Visualising Information Structures

Aspects of Information Visualisation

Keith Andrews

Visualising Information Structures

Aspects of Information Visualisation

Professorial Thesis

13th November 2002

Keith Andrews

IICM
Graz University of Technology
Inffeldgasse 16c
A-8010 Graz

This thesis is available online at

`http://www.iicm.edu/keith/`

Abstract

Information visualisation involves the visual presentation of abstract information spaces and structures to facilitate their rapid assimilation and understanding. Information structures include hierarchies, networks, multidimensional tables, and collections of text documents. Techniques from information visualisation are finding increasing application in these and other areas.

This thesis presents my own work in information visualisation, embedded in the context of related work from the field as a whole.

For search engine users across the pond, the field is sometimes also referred to as information visualization, with a zed.

Contents

List of Figures

List of Tables

Credits

I would like to thank the following individuals and organisations for permission to use their material:

- Figure 2.9 of Sunburst is used with kind permission of John Stasko, Georgia Tech.
- Figure 2.7 of a Tree Map is used with permission of the University of Maryland.
- Figures 2.11 of the Cone Tree and 2.12 of the Cam Tree are used with kind permission of Stu Card, PARC.
- Figure 2.18 of the SInVis Magic Eye View is used with kind permission of Matthias Kreuseler, University of Rostock.
- Figure 2.19 of a Botanic Visualisation is used with kind permission of Jack van Wijk, Eindhoven University of Technology.
- Figure 3.4 and 3.5 of SemNet were scanned from original photographs with kind permission of Kim Fairchild.
- Figure 4.3 of the FilmFinder is used with permission of the University of Maryland.
- Figure 5.2 of SPIRE is used with kind permission of Pacific Northwest National Labs.

The following figures are used subject to the ACM Copyright Notice [1]:

- Figure 2.10 of a CyberGeo Map was extracted from Proc. CHI 2000.
- Figure 3.6 of Galaxy of News was extracted from Proc. UIST 94.
- Figure 4.2 of the Attribute Explorer was extracted from the CHI 94 Electronic Proceedings.
- Figure 4.4 of Envision was extracted from the CHI 97 Electronic Proceedings.
- Figure 5.1 of Bead was extracted from the Proc. UIST 96.

Acknowledgements

It is not possible to write a thesis like this without the help and collaboration of dozens of colleagues and students. At the risk of forgetting to mention one or two, I would like to at least attempt to mention them all. Should the frailty of human memory and the lateness of the hour conspire to confound my noble intentions, I tender my apologies in advance.

Hermann Maurer, my mentor and boss at the IICM, gently encouraged and nurtured me along the rocky road to academia, whilst leaving ample room for other things. For that, I am greatly indebted. My appreciation also to Dieter Fellner, my supervisor and mentor during my first years in Graz.

Jutta Becker, Klaus Tochtermann, Vedran Sabol, Wolfgang Kienreich, and Michael Granitzer of the Know-Center and Wolfgang Kienreich, Frank Kappe, and Georg Droschl at Hyperwave make InfoSky and galactic Voronois both possible and fun.

Christian Gütl, Vedran Sabol, Erwin Weitlaner, Josef Moser, and Wilfried Lackner were instrumental in the visualisation work on VisIslands and the Search Result Explorer for xFIND. Harald Köhler implemented the 3D File Attribute Explorer. Jürgen Schipflinger implemented the original work on the Harmony Collection Browser and on focused graph drawing in the Harmony Local Map, which Alexander Stedile continued with JMFGraph. Hans Madlberger built the Information Tunnel and Helmut Heidegger implemented Information Slices. Stefan Petrik built the IICM version of Magic Eye View.

Josef Wolte built the first version of Information Pyramids, which Michael Welz extended. Martin Eyl and Peter Wolf built the Harmony Information Landscape and Harmony 3D Local Map. Michael Pichler played a leading role in all of my early activities in information visualisation. He also helped build the VRwave and VRweb VRML browsers with Andreas Pesendorfer, Karl Heinz Wagenbrunn, and Georg Meszaros, and Gerbert Orasche.

Nick Scherbakov helped me understand composite hypermedia models. Klaus Schmaranz, Christof Dallermassl, Harald Krottmeier, Malou Lampl, Helmut Leitner, Karl Trummer, and all my other colleagues at the IICM, HMS, WAG, and Know-Center provided support and infrastructure along the way.

I have benefited from fruitful discussion with, and have been inspired by, many colleagues from around the world, including: Franz Aurenhammer, Konrad Baumann, Stu Card, Matthew Chalmers, Andreas Dieberger, John Dill, Andrew Dillon, Steve Eick, Kim Fairchild, Steve Feiner, Nahum Gershon, Ben Gross, Gene Golovchinsky, Hans Hagen, Ivan Herman, Andreas Holzinger, Keith Instone, Daniel Keim, Franz Leberl, Jock Mackinlay, Tamara Munzner, Jakob Nielsen, Stephen North, Lucy Nowell, Jim Pitkow, Steve Poltrock, Ramana Rao, Harald Reit-

erer, George Robertson, Steve Roth, Ben Shneiderman, Bob Spence, John Stasko, Jim Thomas, Matt Ward, Jack van Wijk, Graham Wills, Jim Wise, Pak Chung Wong, Bill Wright, and Brian Wylie.

Thanks to Rich and Rob Cheese, Ross Little, the Anglo-Austrian crew, my ex-wife Elisabeth, and all my friends in Graz for making it a really cool place to live …

And finally, to David and Kristin: I love you and I dedicate this work to you.

Keith Andrews

Graz, Austria, November 2002

Preface

This thesis presents work that I have been involved with over the past seven years. The intention is to place my own contributions in the wider context of the field of information visualisation.

To that end, each of the main chapters begins with a discussion of relevant related work. I make no claim that the relevant work presented is a comprehensive survey. Indeed, I am sure that many more systems and techniques could and should have been included. For now though, the constraints of time dictated the present selection.

The related work in each chapter is followed by a section, roughly corresponding to an individual publication, for each of the projects or systems with which I have been involved. Of course, such involvement is rarely in isolation. Most, if not all, of the projects were collaborative ventures and my collaborators and co-authors are acknowledged either in the references for each section, or in the Acknowledgements section of the thesis.

Chapter 1

Introduction

As information spaces become ever larger and even the number of files and directories on our desktop PC threatens to overwhelm us, it is becoming increasingly important to have ways of managing the flood. One of these is *information visualisation*.

Information visualisation, to my mind, can be summed up as:

> "the visual presentation of abstract information spaces and structures to facilitate their rapid assimilation and understanding".

With appropriate information visualisation tools, it is possible to gain an overview of very large information spaces, to smoothly zoom and filter, and to call up details on demand.

Humans have remarkable perceptual abilities to scan, recognise, and recall images rapidly. The human visual perception system can rapidly and *automatically* detect patterns and changes in size, colour, shape, movement, or texture. Interactive information visualisation systems can exploit this pre-attentive processing to condense large amounts of information into manageable displays. Text-based interfaces require cognitive effort to understand their information content. Information visualisation seeks to present information *visually*, in essence to offload cognitive work to the human visual perception system.

Information visualisation is not quite the same as traditional scientific visualisation. In scientific visualisation, the data generally has inherent 2d or 3d geometry and hence a fairly obvious representation in 2d or 3d space. In information visualisation, the information structures are entirely abstract, and could potentially be visualised in any of a multitude of ways. The skill is to carefully design or *invent* an appropriate visual representation for the information space and context at hand.

That said, as with most attempts to define a fuzzy concept, the boundaries of the field of information visualisation are excitingly blurred. Multidimensional datasets are often both scientific and abstract. The field of bioinformatics is proving an especially rich melting pot for techniques from both traditional scientific visualisation and information visualisation.

The visual representation is, in fact, only half of the story. A static picture without any interaction might be aesthetically pleasing but is generally only of limited utility. Stu Card has a saying:

> "Computers are all well and good in their place, but to get real work done you need a dining room table."

Computer screens, having perhaps a million pixels, are simply not big enough. We have to exploit the dynamic characteristics of the computer display. With modern computer systems, we can design interactive interfaces capable of changing and adapting to the changing needs of the user.

When designing animated transitions between displays, work at PARC [Robertson et al., 1991] suggests that the transition should last about a second. If the transition is too fast, users cannot follow the movements of objects and have to re-assimilate the new scene afresh. If the transition is too slow, the lag is perceptible and the system feels slow and unresponsive.

Ben Shneiderman [Shneiderman, 1996] formulated a mantra, capturing the essence of information visualisation:

> "Overview first, zoom and filter, details on demand."

Alluded to earlier, it is the blueprint for most information visualisation systems.

Another general principle, often cited, is that of *focus plus context*. The simultaneous provision of both local focus and surrounding context can help users maintain a sense of orientation. For example, in *3d perspective*, the user's attention is naturally focused on objects in the foreground, but the user is still aware of the background context. In a *fisheye view*, geometric distortion is used like a magnifying glass to focus on an area of interest while still maintaining context. In *overview plus detail* displays, a separate overview map complements a detailed view.

In order to squeeze the maximum utility out of every available pixel of the display, *space-filling* techniques have emerged. Items are condensed down to single pixels or sub-pixels on the display and (almost) every pixel on the screen is used to convey information. Jerding and Stasko [1988] describe an efficient method for generating such sub-pixel displays of an information space.

In can sometimes be useful to think about information visualisation in terms of the types of information being displayed. In analogy to Shneiderman [1996], information can be classified into the following types:

- *Linear*: Tables, program source code, alphabetical lists, chronologically ordered items, etc.
- *Hierarchies*: Tree structures.
- *Networks*: General graph structures, such as hypermedia node-link graphs, semantic networks, webs, etc.
- *Multidimensional*: Metadata attributes such as type, size, author, modification date, etc. Items with n attributes become points in n-dimensional space.
- *Vector Spaces*: A vector generated from the content of an item is used to represent its characteristics. In a corpora of text documents, vectors embodying the frequencies of

(key)words represent each document. The cosine or other similarity metrics between vectors can be used for document clustering.

- *Spatial*: Inherently 2d or 3d data such as floor plans, maps, CAD models, CAT scans, etc.

Spatial information has an obvious natural representation, so I do not consider it to be information visualisation in the strict sense, but techniques from information visualisation are often used with spatial data.

When working with an information visualisation system, users generally have a particular information need or a particular purpose in mind. Expanding on the task taxonomy of Shneiderman [1996], typical tasks a user might perform include:

- *Overview*: Gain an overview of the entire collection.
- *Zoom*: Zoom in on items of interest.
- *Filter*: Filter out uninteresting items.
- *Details on Demand*: Select an item or group of items and obtain details when needed.
- *Relate*: View relationships between items.
- *History*: Undo, redo, and progressive refinement.
- *Extract*: Search for particular subsets of items.
- *Organise*: Manually organise items in a particular way, so as to find them more easily next time.

It is the job of information visualisation systems to support users as far as possible in these tasks.

The rest of this thesis is structured according to information type. I do not consider linear structures in this thesis. Chapter 2 deals with visualisation techniques for hierarchical information structures, Chapter 3 deals with visualisation techniques for networks and general graphs, Chapter 4 discusses multidimensional information visualisation, and Chapter 5 covers techniques for large collections of text documents. Closing remarks are made in Chapter 6.

For further reading and information, the reader is referred to the resources in Appendix A.

Chapter 2

Visualising Hierarchies

Hierarchical structures are increasingly common in our information rich society as a means for organising large amounts of information. Examples of information hierarchies include the hierarchical organisation of employees in a company and the files and directories stored on a hard disk. Hierarchies are becoming ever larger; even the hard disk of an ordinary home PC may contain many thousands of files. Hence, the provision of explorable, visual representations is essential to make large hierarchies accessible and understandable.

A tree has an order from top to bottom. *Inner nodes* have further children, *leaf nodes* are final nodes having no children of their own. In a tree, there exists a unique path from the *root* of the tree to every leaf node. For example, in a file system hierarchy, the inner nodes are directories and subdirectories, and the leaf nodes are files. Every file can be reached from the root of the file system by opening a path through directories and subdirectories. In a document management system, the inner nodes might be called collections and subcollections and the leaf nodes might be documents. In a decision tree, inner nodes represent choices and leaf nodes final decisions.

Traditional visualisation techniques, such as outlines or tree diagrams, are perfect for visualising small, compact hierarchies, which can be drawn on one sheet of paper or presented on a computer screen without having to scroll. However, if the hierarchy is too large, the visualisation fails to be simple and understandable.

