An Interaction View on Information Visualization

Robert Kosara Helwig Hauser

VRVis Research Center, Vienna, Austria
http://www.VRVis.at/vis/
Kosara@VRVis.at, Hauser@VRVis.at

Donna L. Gresh

IBM Thomas J. Watson Research Center
http://www.research.ibm.com/people/g/donnagresh
gresh@us.ibm.com

Abstract

Information Visualization (InfoVis) encompasses techniques of visualization that deal primarily with abstract data, that is, data for which the user has no preconceived mental model. This is in contrast to, for example, volume or flow data.

For this reason, interaction is particularly important in InfoVis: for exploration, analysis, and presentation of data. Interaction allows the user to implicitly form mental models of the correlations and relationships in the data, through recognition of patterns, marking or focusing in on those patterns, forming mental hypotheses and testing them, and so on. Some interaction techniques are very specific to InfoVis (even though they can be and are applied to other areas as well), such as Focus+Context and Linking+Brushing.

This paper surveys InfoVis techniques with an orientation toward interaction aspects, rather than data model or display dimension. It also tries to put the work into perspective by including aspects such as user studies for the evaluation of methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Visualization

1. Introduction

To visualize a set of data is to produce an image from it, with the goal of elucidating the character of the data and deriving insight. The challenge in visualization is to find a visual metaphor that the user can understand and perceive effectively, and that conveys the information the user is looking for – and to provide interaction methods that make it possible for the user to work with and probe the data as effectively and effortlessly as possible.

1.1. What is Information Visualization?

Information Visualization (InfoVis) deals with data that is usually abstract, high-dimensional, and structured in a complex way. Visualization of such data is difficult because the user does not have any preconception of how such data could look, so no “natural” display is possible (in contrast to volume visualization, where, for example, isosurfaces at a constant data value can pull out representative “objects" such as the core of a thunderstorm or the region of bone in the human body). In many domains, there are also physical methods for making data visible (*e.g.*, in flow visualization by injecting dye or smoke into a flow), which lead to natural analogies in visualizing the data (such as arrows or streamlines), which is typically not the case in InfoVis.

Data in scientific visualization (which subsumes volume and flow visualization) also usually contains objects or structures that have a certain minimal size and that have certain features to look for. This is not necessarily the case in InfoVis, where the character of the relationships to be found

are not at all obvious *a priori*. Thus methods must be able to deal with data that appears to be random, but still contains valuable information.

1.2. Structure of this Survey

Section 2 presents some thoughts on the nature of interaction and its importance in visualization. We follow this by descriptions of interaction methods that are used in InfoVis, such as F+C (Section 3) and multiple views (Section 4). Visualization methods for specific data are discussed in the following sections, such as high-dimensional data (Section 5), trees (Section 6), and temporal data (Section 7). Section 8 deals with user studies that have been conducted as part of InfoVis research. A (very subjective) itemization of what the future of InfoVis might hold (Section 9) rounds off this survey.

2. Interaction

Interaction in InfoVis has many facets, but it generally serves one goal: The user is able to understand the information better (or at all!), if the image is not simply static, but rather he or she can interact with it.

Many interaction methods are simple. Linking and Brushing (Section 4.2) or the use of multiple views (Section 4) might even seem almost trivial. They do, however, provide tremendous power to the user, and it is often much more difficult than it seems to implement them in a way that the user can actually benefit from them.

2.1. Taxonomies of Interaction

InfoVis so far has resisted attempts to put it into one coherent and complete taxonomy. This is perhaps partly due to the fact that it is still a very young science, which will evolve and be better understood with time.

The tendency has been to think of visualization methods in terms of the type of data to which they are applied; for example, new methods are often introduced as methods "for" a certain data type. Trying to put methods into a structure that is more oriented towards how the user works with the data is much more difficult.

The classic taxonomy[71] contained a mix of both tasks and datatypes, by offering a task-by-datatype taxonomy. It consists of seven data types and seven tasks – these seven tasks (Overview, Zoom, Filter, Details-on-Demand, Relate, History, and Extract) summarize and organize the important sorts of interactions users wish to have with their data.

Another work[35] also focussed on the tasks that users wish to perform during visualization of information, organizing them in a higher-level way (e.g. prepare, plan, explore, present, overlay, re-orient). The subtasks identified, however, fit the sorts of activities outlined in the classic taxonomy[71].

Another work[15] described a framework for the variety of possible operations within the overall visualization task, and showed how a number of visualization environments could be characterized by which of these operations they provided.

3. Focus+Context Visualization

One of the most common interaction techniques in InfoVis is Focus+Context (F+C) visualization. Usually, the amount of data to be displayed is too large to fit on the screen completely. It is also useful to be able to zoom in on certain parts of the data. In such cases, one wants to focus on certain data, without losing track of where he or she is in the whole data set.

We differentiate between four groups of F+C techniques:

Distortion-oriented. The best known group are techniques that geometrically distort the image, so that more space is available for the more important parts.

Overview Methods. Context can also be shown separately, in a different window or another layer in the same space. There is no seamless integration of overview and focus.

Filtering. Instead of providing more space, filtering techniques provide more or different information in certain parts of the image, without overloading it.

In-Place Techniques. Pointing out information to the user (instead of changing the amount of information) is also possible. This way, the answer to queries can be seen quickly, and interaction can be done in ways that are not directly visual.

These techniques are presented in the following subsections.

3.1. Distortion-Oriented F+C Methods

Perhaps the best metaphor for distortion-oriented F+C methods is the stretchable rubber sheet: Imagine an image printed onto a thin piece of rubber that is mounted onto a frame. By dragging on parts of it, you can deform it, and thus distort the image to provide more or less room for its different parts. The total area of the image (limited by the frame) remains the same, however.

3.1.1. Classical geometric distortion

The classic method for geometric F+C visualization is that of fisheye views[28]. These use a concept from photography, namely that of the fisheye lens, which has a different magnification in different parts of the image. By applying a similar distortion to the visualized data, more space is provided for important information. The degree of the distortion, as well as the center of interest can be changed.

A different way of distorting an image is by drawing in hyperbolic space[49, 50], and projecting it to (Euclidean) screen

space. This generates a circular display with higher magnification in the center and lower magnification close to the border. This idea can also be extended to 3D hyperbolic space, which has been used for the layout of graphs[61] and trees[62].

An effectively similar approach is to project the information onto a sphere[48], which also makes it possible to easily move the focus out of the center of the display. Also, concentric rings are drawn as an indication of the distortion, which decreases confusion. Another advantage of this method is that it can show the focus with very little distortion, which is very useful when working with text, for example (see below).

Fisheye views have been extended and generalized[39]. The extension not only provides means to do the distortion on a different level than just geometrically (*e.g.*, by replacing images with icons when they are in the context), and to compose the image of different versions of the undistorted image (*e.g.*, to provide color discrimination in addition to the distortion). The problem of the distorted focus area is also addressed by using a distortion function that has a "flat" area around the point of maximum interest.

Fisheye and related distortions have two drawbacks. They distort the whole image, even the region of interest, and the distortion is not well supported by graphics hardware.

The perspective wall[55] uses an approach that consists of three rectangular parts: the central, most magnified one, and two adjacent parts on the sides with decreasing magnification. All the parts are flat, and therefore can be easily implemented using very simple geometry and texture mapping. The central part is also completely undistorted – which is especially important when text is displayed.

The document lens[67] for the display of large amounts of text (in a way similar to microfiche) therefore uses a very similar approach, with context parts also extending on the top and bottom of the focus area. In addition to being able to read the text, the user is also given the opportunity to make use of his or her mental image of the pages surrounding the one of interest. This way, navigation through large amounts of text (*e.g.*, newspaper archives) is a lot easier than when using conventional microfiche readers.

