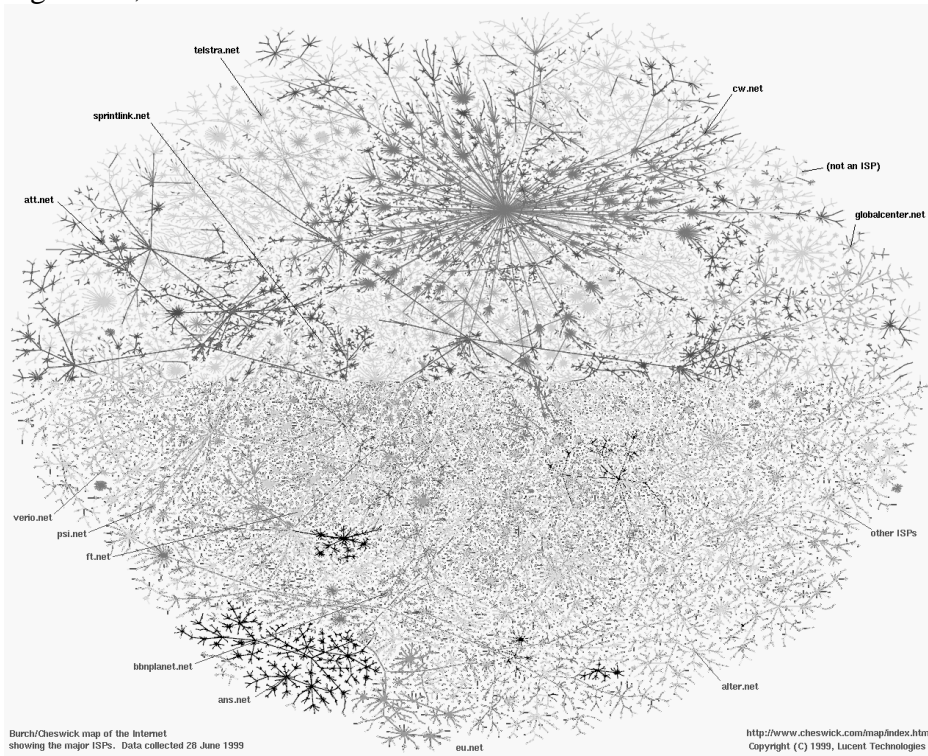
### 3.3.2 Networks

For visualizing networks, the node-link approach is dominant. Many algorithms have been devised to spatially layout network diagrams [Herman et al., 2000], and are increasingly tuned to specific types of networks (Figure 7a). Designs must consider network features such as number of nodes and links, directedness of links, node degree, any common patterns within the network structure, and attributes of nodes and links that should be visible. Links can be drawn as straight lines, arcs, or orthogonal polylines. Algorithms can apply aesthetic constraints such as minimizing link crossings, minimizing link lengths, maximizing symmetries, etc. [Ware et al., 2002]. In general, the goal is to layout the network to reveal hidden network patterns, and avoid the ‘bowl of spaghetti’ phenomenon. Some common layout algorithms include circular, layering, concentric layering, force directed, and clustering. SeeNet [Becker et al., 1995] arranges communications nodes according to geographical position, and raises links off the surface as 3D arcs. Arc properties such as color and line thickness are used to represent communications type and bandwidth. H3 [Munzner, 1998] arranges nodes inside a 3D sphere that can be rotated and navigated similar to Hyperbolic Trees. Navigation can be used to reduce network complexity by representing the network from the perspective of one node in focus [Andrews et al., 1995].

A different approach is to visualize the network as an adjacency matrix [Hetzler et al., 1998] (Figure 7b). An adjacency matrix emphasizes the connections instead of the nodes, mapping each potential connection to a cell in the matrix. Finally, it is also possible to reduce networks to trees using minimum spanning trees or hierarchical aggregation [Feiner, 1988], thereby enabling the use of tree visualization methods.

<<Figure 7: (a) (b) here >>

Figure 7 a,b:





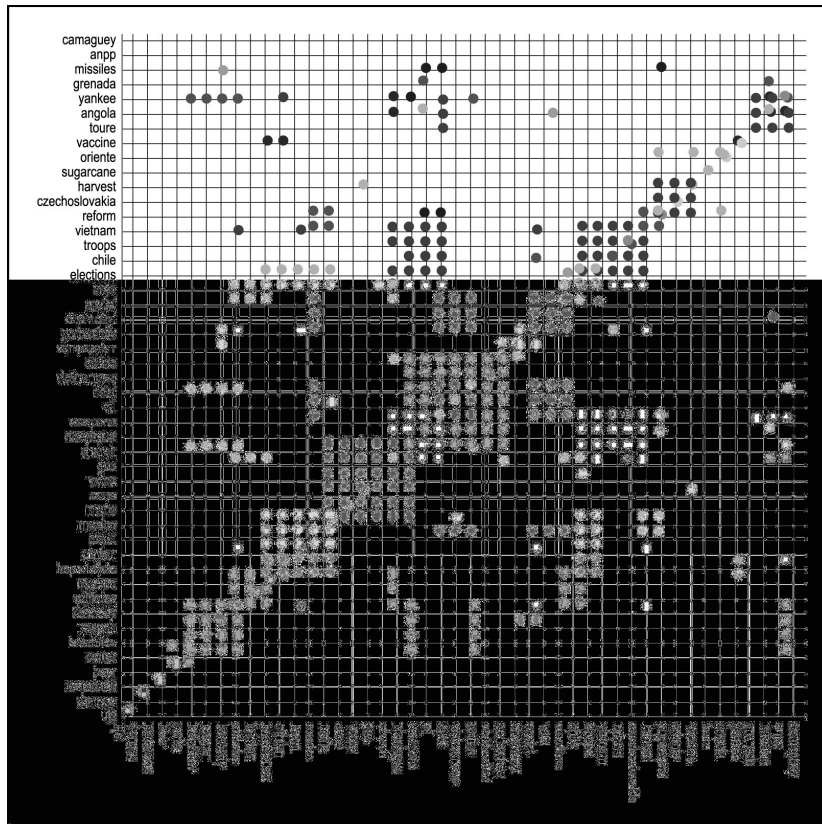


Figure 7: (a) Map of the internet [Cheswick, 1998] color coded by major ISPs (Courtesy William Cheswick).  
 (b) Connex [Hetzler et al., 1998] adjacency matrix representation of a relationship network (Courtesy Pacific Northwest National Laboratory).

### 3.4 Text and Document Collection Structure

This structure consists of arbitrary collections of documents, often text. Examples include digital libraries, news archives, digital image repositories, and software code. Of the four types of information structure, this type is the least structured and, hence, can be the most difficult to design visualizations for. Text is particularly challenging to map to visual form, because it is not obvious how text can be the input to a mapping function as described in section 2. Mapping functions must take advantage of the minimal structure and other characteristics of text to generate useful data for computing visual representations. External structures of text or document collections such as tables of contents (tree structure), meta-data (tabular structure), or citations (network structure) are categorized as other types of information structures and discussed in the previous sections. The emphasis of the structure described in this section is on the full text or the documents themselves. Solutions range from the macro scale (overview of large collections) to the micro scale (a single document fragment).

A major class of text visualizations focuses on providing semantic maps of large document collections based on document topics. Generally, the goal is to spatially cluster documents in the visualization such that similar documents (documents containing similar content) are near each

other, while dissimilar documents are distant. This creates a map of the document space based on the metaphor of a physical library in which books are carefully arranged by topic. Similarity between documents can be measured in many ways, and is generally the domain of *information retrieval* [Baeza-Yates & Ribiero-Neto, 1999]. A common method is to compare the frequency of occurrence of dictionary words or phrases between the two documents. Densities of document clusters can then be analyzed to extract topic keywords for labeling the map. Document Galaxies [Wise et al., 1995] and Kohonen self-organizing maps [Lin, 1992] map individual documents to tiny dots that are clustered by text content. Selecting a dot from the map reveals a document summary or opens the full document. ThemeView [Wise et al., 1995] (Figure 8a) emphasizes the documents' topics, creating a 3D terrain landscape representing the themes in the collection. Themes are mapped to terrain landscapes of mountains, with theme strength mapped to mountain height. Mountains that are adjacent or joined indicate the presence of documents that span both themes.

The keyword query approach provides a more focused map, based on keywords specified by the user. VIBE [Korfhage, 1995] visualizes how documents relate to the keywords. It spreads the user's keywords around the periphery of the display. Then, document dots are mapped into the space according to their strength of match to each keyword, using a spring-based attraction model. TileBars [Hearst, 1995] inverts the map, showing how the keywords relate to each document. Document hits are listed as in a normal textual search engine, but each document has a tilebar that shows the density of the keywords in each section of the document.

Lastly, documents can be arranged by the users themselves or by some default order. Miniature representations of the documents can be displayed to promote browsing by content. Web Book and Forager [Card et al., 1996] collects favorite web pages in a virtual 3D book that users can quickly flip through and visually scan. Books can be arranged on a virtual bookshelf. With DataMountain [Robertson et al., 1998], users arrange images of favorite web pages or photos on an inclined plane (Figure 8b), taking advantage of spatial memory for recall. At the lowest level of text visualization, SeeSoft [Eick et al., 1992] visualizes the text of software code using a miniaturized representation. It displays each line of text as a tiny colored line segment (more on SeeSoft in Section 4.2).