Many general graphs (networks) can also be transformed to a spanning tree plus backlinks, allowing hierarchical visualisation techniques to be applied to general graphs and networks by extracting and visualising a spanning tree and then overlaying any backlinks.

2.1 Related Work

Herman et al. [2000] provides a good overview of work on visualising hierarchies and more general graphs. Young [1996] and Dodge [2002] also include several examples.

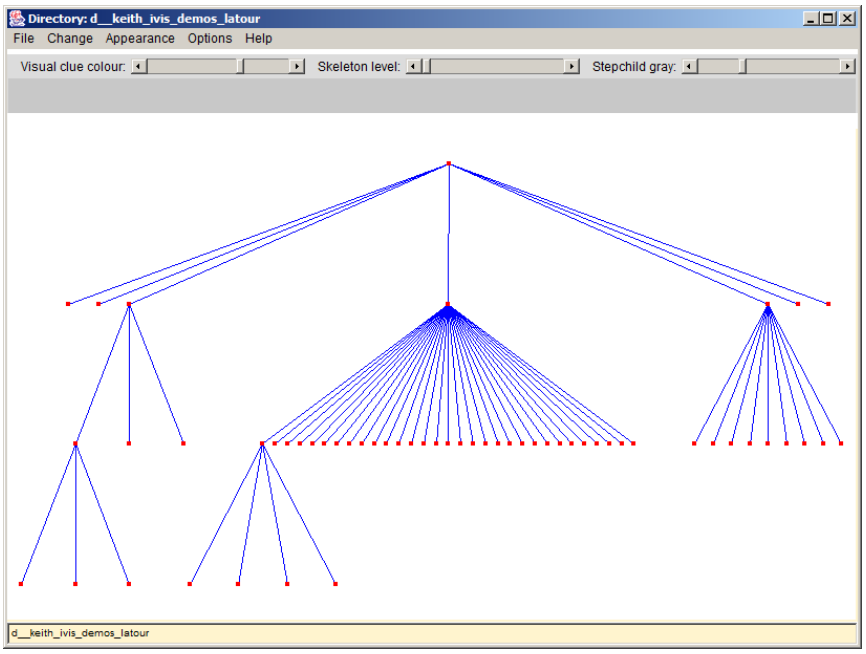


Figure 2.1: The classic Reingold and Tilford tree layout. The tree is drawn recursively upwards from bottom to top.

2.1.1 Classic Tree Drawings

The classic tree drawing algorithm shown in Figure 2.1 was formulated by Reingold and Tilford [1981]. In simple terms, the tree is drawn recursively upwards from bottom to top. Leaves are placed in layers according to their level in the tree. Their x-coordinate is determined so that all nodes in a certain layer are as flush to one another as possible. Improvements to the original algorithm were proposed by Walker [1990] and later Buchheim et al. [2002].

2.1.2 Tree Browsers

Traditional tree browsers, such as the Macintosh File Finder or the Windows Explorer, use an outline method to visualise hierarchies and are used by millions of people on a daily basis. They are simple to implement and can also be used in text-based systems. Typically, not all subtrees of the hierarchy are expanded initially. Instead, the user navigates by specifically expanding and collapsing subtrees, and hence decides explicitly how broadly or deeply the hierarchy is displayed at any one time.

As can be seen in Figure 2.2, the lefthand panel of the Windows Explorer shows the hierarchical structure, the righthand panel shows the items (files, documents) at a particular level. The lefthand panel provides context, whilst the righthand panel provides focus. The Java Swing JTree component shown in Figure 2.3 is very similar.

WebTOC [Nation et al., 1997; Nation, 2002] allows navigation through a web site using a pre-generated hierarchical structure (table of contents). When the hierarchy is generated,

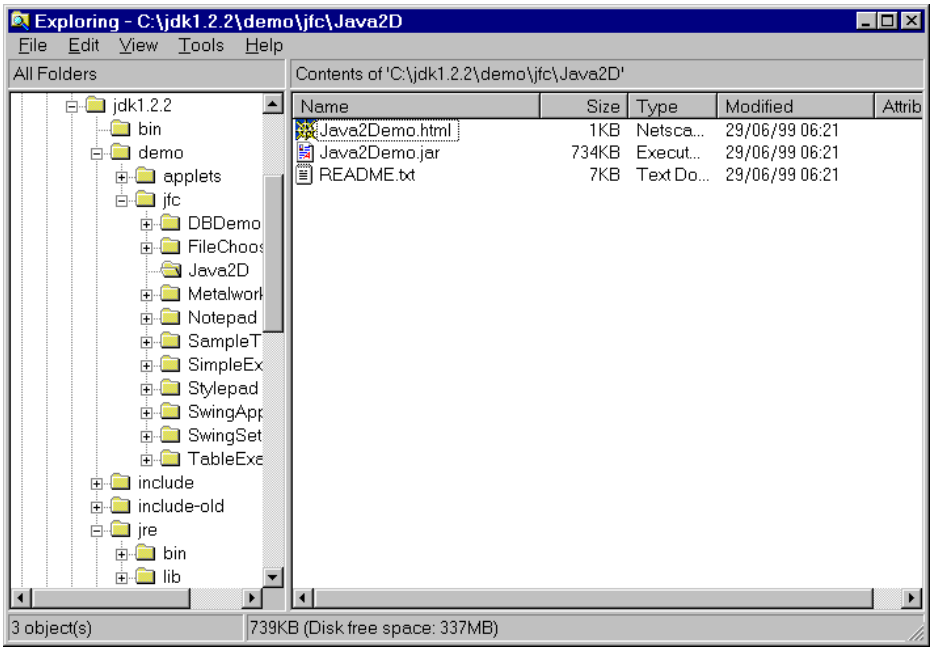


Figure 2.2: The Windows Explorer. A view of directories on the left and their contents on the right.

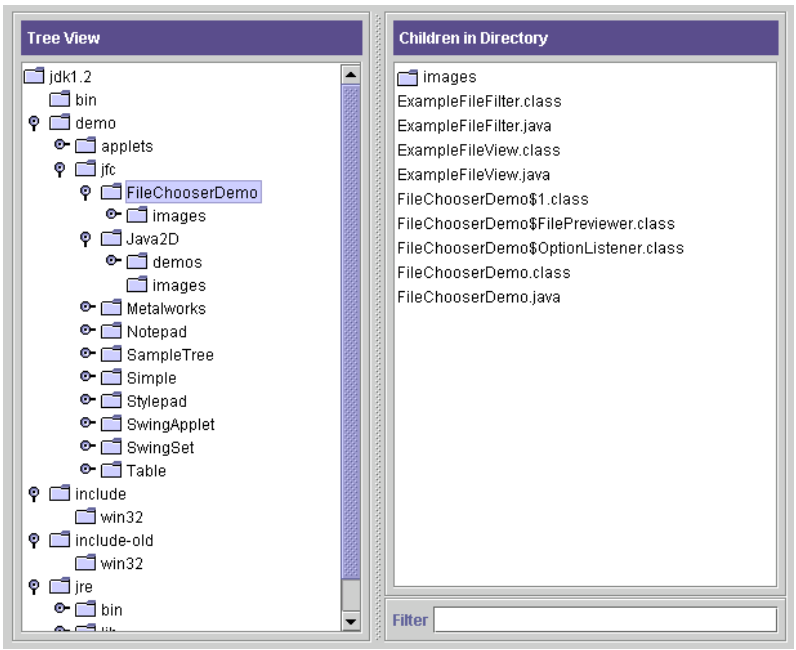


Figure 2.3: The Java Swing JTree tree browser component. A view of directories on the left and their contents on the right.

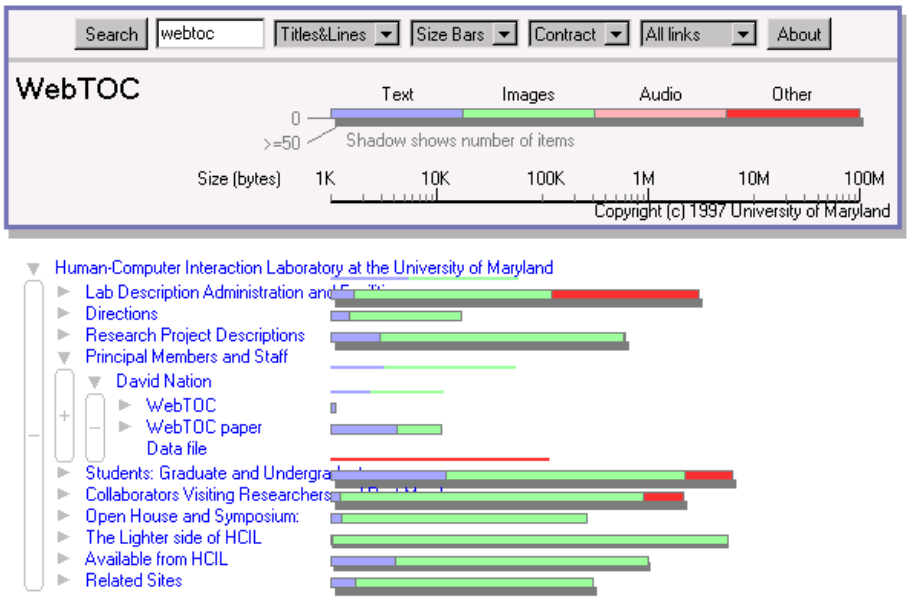


Figure 2.4: A WebTOC table of contents for the University of Maryland's HCI Lab web site. A row to the right of each branch of the tree view indicates the proportion of text, image, audio, and other media on a logarithmic scale. The shadow indicates the number of individual files.

statistical information regarding file types and sizes is also saved. The WebTOC applet uses this information to provide a preview of the contents of each directory. Coloured bars indicate the proportion of various media types, and shadows indicate number of files, as shown in Figure 2.4. The overall effect is of a traditional tree view supplemented with statistical preview information. It is also possible to search for items within the hierarchy.

In a similar vein, the File Magnitude Viewer [Smith and Clark, 2001] is a Java application which scans a file system and compiles a preview pie icon of the relative size of each subdirectory. This icon is then used in a modified JTree tree view component to indicate how much is contained in each subdirectory, as shown in Figure 2.5.

2.1.3 Tree Maps

Xdu is the X Windows disk usage utility [Dykstra, 1991], which displays a graphical overview of disk usage in a Unix file system. Rectangles are stacked from left to right as the directory tree is descended, as shown in Figure 2.6. Vertical space at each level is allocated in proportion to the total size of each subdirectory.

Tree maps [Johnson and Shneiderman, 1991; HCIL, 2002] are a space-filling technique for hierarchical structures. Traditional tree maps divided the available screen space by slicing and dicing alternately into vertical and horizontal strips, as shown in Figure 2.7. The size of each strip is proportional to its *weight*, whereby the weight might typically be the total number or

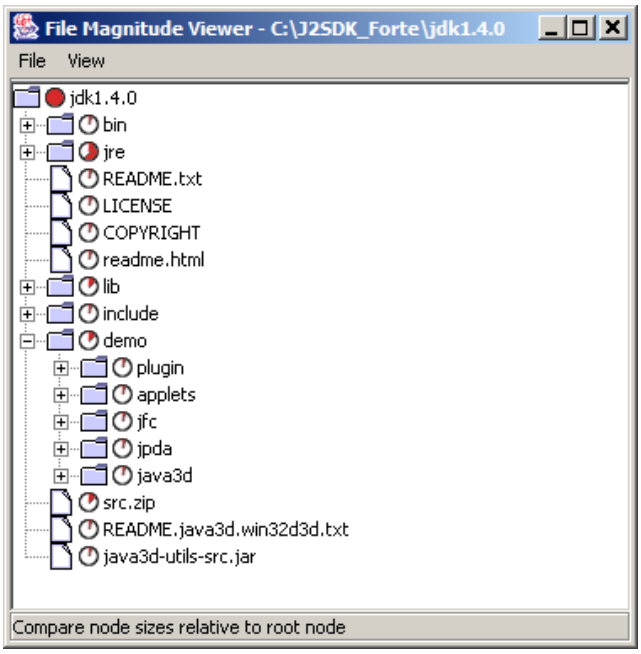


Figure 2.5: The File Magnitude Viewer scans a file system and generates a pie graphic to indicate the relative size of each subtree.