A (now somewhat dated) taxonomy[51] of some of the key techniques in distortion-oriented F+C visualization is available.

3.1.2. Generalized Distortion

The idea of geometric distortion can be taken further to be applied to information that cannot be distorted in the usual sense (without losing readability), such as tables of data.

The table lens[65] shows the entries in every field not as a number, but as a bar that represents the size of the number This way, many more values can be displayed then when each value must be displayed as a number and be readable. The list can be sorted by any of the rows, making it possible for the user to get an overview of the data and to see correlations between rows. The symbolic representation also allows the user to judge the distribution of values much quicker than when having to read numbers. Further work[81] increased the number of values that could be presented in the same screen space while retaining enough precision to be able to get a good overview (even when there are more values than pixels).

INSYDER[41] extends the ideas of the table lens with a combination of different views, including scatterplots, thumbnails of text documents, and the so-called SuperTable (which is very similar to the table lens).

The LensBar[57] was designed for working with long lists of text, like dictionaries, emails, or program code. There is a scrollbar next to the display of the actual values, that also shows where data is along the range of values, depending on the current degree of interest function. The user can then navigate between these areas of interest and perhaps change them or find what he or she is looking for.

A similar idea has also been applied in HCI, in the case of fisheye menus[9], which provide easier navigation in long lists of menus without the need for scrollbars. It can be argued, however, that a hierarchical organization might be a better way to organize the information.

A very different visualization is done for the semantic and link relations between documents. Using a concept of "neighborhoods," CardVis[60] shows very related documents on one card in a stack, which the user can browse through. Some information on the other cards remains visible while the user looks at a specific card. Thus, navigation between them is easy.

3.2. Overview Methods

Focus and context can be shown separately from one other, without a transition between them. In the information mural[38], a smaller window shows the structure of the whole dataset, while the focus is given more screen space. A small frame in the overview shows the location and size of the focus region ("you are here"). Because a simple downsampling of the data can lead to overview images that are hard or impossible to read, special care is taken to do the downsampling so that it retains as much information as possible.

Many of the applications of visualization to the problem of software development also use this type of F+C. A simple, straight-forward (and, by InfoVis standards, extremely natural) visualization of program code is to represent each code line with a simple line that can be color-coded. This makes it possible to display a lot of code (about 50,000 lines) on a standard screen, and retain quite a lot of the structure of the code (indentation, line length, blank lines) to be able to recognize structures in it, and the relative sizes of files are also

quite easily visible. A F+C display can also easily be created by providing a "loupe" that is moved over the lines, with the corresponding code shown in readable size in a separate window or display area. Most methods use this idea together with varying interactions and parameters to change the color coding.

Programs such as Seesoft[26, 3] can use code age, number of changes, changes between versions, etc. for the color coding.

A similar approach is followed for assisting program testing[25]. Each line is color-coded by the number of failed tests it was executed in. The more faulty test-cases a piece of code is a part of, the likelier it is that it contains a bug.

In Sunburst, a radial display of tree data[77], the levels of the tree are drawn onto concentric circles, with the root of the tree being in the center. The farther away a node is from the root (and thus usually the more nodes are on that level), the larger the circle it is located on. Additional space for detail and context is made by shrinking the depiction of the whole tree, and only showing the wedge of the circular structure that is currently of interest. Where that part is located on the whole circle is indicated in the overview image.

The overview can also be displayed in the same space as the focus, as in the Macroscope[52]. To differentiate between them, the overview is displayed with a smaller resolution (but the same size as the detail), leading to blockier images. The user can then look back and forth between the zoom levels by looking at the cruder or the finer parts of the image. The differentiation between focus and context is not easy with this technique, however.

3.3. Filtering

Instead of distorting an image, additional information can be shown in certain parts. This is the idea behind magic lenses[12]. They provide an object (which can be of any shape) that can be move over the display and which shows different information in the area that it covers. This can be information about the font used in a word processor, pseudo-color to show an additional data dimension, or annotations attached to different objects.

The original magic lens already worked with a 2D lens in a 3D environment, but an extension to 3D (with a three-dimensional "lens") has also been done[85].

3.4. In-Place F+C

In-place F+C is similar to filtering, but it does not require a lens to be moved. Instead, the program decides which data to show and which to hide, which is of course highly dependent on the user task. But in this way, the program can also take the initiative and point the user to interesting data.

GeoSpace[54] uses color, transparency and font size to point out objects in a GIS visualization. This way, hospitals, pharmacies, or areas with high crime rates can be quickly seen.

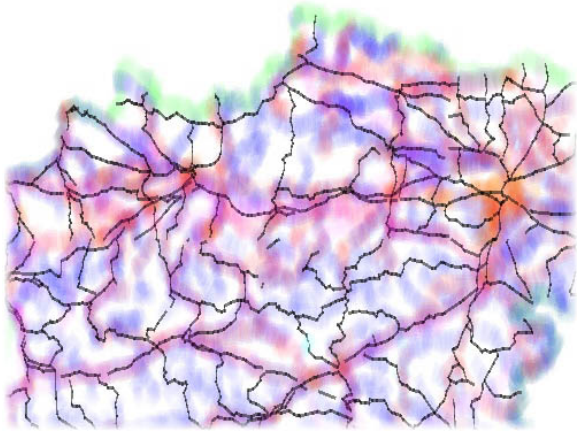

Figure 1: *SDOF[45] in use with a map viewer: The display is focused on streets, with rivers, rail tracks, etc. still being visible as context. The user can easily switch between the layers.*

Semantic Depth of Field (SDOF[45, 46]) is another technique for in-place F+C. It uses the depth-of-field effect from photography to blur objects depending on their distance from the camera. SDOF uses objects' relevance instead of their location to assign blur, and thus directs the user's attention very effectively (Figure 1). The assignment of relevance is done based on other interactions, *e.g.*, telling the program to show files that are larger than one megabyte and older than six months.

The tree browser Cheops[6] also uses in-place F+C, by "stacking" nodes on top of each other, and showing only those that are relevant for the current path. All other nodes are shown as "ghosts" that do not need much space, but that still give the user an idea of the size of the remaining tree.

Another way to provide F+C in-place is a level-of-detail (LOD) approach. In a visualization of real-time process data[58], measurements are shown, where each data value can be measured redundantly by several devices. The data displays are drawn in different ways depending on their current relevance. The user can move the point of interest over the display and thus influence the amount of space each instrument gets. The lowest level of detail is a single "lamp" that shows green or red, depending on a definition of normal values. The second level is a simple numerical display, the third level shows an instrument that displays several values at the same time. The highest level of detail, finally, shows a separate instrument for each data value.

The hardware version of in-place F+C is the Focus Plus Context screen[5]. It consists of a large canvas onto which the

image is projected, and a smaller LCD screen in its middle. The display is integrated so that the LCD area looks almost identical to the projected image, except for the higher resolution. The user can then move parts in and out of the focus area, depending on the current task. In addition to the F+C effect, he or she also gets a very large workspace, compared to current standard monitors.

3.5. Visualization Schemas

In addition to providing interaction to the user, it is useful to have both a formalism for describing interaction, and to be able to save the state of an application after interaction has taken place.

For the Polaris[79] system, a visual formalism[78] was presented that encodes which dimensions are displayed, visual encodings for different dimensions, the display type (map, scatterplot, line plot, etc.), zoom level, etc.