<<Figure 8: (a) (b) here >>

Figure 8 a, b:

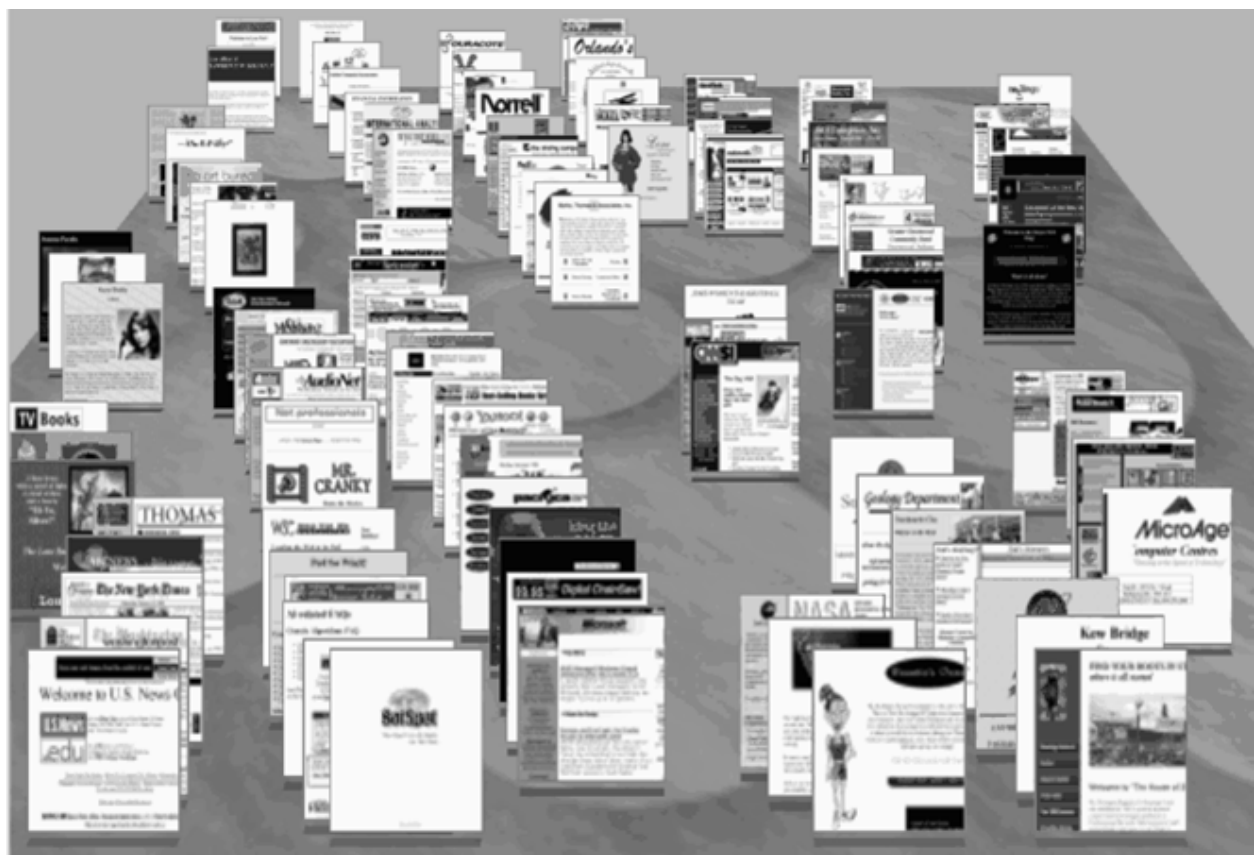
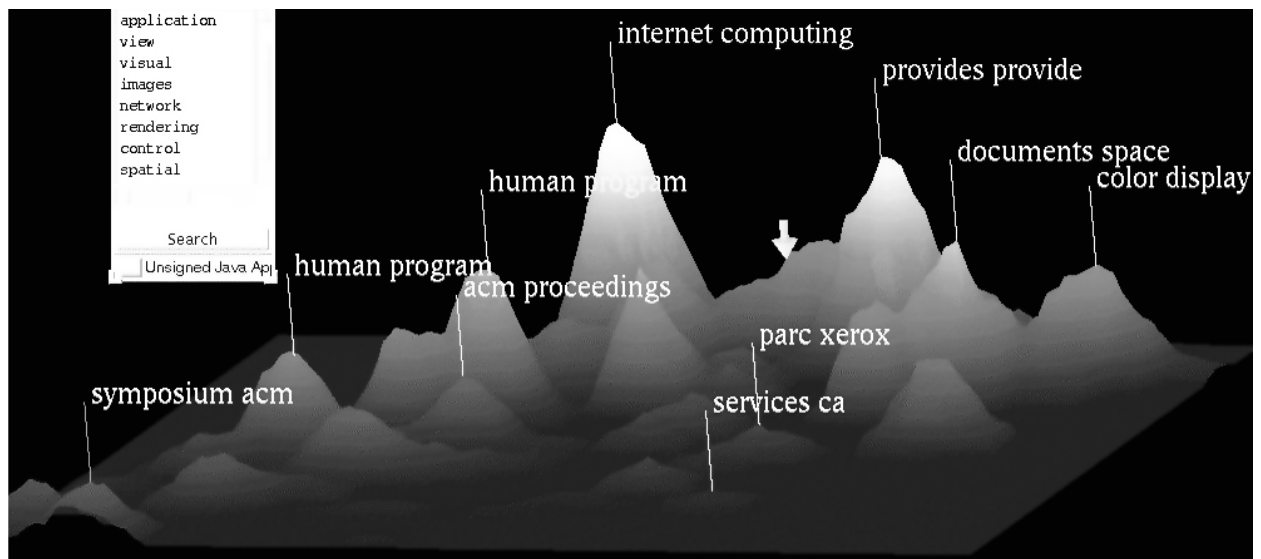


Figure 8: (a) ThemeView [Wise et al., 1995] represents common themes in a large document collection as labeled mountains in a landscape. Probing reveals specific documents. (Courtesy Pacific Northwest National Laboratory)

(b) DataMountain [Robertson et al., 1998] lets users visually organize web favorites, documents, or digital photos for later recall (© 1998 ACM, Inc. Included here by permission).

### 3.5 Combining Multiple Structures

Frequently in real-world applications, information involves complex combinations of multiple information structures. Furthermore, information of one structure type could be computationally massaged into a different structure type to offer new ways to conceptualize the information. For example, a single e-commerce website may consist of a text document collection of product pages, which also contains a network structure of hyperlinks, is organized by a tree structured site map, and has tabular meta-data about product prices and page accesses. A visualization designer should consider each of these separate structures as a potential visual index into the underlying product information.

A frequent insight goal in these situations is to relate the different structures. However, designing a visual representation that effectively combines multiple structures is difficult. Since a structure typically consumes the primary spatial portion of the visual mapping, combining multiple structures in a single mapping can result in conflict. A primary decision is whether to attempt to combine them, or to separate the structures into multiple views [Baldonado et al., 2000]. Multiple views simplify the design, since each structure can independently use its most optimal mapping. The structures can then be related by interactive linking between the views (see Section 6.2). Linking is useful for querying one structure with respect to another. However, because interactive linking only reveals a small number of associations at a time, users must mentally integrate the relationships between the structures over time, and can easily miss interesting associations. On the other hand, integrating two structures into a single view typically requires that one structure be used as the spatial basis, while the other is dismantled and embedded within that space. This enables a clear representation of how the second structure depends on the first, but clarity of the second structure can be lost. Another potential solution is animating or morphing between the two structures [Robertson et al., 2002].

An example is PathSim [Polys et al., 2004] (Figure 9), an information-rich virtual environment for biology simulation, which combines 3D spatial structure of human anatomy with tabular structure of collected data about levels of viral infection within anatomical components. In this design, the tabular data is visually embedded directly within the 3D anatomy as small manipulable visualizations adjacent to their corresponding anatomical components. The tabular structure is dismantled to visually associate portions of the dataset with components in the 3D scene. While this supports the task of understanding the effects in each anatomical component, it does not enable a single overview of all tabular results. To overcome this, a heads-up display is included according to the multiple views approach. It shows aggregated tabular information as a summary of what is visible in the entire scene as users navigate in the 3D anatomy.

<< Figure 9 here >>

Figure 9:

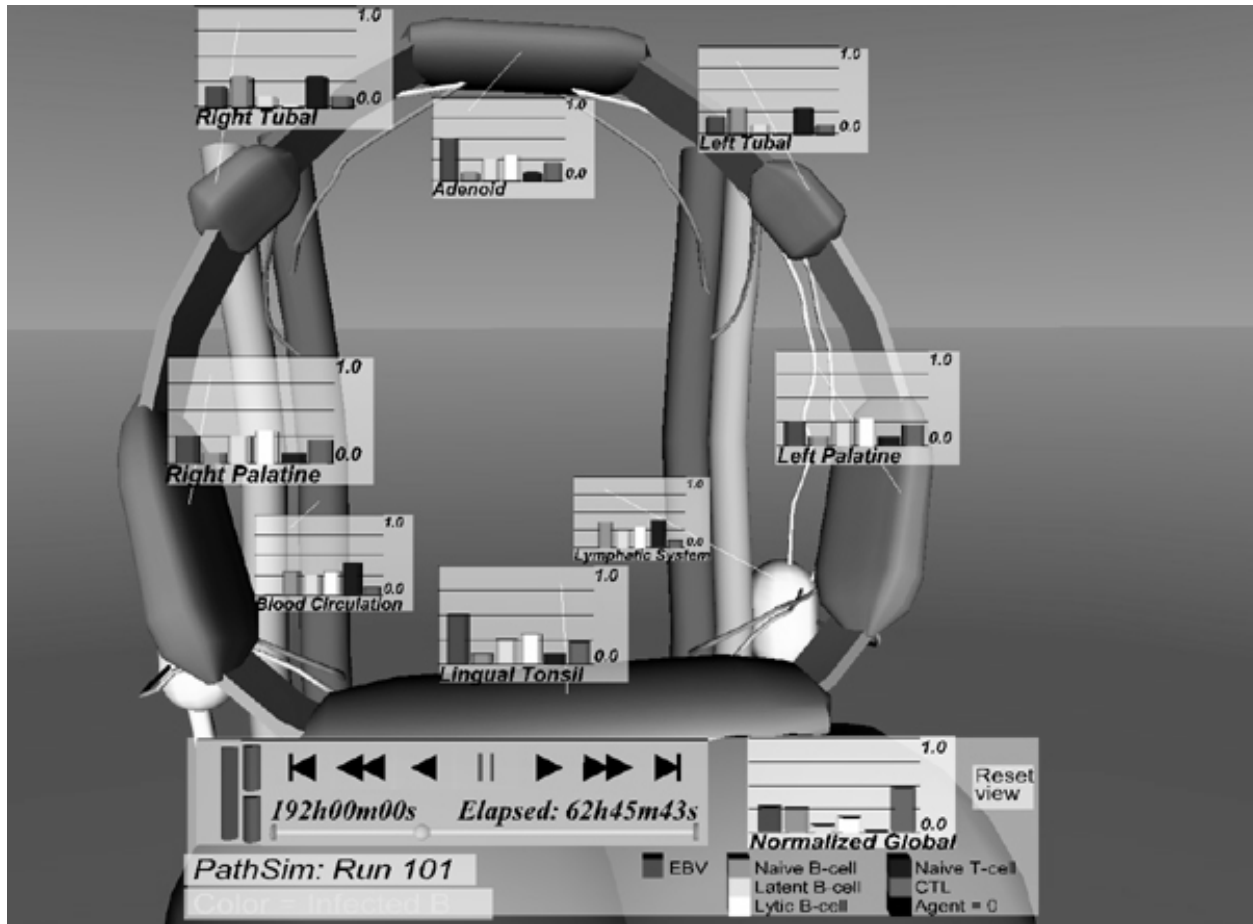


Figure 9: PathSim [Polys et al., 2004] reveals simulated viral infection data within the human anatomy, combining tabular structure with 3D spatial structure. Zooming in or out navigates to lower or higher levels of anatomical structure, and correspondingly shows lesser or higher levels of aggregation of tabular data. Here, we see the effects of an Epstein-Barr virus infection of the tonsils.

## 4 Overview Strategies

Designing methods for the visual representation of very large quantities of information is one of the fundamental problems in visualization research. As information quantity increases, it becomes more difficult to visually pack all the information on the available screen space. There simply are not enough pixels. Even if there were enough pixels, it might appear visually cluttered to include all the details in a single display. In general, a naïve visualization design would be to consume the entire display with the full detail of only a few of the data entities, and thereby limit the display to a relatively small portion of the full dataset. This is analogous to peering into a vast room through a tiny keyhole, and is called the *keyhole problem*. For example, the spreadsheet in Figure 1 shows the detailed numerical data, but the scrolling window only reveals about 40 rows at a time.

To support visualization of very large information spaces, Shneiderman suggests the design mantra “*overview first, zoom and filter, then details on demand*” [Shneiderman & Plaisant, 2005]. The solution to the keyhole problem is to start users with a broad overview of the full information space, sacrificing information details. Then, provide interaction mechanisms that enable users to zoom in on desired information and filter out anything not of interest. Lastly, quickly retrieve and display detailed information about individual data entities when selected by the user. There are several advantages to providing an initial visual overview of the information:

- Supports formation of mental models of the information space.
- Reveals what information is present or not present.
- Reveals relationships between the parts of the information, providing broader insights.
- Enables direct access and navigation to parts of the information, by simply selecting it from the overview.
- Encourages exploration.

Empirical evidence confirms that the use of visual overviews results in improved user performance in various information seeking tasks (some studies are listed in [North, 2001; Hornbæk et al., 2002]). In general, visualization designers should seek to densely pack as much information into the overview as cleanly possible. A major design decision is choosing which information to percolate up to the overview and which information to bury in the lower detail levels reached only through user interaction. This is somewhat analogous to choosing which products to show in the store window. Ideally, an overview should provide some ‘scent’ of all the detailed information hiding beneath it [Pirolli & Card, 1999].

To create overviews that attempt to pack a large dataset onto a relatively small screen, there are two possible approaches in the visual mapping process: (1) reducing the quantity of data in the dataset before the mapping is applied, or (2) reducing the physical size of the visual glyphs created in the mapping.

#### **4.1 Reducing Data Quantity**

One method for reducing the data quantity, while maintaining reasonable representation of the original data, is *aggregation*. Aggregation groups entities within the data set, creating a new dataset with fewer total entities. Each aggregate becomes an entity itself, temporarily replacing the need for all entities within the aggregate. For example, a histogram applies aggregation to represent data distribution on one attribute [Spence, 2001].

When using aggregation, the first design decision is choosing which entities should be grouped together. Entities can be grouped by common attribute values [Stolte et al., 2002], or by more advanced methods such as clustering algorithms [Yang et al., 2003] or nearest neighbors. The next decision is determining the new attribute values of the aggregates. Ideally aggregates’ values should be representative of the contained member entities. Statistical summaries such as mean, minimum, maximum, and count are commonly used. Aggregation can be iteratively applied to generate tree structures of groups and subgroups [Conklin et al., 2002]. The final decision is the visual representation of the aggregates that, ideally, reveals some hint of their contents. Aggregate Towers [Rayson, 1999] (Figure 10a) spatially groups entities if they overlap on a map. The aggregates are shown as towers whose height represents the number of entities in

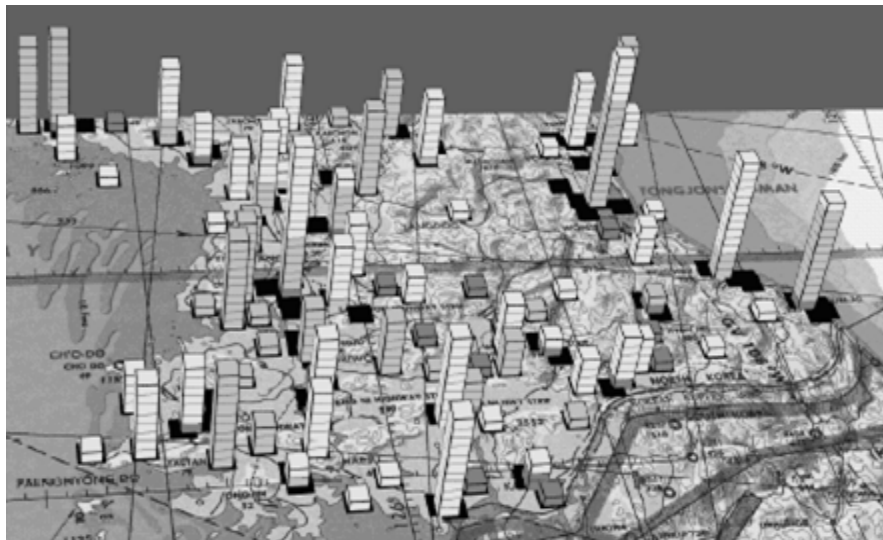
the aggregate. Zooming out of the map causes further aggregation as needed, and zooming in on the map segregates towers until no towers are needed. XmdvTool [Yang et al., 2003] (Figure 10b) clusters tabular data in a parallel coordinates plot. The extent of the contents of each aggregate is revealed by a glowing shadow that emanates from the aggregate's polyline.

Aggregation can also be applied to data attributes. Dimensionality reduction methods reduce the number of data attributes in large multi-dimensional tabular datasets so that they can be more easily visualized [Rencher, 2002]. The reduced set of attributes should approximately capture the main trends found in the full set of attributes. For example, Principal Components Analysis projects the data entities onto a subspace of the original data space that best preserves the variance in the data. Multi-Dimensional Scaling uses measures of similarity between entities, based on their many attribute values, to compute a one-, two-, or three-dimensional mapping that spatially groups similar entities.

Techniques for reducing data quantity based on filtering include: VIDA [Woodruff, 1998] selects a representative subset of data entities, based on data density and entity importance. Spotfire [Ahlberg & Wistrand, 1995] relegates less important data attributes to interactive methods such as dynamic queries, eliminating them from the input to the visual mapping function. Tree structured information is easily reduced by simply filtering deeper levels of the tree to visualize the upper levels as an overview.