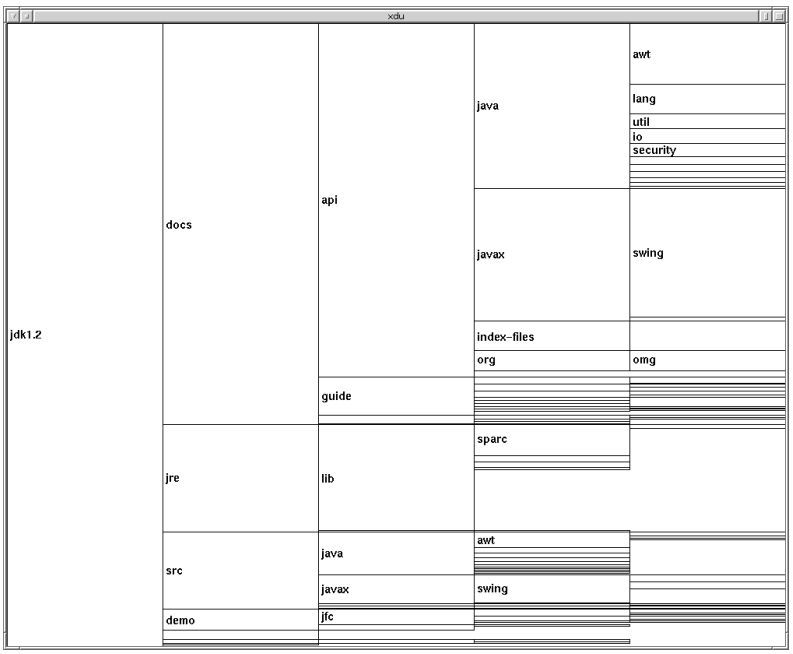


Figure 2.6: An Xdu visualisation of the Java JDK 1.2 distribution. The relative sizes of each rectangle indicate the relative sizes of each subdirectory.

Figure 2.7: A tree map of the Dewey Decimal classification hierarchy widely used in libraries. [Image is used with permission of the University of Maryland. Copyright ©University of Maryland, all rights reserved.]

total size of items within it. The main problem with traditional slice and dice tree maps is that the rectangles often degenerate to very narrow horizontal or vertical strips.

The Market Map [Wattenberg, 1999; Shneiderman and Wattenberg, 2001] is an extension of the tree map idea, aimed at avoiding excessively narrow strips. Essentially, a "squarified" version of a tree map is generated by employing heuristics at each level, as shown in Figure 2.8.

2.1.4 Radial Approaches

Sunburst [Stasko and Zhang, 2000; Stasko et al., 2000] builds on the concept of Information Slices (described in Section2.4). It uses a full disc rather than a semi-disc and provides different kinds of interactive fan-out of subtrees.

CyberGeo Maps [Holmquist et al., 1998; Skog and Holmquist, 2000] use a stars and galaxy metaphor to lay out pages of a web site. First, a manually edited hierarchical categorisation is composed, roughly corresponding to the directory structure of the web site. The root of the hierarchy corresponds to the sun at the centre of the solar system. Dots (stars) representing web pages are placed at orbits around the centre, depending on how many links away they are from the home page. Figure 2.10 shows a CyberGeo Map of a web site as used in the WebAware project. To avoid collisions, a simple hash function based on directory names and random element distributes the stars in their orbits. Levels of the hierarchy form concentric rings around the root, but stars which are near to one another in the visualisation are not necessarily similar in content.

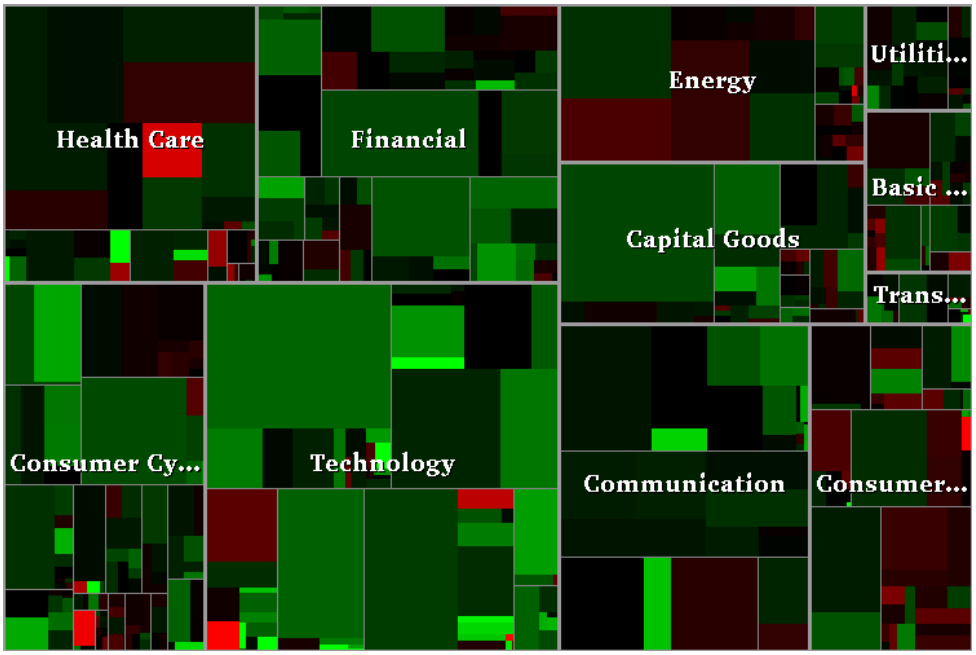


Figure 2.8: A market map of US stocks generated on 17th September 1999. Falling stocks are colour-coded red, rising stocks are green.

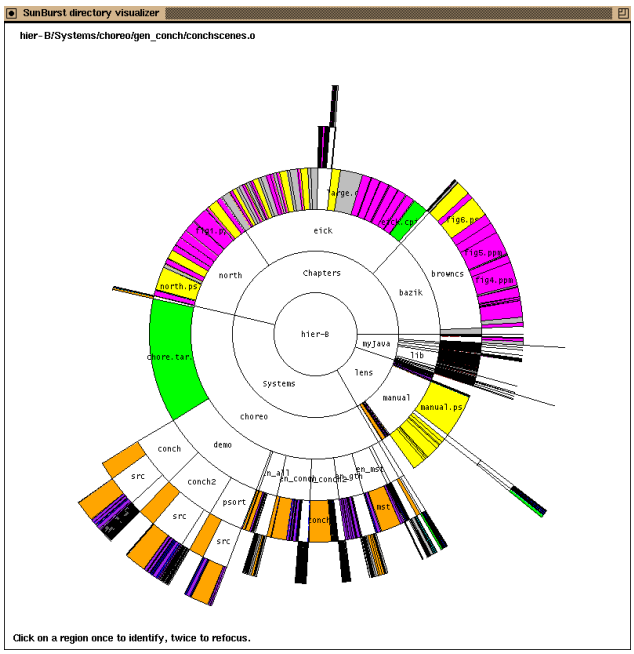


Figure 2.9: Sunburst uses a full radial representation of a tree and provides different kinds of interactive fan-out. [Image used with kind permission of John Stasko, Georgia Tech.]

Figure 2.10: A CyberGeo Map as used in the WebAware project to shows levels of activity on a web site on a public display. [Image extracted from Proc. CHI 2000. Copyright © by the Association for Computing Machinery, Inc.]

2.1.5 Cone Trees

The cone tree [Robertson et al., 1991, 1994], shown in Figure 2.11, is the classical 3d conical representation of a hierarchy. The vertical layout makes labelling of nodes somewhat difficult. A horizontal layout called a cam tree allows better labelling of nodes, as shown in Figure 2.12. Interactive animation of transitions supports object constancy – the human visual perception system can track movement of objects and does not have to reassimilate the scene after each transition. Shadows on the floor plane are used to reinforce the sense of depth.

2.1.6 Landscapes

The File System Navigator (FSN) [Tesler and Strasnick, 1992; Strasnick and Tesler, 1996a,b] is a 3d landscape visualisation of a file system. Files sit atop grey pedestals, subdirectories recede into the background, as shown in Figure 2.13. The height of file icons is logarithmically proportional to their size in bytes. The colour coding indicates the age of a file. Users can freely navigate through the landscape and an overview map can be displayed. It is also possible to search for files and have matching files highlighted in the landscape with a spotlight. FSN gained worldwide attention when it appeared in the film Jurassic Park. It was also used in SGI's MineSet product to visualise decision trees.

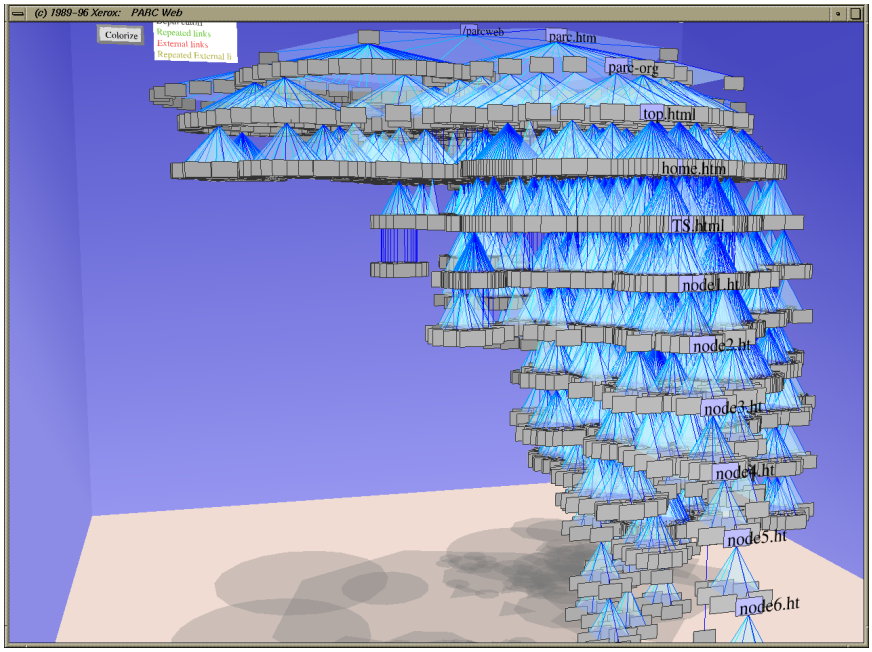


Figure 2.11: A cone tree showing the structure of the Xerox PARC web site. Subtrees can be interactively opened and closed. When a node is selected, that part of the tree rotates around in a smooth animated transition to bring the selected node to the front of the display. [Image used with kind permission of Stuart Card, PARC.]

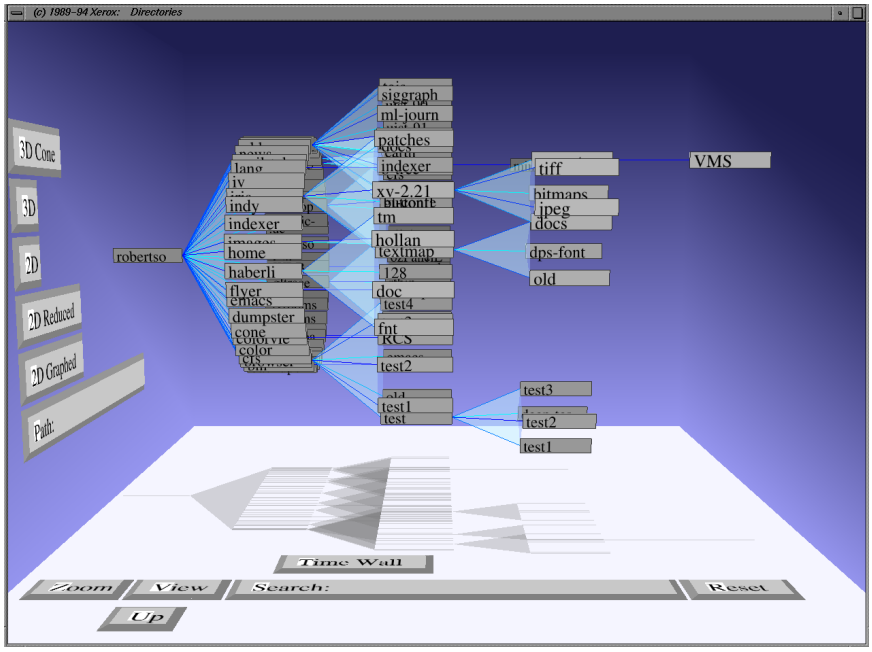


Figure 2.12: A cam tree of a file system directory structure. Note the shadows on the floor reinforcing the sense of depth. [Image used with kind permission of Stuart Card, PARC.]