A similar approach is the feature definition language (FDL[22]), which allows the user to specify features (which are arbitrary logical combinations of selections) that are brushed in the visualization. In this way, an analysis of a data set can be applied to another, similar data set, which saves the user time and repetitive work.

4. Multiple Views

Given today's graphical user interfaces, using several views on the same data instead of just one is trivial. Doing it right, and providing the necessary amount of interaction, however, is not[2].

4.1. Scatterplots and Relatives

Scatterplots show data by drawing a point (or glyph) for each data point. The plane onto which they are drawn has two axes selected from the axes of the data space (thus creating a two-dimensional projection of the *n*-dimensional data space). A mapping of data values to coordinates is also necessary, which usually enables the user to select the part of the data that is drawn, to zoom, pan, etc. Other visual features such as color, size, etc. can also be controlled by other selected data dimensions.

Scatterplots are extremely useful because they provide a good overview of the data, and also show correlations in the data well. They are also one of the better known methods in information visualization (familiar to many people outside the field) and very easy to implement. Applications of scatterplots are rarely published, but the few examples[30] show what can be done by intelligently applying a simple method. But scatterplots only show two data dimensions at a time, and there are also a number of other problems (*e.g.* when many data points are drawn at the same pixel location, the true aggregate nature of the data may be masked by whichever point happens to be drawn on top).

Scatterplot matrices[16] show all two-dimensional projections of a high-dimensional dataset in a grid. In each row, the y axis is the same for all plots, and in each column, the diagrams share an x axis. The projections along the diagonal are used to show a histogram of the projection of the data onto one axis. A system similar to scatterplot matrices, but used to visualize high-dimensional scalar functions instead of data points, is Hyperslice[84].

Hypercell[24] is an extension of Hyperslice into 3D, which makes it possible to see more information at the same time. It uses volume rendering to give the user an impression of the volume without occluding much of the data. In contrast to scatterplot matrices and hyperslice, the user cannot see the entire matrix of projections, but has to select certain views, which he or she can then switch between.

A natural extension of scatterplots is the use of 3D scatterplots[7]. These are not widely used, however, because they can be difficult to interpret, and because the technology used has not allowed for a large number of data points nor easy interaction with the display.

4.2. Linking and Brushing

Multiple views are of little use without being able to interact with them. The user can select points to be investigated further with a technique called brushing. Brushing is usually done directly on the visible structures, by interacting with the view using the mouse (*e.g.*, opening a rectangular region of interest). But it can also be done independently of the display, by entering values, selecting regions of interest with sliders, or by more complex means (such as selecting a cluster, the points inside which are brushed). This makes it possible to brush in dimensions that are not currently shown.

Brushes can also be combined to form composite brushes[87], thus enabling the user to select more complex shapes and to specify very precisely which points he or she is interested in.

When brushing is used, a scalar value called the degree-of-interest (DOI) is associated with each data point. Typically, a value of 1 means that the point is brushed, and a value of 0 means that it is not. Smooth brushing[56] allows the user to specify brushes without the sharp edges a binary classification would create. Such brushes are harder to composite, but provide more flexibility to the user, who might not want (or be able to) specify a sharp boundary in a domain.

Brushing alone is of limited use, however, without view linking. Linked views exchange information about which points are brushed, so the user can easily see the same points brushed in different views on the data. Linking is usually implemented so that it works from any view to all other views, in contrast to slaving (which is one-way). In addition to linkage of brushing information, one can also link other viewing parameters such rotation, zoom, etc. In this way, the user is

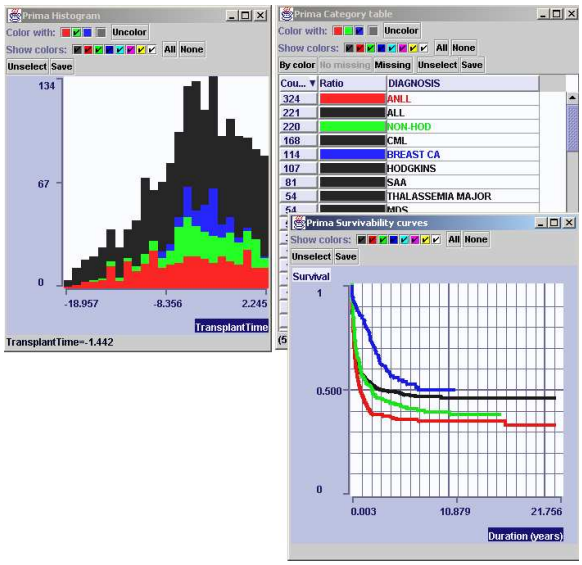

Figure 2: *Multiple window view of medical data. Color is used to distinguish different diagnoses, with color shared across all realizations of the data. Survival curves show the different prognoses for the different diagnoses, while a histogram shows the number of cases of each diagnosis treated over time.*[30]

provided with a consistent view of the data, which can be useful if several similar views are present.

Because Linking and Brushing are usually implemented at the same time, Linking&Brushing (L&B) has become a fixed expression.

4.3. Other Data Realizations

When multiple views of the data are presented to the user, it is common to mix a variety of visual representations. For example scatterplots of numerical variables may be shown at the same time with views more appropriate to categorical variables such as categorical lists or pie charts. Histograms can be used to select, for example, outliers in a particular variable. Several data visualization systems are built upon this concept, *e.g.* Opal[30], illustrated in Figure 2. Multiple, linked views effectively present multidimensional data in a way in which users can find correlations across multiple dimensions. For example, a cluster noted in a two-dimensional scatterplot might be found to correlate to only one or two variables in a categorical string variable, for example. The advantage of multiple views is that presentations appropriate to each sort of data variable may be independently chosen. Of course, interactive linkage of data points is critical to the effectiveness of this method.

4.4. Scientific Visualization and InfoVis

As a special case of multiple views, we describe some work which has sought to bridge the worlds of scientific and information visualization. For example, WEAVE[31] is a system that uses multiple information visualization views such as histograms, parallel coordinates, and scatterplots, along with traditional scientific visualization representations such as surfaces and streamlines. The specific application was a complex simulation of the heart. Brushing and linkage in all views was supported, so that, for example, one could understand the spatial dependence of points with both high voltage and low current, or alternatively, determine the relationship between calcium and potassium in a particular region of the heart.

InfoVis (or at least simple graphical displays) have always been used to support scientific visualization, *e.g.*, for transfer function design in a real-time volume rendering library[59].

More elaborate visualizations have been implemented in scientific visualizations since. A panel for the definition of three-dimensional transfer functions for volume rendering[42] shows a histogram display of the data and two derivatives, so that the user can place the transfer function at boundaries between different materials (which are directly visible).

Similarly, InfoVis also supports studies of high-dimensional flow simulation data[23, 22]. Such data typically consists of many dimensions (such as pressure, temperature, turbulent kinetic energy, etc.), which the engineers are interested in. By providing views on the spatial data, as well as the non-spatial parameter space, it is possible to find out, for example, where in an object the temperatures are too high, and the flow is too slow. Users can, for example, notice interesting patterns in a subset of the parameter space (which is displayed in a scatterplot, for example), and then use brushing and linkage to find out where in the object this configuration is present. From there, they can investigate the parameter values in adjacent regions, for example.

5. Interaction with Dimensions

Moving beyong direct interactions on the data point level, it is possible to gain new insights into data by being able to change the way dimensions are depicted, and also to filter the data prior to display.

5.1. Methods beyond Scatterplots

Prosection views[29] and the prosection matrix[76] are an improvement over standard scatterplots and scatterplot matrices, which allows the user not only to see projections, but the points that actually get projected can be selected in every projection (which then affect all other projections). This also gives this method its name, which is a combination of "projection" and "section."