<< Figure 10: (a) (b) here >>

Figure 10 (a) & (b):



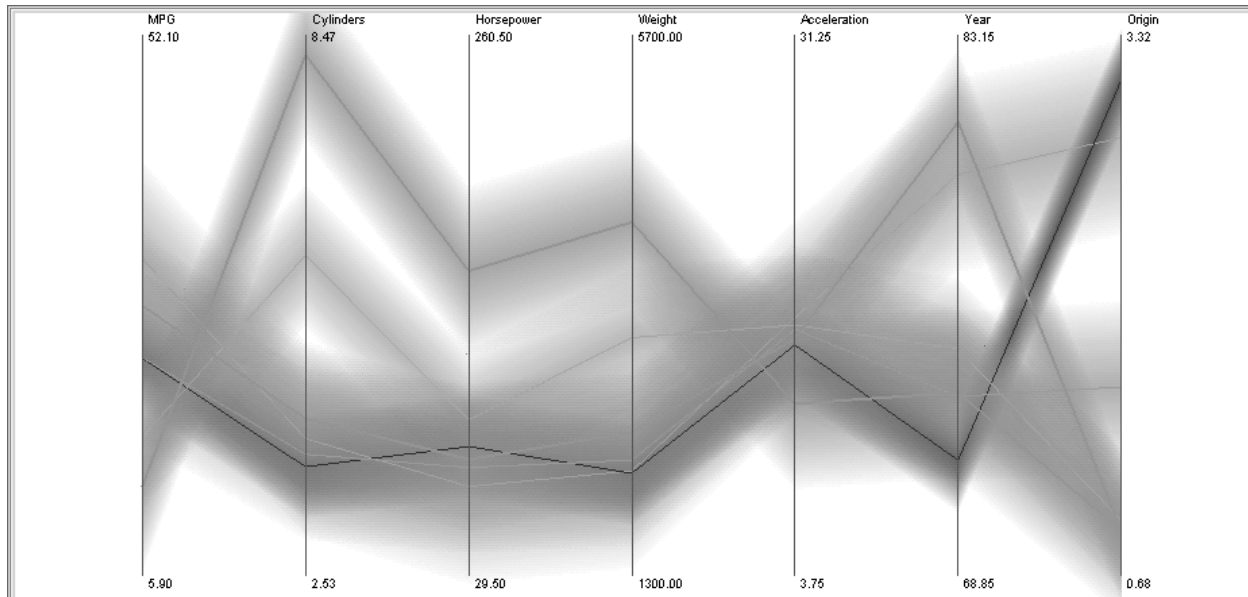


Figure 10: (a) Aggregate Towers [Rayson, 1999] stacks military units that spatially overlap on a map. The black footprints show the spatial coverage of each tower. Zooming in separates the units. (© 1999 IEEE)

(b) XmdvTool's Parallel Coordinates clusters the data from Figure 4(b) into six entities to reduce clutter [Yang et al., 2003]. Colored shadows reveal the approximate spread of each cluster. (Courtesy Matthew Ward)

## 4.2 Miniaturizing Visual Glyphs

Alternatively, emphasis can be placed on miniaturization of the visual glyphs generated by the visual mapping process. Tufte argues for increased data density in visual displays, by maximizing the data per unit area of screen space, and maximizing the data:ink ratio [Tufte, 2001]. Higher data:ink ratio is accomplished by minimizing the quantity of “ink” required for each visual glyph, and eliminating chart junk that wastes ink on unimportant non-data elements.

SeeSoft [Eick et al., 1992] (Figure 11a) provides an overview of textual software code using miniaturization. Similar to TableLens (Figure 4a), each line of code is reduced to a single line segment of colored pixels whose length is proportional the number of characters in the line of code. In this way, large software projects of up to 50,000 lines of code can be viewed in a single screen. Color coding can be used to reveal other attributes of the lines of code, such as which programmer wrote it, whether it has been tested, or the amount of CPU time required to execute the line (code profiling). Pixel Bar Charts [Keim et al., 2002] reduces the size of visual glyphs for tabular data to a single pixel, colored by one attribute and ordered on the display by another attribute. The Information Mural [Jerding & Stasko, 1998] (Figure 11b) takes miniaturization to the sub-pixel level. When many glyphs overlap and occlude each other, Mural visualizes the density of the glyphs like an x-ray image.

<< Figure 11 (a) (b) here >>

Figure 11 (a)(b):



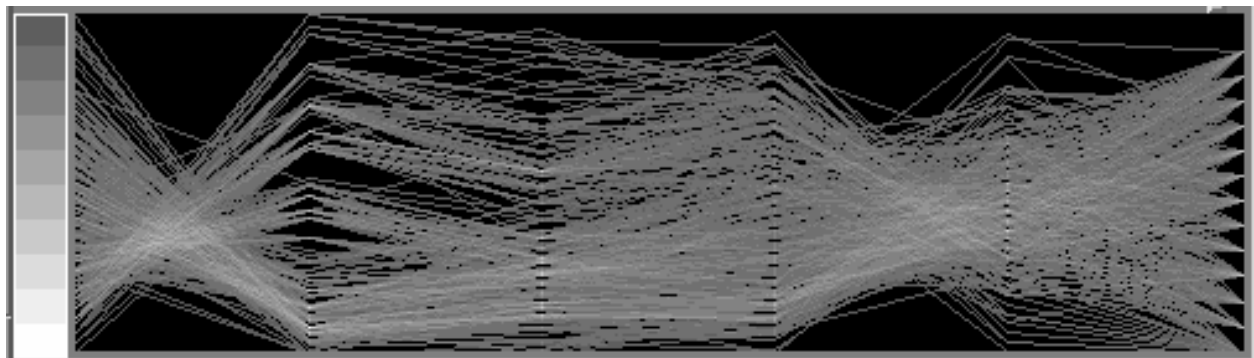
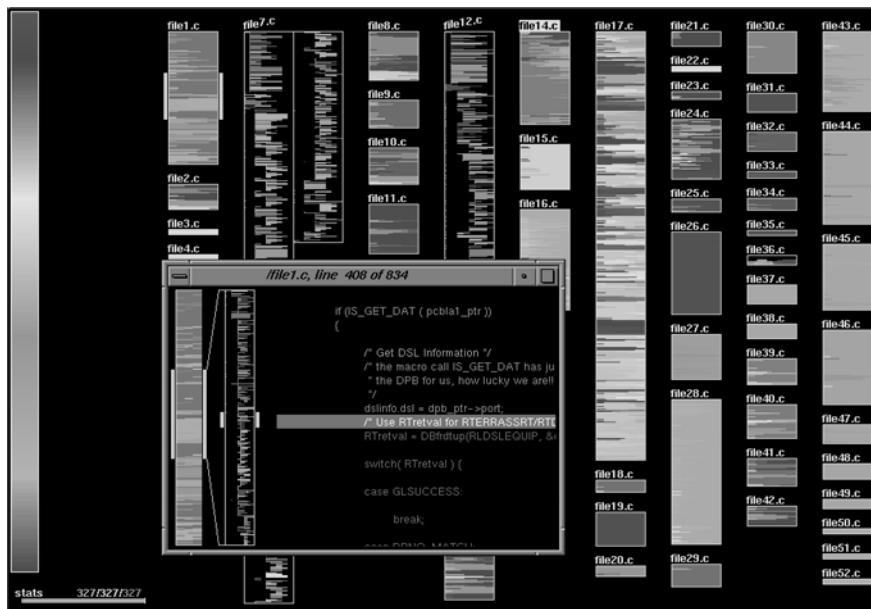


Figure 11: (a) SeeSoft [Eick et al., 1992] provides a miniaturized visual overview of software code, coloring each line of code by an attribute such as date authored (Courtesy Stephen Eick). (b) Information Mural [Jerding & Stasko, 1998] shows the density of a parallel coordinates plot, enabling users to see hidden patterns within dense clutter that would otherwise be occluded as in Figure 4b (Courtesy John Stasko).

## 5 Navigation Strategies

After employing an overview strategy to provide a broad view of a large information space, the next design concern is that of navigation. Interactive methods are needed to support the navigation between broad overview and the details of the entire information. To support this need, three primary navigation design strategies have evolved: *Zoom+Pan*, *Overview+Detail*, and *Focus+Context*. These strategies reside at the third stage of the visualization pipeline, view transformation (see Figure 2).

These navigation strategies should be contrasted with the naïve strategy called *Detail-Only*. Detail-only is the baseline strategy that does not employ an overview. It provides only the

detail-level view of a portion of the information space (like the spreadsheet in Figure 1a). Users can navigate by scrolling or panning to access the rest of the information space. In general, the detail-only strategy should be avoided. The principle disadvantage is disorientation due to lack of overview, leaving the user lost in the information space and wondering: Where am I? Where do I want to go? How do I get there?

## 5.1 Zoom+Pan

Zoomable visualizations begin with the overview, and then enable users to dynamically zoom into the information space to reach details of interest. Users can zoom back out to return to the overview, and again zoom in to another portion of detail. Users can also pan across the space without zooming out. Zooming can be a smooth continuous navigation through the space as in Pad++ [Bederson et al., 1996] (Figure 12), or can be used to drill down through discrete levels of scale as in Treemaps [Johnson & Shneiderman, 1991]. While the zooming strategy provides for an overview, the primary problem that remains is disorientation when zooming in. It is easy for users to become lost in the space when zooming in and panning, since the overview is no longer present.

<< Figure 12 here >>

Figure 12:

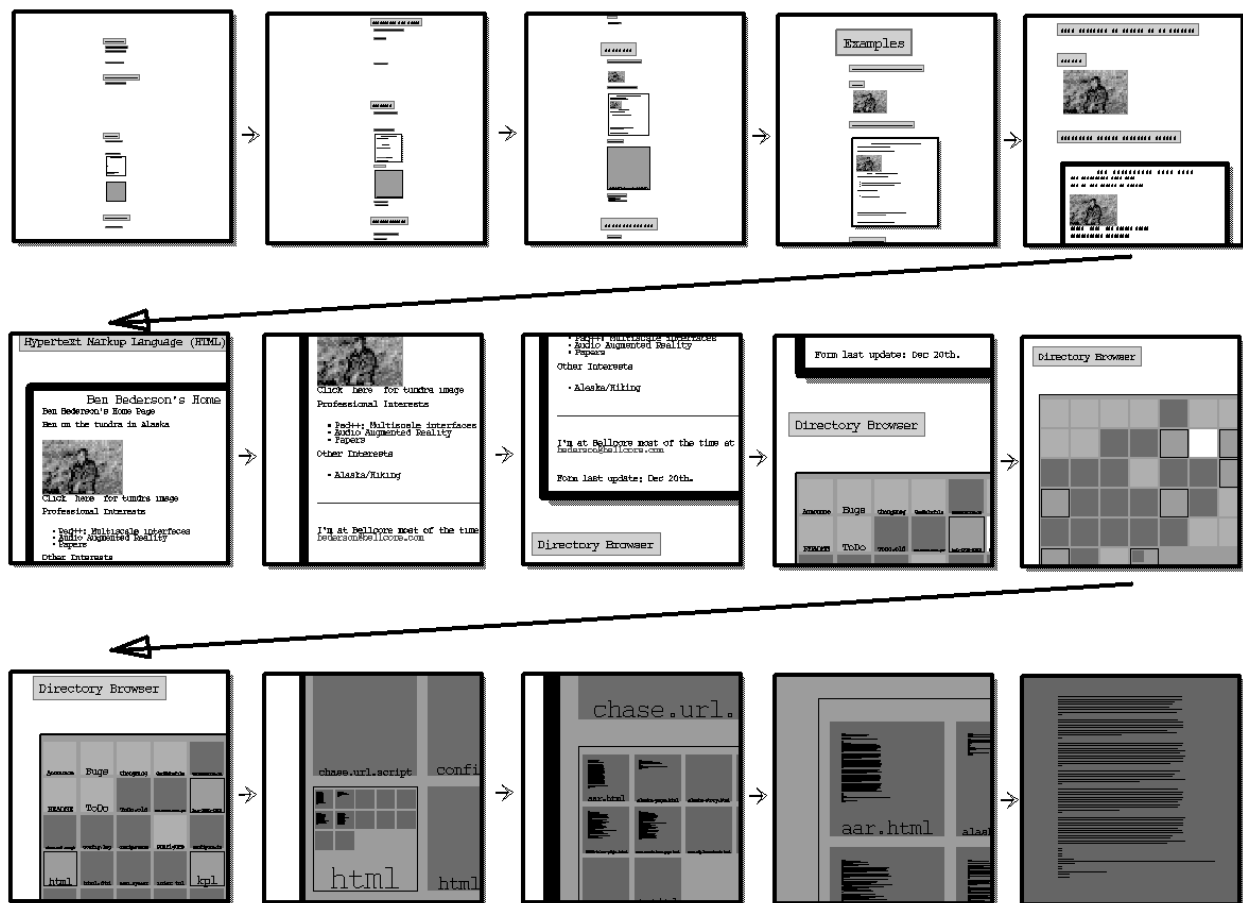


Figure 12: A zooming sequence in Pad++ [Bederson et al., 1996], from a web page to an embedded folder to an embedded text file (Courtesy Ben Bederson).

## 5.2 Overview+Detail

This strategy uses multiple views to simultaneously display an overview and a detail view. A field-of-view indicator in the overview indicates the location of the detail view within the information space. The views are linked such that manipulating the field-of-view in the overview causes the detail view to navigate accordingly. Likewise, when users navigate directly in the detail view, the field-of-view updates to provide location feedback. This strategy is commonly found in various map and image browsing software [Plaisant et al., 1995]. In SeeSoft [Eick et al., 1992], the miniaturized overview of text operates as a scrollbar for the detail view of the actual text (Figure 11a, center). A zoom factor of 30:1 between overview and detail view is the usability limit for navigating 2D images, but intermediate views can be chained to reach higher total zoom factors [Plaisant et al., 1995]. In navigating 3D worlds, Worlds in Miniature [Stoakley et al., 1995] provides a small 3D overview map attached to a virtual glove (Figure 13, bottom center) to help orient users within the world. Overview+detail preserves overview to avoid disorientation in the detail view, but suffers from a visual discontinuity between the overview and detail view.

<< Figure 13 here >>

Figure 13:

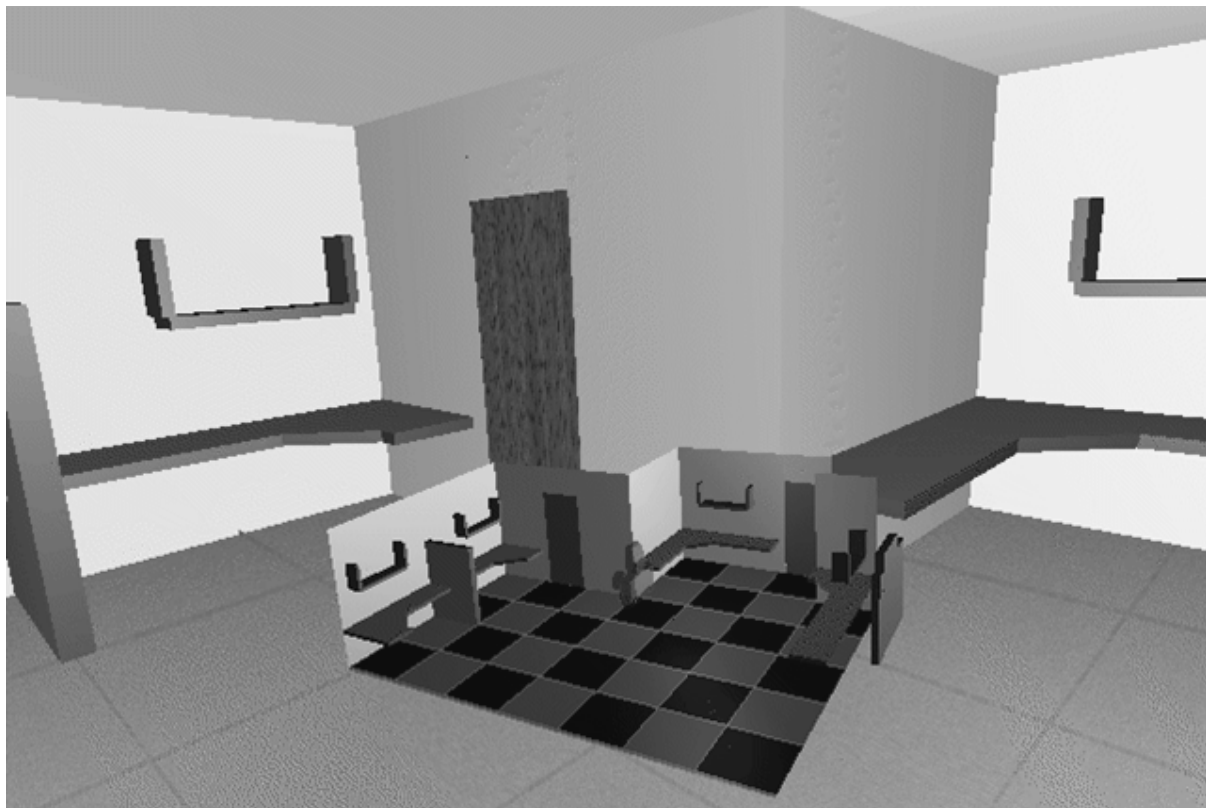


Figure 13: Worlds in Miniature [Stoakley et al., 1995] uses overview+detail for navigating 3D virtual environments (© 1995 ACM, Inc. Included here by permission).

### 5.3 Focus+Context

This strategy expands a focus region directly within the overview context. The focus is enlarged and magnified to provide detailed information for that portion of the information space. Users can navigate by simply sliding the focus across the overview to reveal details for other portions of the space. To make room for the expanded focus region, the surrounding overview must be partially pushed back by distorting or warping the overview. For this reason, this strategy is also referred to as fisheye [Furnas, 1986] or distortion-oriented [Leung & Apperley, 1994] displays. Without distortion, the magnified region would occlude the adjacent context like a magnifying glass. Since the near context is the most important part of the context, the magnifying glass effect is undesirable, and distortion is required to preserve the overview. In general, the focal point is magnified the most, and the degree of magnification decreases with distance from the focal point. Careful design based on a variety of metaphors can help to minimize the negative effects of the distortion.

Several variants of the focus+context strategy have been developed for navigating 1D and 2D spaces, including: *Bifocal* [Spence, 2001] uses two distinct levels of magnification, like TableLens [Rao & Card, 1994] (Figure 4a). *Perspective* wraps information on 3D angled surfaces, like the Perspective Wall [Robertson et al., 1993] (Figure 14a). *Wide-angle* lens creates a classic visual fisheye effect, like Hyperbolic Trees [Lamping et al., 1995]. *Non-linear* uses more complex magnification functions to create a magnified bubble effect [Keahey & Robertson, 1996] (Figure 14b). As an alternative to spatial distortion, “focus+context screens” [Baudisch et al., 2002] offer *resolution distortion* which may provide a better match to the human visual system. Fisheyes for navigating 3D spaces have also been developed [Carpendale et al., 1997]. The focus+context strategy offers continuity of detail within overview context, but suffers from disorientation caused by distortion.

<< Figure 14: (a) (b) here >>

Figure 14 (a)(b):



(b) Non-linear magnification [Keahey & Robertson, 1996] can create this bubble effect which magnifies the focus region, squeezes the near context, and maintains an otherwise stable far context (© 1996 IEEE).

While studies have repeatedly shown advantages of these three strategies over the detail-only strategy, comparisons between the three are inconclusive and highly dependent on the specifics of the individual designs, data domains, and user tasks (e.g. [Hornbæk et al., 2002]). An analytic summary follows:

**Zoom+Pan:**

- + Screen space efficient
- + Infinite scalability
- Lose overview when zoom in
- Slower navigation

**Overview+Detail:**

- + Stable overview
- + Scalable; chained views; multiple overviews or foci
- Visual disconnect between views; back-and-forth
- Views compete for screen space; smaller overview

**Focus+Context:**

- + Detail visually connected to surrounding context
- Limited scalability; typically under 10:1 zoom factor
- Distortion; unstable overview

## 6 Interaction Strategies

Interaction strategies support further scalability and complexity of visualized information. While it is preferable to visually map all data onto the display in a form that effectively reveals all desired insights without interaction, this is generally impossible for data of even modest complexity. Interaction strategies overcome this limitation by enabling users to interactively explore additional mappings and insights over time. Many interactive techniques exist. A few major categories of interaction strategies should be considered in every visualization design.