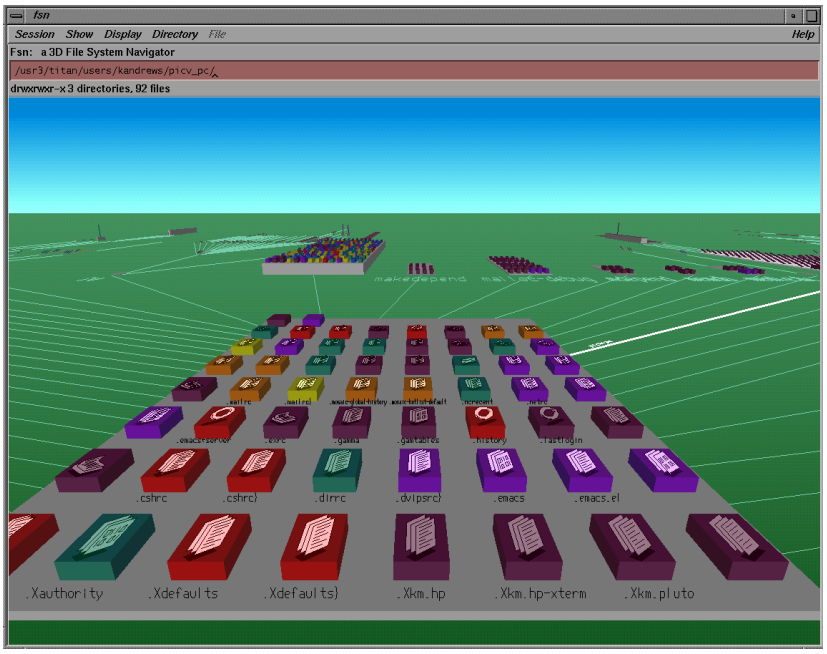


Figure 2.13: An FSN landscape visualisation of a file system hierarchy. Files sit atop pedestals, subdirectories recede into the background.

2.1.7 Hyperbolic Browsers

The Hyperbolic Browser [Lamping and Rao, 1994; Lamping et al., 1995; Lamping and Rao, 1997] is probably the most well-known information visualisation technique. Using a layout based on hyperbolic geometry, the hyperbolic browser produces a space-filling, focus plus context visualisation. Initial layout is done on a hyperbolic plane, a feature of which is that it provides unlimited space in each direction, so the entire hierarchy can always be laid out. A mapping then transforms the layout to the unit disc, which is displayed on screen. Each child places its children in a wedge of space, as shown in Figure 2.14. The software is now sold as a component by Inxight, who also have an online demo [Inxight, 2002].

Similar to the 2d hyperbolic browser, the 3d hyperbolic browser [Munzner and Burchard, 1995] makes use of 3d hyperbolic space for layout, which is then mapped to the surface of a unit sphere. For web sites or more general graphs, a spanning tree is generated and laid out and any cross-links are displayed on request, as shown in Figure 2.15. First presented in 1995 [Munzner and Burchard, 1995; Munzner, 1997], software and videos are available from [Munzner, 2002].

The Walrus system builds on the 3d hyperbolic browser but uses Java and Java3D [Hyun, 2002].

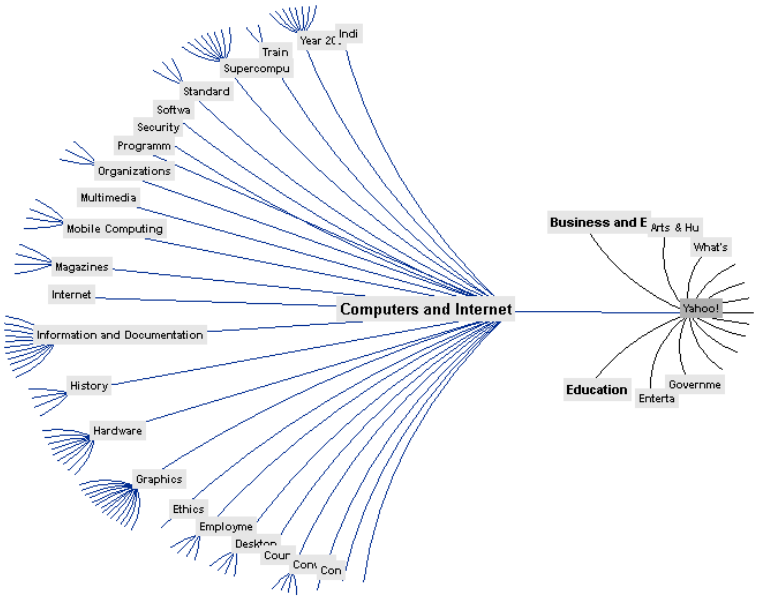


Figure 2.14: The hyperbolic browser always displays the entire hierarchy. However, subtrees around the edge of the disk become so small they are invisible. Here the top levels of the Yahoo hierarchy are shown.

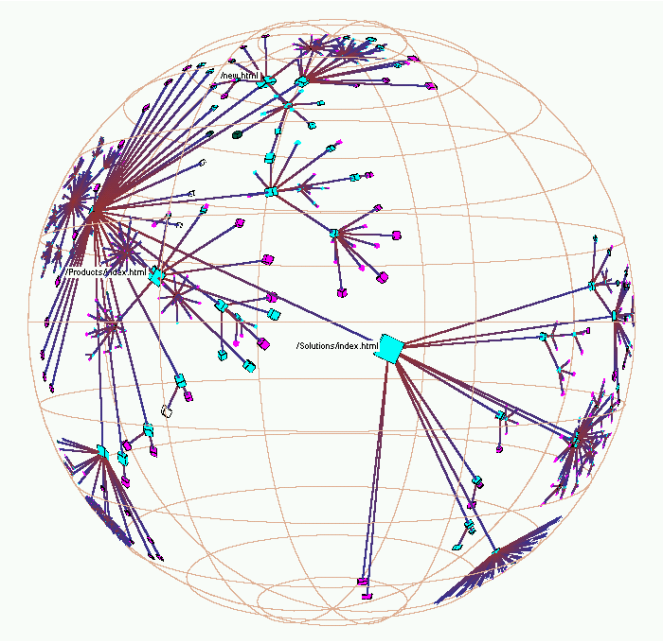


Figure 2.15: The H3 3d hyperbolic browser.

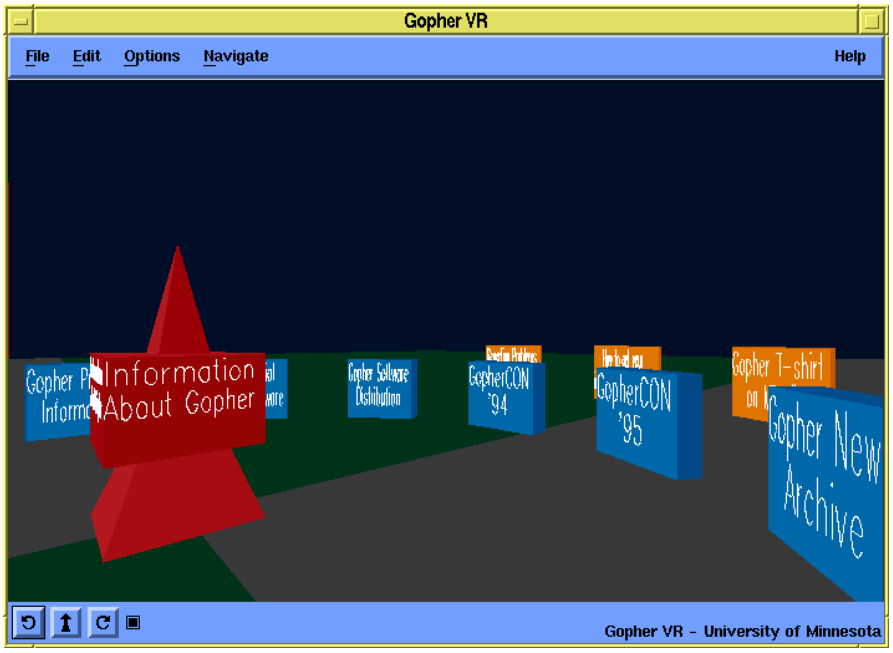


Figure 2.16: GopherVR visualises one level of a Gopher hierarchy at a time. The central pyramid bears the name of the current level, clicking on it returns the user to the next higher level.

2.1.8 Other Approaches

GopherVR [McCahill and Erickson, 1995; Goerzen, 2002] is a browser for Gopher servers. Gopher is a hierarchical information system which preceded the web. GopherVR provides a 3d landscape visualisation of *individual* levels of a Gopher hierarchy. Members of a collection are arranged in a stonehenge-like circle, as shown in Figure 2.16. Gopher search result sets are visualised in a spiral, spiralling out from centre with decreasing relevance to the query. A variation [Iacovou and McCahill, 1995] allows the possibility to hand-place items, for example in order to manually group related items. The main problem with GopherVR is that it only visualises one level of the hierarchy at a time and users can easily become disoriented.

The Cheops [Beaudoin et al., 1996; CRIM, 2002] system uses stacked triangles to compactly display a hierarchy. As shown in Figure 2.17, triangles representing children at a particular level are overlaid (squashed together) horizontally to save on screen space. Navigation is performed from top to bottom: only those children of the currently active branch are available for selection at a particular level (shown in medium blue).

The Magic Eye View [Burger, 1999] is part of the SInVis system [Kreuseler and Schumann, 1999; Kreuseler et al., 2000]. The hierarchy is first laid out in 2d space according to the classic Reingold and Tilford [1981] or Walker [Buchheim et al., 2002] algorithms. It is then mapped geometrically onto the surface of a hemisphere, as shown in Figure 2.18. Smooth animated transitions are possible. The visual effect is very similar to the patented hyperbolic browser, but hyperbolic geometry is not used.

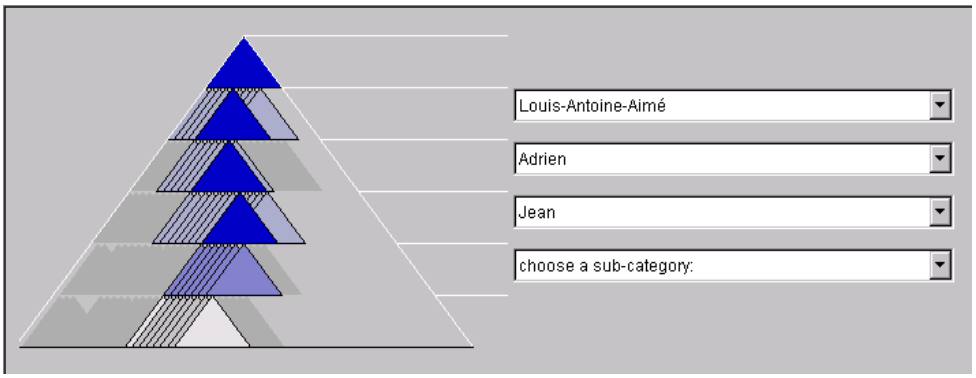


Figure 2.17: Cheops uses stacked triangles to compactly display a hierarchy. Navigation proceeds top-down: only the children of the currently active branch, shown here in medium blue, are available for selection at any one time.

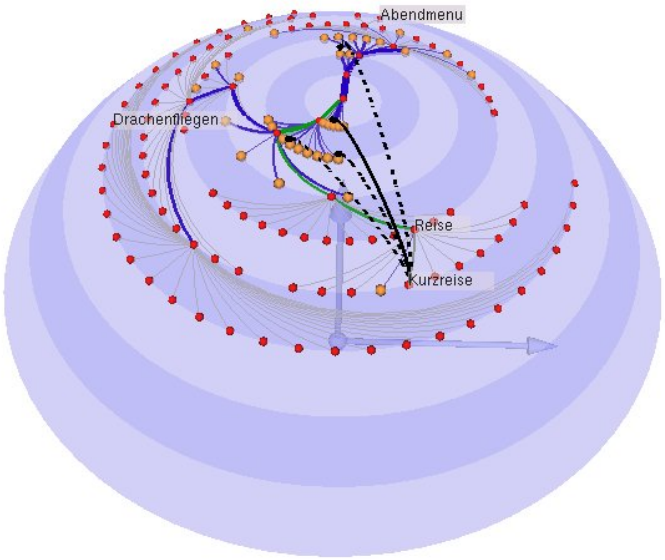


Figure 2.18: The SInVis Magic Eye View uses a geometric mapping from a simple 2d tree layout to the surface of a hemisphere. [Image used with kind permission of Matthias Kreuseler, University of Rostock.]