The reorderable matrix is an idea that was originally used

with cardboard cards[11], and was later implemented in computer programs[73]. It shows multidimensional data as filled circles, one for each attribute of each data point, with the size corresponding to the attribute value. Users can sort the objects by any attribute, and move objects and attributes around. In this way, correlations between attributes (typically between two attributes) can be found quite easily and naturally.

5.2. Rearranging Dimensions

In 2D or 3D visual space, only two or three axes, respectively, can be drawn perpendicular to each other. A number of techniques display more dimensions by working around this limitation.

The best-known method in this group is that of parallel coordinates[37], which draws the axes parallel to each other. Each point in *n*-space is represented by one value on each axis, which are connected with lines so that the values for one point can be seen.

The main interaction with parallel coordinates is brushing on one or more axes. By selecting a value range on one axis, the user can see possible correlations between this axis and other ones.

In addition to brushing, other possible interaction mechanisms are reordering axes (so that axes that the user wants to compare are directly next to each other), scaling, panning (moving the origin of the axis), and changing the orientation of axes.

Parallel coordinates have been extended and used in many areas. In addition to the data itself, other information can be shown[74]. Quartiles of the data along each axis can be displayed, as well as the average of a selection (called *polyline averaging*), and the correlation between areas on different axes that the user selects.

A different work[32] provides an additional brushing method, a technique called angular brushing, which allows the user to select points based on an angle range between two axes, instead of their value on a single axis (Figure 3). This makes it possible to directly brush points that have a certain correlation to each other. Additionally, histograms can optionally be drawn on the axes instead of quartiles.

By applying a color scheme based on proximity, the display of hierarchical, high-dimensional data is also possible[27] with parallel coordinates, as well as allowing movement through this hierarchy to see the data at different levels of magnification.

Bargrams[88] are more an interaction technique than a visualization, but follow a concept somewhat similar to parallel coordinates. A bargram is a histogram where the relative number of objects in each bin is shown by a horizontal bar, with all bars next to each other on one line (empty bins are not shown). Small icons are drawn above these bargrams, so

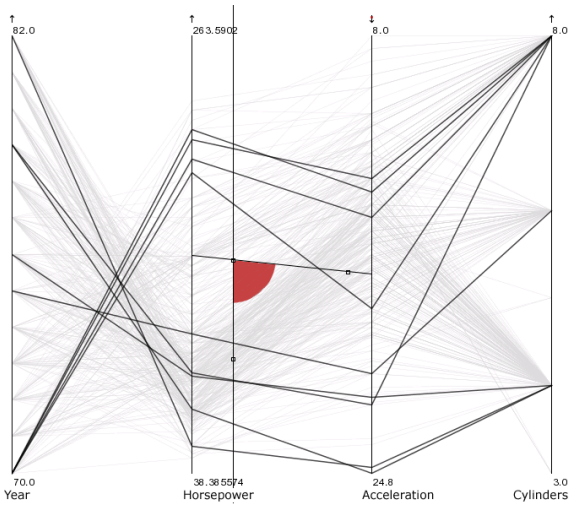

Figure 3: *Angular brushing in Parallel Coordinates*[32]. *All data points (represented by polylines) whose angle between the two axes of interest lies within the angle range specified by the red wedge are brushed.*

that when an object is brushed, it can be seen where in the data it is (the application shows icons for all objects, *e.g.*, cars, that can be selected). Objects can also be brushed by selecting their bin in the bargram, which then shows the bins on the other axes that contain objects from this bin.

Sparkler[33] is a visualization of query results from WWW search engines. Each result is represented by a point, along a line (spreading out perpendicularly to the line to avoid overplotting), the position on which is determined by its relevance. Several search results are put on a disc, with all lines starting out in its center. This creates an image that is similar to a parallel coordinate plot with the axes being drawn in like the spokes of a wheel. The structure of all the documents from a query gives the user an overview of the overall quality of the results, and can give hints whether refinement is needed. Single documents can be selected, and are then highlighted in all the result sets – making it possible to visually combine queries.

A visualization of data that has internal structure (other than the scattered, unstructured data, that parallel coordinates are typically used for) is SeeNet[8]. It is a non-typical visualization of networks, which in addition to displaying the graph in the usual way (showing its nodes and edges), also shows data about the graph, *e.g.* the data transfer volume between nodes, as a matrix. In a way, this is also similar in its intention to parallel coordinates: Overcome the problems with the standard way of looking at the data by rearranging it into a completely different shape.

6. Trees and Networks

Trees (or hierarchical data) are perhaps the data structure for which the most methods exist. This may be due to the fact that trees are, indeed, a very important concept in computer science. Another reason is likely that tree data is the easiest to come by for testing the methods (*e.g.*, the directory tree of a researcher's workstation). At the same time, trees are perhaps the data structure for which the fewest methods exist that are truly interactive.

Cone and cam trees[68] are among the best-known methods in this area. The root node of a subtree is put onto the tip of a cone, and its child nodes along the rim of its base. Leaves are displayed as rectangles with text on them. While cone trees grow from the top to the bottom of the display, cam trees grow from left to right. The latter have the advantage that the labels at their nodes can be more easily displayed (because they do not overlap).

The user can select a node in a cone tree, which is then brought to the front, so that it faces the user. This is done in a smooth animation, where all levels of the tree rotate at the same time, so that the user can easily follow the motion.

A visualization method that is similar to cam trees is that of collapsible, cylindrical trees[21]. Cylinders are drawn next to each other, with each cylinder displaying the subnodes of a node as text labels. The cylinders can be spun so that the label of interest is shown in the middle of the display, and some of the others are still visible as context above and below. The node in the middle is also expanded in the cylinders to the right, so that a complete path through the tree is displayed. This also makes the tree quite compact so that it can be displayed below a text document whose structure it represents, for example.

SpaceTree[64] is a visualization of trees that provides information about the parts of the tree that are not currently visible (by drawing icons for the subtrees of folded nodes). Animation is also used to make the transitions easier to understand when folding and unfolding subtrees. This allows the user to more effectively perform some tasks, like estimating the number of nodes under a given node. Most of the enhancements of SpaceTree can also be made quite easily for other methods, and thus more general ideas for improving tree visualizations are given.

Ordered treemap layouts[72] are based on the rather well-known (but almost totally un-interactive) treemaps[70]. Their layout algorithm tries to preserve the location of the individual objects, so that data that varies over time (such as stock market data) can be displayed in a way that makes tracing them possible, when the user moves from one point in time to another.

Another three-dimensional visualization of trees is that of Beamtrees[83], which are a modification of treemaps for three-dimensional display. The nodes of the treemap are scaled and combined, and represented by cylinders, whose depth structure (orthogonal to the layout dimensions of the treemap) represents the tree depth. The user can rotate the object and thus see the tree from different sides (Figure 4).