### 6.1 Selecting

The most fundamental need in visualization is interactive selection of individual data entities or subsets of data entities. Users select entities to identify data that is of interest to them. This is useful for many reasons, including: viewing detailed information about the entities (details on demand), highlighting entities that are obscured or occluded in a crowded display, grouping a set of related entities, or extracting entities for future use.

In general, there are two possible criteria by which users can specify selections. First, users can select data entities directly. Direct manipulation visualizations enable users to directly select entities in a visualization using a variety of techniques [Wills, 1996], such as pointing at individual entities' glyphs (as in Figure 4a), lassoing a group of glyphs, etc. Second, users can

indirectly select data entities through selection criteria on information structures (Section 3). For example, XmdvTool [Ward, 1994] enables users to make selections in tabular data in parallel coordinates by specifying range criteria on data attributes. In Figure 4b, all American-made cars are highlighted by selecting the ‘USA’ range on the ‘Origin’ axis. Other structure-based selection techniques include selecting an entire branch in a tree structure, selecting a path in a network structure, or selecting a ThemeView mountain [Wise et al., 1995] in a collection structure. Selection techniques should be designed to enable users to easily select entities, select additional entities into the current selection, remove entities from the current selection, and clear the selection. Selecting is sometimes called *brushing*, because it is like painting glyphs with a special type of paint brush that behaves according to the selection technique.

## 6.2 Linking

Linking is useful to interactively relate information between multiple views [Baldonado et al., 2000; North, 2001]. Information can be mapped differently into separate views to reveal different perspectives or different portions of the information. The most common form of linking is called *brushing and linking* [Becker & Cleveland, 1987]. Interactive selections of entities in one view are propagated to other views to automatically highlight corresponding entities, enabling users to recognize relationships. This strategy enables users to simultaneously take advantage of the different strengths of different visual representations. This is particularly useful for relating between different information structures (Figure 15), essentially using one structure to query another. Users can select entities according to criteria in one structure, which then shows the distribution of those entities within the other structure. While linking is commonly used to relate two views of the same dataset in a one-to-one fashion, it can also be used to relate entities across many-to-many database relationships for more complex scenarios. Linking can also be combined with navigation strategies to enable other useful techniques such as synchronized scrolling, or specialized overview+detail combinations for drilling down through hierarchical data [North et al., 2002].

<<Figure 15 here >>

Figure 15: <<(top) and (bottom) parts should be placed together>>

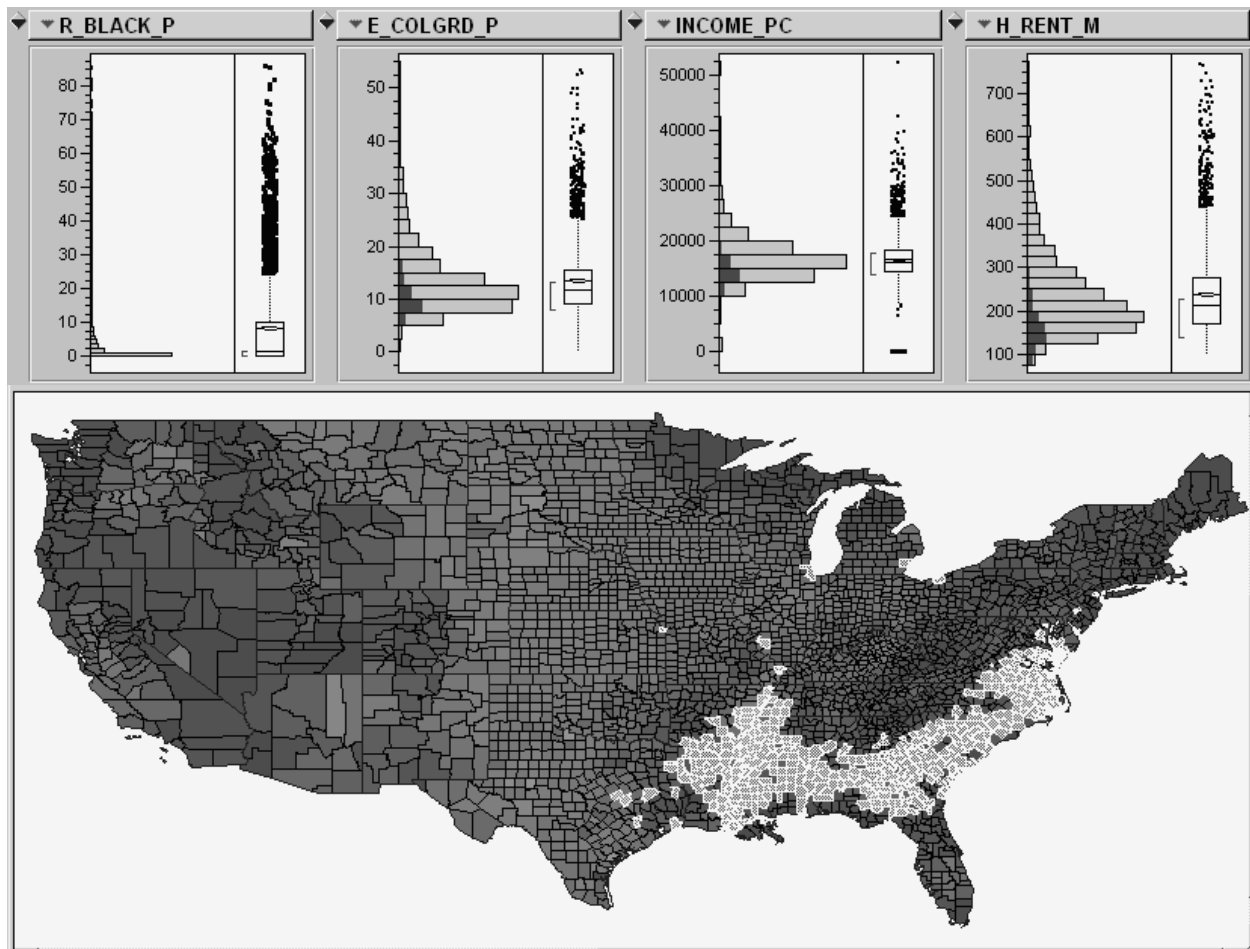


Figure 15: Interactive brushing and linking between histogram plots (top) and geographic map (bottom) of a census counties dataset. Histograms (tabular information structure) show data distributions for four county attributes, ‘percent population black’, ‘percent population with college degree’, ‘income per capita’, and ‘median rent’. The map (2D spatial information structure) is colored by ‘percent farmland’. Selecting the counties that are more than 25% African American in the first histogram also highlights those counties in the other views. The highlighting in the map reveals that those counties are clustered in the south and south-east regions. Selecting counties in the map would similarly highlight them in the histograms. The histograms are generated by JMP [SAS, 2004], and map by ArcView [ESRI, 2004]. They are linked by Snap-Together Visualization [North et al., 2002].

### 6.3 Filtering

Interactive filtering enables users to dynamically reduce information quantity in the display, and focus in on information of interest. *Dynamic queries* [Ahlberg & Wistrand, 1995] apply direct manipulation principles to querying attribute values. Visual widgets, such as the range slider (Figure 1b, right), enable users to rapidly adjust query parameters and immediately view filtered results in the visualization in real time. The widgets also provide a visual representation of the current query parameters. Because of the rapid feedback, dynamic query filters can be used not just for reducing information quantity but also for exploring relationships between mapped



attributes and query attributes. For example, in Figure 1, filtering with the query slider for “unemployment” to eliminate the low unemployment counties from the display reveals that counties with high unemployment are all in the low income and low education area of the plot. The rapid query feedback also eliminates the difficulty of zero-hit or mega-hit query results, because users can quickly adjust the query parameters to until a desirable number of hits is acquired. For example, by further filtering on “unemployment” in Figure 1, users find that there are 11 counties with unemployment rate over 20%, mostly located near the border with Mexico. Dynamic queries are the inverse of brushing; brushing highlights selected data, while dynamic queries elide non-selected (filtered) data. Magic Lenses [Fishkin & Stone, 1995] offer a spatially localized form of filter. For more advanced queries involving complex combinations of boolean operations, metaphors such as Filter Flow [Young & Shneiderman, 1993] enable users to construct virtual pipelines of filters.

#### **6.4 Rearranging and Remapping**

Since a single mapping of information to visual form may not be adequate, it is straightforward to enable users to customize the mapping or choose among several mappings. Since the spatial layout is the most salient visual mapping, rearranging the spatial layout of the information is the most potent for generating different insights. For example, TableLens [Rao & Card, 1994] (Figure 4a) can spatially rearrange its view by choosing a different attribute to sort by, and Parallel Coordinates [Inselberg, 1997] (Figure 4b) can rearrange the left-to-right order of its axes. This enables users to explore relationships between different attributes.

In general, any part of the mapping process throughout the visualization pipeline can be under user control. For example, Spotfire [Ahlberg & Wistrand, 1995] users can customize the scatterplot view (Figure 1b) by choosing data attributes to map to various visual properties such as x, y, color, and size. It also provides a variety of different visual representations to choose from, including heat maps (colored spreadsheets), parallel coordinates, histograms, and pie and bar charts. Visage [Roth et al., 1996] emphasizes a technique called ‘data-centric’ interaction, in which users can directly select data entities and drag them to different views to display them in new ways. At the extreme are systems such as Sage and SageBrush [Roth et al., 1994] that let users design new visual mappings for a dataset, using a set of basic primitives as described in Section 2. Sage can also automatically generate certain visual mappings for a given dataset and task, using a rule-based expert system.