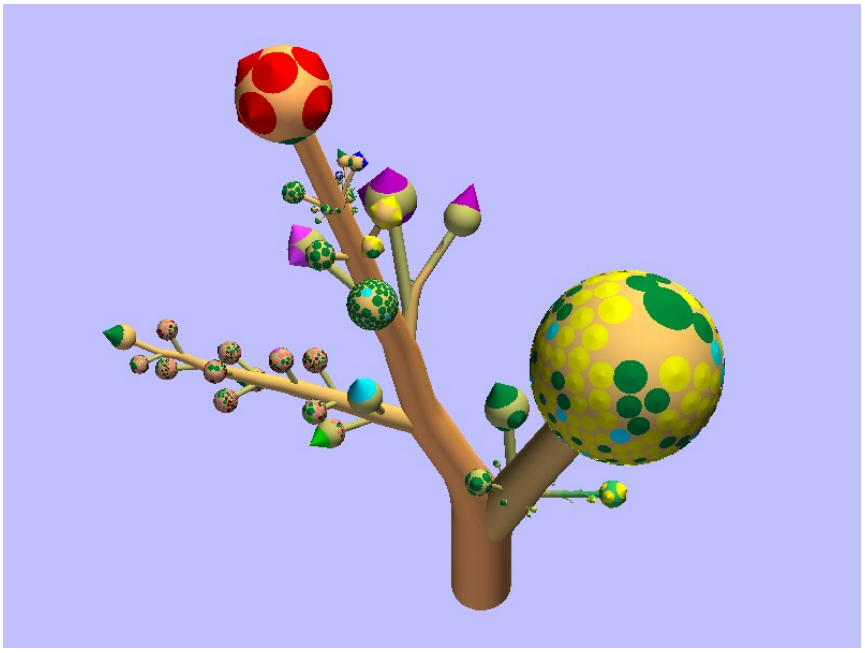


Figure 2.19: Botanical visualisation of a hierarchy. Leaf nodes are displayed as fruit. [Image used with kind permission of Jack van Wijk, Eindhoven University of Technology.]

Botanical Visualisation [Kleiberg et al., 2001] transforms a hierarchy into a geometric model of branches and leaves, which is then rendered. For better aesthetics, continuing branches are emphasised, long branches are contracted, and leaves are shown as fruit.

2.2 Harmony Collection Browser

Harmony [Andrews, 1995; Andrews and Kappe, 1994] was the X Windows browser client and authoring tool for Hyper-G [Andrews et al., 1995; Maurer, 1996], an information server which provided numerous information management and visualisation tools. The Hyper-G research prototype has since been turned into a successful commercial product under the name Hyperwave [Hyp, 2002].

Hyper-G supports a-priori organisation of information into hierarchies (called *collections*), provides rich sets of standardised meta-data, and a standard representation for search results. In addition, Hyper-G uses a separate link database for hyperlink facilities, has typed links, and has fully integrated search facilities. With such rich supporting infrastructure is it possible to generate tightly-coupled, multi-faceted visualisations of information spaces.

The Harmony Collection Browser [Schipflinger, 1998] is a traditional tree browser, but with an additional feature known as *location feedback*. Selecting a document in the search result set or any other Harmony window opens up the path(s) to that document in the collection browser, providing the user with a sense of the context of that document without having to open it. This

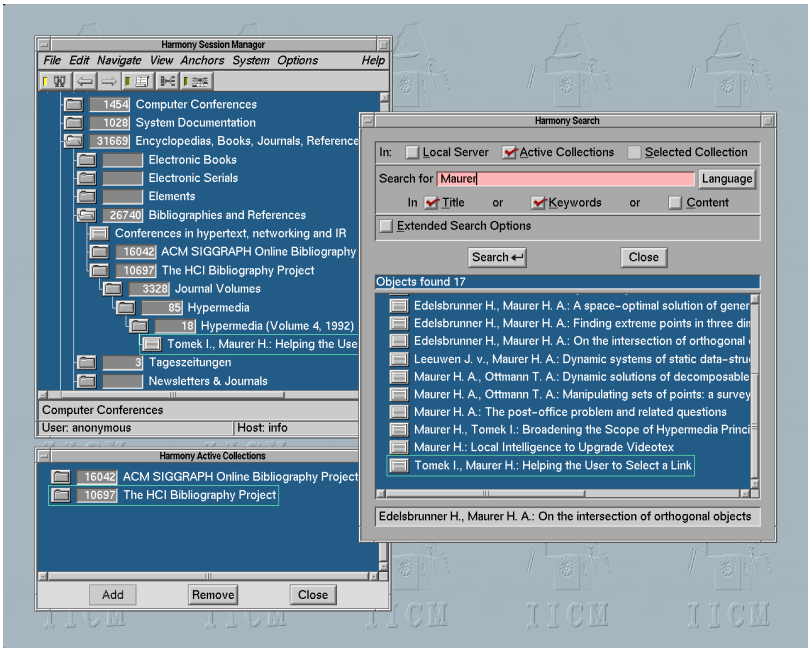


Figure 2.20: The Harmony Collection Browser. Clicking on the document "Helping the User to Select a Link" in the search result set opens up the path(s) to the document in the collection browser.

can be seen in Figure 2.20.

2.3 Harmony Information Landscape

The Harmony Information Landscape [Eyl, 1995; Wolf, 1996; Andrews, 1996; Andrews et al., 1996; Andrews, 1999] is a *rich information landscape* for visualising the hierarchical structure of Hyper-G (and Gopher) spaces on a plane in three-dimensional space. Similar to FSN [Tesler and Strasnick, 1992], documents sit atop pedestals and subcollections recede into the background. Underlying meta-data is used to encode visual attributes of various elements in the landscape display. Going beyond FSN, hyperlink relationships may be selectively superimposed upon the hierarchy map and documents returned by search queries may be selectively "plotted" in the landscape (location feedback), indicating their whereabouts in a broader context.

As can be seen in Figure 2.21, Harmony's Information Landscape is an interactive, three-dimensional visualisation of the Hyper-G collection structure. The collection hierarchy is mapped out onto a plane, documents within a collection or cluster are arranged on top of the corresponding block. The standard default mapping of document attributes to visual representation is to encode document type through the shape of the corresponding 3d icon, document size as the size of the icon, and use colour either to reinforce the document type or to convey the age of documents.

Collections can be opened and closed and documents accessed through mouse clicks, menu

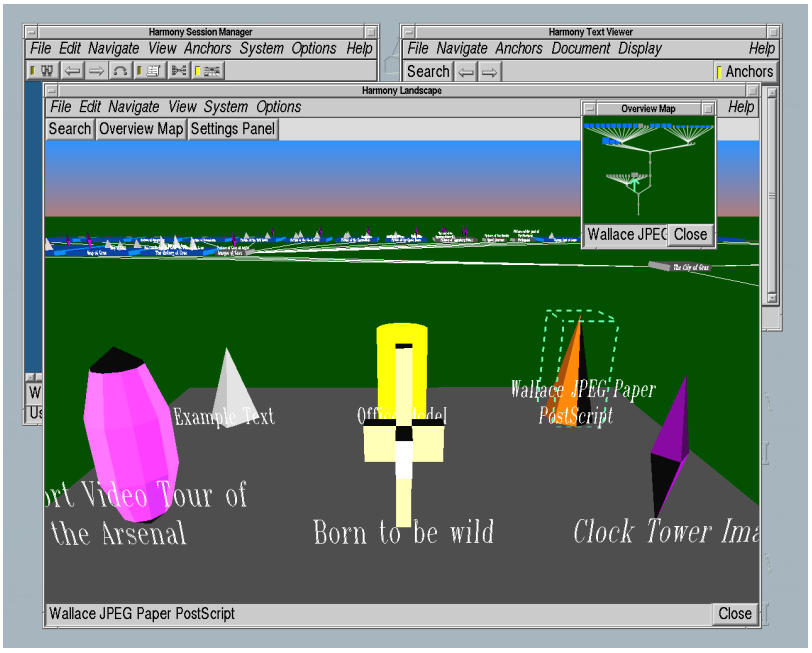


Figure 2.21: The Harmony Information Landscape. Blocks on the plane represent collections. The default style represents documents as simple 3d glyphs. Note the overview window in the top right.

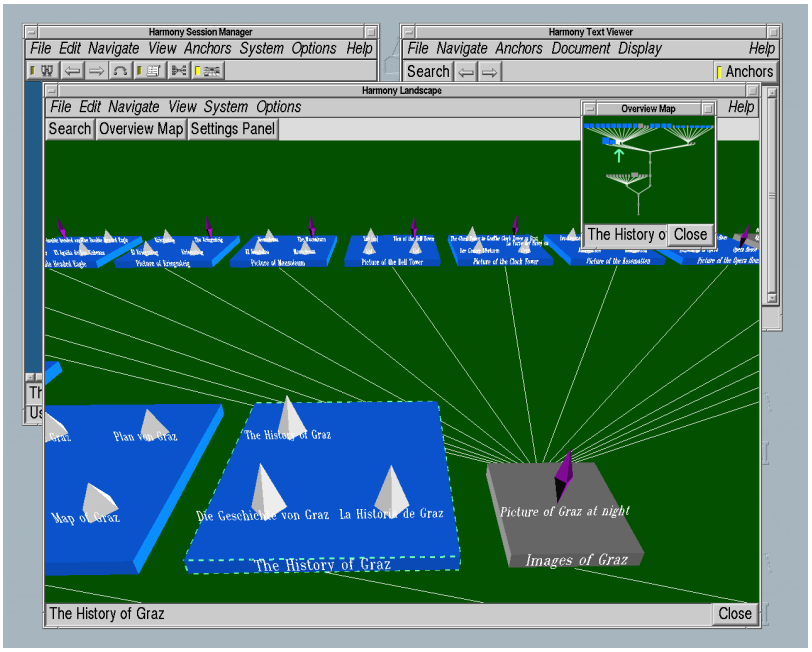


Figure 2.22: Harmony Information Landscape. Document clusters containing text and image documents. A *cluster* in Hyper-G is a closely bound set of documents, such as different language versions of the same text.

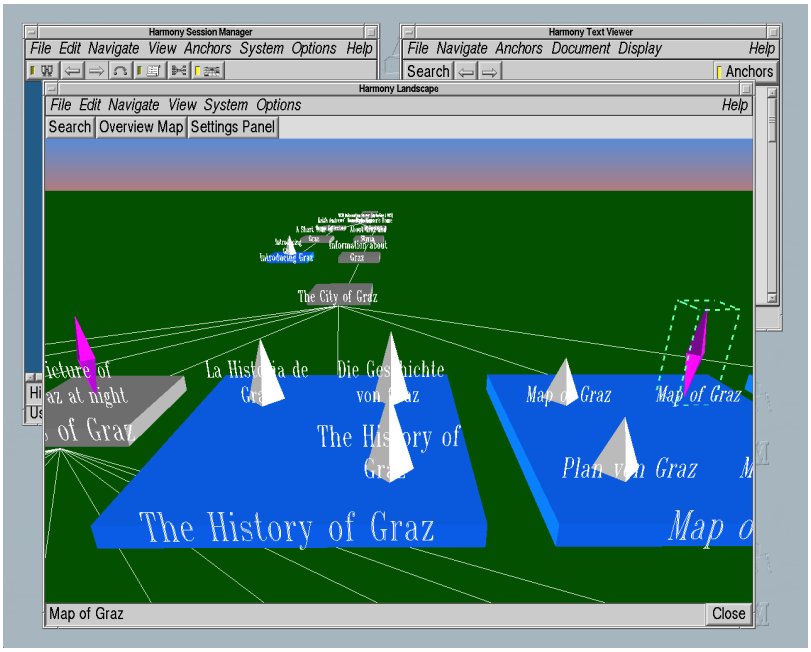


Figure 2.23: Harmony Information Landscape. Looking back to the parents of a collection.

commands, and keyboard shortcuts. As more collections are opened, the landscape fans out to accommodate them (see Figure 2.22). Middle-clicking a collection or document symbol flies the user gently to down to that object. Right-clicking brings up Harmony's attribute window to display the object's metadata. The overview window provides both orientation and an alternative form of navigation.

