Guidelines for Designing Information Visualization Applications

David A. Carr
Institutionen för Datavetenskap
Linköpings Universitet
S-581 83 Linköping, Sweden
+46 13 28 24 25
davca@ida.liu.se

ABSTRACT

Information overload is considered one of the fundamental human-computer interaction problems today. The Internet provides access to millions of documents. Closer to home, average desktop users may have thousands of files and documents stored on their systems. Users may deal with budgetary information distributed amongst hundreds of categories or access databases of enormous size. The computer can, in a few seconds, retrieve information that humans would take years to digest. Often, much of this information is irrelevant. In an effort to cope with this problem, more and more users are turning to information visualization. This paper surveys the area of Information Visualization. It discusses visualization designs and techniques from a prospective based on Shneiderman's abstract user-task taxonomy for information visualization. It then presents some design guidelines that should be considered when designing information seeking applications based on visualization techniques. It also observes that little scientific study has been performed on the usability of information visualization and calls for study in realistic work situations.

Keywords

Information visualization, guidelines, user tasks, data types.

INTRODUCTION

Information overload is considered one of the fundamental human-computer interaction problems today. The Internet provides access to millions of documents. Closer to home, average desktop users may have thousands of files and documents stored on their systems. They may deal with budgetary information distributed amongst hundreds of categories or access databases of enormous size. The computer can, in a few seconds, retrieve information that humans would take years to digest. Often, much of this information is irrelevant. In an effort to cope with this problem, more and more users are turning to information visualization. By graphically displaying the results of an information search, they hope to be able to understand the vast amounts of data. Unfortunately, this all too often ends in disappointment. This does not mean that visualization is not the solution. However, quite often the visualization tool does not fit the user's needs.

Information visualization is the presentation of abstract data in a graphical form so that the user may use his visual perception to evaluate and analyze the data. It differs from scientific visualization in that the data does not involve natural phenomena. In many instances, this link to nature or the physical world suggests a way to organize and present the information that is immediately recognized by the user – a map of population density for example. Information visualization is characterized by the need for the designer to invent a way to transform the data into graphical representation. This representation must express the important properties of the data and express how different items are related to one another. Sometimes, this is relatively straightforward. For example, a governmental budget can be represented as a pie diagram. Sometimes this is not easy. For example, displaying the interconnection of pages on a web site with a few thousand nodes. Often, the problem is designing a representation that works for large data sets. Size generally gives rise to the need to limit the amount of data presented through a user-directed filter.

The design of an information visualization application is not a simple task. The designer must consider how best to map the abstract information into a graphic that conveys information to the user. In addition, the designer must provide dynamic actions which limit the amount of information the user receives while at the same time keeping the user informed about the data set as a whole. Good design is especially difficult because the designer lacks guidance that is grounded in comparative evaluations of different information visualization designs. The designer cannot usually compare different visualizations on the same data sets. As we noted in a comparison of several three-dimensional visualizations[20], the data sets presented in papers are often those for which the visualization is well suited. We found that unbalanced hierarchies adversely affected the utility of the studied hierarchical visualization methods.

In this paper, I will advocate a user-oriented approach to information visualization where one starts with the user's tasks and designs the visualization to fulfill them. First, I

Proceedings of the 1999 Ericsson Conference on Usability Engineering.

will survey the area of visualization with emphasis on those aspects related to user tasks. Then, I will propose some guidelines for designers of information visualization applications. Finally, I will outline work that needs to be done in order to improve the design and use of information visualization.

USER TASKS IN VISUALIZATION

At first glance, an overview of user tasks in visualization seems impossible. Each situation has its own set of tasks, and one would have to study all uses of visualization in all application areas. Dozens of different designs have been devised[4]. In order to simplify things, abstraction is required. Shneiderman[17] presents seven abstract tasks that visualization users perform: overview, zoom, filter, details-on-demand, relate, history, and abstract.

Overview: Users need to gain an overview of the entire data collection. This overview concentrates on showing how the data items are related for a limited number of parameters. This limit is imposed by both the limits of the graphical device and the human perceptual system. The graphical presentation is limited to a spatial substrate, marks, and graphical properties of the marks[3, 4]. The most common graphical attributes are location (3 dimensions), color, mark type, and mark size. Perhaps a half dozen data parameters can be displayed; however, there is some question whether the user can perceive and cognitively process all of them. Also, certain types of data are better suited to certain types of coding[4, 10]. For example, continuous values map best to position along one axis while type classifications map best to a discrete property such as mark type.