### **7 The Future**

Information visualization is a relatively young field (e.g. the IEEE Symposium on Information Visualization started in 1995). Significant further research is needed on new visual mappings, overview strategies, interaction and navigation strategies, evaluation methods, and guidelines. A few critical areas of need that should be explored in the foreseeable future include:

- Visualization of massive data: Recent developments in intelligence analysis and homeland security require new abilities to analyze terabytes of textual, voice, and video data in unstructured collections. Bioinformatics is driving the need for new methods to visualize mega-dimensional tabular datasets, containing 1000s or 1,000,000s of data attributes, and huge networks.

- Integrating visualization with the analysis context: Visualization is not an independent task, but must be integrated with data management, information retrieval, statistical analysis, data mining [Shneiderman, 2002], decision support, task management, and content authoring and publishing.
- High-resolution visualization: New high-resolution display technologies are becoming more common [Funkhouser & Li, 2000]. New visualization strategies must be devised to effectively take advantage of significantly increasing quantity of pixels and large physical screen sizes, and to explore the limits of visualization.

## References

Ahlberg, C., Wistrand, E. (1995). IVEE: An Information Visualization and Exploration Environment. *Proc. Symposium on Information Visualization 1995*, Atlanta, GA, pp. 66-73. <http://www.spotfire.com/>.

Andrews, K., Kappe, F., and Maurer, H. (1995). Hyper-G and harmony: towards the next generation of networked information technology. *Conference companion on Human factors in computing systems 1995*, 33—34, Denver, Colorado.

Baeza-Yates, R., Ribiero-Neto, B. (1999). *Modern Information Retrieval*. Addison-Wesley.

Baldonado, M., Woodruff, A., Kuchinsky, A. (2000). Guidelines for using multiple views in information visualization. *Proc. ACM Advanced Visual Interfaces '00*, pp. 110-119.

Baudisch, P., Good, N., Bellotti, V., Schraedley, P. (2002). Keeping things in context: a comparative evaluation of focus plus context screens, overviews, and zooming. *Proceedings of the SIGCHI conference on Human factors in computing systems 2002*, 259—266, Minneapolis, Minnesota.

Becker R., and Cleveland, W. (1987). Brushing scatterplots. *Technometrics*, 29(2):127-142.

Becker, R. A., Eick, S. G., and Wilks, A. R. (1995). Visualizing network data. *IEEE Transactions on Visualization and Graphics*, 1(1):16-28.

Bederson, B.B., Hollan, J., Perlin, K., Meyer, J., Bacon, D., and Furnas, G. (1996). Pad++: A Zoomable Graphical Sketchpad for Exploring Alternate Interface Physics. *Journal of Visual Languages and Computing*, 7(1):3-32.

Bederson, B., Shneiderman, B., and Wattenberg, M. (October 2002). Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies. *ACM Transactions on Graphics*, 21(4):833-854.

Bertin, J.B. (1983). *Semiology of Graphics: Diagrams, Networks, Maps*. Translated by Berg, W.J., The University of Wisconsin Press.

Beshers, C. and Feiner, S. (1993). AutoVisual: Rule-based design of interactive multivariate visualizations. *IEEE Computer Graphics and Applications*, 13(4): 41-49.

Card, S., Mackinlay, J., Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*, Morgan Kaufmann.

Card, S., Robertson, G., York, W. (1996). The WebBook and the Web Forager: an information workspace for the World-Wide Web. *Proceedings of the SIGCHI conference on Human factors in computing systems 1996*, pp. 111, Vancouver, British Columbia.

Carlis, J., Konstan, J. (1998). Interactive visualization of serial periodic data. *Proc. ACM User Interface Software and Technology -- UIST '98*, pp 29-38.

Carpendale, M. S. T., Cowperthwaite, D. J. and Fracchia, F. D. (1997). Extending Distortion Viewing from 2D to 3D. *IEEE Computer Graphics & Applications*, 17(4): 42-51.

Carswell, C. (1992). Reading graphs: Interactions of processing requirements and stimulus structure. In B.Burns (Ed.), *Percepts, Concepts, and Categories: The Representation and Processing of Information*, pp. 605-645. Amsterdam: Elsevier Science Publishers.

Chambers, J., Cleveland, W., Kleiner, B., and Tukey, P. (1983). *Graphical Methods for Data Analysis*, Wadsworth.

Chen, C. (1999). *Information Visualisation and Virtual Environments*, Springer-Verlag.

Chen, C. and Yu, Y. (2000). Empirical studies of information visualization: a meta-analysis. *Int. J. Human-Computer Studies*, 53(5): 851-866.

Cherno?, H. (1973). The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68: 361–368.

Cheswick, B. (1998). The Scenic Route. *Wired Magazine*, 6(12), Dec. 1998. Also: Internet Mapping Project, <http://research.lumeta.com/ches/map/>.

Chewar, C. M., McCrickard, D. S., Ndiwalana, A., North, C., Pryor, J., and Tessendorf, D. (2002). Secondary task display attributes: optimizing visualizations for cognitive task suitability and interference avoidance. *Proceedings of the Symposium on Data Visualisation 2002*, pp 165 – 171, Barcelona, Spain.

Cleveland, W. (1993). *Visualizing Data*, Hobart Press.

- Cleveland, W. S., and McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of American Statistical Association* 79(387): 531-554.
- Conklin, N., Prabhakar, S., North, C. (2002). Multiple Foci Drill-Down through Tuple and Attribute Aggregation Polyarchies in Tabular Data. *Proc. IEEE Symposium on Information Visualization 2002*, pg. 131-134.
- ESRI. (2004). ArcView desktop GIS. Environmental Systems Research Institute, Inc., Redlands, CA. <http://www.esri.com/>.
- Eick, S. G., Steffen, J. L., and Sumner, E. E. Jr. (1992). SeeSoft---A Tool for Visualizing Line Oriented Software Statistics. *IEEE Transactions on Software Engineering*, 18 (11): 957—968.
- Elliott, E., and Davenport, G. (1994). Video streamer. *ACM Conference Companion on Human Factors in Computing Systems 1994*, pp 65—68, Boston, Massachusetts.
- Fayyad, U.M., Grinstein, G., Wierse, A., Fayyad, U. (2001). *Information Visualization in Data Mining and Knowledge Discovery*, Morgan Kaufmann.
- Feiner, S. (1988). Seeing the forest for the trees: hierarchical displays of hypertext structures. *Conference Sponsored by ACM SIGOIS and IEEECS TC-OA on Office Information Systems*, pp. 205—212, Palo Alto, California.
- Fishkin, K., and Stone, M. (1995). Enhanced Dynamic Queries via Movable Filters. *ACM Conference on Human Factors in Computing Systems*, Denver, Colorado, pp. 415-420.
- Funkhouser, T., and Li K. (2000). Onto the Wall: Large Displays, *Special issue of IEEE Computer Graphics and Applications*, 20(4).
- Furnas, G. (1986). Generalized fisheye views. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems 1986*, pp. 16—23, Boston, Massachusetts.
- Furnas, G., and Zacks, J. (1994). Multitrees: enriching and reusing hierarchical structure. In *Conference proceedings on Human Factors in Computing Systems (CHI '94)*, pp. 330-336.
- Gillan, D., Wickens, C., Hollands, J., Carswell, C. (1998) Guidelines for presenting quantitative data in HFES publications. *Human Factors* 36, pp 419-440.
- Grosjean, J., Plaisant, C., Bederson, B. (2002). SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. *Proceedings of IEEE Symposium on Information Visualization*, pp. 57-64, Boston.
- Healey, C. G., Booth, K. S., and Enns, J. T. (1996). High-Speed Visual Estimation Using Preattentive Processing. *ACM Transactions on Human Computer Interaction* 3(2): 107-135.

- Hearst, M. (1995). TileBars: Visualization of Term Distribution Information in Full Text Information Access, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pp. 59-66, Denver, CO.
- Herman, I., Melancon, G., and Marshall, M. S. (2000). Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24-43.
- Hetzler, B., Harris, W.M., Havre, S., Whitney, P. (1998). Visualizing the Full Spectrum of Document Relationships. In *Structures and Relations in Knowledge Organization. Proc. 5th Int. ISKO Conf.* Wurzburg: ERGON Verlag, pp. 168-175.
- Hornbæk, K., Bederson, B., and Plaisant, C. (2002). Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4): 362—389.
- Inselberg, A. (1997). Multidimensional detective. *Proc. IEEE Symposium on Information Visualization '97*, pp 100-107.
- Jerding, D. F., and Stasko, J. T. (1998). The Information Mural: A Technique for Displaying and Navigating Large Information Spaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3): 257—271.
- Johnson, B., Shneiderman, B. (1991). Treemaps: a space-filling approach to the visualization of hierarchical information structures. *Proc. of the 2nd International IEEE Visualization Conference*, pp 284-291, San Diego.
- Kandogan, E., (2000). Star coordinates: a multi-dimensional visualization technique with uniform treatment of dimensions. *LBHT Proc. IEEE Symposium on Information Visualization 2000*, pp 9-12.
- Keahey, T.A., Robertson, E. (1996). Techniques for non-linear magnification transformations. In *Proceedings IEEE Symposium on Information Visualization '96*, pp 38-45.
- Keim, D., Hao, M., Dayal, U., Hsu, M. (2002). Pixel bar charts: a visualization technique for very large multi-attribute data sets. *Information Visualization*, 1(1): 20—34.
- Kniss, J., Kindlmann, G., and Hansen, C. (2001) Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proc. of IEEE Visualization*, pp 255--262, San Diego, CA.
- Korfhage, R. (1995). VIBE: Visual Information Browsing Environment. *SIGIR 95*, pp. 363.
- Lamping, J., Rao, R., and Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp 401—408, Denver, Colorado.