The Harmony Information Landscape is tightly coupled with other components of Harmony. Single-clicking an object in a search result list or in Harmony's Local Map (see Section 3.2) activates *location feedback*: the path(s) to the object are opened up in the landscape, giving users visual feedback about the context of its use, its type, and its size, before any decision on whether to retrieve it.

The user may wander freely in any direction. Figure 2.23 shows the view looking back to the parents of a set of objects. The textual labels in the display are rotated and tilted as necessary to match the user's viewing position and angle.

Figure 2.24 shows more complicated glyphs being used to represent individual document types. The 3d icon chooser can be used to select an appropriate glyph, similar choosers are available to select colours and fonts.

Figure 2.25 shows an experiment in the use of textured information landscapes. Thumbnail images pasted to document icons could be used to give an indication of the document's content. For example, a thumbnail of the actual image in an image document or a thumbnail of the first page of a PDF document could be pasted onto the document's glyph in the landscape, giving users a preview of the document's contents.

Another planned feature, which however was not implemented before the project finished,

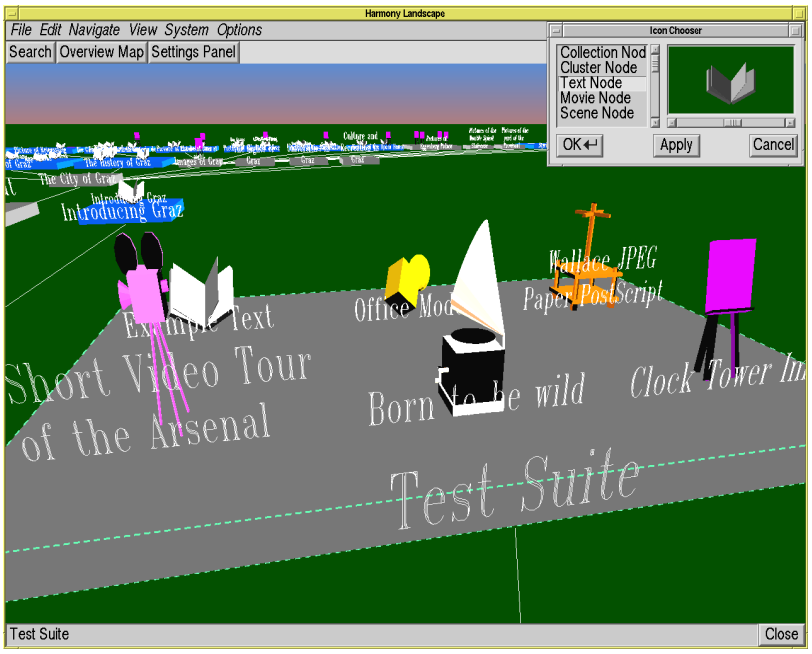


Figure 2.24: Harmony Information Landscape. A geometrically more complicated set of document icons and the 3d icon chooser (top right).

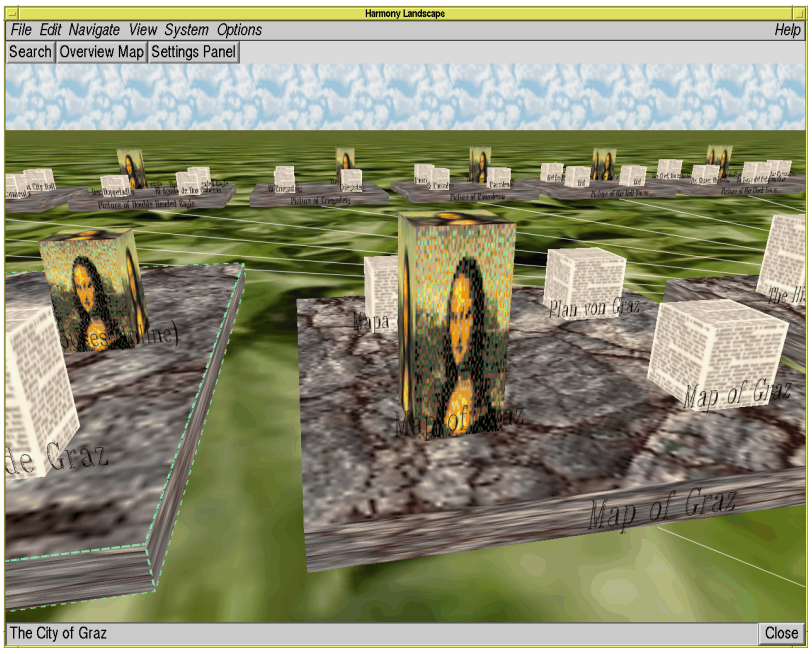


Figure 2.25: Harmony Information Landscape. A textured version of the information landscape. Thumbnail images pasted to nearby document icons give an indication of their content. “Landmark” 3d icons representing whole collections could be visible from afar.

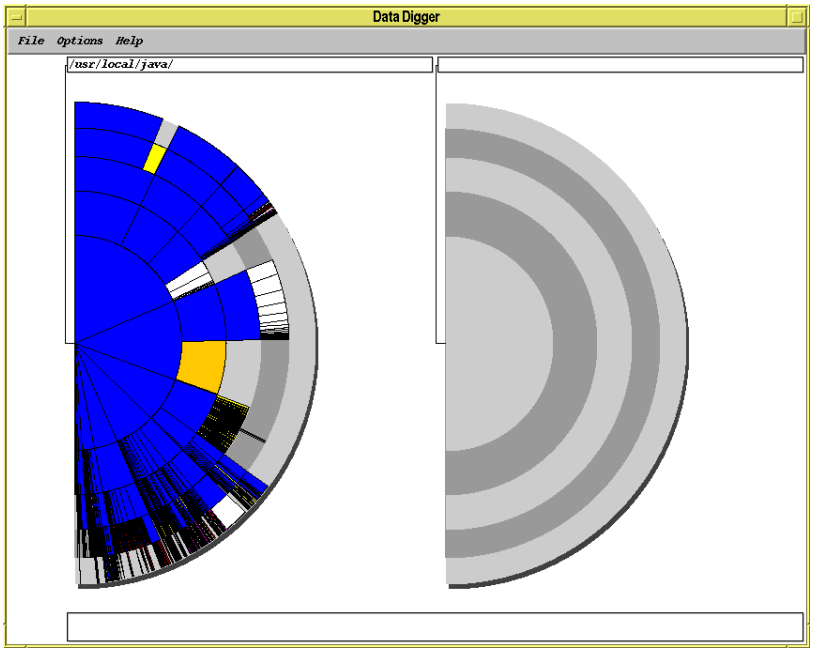


Figure 2.26: Information Slices visualisation of the JDK 1.1.6 distribution, showing the root directory and five levels of its subdirectories.

was the ability to assign a “landmark” 3d icon to a collection, to represent the collection as a whole and which would be visible from afar. For example, a model of the Eiffel Tower might be assigned to represent a collection about Paris.

A further significant innovation in the Harmony Information Landscape is the combined display of hierarchical structuring and hyperlink relationships in one visualisation, the Harmony Local Map 3D, which is described in Section 3.3.

In summary, the Harmony Information Landscape provides a rich information landscape, which builds on the lavish underlying infrastructure provided by a Hyper-G server. It presents combined structure and link maps, and uses Hyper-G’s metadata to encode visual attributes of various elements in the landscape display.

2.4 Information Slices

Information Slices [Andrews and Heidegger, 1998; Andrews, 1998] are an approach to visualising large hierarchies which works especially well for deep hierarchies. The information slices technique uses one or more semi-circular discs to compactly visualise large hierarchies in two-dimensional space. Each disc represents multiple levels of a hierarchy; typically between 5 and 10 levels are visualised per disc, the exact number being user configurable. Deeper hierarchies are represented using a cascading series of discs. At each level of the hierarchy, the children are fanned out in the available space according to the total size of each child.

A prototype of information slices was implemented for visualising the hierarchical tree struc-

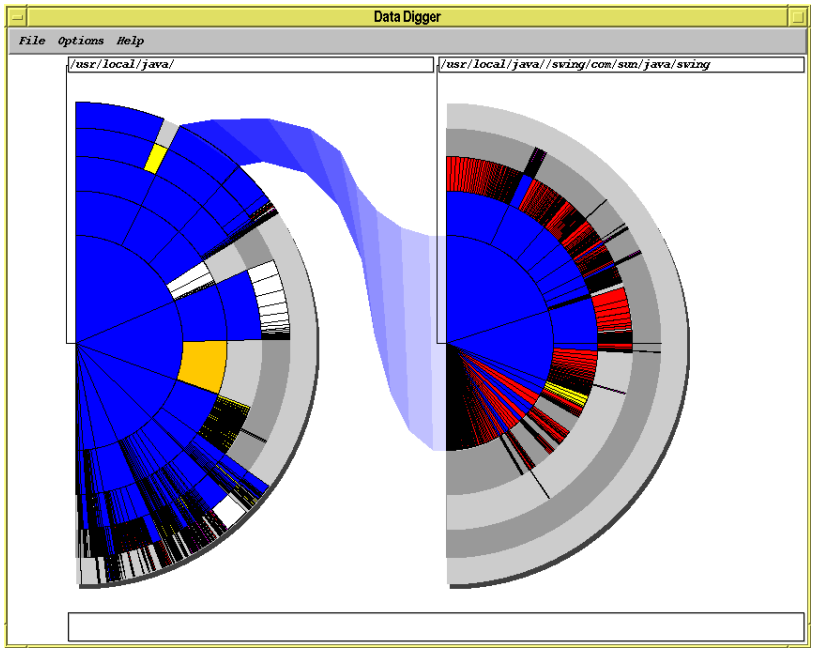


Figure 2.27: Information Slices. Expanding the `swing/com/sun/java/swing` subdirectory into the righthand disc.

ture of directories and files in a file system. Figure 2.26 shows the information slices representation of Sun Microsystem's JDK 1.1.6 distribution for Solaris, which has 6158 files, arranged in 502 directories, with a maximum depth of 9. The lefthand disc shows the top 5 levels of the JDK hierarchy, whilst the righthand disc is as yet unused.

Figure 2.27 shows the display after the user has expanded the `swing/com/sun/java/swing` subdirectory into the righthand disc. Directories are coloured blue and other files are colour-coded according to their type. Further expanding the `text/html` subdirectory causes the leftmost disc to be iconified and slide off the screen to the left, the right disc slides over to the left, and a new disc is opened up on the right, as shown in Figure 2.28. This cascading chain of discs supports rapid exploration of very deep hierarchies. An iconified disc can be restored by clicking upon it.

The user can interactively configure how many levels of hierarchy should be displayed on each disc, using the Options panel shown in Figure 2.29. For example, in Figure 2.30, the user has selected 8 levels of hierarchy to be displayed in each disk.

By default, the space allocated to a directory's slice is proportional to the total size in bytes of files belonging to it and all of its subdirectories. The Options panel allows the user to change this mapping to, for example, the total number of files contained in a branch of the tree. Furthermore, the order of listing each child from top to bottom can be changed from descending order of size (the default) to alphabetical by name or chronological by modification time. Figure 2.31 shows the JDK 1.1.6 distribution, where the size of each slice is determined by the total number of files, and children are listed alphabetically by name.

The prototype file system visualiser using information slices is written in Java and uses Java's

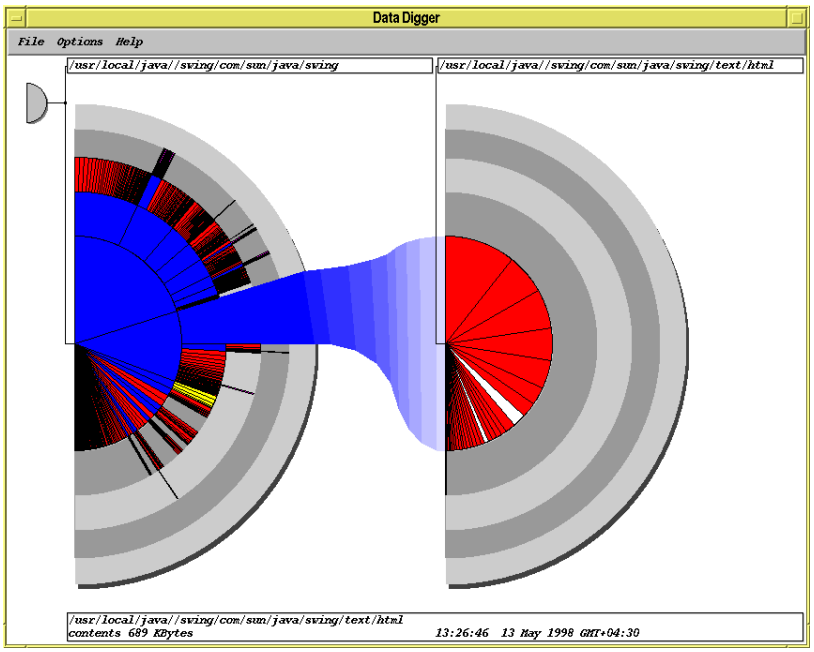


Figure 2.28: Information Slices. Further expanding the `text/html` subdirectory causes the lefthand disc to be iconified off to the left of the screen and a new disc to be opened to the right.