Zoom: Zooming combines filtering with limited increases in detail. Users zoom for two reasons, to concentrate on a subset of the data and to see more graphical detail. Often, it is important to preserve a context while zooming. This can be accomplished by using distortion techniques[9], including hyperbolic geometry[8], or by using multiple windows.

Distortion techniques magnify a portion of the data set while keeping the entire data set on the display. In order to do this they must distort the portion of the data set that is not the focus of attention. Examples are the hyperbolic-geometry-based MagniFind file browser[7] by Inxight, Inc. (Figure 1) and Document Lens[16] (Figure 2). These techniques are also called fisheye techniques.

An alternative to distortion techniques is to use multiple window browsing. Here, the display is divided into a number of windows, and the magnified portion of the data set is displayed in one while the entire data set is displayed in another. An important function that should be implemented is coordination between the different windows[12]. This means that operations in one window are reflected in the other. For example in order to maintain context, the portion in a magnified view should be marked in the view of the entire data set (perhaps with a rectangle). Scrolling the magnified view will change its position within the view of the entire data set so the rectangle should move in order to maintain context.

Filter: In addition to the graphical data set reduction accomplished by zooming, users often need to reduce the data set size by eliminating items based on their attributes. One of the most efficient ways that filtering can be supported is with dynamic queries[1, 6]. Dynamic queries allow the user to manipulate sliders and buttons in order to specify ranges on data set parameters.

Details-on-Demand (or Drilling-Down): While exploring the data set, users will want to see the details of a particular data item. This is usually accomplished by clicking on one or more items and having the details displayed in a popup or auxiliary window.

Relate: If users discover an item of interest, they may need to know about other items with similar attributes. Alternatively, items in a visualization could be related through being part of the same process or event (a visit to the doctor in a medical database, for example). Users would want to know about related events or items. Here, clicking on one item would highlight related ones. This technique is used in the FilmFinder[1], LifeLines[13], and SDM[5].

History: Users need support for undo, replay, and

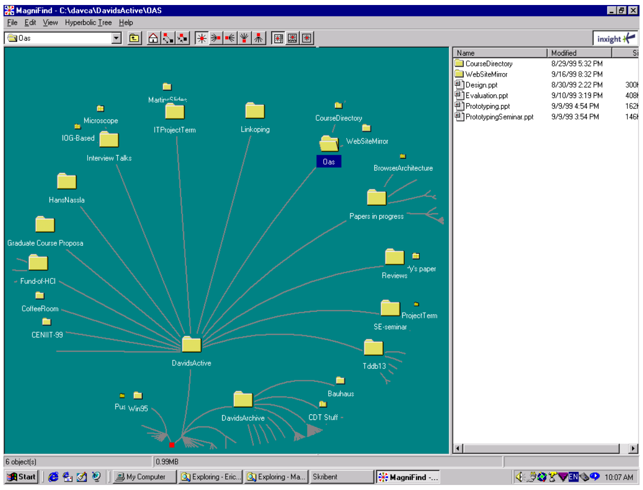

Figure 1 – The MagniFind file browser from Inxight.

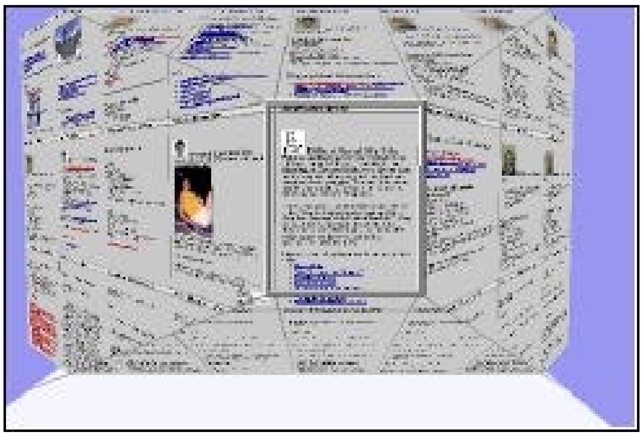

Figure 2 – The Document Lens

progressive refinement. It is therefore important to keep a history of user actions and allow the user to manipulate it. Often visualizations do not support history actions or support them incompletely. For example, browsing operations are often undoable only by other browsing actions and not via the undo command.