Leung, Y. K., and Apperley, M. D. (1994). A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126--160.

Lin, X. (1992). Visualization for the document space. *Proceedings of the 3rd conference on Visualization '92*, pp 274—281, Boston, Massachusetts.

Mackinlay, J. (1986). Automating the design of graphical presentations of relational information, *ACM Transactions on Graphics*, 5(2): 110-141.

Malinowski, S. (2004). Music Animation Machine, <http://www.musanim.com/>.

Mihalisin, T., Timlin, J., Schweigler, J. (1991). Visualizing multivariate functions, data, and distributions. *IEEE Computer Graphics and Applications*, 11(3): 28- 35.

Miller, J. (2004). Daisy Analytics, <http://www.daisy.co.uk/>.

Munzner, T. (1998). Exploring Large Graphs in 3D Hyperbolic Space. *IEEE Computer Graphics and Applications*, 18(4): 18-23.

North, C. (2001). Multiple Views and Tight Coupling in Visualization: A Language, Taxonomy, and System. *Proc. CSREA CISST 2001 Workshop of Fundamental Issues in Visualization*, pg. 626-632.

North, C., Conklin, N., and Saini, V. (2002). Visualization Schemas for Flexible Information Visualization. *Proc. IEEE Symposium on Information Visualization 2002*, pp 15-22.

North, C., Shneiderman, B., Plaisant, C. (1996). User Controlled Overviews of an Image Library: A Case Study of the Visible Human. *Proc. ACM Digital Libraries '96 Conference*, pg. 74-82.

Nowell, L., Schulman, R., Hix, D. (2002). Graphical encoding for information visualization: an empirical study. *IEEE Symposium on Information Visualization*, 2002, pp 43- 50.

Pirolli, P., Card, S. (1999). Information Foraging. *Psychology Review* Vol. 106, No. 4. (pp. 643-675).

Plaisant, C. (2004). The Challenge of Information Visualization Evaluation. *Proc. of Advanced Visual Interfaces --AVI'04*, pp 109 – 116.

Plaisant, C., Carr, D., Shneiderman, B. (1995). Image browsers: taxonomy, guidelines, and informal specifications. *IEEE Software*, 12(2), pp. 21-32, (March 1995).

Plaisant, C., Milash, B., Rose, A., Widoff, S., Shneiderman, B. (1996). LifeLines: Visualizing personal histories. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems '96*, pp 221-227.

Polys, N., Bowman, D., North, C., Laubenbacher, R., and Duca, K. (2004). PathSim Visualizer: an Information-Rich Virtual Environment Framework for Systems Biology. *Proceedings of ACM Web3D Symposium*, pp. 7-14.

Rao, R., Card, S. (1994). The Table Lens: Merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. *Proc. Proceedings of the SIGCHI conference on Human Factors in Computing Systems '94*, pp 318-322.  
<http://www.tablelens.com>.

Rayson, R. (1999). Aggregate Towers: Scale Sensitive Visualization and Decluttering of Geospatial Data. *Proceedings of the 1999 IEEE Symposium on Information Visualization*, pp 92-99.

Rencher, A. (2002). *Methods of Multivariate Analysis*, Second Edition, Wiley.

Robertson, G., Cameron, K., Czerwinski, M., and Robbins, R., (2002). Polyarchy visualization: visualizing multiple intersecting hierarchies. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp 423—430, Minneapolis, Minnesota.

Robertson, G., Card, S., Mackinlay, J. (1993). Information visualization using 3D interactive animation, *Communications of the ACM*, v.36 n.4, p.57-71.

Robertson, G., Czerwinski, M. Larson, K., Robbins, D., Thiel, D., van Dantzich, M. (1998). Data mountain: using spatial memory for document management. *Proceedings of the 11th annual ACM symposium on User Interface Software and Technology*, pp 153—162, San Francisco, California.

Rosenblum L., Earnshaw R., Encarnacao J., Hagen H., Kaufman A., Klimenko S., Nielson G., Post F., Thalmann D. (1994). *Scientific Visualization: Advances and Challenges*. Academic Press in association with IEEE Computer Society, USA.

Rosson, M.B., Carroll J. (2001) *Usability Engineering: Scenario-based Development of Human Computer Interaction*. Morgan Kaufmann.

Roth, S.F., Kolojejchick, J., Mattis, J., and Goldstein, J. (1994). Interactive Graphic Design Using Automatic Presentation Knowledge. *Proceedings of the Conference on Human Factors in Computing Systems (SIGCHI '94)*, Boston, MA, pp. 112-117.

Roth, S. F., Lucas, P., Senn, J. A., Gomberg, C. C., Burks, M. B., Stroffolino, P. J., Kolojejchick, J. A., and Dunmire, C. (1996). Visage: A user interface environment for exploring information. In *Proceedings of IEEE Symposium on Information Visualization*, pp 3--12. San Francisco: IEEE.

SAS (2004). *JMP: The statistical discovery software*. SAS Institute Inc., Cary, NC.  
<http://www.jmp.com/>.

Saraiya, P., North, C., Duca, K., (2004). An Evaluation of Microarray Visualization Tools for Biological Insight. *Proceedings IEEE Symposium on Information Visualization 2004*, pp 1-8.

Shneiderman, B. (1995). The eyes have it: a task by data type taxonomy for information visualization. *Proceedings of the IEEE Visual Languages* (Boulder, CO, 1996). IEEE Computer Society, pp 336-343.

Shneiderman, B. (2002). Inventing discovery tools: combining information visualization with data mining. *Information Visualization*, 1(1): 5-12.

Shneiderman, B., Plaisant, C. (2005). *Designing the User Interface: Strategies for effective human-computer interaction*, Fourth Edition, Addison-Wesley.

Spence, R. (2001). *Information Visualization*, Addison-Wesley.

Stasko, J., Catrambone, R., Guzdial, M., McDonald, K. (2000). An evaluation of space-filling information visualizations for depicting hierarchical structures. *Int. J. Human-Computer Studies*, 53(5): 663-694.

Stoakley, R., Conway, M. J., and Pausch, R. (1995). Virtual reality on a WIM: Interactive worlds in miniature. *Proceedings of the Conference on Human Factors in Computing Systems '95*, pp. 265--272.

Stolte, C., Tang, D., and Hanrahan, P. (2002). Polaris: A System for Query, Analysis and Visualization of Multi-dimensional Relational Databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1): 52-65

Tory, M., Möller, T. (2004a). Human Factors in Visualization Research. *IEEE Transactions on Visualization and Computer Graphics*, TVCG 10(1): 72-84.

Tory, M., Möller, T. (2004b). Rethinking Visualization: A High-Level Taxonomy. *Proceedings of IEEE Symposium on Information Visualization (InfoVis) 2004*, pp 151- 158.

Tufte, E. (2001). *The Visual Display of Quantitative Information*, Graphics Press, 2nd edition.

van Wijk, J., van de Wetering, H. (1999). Cushion Treemaps: Visualization of Hierarchical Information. *IEEE Symposium on Information Visualization (INFOVIS'99)*, pp 73-80, San Francisco. <http://www.win.tue.nl/sequoiaview/>.

van Wijk, J., van Liere, R. (1993). HyperSlice: visualization of scalar functions of many variables. *Proc. IEEE Visualization '93*, pp 119-125.

Ward, M. (2002). A taxonomy of glyph placement strategies for multidimensional data visualization. *Information Visualization*, 1(3): 194-210.



- Ward, M. (1994). XmdvTool: Integrating Multiple Methods for Visualizing Multivariate Data. *Proc. of Visualization '94*, p. 326-33. <http://davis.wpi.edu/~xmdv/>.
- Ware, C. (2004). *Information Visualization: Perception for Design*, Morgan Kaufmann.
- Ware, C., Purchase, H., Colpoys, L., and McGill, M. (2002). Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2), 103—110.
- Wehrend, S. and Lewis, C. (1990). A problem-oriented classification of visualization techniques. *Proceedings of IEEE Visualization (Vis'90)*, pp 139-143.
- Wickens, C., and Hollands, J. (2000). *Engineering Psychology and Human Performance*, Prentice-Hall.
- Wilkinson, L. (1999). *The Grammar of Graphics*, New York: Springer-Verlag.
- Wills, G. (1996). Selection: 524,288 ways to say 'this is interesting'. In *Proceedings of the IEEE Symposium on Information Visualization*, pp 54--60.
- Wise, J., Thomas, J., Pennock, K., Lantrip, D., Pottier, M., Schur, A., Crow, V. (1995). Visualizing The Non-Visual: Spatial Analysis And Interaction With Information From Text Documents. *Proceedings of IEEE Symposium on Information Visualization*, pp 51-58.
- Wiss, U., Carr, D.A. (1999). An empirical study of task support in 3D information visualizations. *Proceedings IEEE International Conference on Information Visualization*, pp 392 – 399.
- Woodruff, A., Landay, J., Stonebraker, M. (1998). Constant Information Density in Zoomable Interfaces. *Proc Advanced visual interfaces-- AVI'98*, pp 57 – 65, L'Aquila, Italy.
- Yang, J., Ward, M., and Rundensteiner E. (2003). Interactive Hierarchical Displays: A General Framework for Visualization and Exploration of Large Multivariate Data Sets. *Computers and Graphics Journal*, 27(2): 265-283.
- Young, D., and Shneiderman, B. (1993). A Graphical Filter/Flow Representation of Boolean Queries: A Prototype Implementation and Evaluation. *Journal of the American Society of Information Science*, 44(6), 327-339.
- Zhou, M., and Feiner, S. (1998). Visual task characterization for automated visual discourse synthesis. *Proceedings of the ACM Human Factors in Computing Systems Conference (CHI'98)*, pp 392-399.