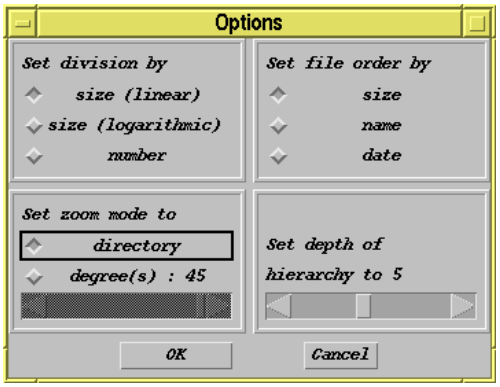


Figure 2.29: Information Slices. The Options panel allows various setting to be configured by the user interactively.

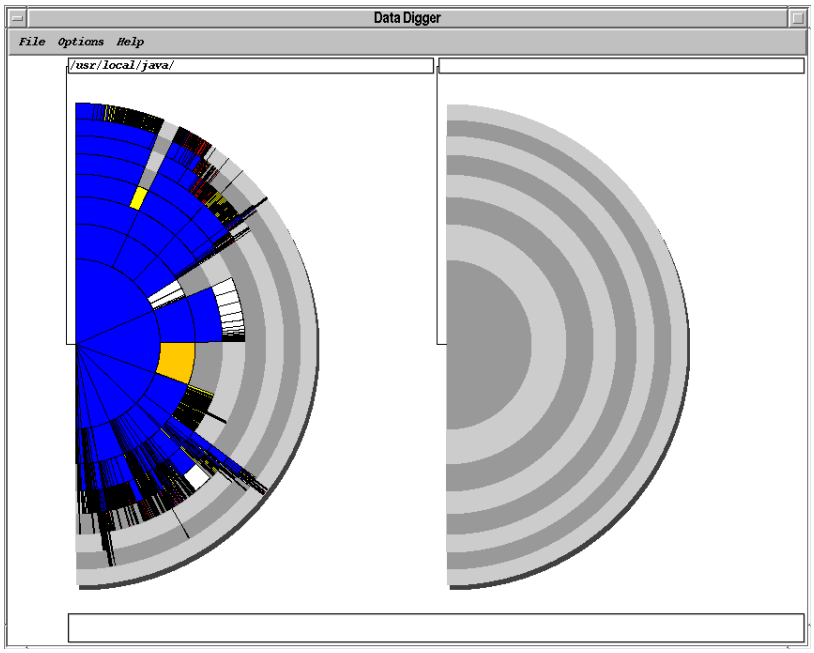


Figure 2.30: Information Slices. The JDK distribution shown on a single disc of 8 levels.

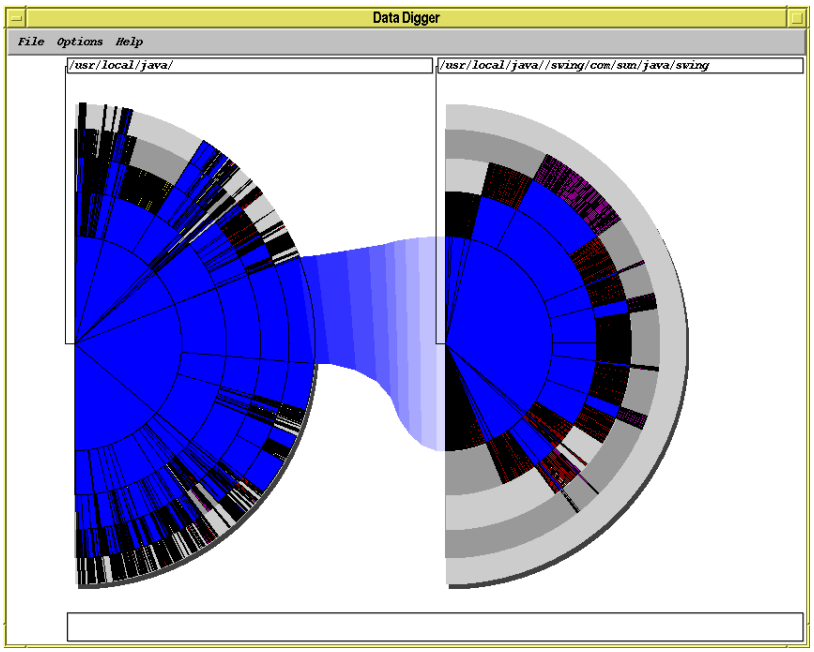


Figure 2.31: Information Slices. The JDK distribution, where the size of each slice is determined by the total number of files, and children are listed alphabetically by name.

Swing GUI components. From early experience and formative usability testing, the information slices technique appears to be particularly well-suited to the rapid navigation of deep hierarchies. It is very easy to rapidly traverse many levels of a hierarchy. It is also easy to gain an overview of the relative sizes of parts of a tree.

Broad hierarchies, however, can result in dense, thin slices, which are sometimes initially overwhelming. In such cases, users can select particular (dense) slices of interest and fan them out in 180 degrees of their own in the righthand disc. A form of "handles" for interactive fanout of selected slices might help alleviate this problem.

Currently, work is underway to integrate an information slices disc with a conventional tree browser (the JTree component of Swing), so as to combine the advantages of both techniques. Users will hopefully be able to gain an overview and rapidly traverse in the disc, and then explore in detail in the tree browser. As space permits, the names of files and directories will also be displayed in the corresponding sector of the disc, in addition to being displayed in the status bar. Stasko and Zhang [2000] have already developed the some of these ideas further.

Although the example application of information slices presented here refers to the tree structure of files and directories in a file system, the technique is of course applicable to any kind of hierarchical structure.

2.5 Information Pyramids

The information pyramids approach [Andrews et al., 1997; Wolte, 1998; Andrews, 2002] utilises three dimensions to compactly visualise large hierarchies. The first prototype of information pyramids was the 3D file system browser [Andrews et al., 1997], shown in Figure 2.32. It is implemented in Java and uses OpenGL for its 3d graphics output. A plateau represents the top of the hierarchy (or root of the tree). Other, smaller plateaus arranged on top of it represent its subtrees. Separate icons are used to represent non-subtree members (leaves) of a node such as files or documents. The general impression is that of pyramids growing upwards as the hierarchy grows deeper.

The next version, called the 3D Explorer [Wolte, 1998] and shown in Figure 2.33, features a much more extensive user interface and textual labels. By default, the space allocated to a directory's plateau is proportional to the total number of files belonging to it and all of its subdirectories. The grey plateaus indicate subdirectories whose contents have not yet been fully scanned.

In order to maintain interactive frame rates, the quality of rendering gracefully degrades during movement. Drawing performance is monitored, and less detailed representations are used when time becomes scarce, as shown in Figure 2.34. Once the user comes to a halt, the scene is rendered in its entirety once more.

Users can freely navigate around the pyramid landscape. As pyramids are approached, more details are revealed. In Figure 2.35 the user is approaching a close-up of the particular pyramid representing the "javarsc" subdirectory and its children. The ordering of children atop a plateau can be based on alphabetical, chronological, or other criteria. In Figure 2.36, colour coding is used to indicate a file's type, but could also be mapped to age, or any of a number of other

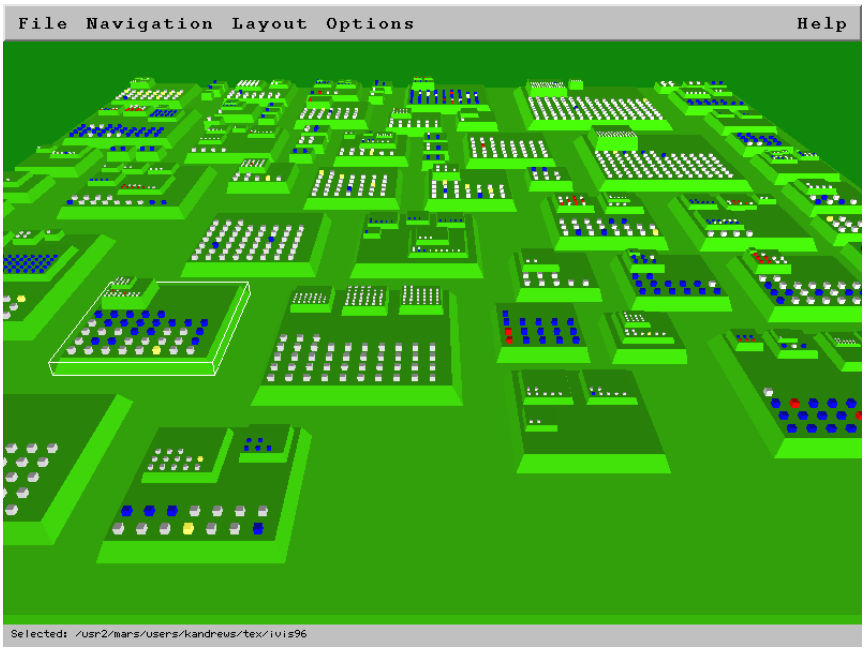


Figure 2.32: The original Information Pyramids prototype.

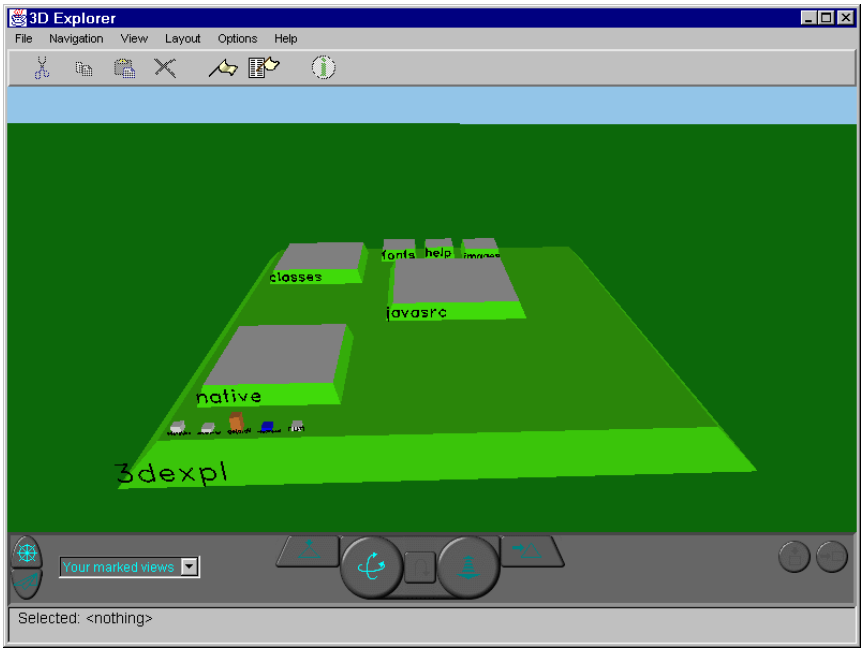


Figure 2.33: Information Pyramids. The 3D Explorer displaying the source and class files of its own implementation.