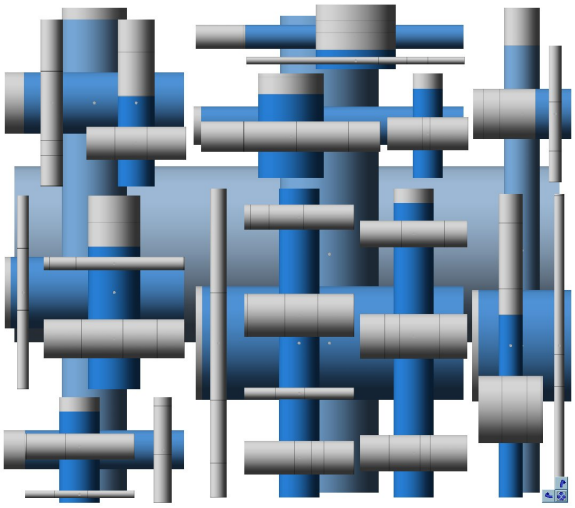

Figure 4: *An example of Beamtrees (Image courtesy of F. van Ham[83], Technische Universiteit Eindhoven)*

Radial layout of a graph[89] puts the nodes onto concentric circles. One node (the one the visualization focuses on) is displayed in the center of the circles, nodes that are connected to it with an edge are drawn on the innermost circle, nodes connected to them on the next circle, etc. This provides a direct visualization of the distance of a node from the focus node. Transitions between different focus nodes are animated by interpolating the polar coordinates of the old and new locations, thus minimizing the number of nodes that cross each other's path.

Computing clusters are a special kind of computer network, that consists of a large number of CPUs and a high-bandwidth connection, that is usually tightly woven and has a regular shape. Debugging of such hardware is supported by a visualization[20] that shows the network connections and the processor and network ports. Errors are visible on the ports and connections, and can be easily spotted and traced over time – much easier than reading the log files of the hardware.

7. Temporal Data

While time simply adds another dimension to data, it is quite a special one. In InfoVis, where most data is abstract, time adds one very concrete dimension, that is generally treated differently from all the others. The methods can be put into two groups: visualization of recorded data from the past, and visualization of plans and expected data (usually with uncertainty) in the future.

Events or data in a time series are essentially linear, which

can be a poor use of two-dimensional screen space – at least in the time dimension – when time is simply drawn as a line.

Spirals are one way for more efficient use of space. The time axis is "rolled up" into a spiral, and the data drawn along it. The data objects can also extend into 3D and can be drawn in layers above the time axis[13]. In temporal data, the user often wants to find recurring patterns, like changes with the seasons. This is possible with spirals because each rotation covers one time unit (like a year), and thus events that happen in the same part of that time unit are close together, so that eventual trends can be seen. An example for such a pattern is the accumulation of horror movies coming out shortly before Halloween (finding this requires a spiral display and the ability to filter the movie data by film type).

When the application is not to find patterns in the data using a "natural" time scaling (*e.g.*, one revolution for one year for movie release data), but to find patterns with an unknown frequency, a spiral that can be changed interactively is extremely useful and the patterns can be seen very easily[86].

While visualization is quite basic in TimeFinder[36], it allows the user to find certain temporal patterns in the data. These patterns are specified with simple rectangles which cover a certain time and value range, and which are compared against the data. All data that fit into these windows are brushed in the display. More complex shapes can be pieced together with several rectangles (the intersection of all selected data values is brushed).

8. Perception and Usability Studies

More than any other branch of computer science, visualization means communication with the user. No matter how well devised and carefully designed a visualization might be, if it does not convey the information intended, it is useless. Recently, researchers have begun to test their own designs in user studies, and to perform studies to base new methods on. A few of these studies will be surveyed here. A short introduction to user studies, together with practical tips and experiences from past studies is available[44].

8.1. Basic Perception

One feature of the human visual system that has received much attention in visualization is preattentive processing[82]. Certain features (*e.g.*, color, closure, orientation) can be perceived in a very short amount of time (about 200 ms) and without the need for serial search. People can tell where objects were located, and approximately how many there were, even after having seen an image for only a fraction of a second.

One example of the use of preattentive features are bars of different height, color and density[34], which represent different data channels in a salmon tracking system (*e.g.* plankton density, ocean current strength, surface temperature). These

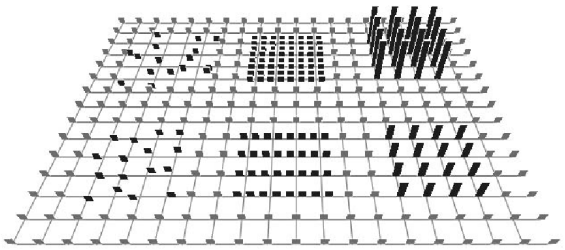

Figure 5: *Pexels representing different types of data points (Healey et al.,[34] used with permission)*

are called pexels, for "perceptual element", for an example see Figure 5.

Semantic depth of field (SDOF, Figure 1), which uses blurring to direct the users' attention to relevant objects (which are crisp), was also tested and found to be a preattentive feature[47]. This makes it a very efficient tool for pointing out information.

Another well-known effect in psychology is change blindness. After an interaction, users may not be able to tell what has changed on screen, so the interaction has to be repeated, several times perhaps. A way to conquer change blindness is by using a transition between the old and new display[63]. In addition, an "asymmetry" is created, by using a different method when fading out the old image, and fading in the new one (transparency and drawing a wireframe). This way, even for a static frame (*e.g.*, a screenshot) one can tell which objects are disappearing, and which are new.

An interesting aspect of InfoVis, that is very likely to become much more important in the future, is the use of visualization as a secondary task. When visualization is used in cars, appliances, etc., the user will concentrate on a task, and only use the visualization occasionally, when the need arises. In such a case, it is necessary to convey the information without distracting the user too much from the primary task. Visual attributes, such as color, size, closure, etc., that can be used for such a purpose were tested in user studies[14, 75].

Colormaps that are monotonically uniform in luminance are widely recommended for use as a visualization dimension. A simple method for generating such colormaps without the need to calibrate the display, but with user participation, was developed, the "Which Blair Project"[69]. A black and white photograph of a face is colored with the colormap to be tested, and presented to the user. Because humans are very sensitive to faces, they can easily tell if such a colormap is increasing in luminance or not, by judging how natural the image looks.

Based on this idea, a system for face-based colormap generation[40] was developed, which uses two binary images of a face (one the inversion of the other) to generate percep-

tually uniform colormaps on any display. The comparison of the two faces is intuitive and quick

8.2. Application-level studies

Different from low-level perception studies, application-level studies are aimed at more complex visualizations and interaction with a system. Such studies are much more difficult to do than low-level studies, but have the advantage of being directly applicable to the complete application, without the danger of other effects interfering with the desired ones. Studies can be done to compare a method to another one, or to find out how users work with a method and what could be improved.

A large user study (with 83 participants) of different visualization applications (Eureka, InfoZoom[65], Spotfire[1]) was performed[43] to compare how quickly users could solve different tasks with them. Not surprisingly, different tools were better for different types of tasks. For example, for tasks where it was necessary to ascertain the existance of a correlation between two variables, users found Spotfire's default visualization of a scatterplot very useful, while InfoZoom users had difficulty seeing correlations due to some misunderstandings of how to manage the interface. On the other hand, a task that involved finding the answer to a question about the proportion of cases with a particular attribute found that InfoZoom users obtained the result much faster than either Eureka or Spotfire users, because a simple combination of interactions leads directly to the answer, while Spotfire users first had to determine the appropriate visual represenation. Eureka users had other, different issues involving being able to backtrack easily from mistakes. The point seems to be that the more "natural" a particular task is for a particular interface, the more easily a user will perform it. Thus the issue is not that an visualization system *can* do any particular task, but whether a user will be able to quickly find and apply the appropriate tool.

An evaluation of cone trees[17] showed that while users liked cone trees (because of their appearance and interaction), they were significantly slower when navigating through cone trees than when using a more traditional 2D visualization.

Another study of different tree visualization methods[4] showed that different visualizations scored very differently for certain tasks – a result similar to the study done for SpaceTree[64].