Extract: The results of visualization operations need to be extracted from the visualization. This might be a subset of the data or the query parameters and operations which produced the results. In this way, users can extract subsets of the data for further analysis and apply the results of an analysis to other data sets.

Finally, one should consider how users interact with visualization applications. Part of this can be summed up in Shneiderman's visual information seeking mantra[17]: "Overview first, zoom and filter, then details on demand." However, one should add that history operations are needed throughout and that afterwards, relate and extract operations are common.

GUIDELINES

Information Visualization is a young discipline, and little evaluation has been done to test the efficacy of various designs. However, designers need to help users cope with information overload. So, here are seven general guidelines to help design information seeking applications using Information Visualization.

Visualization Is Not Always the Best Solution

Visualization is designed to assist users who are pursuing a goal that is unknown to the designers and requires sifting through a lot of data. If the goal of the user can be determined at design time, then an algorithm to meet that goal will be faster and more accurate. For example, one of the canonical data sets for testing hierarchical visualizations is a computer file system. File systems are used because they are easy to find, large, and representative of the class of hierarchies. (Every modern PC has a file system of thousands of nodes that is organized in a strict hierarchy.) Researchers usually use test tasks such as: "What is the largest file?", "How many files are in directory X?", and "Find file Y." This tasks are precisely the tasks that are best solved by a program written specifically for that task.

This is not to say that file browsers are not useful. When the goal is to find a file whose name and location are approximately known, a file browser is a very good tool. The list of files and directory names helps the user recognize the specific file that is wanted.

In summary, consider visualization if there are large amounts of data, the user goals are not easily quantifiable, and there are no simple algorithms to accomplish the goals.

User Tasks Must Be Supported

If visualization is being used, then users have large amounts of data to consider. In order to improve the efficiency of general visualization techniques, support for specific user tasks should be included. The abstract tasks given above provide a framework, but users will appreciate more specific support. For example, if users frequently compare the details of two nodes in a network, then providing specific support for this task is a good idea. A design might include a browser with three windows, one for the network and two for node detail. The user could then point at a node and display its details by dragging the node to one of the node-detail windows and dropping it. The design should also support some type of "gravity" function that attracts the cursor to a node in order to aid in selecting nodes. In general, well-designed support for user tasks will be faster and less error prone than a general visualization.

The Graphic Method Should Depend on the Data

When choosing a visualization method carefully consider the data type. The first question to ask is "Does the data have some tie to the physical world?" If so, displaying the data on a representation of the physical world can help orient users. For example, a communications network may have nodes in different cities. While geography may not be important for identifying bottlenecks, displaying the data on a map will give the user a strong point-of-reference. Shneiderman[17] identifies seven data types and describes different visualizations for each. The data types along with a brief description are:

- **1-Dimensional:** This data type represents text, program source code, and similar data. There may be additional data such as last modification date associated with each item. This type has basically been viewed in one of two ways. The first is a line or band with color or other coding. There may also be some type of a magnifying tool. Long data sets are mapped onto the display by wrapping the lines. The second technique has been to place the data in two dimensions and use a magnification tool. The Document Lens (Figure 2) is an example of this.
- **2-Dimensional:** This type includes geographic data, floor plans, and similar data. Since each point in the plan has some number of attribute values associated with it, the natural thing is to lay out the points in a plane and use color, mark type, or a similar graphical primitive to encode the attribute's value. Design problems occur when users need to analyze two or more attributes simultaneously. Also, users may need to find paths between items or containment relations. Support for zooming, filtering, and relating are required along with user specification of coding schemes. Techniques such as scatter plots and dynamic queries work well[1, 6].
- **3-Dimensional:** This is real world data tied to objects with volume data. Here, it is important to connect the data to the real world object. However, users will need to look at the data closely, and volume means that one part of the data may obscure others. Techniques such as transparency, slicing, and multiple views will be required.