A study of an implementation of the reorderable matrix[73] was done to discover usage patterns and possible improvements of the method. The results include some interesting ideas for new interactions (such as moving groups of rows and columns, not just single ones), as well as different strategies when comparing data rows and columns.

The effect of animation on building a mental map was investigated in another study[10], which used family trees that the users had to memorize from only seeing a zoomed-in version that they could pan. Interestingly, only the reconstruction of the whole tree was aided significantly by animation, not the answers to specific questions. The high level of dissatisfaction of users with the task also showed that not providing an overview of the whole data makes the user extremely uncomfortable.

8.3. 2D vs. 3D

The question as to whether 2D or 3D visualizations are more useful was investigated in a number of studies. While 3D theoretically provides more space and enables the user to make use of his or her spatial memory, it brings with it problems such as depth perception and occlusion.

One study[53] tested participants' capability to perceive the space around them as "Euclidian", i.e., to estimate sizes equally well in all directions, including the one pointing away from them. They were shown (physical) objects at a relatively flat angle, and had to point to the 2D shape that matched them. The result was a high error in the perceived shape. Even when a comparison object (which was the same in all cases) was presented, accuracy did not improve significantly. The results of this study very clearly show the possible problems with 3D visualization methods, which restrict what can be done in practice. This does not mean that 3D could not be used in InfoVis at all, however.

The Data Mountain[66], is a user interface for storing and retrieving documents on a tilted 2D plane. Documents can be moved in two dimensions on the plane, and thus be arranged from back to front and in groups. An initial study showed that this method was clearly superior to simple lists (like bookmarks or favorites). Another study[18] later compared this method to a version that had fewer depth cues (no tilted plane, no perspective projection). Interestingly, no significant differences were found in retrieval time between the interfaces, not even when the display got cluttered by many objects (where the authors had expected the 3D version to be clearly superior).

Similar techniques were implemented and tested against real-world interfaces made from steel frames and fishing lines, where little printouts of web pages could be placed[19]. To interact with pages in the real world, study participants pointed at them with a laser pointer, and told the experimenter where to put them. The results of this study showed a decrease in performance and subjective assessment of effectiveness from 2D to 3D. It also turned out that occlusion in 3D was less a problem than the subjects forgetting page locations in 3D; and also the gained flexibility, which led to less efficiency.

A different study[80] came to the conclusion that 3D can indeed increase performance for special tasks, in this case memorizing the structure and contents of a small tree. The 2D display was a display similar to the Windows Explorer,

while the 3D version showed the depth structure of the tree as the distance of the objects from the screen, and presented the information on vertical "cards" which did not overlap. The choice of visualization (no overlapping objects) and task make it questionable, however, if the results can be generalized.

The question of whether to use 2D or 3D is quite a crucial one, that will need many more studies to gain more insight and concrete recommendations what to use for which application.

9. The Future of Information Visualization

Even though InfoVis can still be considered to be in its infancy, with many directions being tried out and completely new directions evolving, there are a few things that we consider very likely to happen in InfoVis in the next years.

Visualizing large amounts of data will be an issue in the near future, and this will mean not only fast rendering, but also new and improved interaction to work with such vast amounts of data. Given that screen sizes will most probably not grow significantly in the next years, working with more data points than pixels on the screen will require good mechanisms to orient the user, filter data, and present it succinctly.

More interaction and putting the user more into the center of InfoVis methods (*e.g.*, by developing methods based on knowledge from psychology and testing them in user studies) will be necessary. Different visualizations will need to be integrated seamlessly, so the user can easily switch, depending on the task that needs to be done.

Also very simple interactions like undoing actions, backtracking to a certain point, etc., will be very important for InfoVis applications that are used by a large number of people. Ways of saving and loading settings for views, brushes, etc., will be important for users who are not interested in InfoVis as such, but in using it as a vehicle to work with their data. Providing such simple services will make a big difference for their everyday work with InfoVis systems.

InfoVis as a secondary task is also a very promising direction. InfoVis will only become common to many people if it is a simple but useful part of a larger system, that makes dealing with information easier. For this, methods are needed that are simple enough to be immediately understood, and that can also inform the user about data in a non-distracting way.

An open question is how to determine the most relevant uses of visualization. For example, visualization can clearly be used to see correlations between variables in data. However, statistical techniques of long standing may also be used, and can arguably find more subtle correlations, and in addition give a more rigorous measure of significance. Thus it is important to be clear on when visualization adds value, and when there are better techniques to use. It would seem that in order to be most useful, the user must be able to quickly and easily make a variety of comparisons and investigations that would otherwise be tedious or difficult to do algorithmically. Clearly, visualization is of most use when the user discovers something that he or she was not already looking for (for otherwise an algorithmic technique would be applied). Thus ease of use and flexibility is of paramount importance.

Acknowledgements

This work has been done in the scope of the basic research on visualization (`http://www.VRVis.at/vis/`) at the VRVis Research Center in Vienna, Austria (`http://www.VRVis.at/`), which is funded by an Austrian research program called Kplus.

References

1. Christopher Ahlberg and Erik Wistrand. IVEE: an Information Visualization and Exploration Environment. In N. Gershon and S. Eick, editors, *Information visualization: proceedings, October 30–31, 1995, Atlanta, Georgia, USA*, pages 66–73, 142–143. IEEE Computer Society Press, 1995.

2. Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pages 110–119. ACM Press, 2000.

3. Thomas Ball and Stephen G. Eick. Software visualization in the large. *Computer*, 29(4):33–43, April 1996.

4. Todd Barlow and Padraic Neville. A comparison of 2-D visualizations of hierarchies. In *2001 IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 131–138. IEEE Computer Society Press, 2001.

5. Patrick Baudisch, Nathaniel Good, and Paul Stewart. Focus plus context screens: combining display technology with visualization techniques. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 31–40. ACM Press, 2001.

6. Luc Beaudoin, Marc-Antoine Parent, and Louis C. Vroomen. Cheops: A compact explorer for complex hierarchies. In Roni Yagel and Gregory M. Nielson, editors, *Proceedings of the IEEE Conference on Visualization (Vis'96)*, pages 87–92, Los Alamitos, October 27–November 1 1996. IEEE.

7. Barry G. Becker. Volume rendering for relational data. In *IEEE Symposium on Information Visualization (InfoVis '97)*, pages 87–91, Washington - Brussels - Tokyo, October 1997. IEEE.

8. Richard A. Becker, Stephen G. Eick, and Allan R. Wilks. Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28, March 1995.

9. Benjamin B. Bederson. Fisheye menus. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 217–225. ACM Press, 2000.

10. Benjamin B. Bederson and Angela Boltman. Does animation help users build mental maps of spatial information? In *IEEE Symposium on Information Visualization (InfoVis 1999)*, pages 28–35. IEEE Computer Society Press, 1999.

11. Jacques Bertin, editor. *Graphics and Graphic Information Processing*. Walter de Gruyter, 1981.

12. Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. Toolglass and magic lenses: The see-through interface. *Computer Graphics*, 27(Annual Conference Series):73–80, 1993.

13. John V. Carlis and Joseph A. Konstan. Interactive visualization of serial periodic data. In *Proceedings of the 11th annual ACM symposium on User interface software and technology (UIST'98)*, pages 29–38. ACM Press, 1998.

14. C. M. Chewar, D. Scott McCrickard, Ali Ndiwalana, Chris North, Jon Pryor, and David Tessendorf. Secondary task display attributes – optimizing visualizations for cognitive task suitability and interference avoidance. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization 2002 (VisSym'02)*, pages 165–171, 2002.

15. Ed H. Chi and John T. Riedl. An operator interaction framework for visualization systems. In *IEEE Symposium on Information Visualization (InfoVis '98)*, pages 63–78, Washington - Brussels - Tokyo, October 1998. IEEE.

16. William S. Cleveland. *The Elements of Graphing Data*. Wadsworth Inc, 1985.

17. Andy Cockburn and Bruce McKenzie. An evaluation of cone trees. In *People and Computers XIV: British Computer Society Conference on Human Computer Interaction*, pages 425–436. Springer Verlag, 2000.

18. Andy Cockburn and Bruce McKenzie. 3d or not 3d?: evaluating the effect of the third dimension in a document management system. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 434–441. ACM Press, 2001.

19. Andy Cockburn and Bruce McKenzie. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments. In *Proceedings of ACM CHI'2002 Conference on Human Factors in Computing Systems*, pages 203–210. ACM Press, 20–25April 2002.

20. Patricia Crossno and Rena Haynes. Case study: Visual debugging of cluster hardware. In Thomas Ertl, Ken Joy, and Amitabh Varshney, editors, *Proceedings Visualization 2001*, pages 429–432. IEEE Computer Society Technical Committee on Visualization and Graphics Executive Committee, 2001.

21. Raimund Dachselt and Jürgen Ebert. Collapsible cylindrical trees: a fast hierarchical navigation technique. In *2001 IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 79–86. IEEE Computer Society Press, 2001.

22. Helmut Doleisch, Martin Gasser, and Helwig Hauser. Interactive feature specification for focus+context visualization of complex simulation data. In *Proceedings of the 5th Joint IEEE TCVG - EUROGRAPHICS Symposium on Visualization (VisSym 2003)*. Eurographics, 2003.

23. Helmut Doleisch and Helwig Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. In *10th International Conference in Central Europe on Computer Graphics, Visualization, and Computer Vision (WSCG 2002)*, pages 147–155, 2002.

24. Selan R. dos Santos and Ken W. Brodlie. Visualizing and investigating multidimensional functions. In *Joint Eurographics – IEEE TCVG Symposium on Visualization (VisSym 2002)*, pages 173–182. ACM Press, 2002.

25. James Eagan, Mary Jean Harrold, James A. Jones, and John Stasko. Technical note: visually encoding program test information to find faults in software. In *IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 33–36. IEEE Computer Society Press, 2001.

26. Stephen G. Eick, Joseph L. Steffen, and Eric E. Sumner Jr. Seesoft — A tool for visualizing line oriented software statistics. *IEEE Transactions on Software Engineering*, pages 957–68, November 1992.

27. Ying-Huey Fua, Matthew O. Ward, and Elke A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In David Ebert, Markus Gross, and Bernd Hamann, editors, *IEEE Visualization '99*, pages 43–50, San Francisco, 1999. IEEE.

28. George W. Furnas. Generalized fisheye views. In *Proceedings of the ACM Conference on Human Factors in Computer Systems*, SIGCHI Bulletin, pages 16–23. ACM Press, 1986.

29. George W. Furnas and Andreas Buja. Prosection views: dimensional inference through sections and projections. with a discussion by John F. Elder IV, Shingo

Oue and Daniel B. Carr and a rejoinder by the authors. *Journal of Computational and Graphical Statistics*, 3(4):323–385, December 1994.

30. Donna L. Gresh, David A. Rabenhorst, Amnon Shabo, and Shimon Slavin. Prima: A case study of using information visualization techniques for patient record analysis. In *Proceedings of the IEEE Conference on Visualization (Vis'02)*, pages 509–512. IEEE Computer Society Press, 2002.

31. Donna L. Gresh, Bernice E. Rogowitz, R. L. Winslow, D. F. Scollan, and C. K. Yung. WEAVE: A system for visually linking 3-D and statistical visualizations, applied to cardiac simulation and measurement data. In T. Ertl, B. Hamann, and A. Varshney, editors, *Proceedings IEEE Visualization 2000 (Vis'2000)*, pages 489–492. IEEE Computer Society Press, 2000.

32. Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing of extended parallel coordinates. In *IEEE Symposium on Information Visualization 2002 (InfoVis 2002)*, pages 127–130. IEEE Computer Society Press, 2002.

33. Susan Havre, Elizabeth Hetzler, Ken Perrine, Elizabeth Jurrus, and Nancy Miller. Interactive visualization of multiple query results. In *IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 105–112. IEEE Computer Society Press, 2001.

34. Christopher G. Healey and James T. Enns. Large datasets at a glance: Combining textures and colors in scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):145–167, April 1999.

35. Stacie L. Hibino. Task analysis for information visualization. In *Visual Information and Information Systems (VISUAL'99)*, pages 139–146, 1999.

36. Harry Hochheiser and Ben Shneiderman. Visual specification of queries for finding patterns in time-series data. In *Proceedings of Discovery Science 2001*, pages 441–446. Springer, 2001.

37. Alfred Inselberg and Bernard Dimsdale. Parallel coordinates for visualizing multi-dimensional geometry. In Tsiyasu L. Kunii, editor, *Computer Graphics 1987 (Proceedings of CG International '87)*, pages 25–44. Springer-Verlag, 1987.

38. Dean F. Jerding and John T. Stasko. The information mural: A technique for displaying and navigating large information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3):257–271, July/September 1998.

39. T. Alan Keahey. The generalized detail-in-context problem. In *Proceedings IEEE Symposium on Information Visualization 1998*, pages 44–51. IEEE Computer Society Press, 1998.

40. Gordon Kindlmann, Erik Reinhard, and Sarah Creem. Face-based luminance matching for perceptual colormap generation. In *Proceedings IEEE Visualization 2002 (Vis'02)*, pages 299–306. IEEE Computer Society Press, 2002.

41. Peter Klein, Frank Müller, Harald Reiterer, and Maximilian Eibl. Visual information retrieval with the supertable + scatterplot. In *Sixth International Conference on Information Visualisation (IV'02)*, pages 70–75. IEEE Computer Society Press, 2002.

42. Joe Kniss, Gordon Kindlmann, and Charles Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In Thomas Ertl, Ken Joy, and Amitabh Varshney, editors, *Proceedings IEEE Visualization 2001 (Vis'01)*, pages 255–262. IEEE Computer Society Press, 2001.

43. Alfred Kobsa. An empirical comparison of three commercial information visualization systems. In *2001 IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 123–130. IEEE Computer Society Press, 2001.

44. Robert Kosara, Christopher G. Healey, Victoria Interrante, David H. Laidlaw, and Colin Ware. Thoughts on user studies: Why, how, and when. *IEEE Computer Graphics & Applications (CG&A), Visualization Viewpoints*, 23(4), July/August 2003.

45. Robert Kosara, Silvia Miksch, and Helwig Hauser. Semantic depth of field. In *IEEE Symposium on Information Visualization 2001 (InfoVis 2001)*, pages 97–104. IEEE Computer Society Press, 2001.

46. Robert Kosara, Silvia Miksch, and Helwig Hauser. Focus and context taken literally. *IEEE Computer Graphics & Applications, Special Issue on Information Visualization*, 22(1):22–29, January/February 2002.

47. Robert Kosara, Silvia Miksch, Helwig Hauser, Johann Schrammel, Verena Giller, and Manfred Tscheligi. Useful properties of semantic depth of field for better f+c visualization. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization 2002 (VisSym'02)*, pages 205–210, 2002.

48. Matthias Kreuseler, Norma López, and Heidrun Schumann. A scalable framework for information visualization. In *IEEE Symposium on Information Visualization 2000*, Salt Lake City, UT, USA, October 8–13, 2000. IEEE.