- **Temporal:** This data is either, any of the above data types with time added, or time itself may form a second dimension for measurement at a single point. If the number of points that need to be considered at one time is small, value versus time plots work well. When the number of points becomes large, the temptation is to use animation. However, one should be aware that users may need to compare static images taken at several points in time and to control the speed of animation. Static tools such as LifeLines[13] may work better, especially when users must study the data.
- **Multi-Dimensional:** Relational and statistical databases can be conveniently considered as points in a multi-dimensional space. With this type of data, techniques such as dynamic queries and scatter plots can be useful[1, 6]. It is especially important that the user be able to select the attributes to be manipulated by the visualization.
- **Hierarchical:** Hierarchies are one of the most important classification methods. They are usually represented as node-link diagrams. The leaves may or may not be treated specially. Sometimes, the containment is represented graphically. Examples include:
  - The Microsoft Explorer, a node-link diagram over interior nodes with a special window for the children of a selected node (Figure 3).
  - The Inxight MagniFind which has same strategy except the diagram uses hyperbolic geometry (Figure 4).
  - The Cam Tree[15], a 3D node-link diagram using the surface of a cone as its geometric base (Figure 5).
  - The Information Landscape[2, 18], a 3D node-link diagram using a plane as its geometric base (Figure 6).
  - The Information Cube[14], a 3D visualization that uses cubes within cubes to show containment relations (Figure 7).

  Note that Figures 3 through 7 all depict the same file structure.
- **Network:** Network data is a group of nodes connected by arbitrary links. The links cannot be arranged into a tree or hierarchy. The usual method of handling networks is to place the focus on one node and generate a node-link diagram from it as if it were the root of a tree. The problem is to represent the extra links without getting “spaghetti”. The usual solutions are to use circles, spheres, or hyperbolic geometry as a base and hope that the extra links do not cause too many problems. An important support for users is the ability to prune links and nodes with a filter.

Three Dimensions Are Not Necessarily Better Than Two

Many designers will advocate the use of three dimensions because one gets an extra, continuous, data channel. However, this does not come without a price. Introducing three dimensions will slow the repainting of the display. When the user must navigate this will take longer than in two dimensions. Data will also be occluded, thus increasing the chances of errors and tending to force the user to navigate. Finally, perspective viewing makes judging size more difficult removing it as a effective means of displaying an attribute.

In our comparative study[21] of the Information Landscape, Cam Tree, and Information Cube, we found that users performed simple tasks significantly faster on the Information Landscape than on the other two, and the Cam Tree was significantly faster than the Information Cube. The main reasons for this were problems with occlusion and navigation. The Information Cube was particularly difficult as the users were constantly becoming disoriented. We did not include a 2D visualization in the study. However, we did note that many users of the Information Landscape positioned themselves over the landscape so that it effectively became a 2D visualization. Also, our data sets were small enough that a simple node-link diagram would have fit on the screen. Users could then have preformed the test tasks by inspection.

Our experience should not be considered the last word. We used a simple perspective view. Another study[19] tested 2D against three different virtual reality techniques: stereo display, head-coupled display, and both in combination. Each display gave successively better results about the size of network that users could understand. Results as a factor of network size on the 2D display at identical error rates ranged from 1.6 for stereo displays to 3 for stereo and head-coupled in combination. It should be noted that the 2D diagram was static and apparently did not have zoom and filter support.

In summary, the case for or against using 3D graphics is not clear. It would seem that, for data with a physical representation, three dimensions is okay, but for abstract data it helps to tie the data to some geometric structure such as a plane (so called, “2.5D” graphics) or to a sphere.

Navigation and Zooming Do Not Replace Filtering

There is a great temptation to use only graphical methods to allow the user to explore data. This should be avoided. Users need to eliminate data items from consideration in order to concentrate on items of interest. While navigating and zooming through the graphic display does this, there are times when this is not adequate. It often occurs that users will eliminate a whole class of data items that are scattered throughout the display. The only way to accomplish this is with a filter. There are also times when the relation of interest is not one of the primary axes. If a way to highlight points satisfying the criteria or remove points not satisfying the criteria is not available, the user is reduced to tediously examining the data item by item.

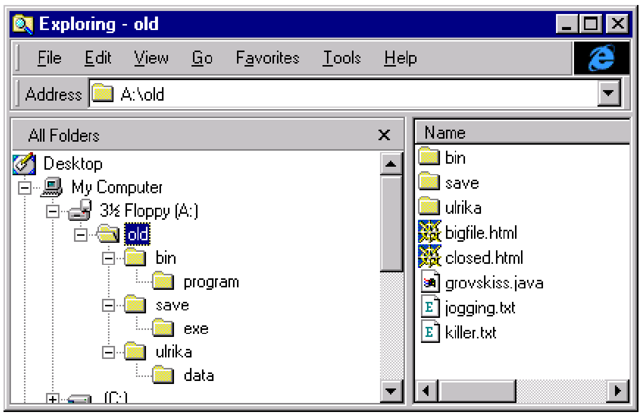

Figure 3 – Microsoft Windows Explorer.

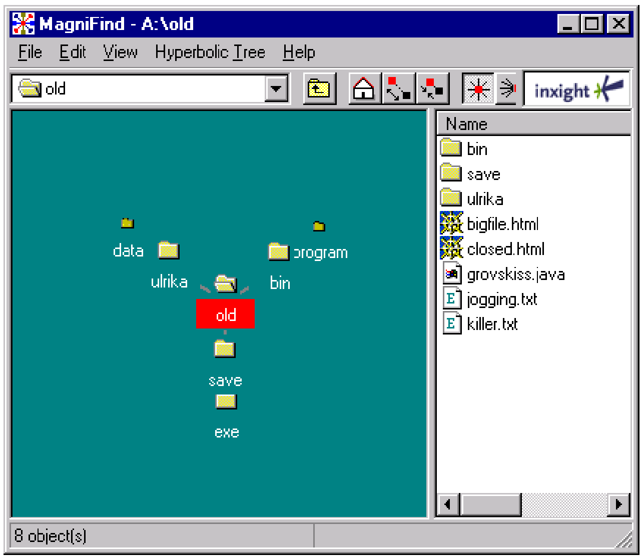

Figure 4 - Inxight MagniFind.

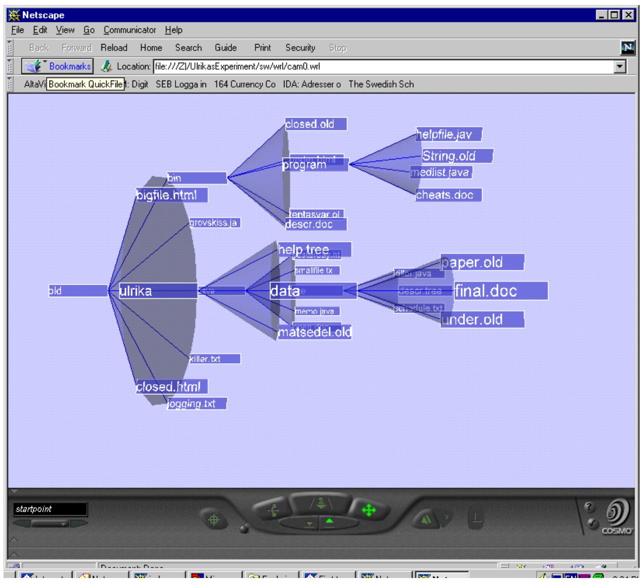

Figure 5 – Cam Tree.