49. John Lamping and Ramana Rao. Laying out and visualizing large trees using a hyperbolic space. In *Proceedings of the ACM Symposium on User Interface Software*

and Technology (UIST'94), pages 13–14. ACM Press, 1994.

50. John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings Conference on Human Factors in Computing Systems (CHI'95)*. ACM, 1995.

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

52. Henry Lieberman. Powers of ten thousand: Navigating in large information spaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology '94*, Visualization I, pages 15–16, 1994. TechNote.

53. Mats Lind, Geoffrey P. Bingham, and Camilla Forsell. The illusion of perceived metric 3d structure. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, pages 51–56. IEEE CS Press, 2002.

54. Ishantha Lokuge and Suguru Ishizaki. Geospace: An interactive visualization system for exploring complex information spaces. In *CHI '95 Proceedings*, 1995.

55. Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The perspective wall: Detail and context smoothly integrated. In *Proceedings of ACM CHI '91 Conference on Human Factors in Computing Systems*, pages 173–179, 1991.

56. Allen R. Martin and Matthew O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Proceedings IEEE Visualization (Vis'95)*, pages 271–278. IEEE Computer Society Press, 1995.

57. Toshiyuki Masui. Lensbar - visualization for browsing and filtering large lists of data. In *Proceedings IEEE Symposium on Information Visualization 1998*, pages 113–120. IEEE, 1998.

58. Kresimir Matkovic, Helwig Hauser, Reinhard Seinitzer, and Eduard Gröller. Process visualization with levels of detail. In *IEEE Symposium on Information Visualization 20002 (InfoVis 2002)*, pages 67–70. IEEE Computer Society Press, 2002.

59. Lukas Mroz and Helwig Hauser. RTVR - a flexible java library for interactive volume rendering. In *IEEE Visualization '01 (VIS '01)*, pages 279–286. IEEE, 2001.

60. Sougata Mukherjea and Yoshinori Hara. Visualizing world-wide web search engine results. In *1999 International Conference on Information Visualisation (IV'99)*, pages 400–407. IEEE Computer Society Press, 1999.

61. Tamara Munzner. H3: Laying out large directed graphs in 3D hyperbolic space. In *IEEE Symposium on Information Visualization (InfoVis 1997*, pages 2–10, 1997.

62. Tamara Munzner and Paul Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *Proceedings of the 1995 symposium on Virtual reality modeling language (VRML'95)*, pages 33–38. ACM Press, 1995.

63. Lucy Nowell, Elizabeth Hetzler, and Ted Tanasse. Change blindness in information visualization: a case study. In *IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 15–22. IEEE Computer Society Press, 2001.

64. Catherine Plaisant, Jesse Grosjean, and Benjamin B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *IEEE Symposium on Information Visualization (InfoVis 2002)*, pages 57–64. IEEE Computer Society Press, 2002.

65. Ramana Rao and Stuart K. Card. The table lens: Merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of ACM CHI '94 Conference on Human Factors in Computing Systems*, pages 318–322, 1994. Color plates on pages 481-482.

66. George Robertson, Mary Czerwinski, Kevin Larson, Daniel C. Robbins, David Thiel, and Maarten van Dantzich. Data mountain: using spatial memory for document management. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, pages 153–162. ACM Press, 1998.

67. George G. Robertson and Jock D. Mackinlay. The document lens. In *Proceedings of the ACM Symposium on User Interface Software and Technology '93*, Visualizing Information, pages 101–108, 1993.

68. George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: Animated 3D visualizations of hierarchical information. In *Proceedings of ACM CHI '91 Conference on Human Factors in Computing Systems*, pages 189–194, 1991.

69. Bernice E. Rogowitz and Alan D. Kalvin. The "which blair project": A quick visual method for evaluating perceptual color maps. In Thomas Ertl, Ken Joy, and Amitabh Varshney, editors, *Proceedings Visualization 2001*, pages 183–190. IEEE Computer Society Technical Committee on Visualization and Graphics Executive Committee, 2001.

70. Ben Shneiderman. Tree visualization with treemaps: a 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, January 1992.

71. Ben Shneiderman. The eyes have it: A task by data

type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 336–343. IEEE Computer Society Press, 1996.

72. Ben Shneiderman and Martin Wattenberg. Ordered treemap layouts. In *IEEE Symposium on Information Visualization 2001 (InfoVis 2001)*, pages 73–78. IEEE Computer Society Press, 2001.

73. Harri Siirtola. Interaction with the reorderable matrix. In *International Conference on Information Visualization (IV '99)*, pages 272–279, Washington - Brussels - Tokyo, July 1999. IEEE Computer Society Press.

74. Harri Siirtola. Direct manipulation of parallel coordinates. In *International Conference on Information Visualisation (IV2000)*. IEEE Computer Society Press, 2000.

75. Jacob Somervell, D. Scott McCrickard, Chris North, and Maulik Shukla. An evaluation of information visualization in attention-limited environments. In *Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization 2002 (VisSym'02)*, pages 211–216, 2002.

76. Bob Spence, Lisa Tweedie, Huw Dawkes, and Hua Su. Visualization for functional design. In *Proceedings IEEE Symposium on Information Visualization (InfoVis 1995)*, pages 4–10. IEEE Computer Society Press, 1995.

77. John Stasko and Eugene Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *IEEE Symposium on Information Visualization (InfoVis 2000)*, pages 57–68. IEEE Computer Society Press, 2000.

78. Chris Stolte, Diane Tang, and Pat Hanrahan. Multi-scale visualizations using data cubes. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, pages 7–14. IEEE Computer Society Press, 2002.

79. Chris Stolte, Diane Tang, and Pat Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, January 2002.

80. Monica Tavanti and Mats Lind. 2D vs 3D, implications on spatial memory. In *IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 139–148. IEEE Computer Society Press, 2001.

81. Tichomir Tenev and Ramana Rao. Managing multiple focal levels in table lens. In *IEEE Symposium on Information Visualization (InfoVis '97)*, pages 59–64. IEEE Computer Society Press, 1997.

82. Anne Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31:156–177, 1985.

83. Frank van Ham and Jarke J. van Wijk. Beamtrees: Compact visualization of large hierarchies. In *Proceedings IEEE Symposium on Information Visualization (InfoVis'02)*, pages 93–100. IEEE Computer Society Press, 2002.

84. Jarke J. van Wijk and Robert van Liere. HyperSlice - visualization of scalar functions of many variables. In *Proceedings of IEEE Visualization 1993 (Vis'93)*, pages 119–125. IEEE Computer Society Press, 1993.

85. John Viega, Matthew J. Conway, George Williams, and Randy Pausch. 3d magic lenses. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 51–58. ACM Press, 1996.

86. Marc Weber, Marc Alexa, and Wolfgang Müller. Visualizing time-series on spirals. In *2001 IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 7–14. IEEE Computer Society Pres, 2001.

87. Graham J. Wills. Selection: 524,288 ways to say "this is interesting". In *Proceedings of the 1996 IEEE Symposium on Information Visualization (InfoVis'96)*, pages 54–61. IEEE, 1996.

88. Kent Wittenburg, Tom Lanning, Michael Heinrichs, and Michael Stanton. Parallel bargrams for consumer-based information exploration and choice. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 51–60. ACM Press, 2001.

89. Ka-Ping Yee, Danyel Fisher, Rachna Dhamija, and Marti Hearst. Animated exploration of dynamic graphs with radial layout. In *2001 IEEE Symposium on Information Visualization (InfoVis 2001)*, pages 43–50. IEEE Computer Society Press, 2001.