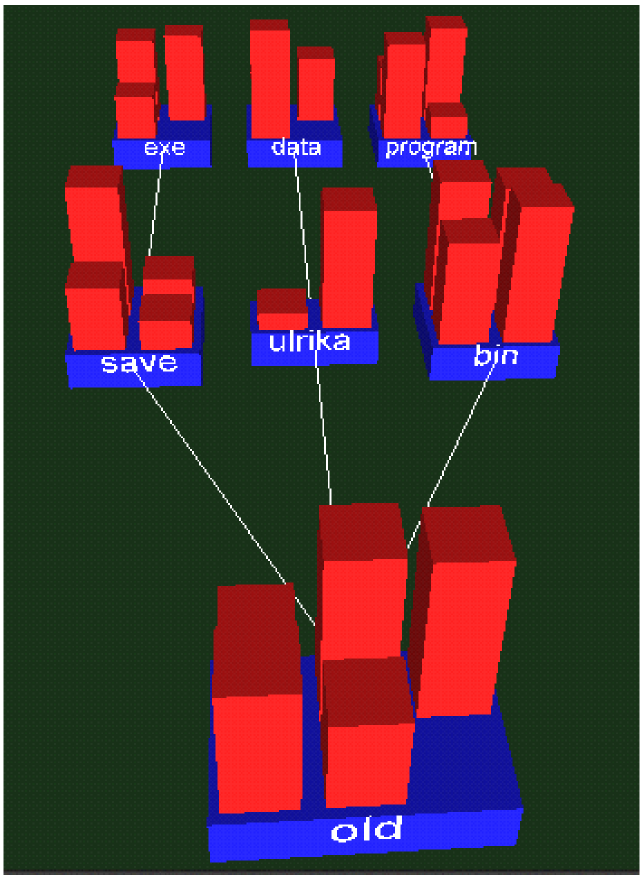

Figure 6 – Information Landscape.

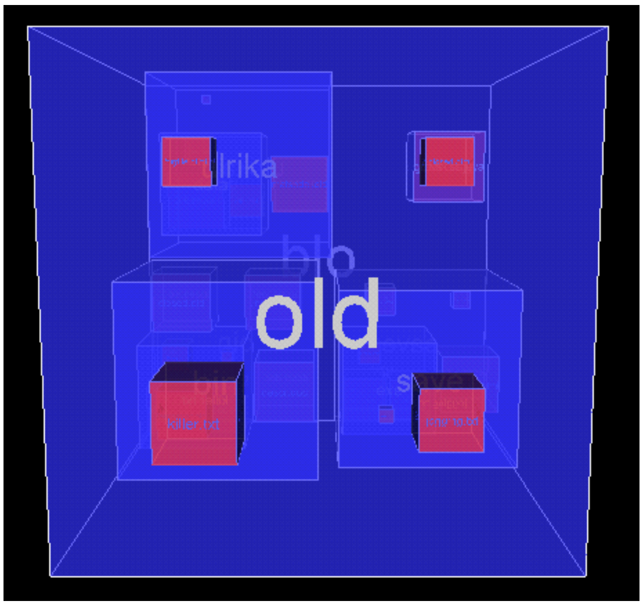

Figure 7 – Information Cube.

Multiple Views Should Be Coordinated

Managing display real estate is always a problem in graphic applications. Information visualization is no exception. The standard way to manage limited display space is through windowing. As the user zooms, filters, and extracts subsets of the data, the results are displayed in auxiliary windows. It is important that the relationships between windows be maintained. For example, users may select an area in one window for zooming and display in another. The selected area should be shown in the first window. If the user grabs the rectangle and moves it, the second window should be updated to show the new area. Similarly, scrolling in the second window should update not only its view, but the location of the rectangle in the first window as well. In the Snap-Together Visualization system[11], this principle is carried one step further. It is possible to construct coordination relationships between windows where filtering and other operations in the first window affect the display of the second. In addition, the system allows more than just simple graphics operations between windows. So, data selected in the first window could be displayed using a different visualization method in the second.

Test Your Designs with Users

Information visualization applications are complex and cannot be designed without user input. Knowledge about the usability of specific designs is scarce. Therefore, it is imperative to test with users under work situations. It is also important to consider that novice users may perform poorly using systems on which more experienced users perform well. Little is known about how much training is needed for users to become productive.

CONCLUSIONS AND FUTURE WORK

There are now many published Information Visualization designs that are designed to help users cope with information overload. When considering a design one should analyze it in terms of how it supports each of Shneiderman's seven abstract information seeking tasks: overview, zoom, filter, details-on-demand, relate, history, and extract. These abstract task categories provide a framework that allows one to determine how completely the user's task is supported.

In addition to the abstract tasks, seven general guidelines for visualization design were presented:

- Visualization is not always the best solution.
- User tasks must be supported.
- The graphic method should depend on the data.
- Three dimensions are not necessarily better than two.
- Navigation and zooming do not replace filtering.
- Multiple views should be coordinated.
- Test your designs with users.

Unfortunately, there has been almost no testing of visualization usability. If one scans the literature, one finds very few studies of the subject. In fact, it is almost impossible to find an example of two different techniques displaying the same data set. Much more study of visualization usability is sorely needed. In addition, there needs to be a study of visualization in realistic work settings. Current work has been limited to small data sets, novice users, and laboratory settings. In order to recognize the true potential of information visualization, more information is needed regarding learning time and usefulness on large data sets.

REFERENCES

1. Ahlberg, C. and Shneiderman, B. Visual information seeking: tight coupling of dynamic query filters with starfield displays, *Proceedings of CHI'94, ACM Conference on Human Factors in Computing Systems,* 365-371, (also [4], 244-250).
2. Andrews, K. Visualizing cyberspace: information visualization in the harmony internet browser, *Proceedings of InfoVis'95, IEEE Symposium on Information Visualization,* 97-104, color plates 147-148, (also [4], 493-502).
3. Bertin, J. *Graphics and Graphic Information Processing,* Berlin, De Gruyter, 1977/81, 24-31 (also [4], 62-65).
4. Card, S., Mackinlay, J., and Shneiderman, B. eds. *Readings in Information Visualization Using Vision to Think,* Morgan Kaufmann, 1999, ISBN 1-55860-533-9.
5. Chuah, M., Roth, S., Mattis, J., and Kolojejchick, J. SDM: selective dynamic manipulation of visualizations, *Proceedings of UIST'95, ACM Symposium on User Interface Software and Technology,* 61-70 (also [4], 263-275).
6. Fishkin, K. and Stone, M. Enhanced dynamic queries via movable filters, *Proceedings of CHI'95, ACM Conference on Human Factors in Computing Systems*, 415-420, (also [4], 253-259).
7. Inxight, Inc. MagniFind file browser available at http://www.inxight.com/MagniFind/MagniFind.html
8. Lamping, J. and Rao, R. The hyperbolic browser: a focus + context technique for visualizing large hierarchies, *Journal of Visual Languages and Computing*, 7(1), 33-55, (also [4], 382-408).
9. Leung, Y. and Apperley, M. A review and taxonomy of distortion-orientation presentation techniques, *ACM Transactions on Human-Computer Interaction,* 1(2), 126-160, (also [4], 350-367).
10. Mackinlay, J. (1986) Automating the design of graphical presentations of relational information, *ACM Transactions on Graphic,* 5(2), 110-141, (also [4], 66-82).
11. North, C. and Shneiderman, B. Snap-together visualization: coordinating multiple views to explore information, University of Maryland Comp. Science Department Technical Report CS-TR-4020, 1999,

(ftp://ftp.cs.umd.edu/pub/hcil/Reports-Abstracts-Bibliography/99-10html/99-10.html).

12. Plaisant, C., Carr, D., and Shneiderman, B. Image browser taxonomy and guidelines for designers, *IEEE Software*, 12(2), (March, 1995), 21-32.

13. Plaisant, C., Milash, B., Rose, A., Widoff, S., and Shneiderman, B. LifeLines: visualizing personal histories, *Proceedings of CHI'96, ACM Conference on Human Factors in Computing Systems*, 221-227, (also [4], 287-294).

14. Rekimoto, J. and Green, M. The information cube: using transparency in 3D information visualization, *Proceedings of the Third Annual Workshop on Information Technologies and Systems (WITS'93),* 125-132.

15. Robertson, G., Mackinlay, J., and Card, S. Cone trees: animated 3D visualizations of hierarchical information, *Proceedings of CHI'91, ACM Conference on Human Factors in Computing Systems*, 217-226.

16. Robertson, G. and Mackinlay, J. The document lens, *Proceedings of UIST'93, ACM Symposium on User Interface Software and Technology,* 101-108, (also [4], 562-569).

17. Shneiderman, B. The eyes have it: a task by data type taxonomy for information visualizations, *Proceedings of 1996 IEEE Visual Languages*, 336-343.

18. Tesler, J. and Strasnick, S. FSN: 3D information landscapes, man page entry for an unsupported but publicly released system from Silicon Graphics, Inc., 1992.

19. Ware, C. and Franck, G. Viewing a graph in a virtual reality display is three times as good as a 2D diagram, *Proceedings of 1994 IEEE Visual Languages*, 182-183.

20. Wiss, U. and Carr, D. Evaluating three-dimensional information visualization designs: a case study of three designs, *Proceedings IEEE Conference on Information Visualization (IV'98),* 137-144.

21. Wiss, U. and Carr, D. An empirical study of task support in 3D information visualizations, *Proceedings IEEE Conference on Information Visualization (IV'99),* 392-399.