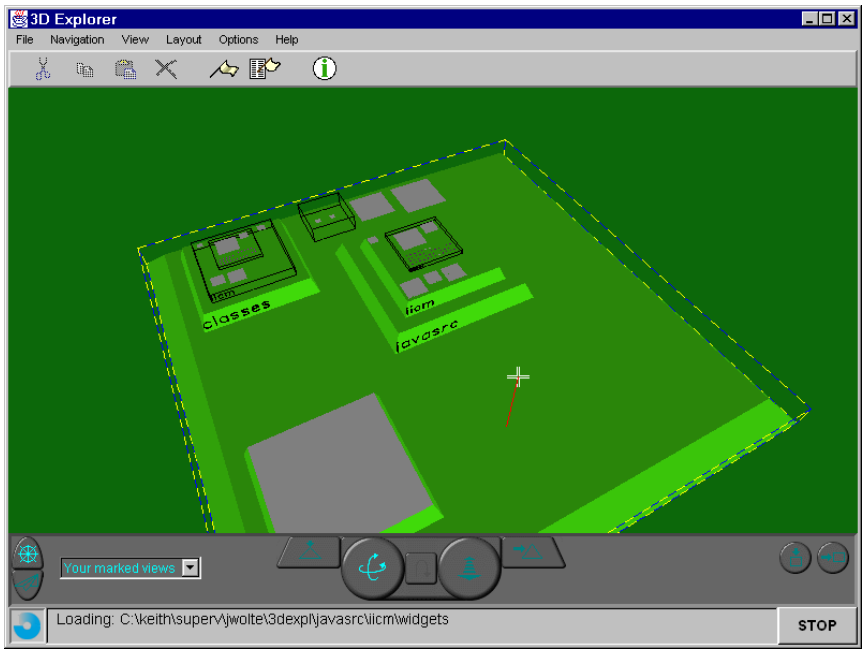


Figure 2.34: Information Pyramids. During navigation interactive frame rates are maintained by drawing in less detail or with wire frames, until the user again becomes stationary.

characteristics.

As the user approaches the “utils3d” subdirectory in Figure 2.37, its contents become identifiable and the file names become readable. Notice how the file name labels are inclined towards the user’s point of view. By default, the height of the glyphs representing files is logarithmically proportional to their size.

A bird’s eye view of the pyramid landscape provides a space-filling, freely zoomable overview. In this view, illustrated in Figure 2.38 for the Java JDK 1.2.2 distribution, it is easy to recognise (visually) which directories contain the largest number of files by their relative areas.

The visual representation is only one component of the interface. Equally important is the provision of suitable navigational facilities, in order support intuitive exploration of the hierarchy. In the 3D Explorer, users can choose between a wide variety of navigation facilities. Clicking on a file glyph or directory plateau flies the user smoothly to view that object. Special facilities in the navigation panel at the bottom of the screen allow users to freely navigate. Examine mode (the crossed semi-circular arrows in the middle of the navigation panel) allows the user to rotate and scale the pyramids visualisation, Explore (the flat striped arrow) mode allows users to freely move their own viewpoint within the visualisation, and Fly mode (the paper airplane at the bottom left of the navigation panel) allows the user to steer and control speed whilst flying over the visualisation. Two buttons provide direct access to a top view and a front view. Finally, the user can define customised viewpoints (placing flagpoles in the ground), which can be returned to at will.

Directories are scanned and loaded in a separate thread, so that users can navigate whilst

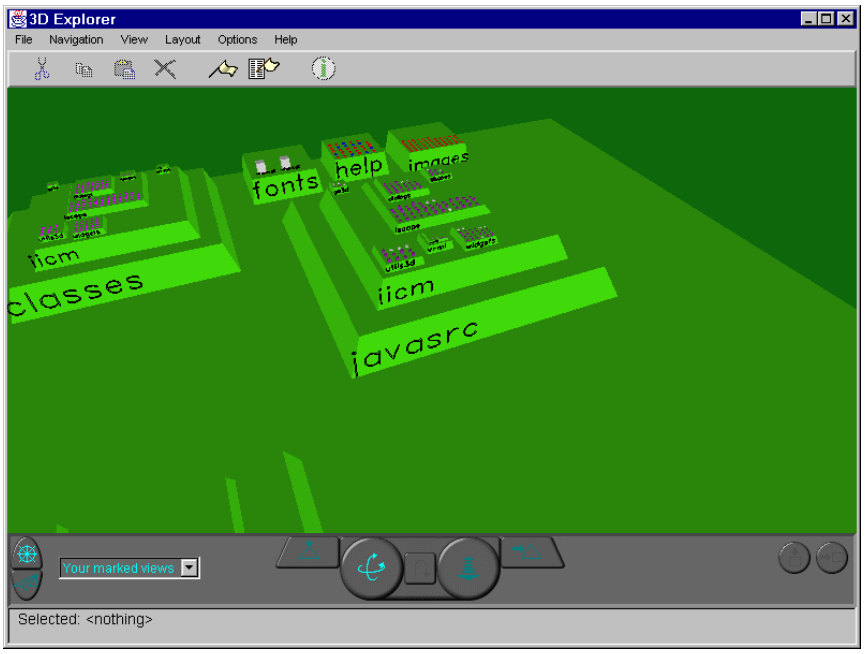


Figure 2.35: Information Pyramids. Zooming in on the subdirectory “iicm” of “javasrc”.

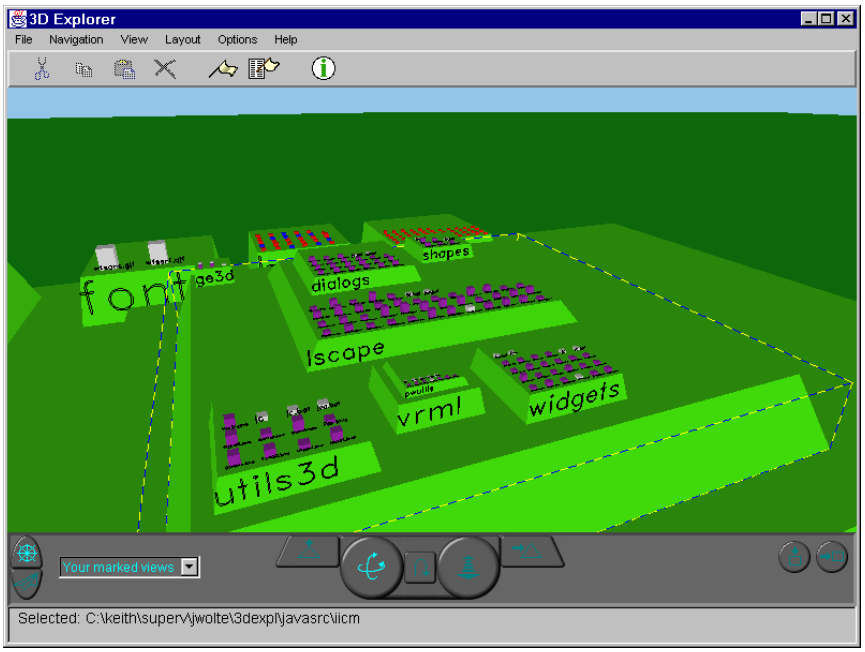


Figure 2.36: Information Pyramids. By default, files are colour-coded according to their extension. Here, for example, purple indicates “.java”, red “.gif”, and blue “.html”. The ordering of children atop a plateau can be based on alphabetical, chronological, or other criteria.

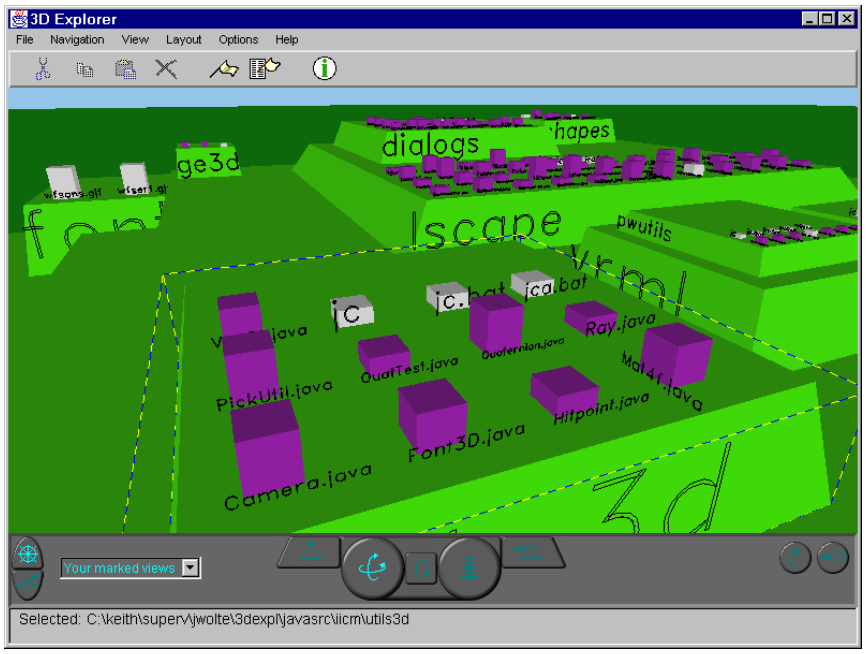


Figure 2.37: Information Pyramids. The file names of individual files are inclined towards the user's viewpoint and are now readable.

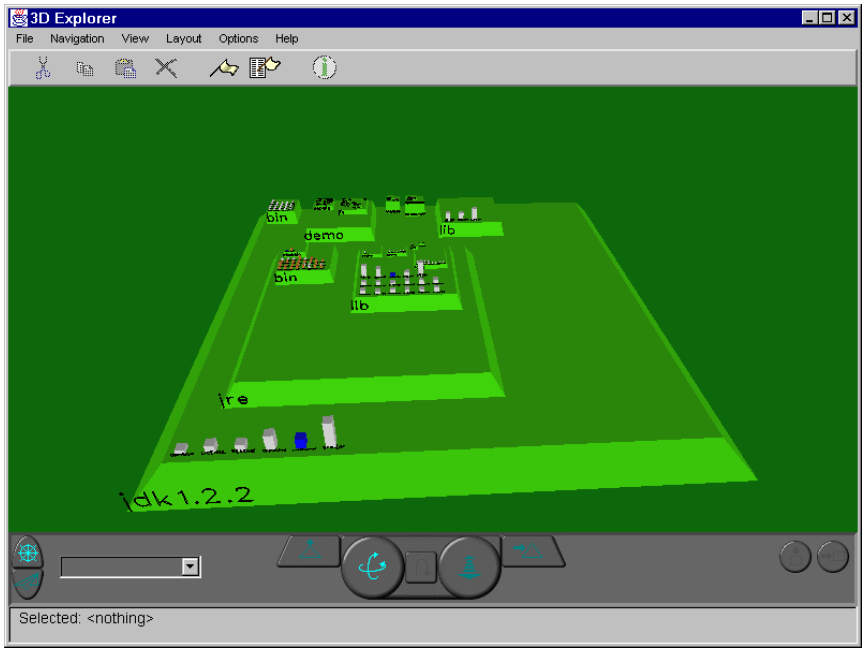


Figure 2.38: Information Pyramids. The hierarchical structure of the Java JDK 1.2.2 distribution, which contains 6 first level subdirectories, extending down to a depth of six. It can be seen at a glance that the subdirectory "jre" is by far the largest and that subdirectories "bin" and "jre/bin" contain many files.

the visualisation is being updated. In order to reduce clutter, users can “prune” the pyramid to a particular directory, making its plateau the current root of the pyramid, and (temporarily) eliminating all other elements from the display. The reverse operation, “unprune”, makes the directory’s parent and siblings visible once more.

Formative usability evaluation of the 3D Explorer suggested that the information pyramids technique scales well to both wide and deep hierarchies. Nodes deep within the tree are not (initially) individually visible but contribute to the overall proportions of their parents. Broad hierarchies result in many pyramids at each tier, but remain manageable. Users were positive in their comments and liked the overview which information pyramids provided.

Users did, however, have some difficulty in locating specific individual files of which they knew the location within the hierarchy. They also displayed a tendency to become immersed in the 3d navigational facilities and rather forget their initial intentions. Several users reported finding the textual labels to be too cluttered.

To incorporate some of the lessons learned, a third information pyramids hierarchy browser was implemented, called the Java Pyramids Explorer [Welz, 1999]. The main changes included:

- To facilitate directed browsing in a known hierarchy, a traditional tree browser (the Java JTree component) was paired and synchronised with the information pyramids browser.
- The extensive 3d navigational facilities were replaced with three simple sliders controlling the 3d view.
- For increased portability, the OpenGL 3d graphics was replaced with Java2d code implementing simple 3d projection.
- In order to reduce clutter, textual labels are now displayed in the margins of the pyramids view, and only for selected nodes and their parent, siblings, and children.

Figure 2.39 shows the Java Pyramids Explorer displaying the Java source tree of its own implementation. Figure 2.40 shows the Java JDK 1.2.2 distribution in the Java Pyramids Explorer. Here, individual files are not being displayed, only the directory structure.

Although the example prototype applications of information pyramids presented here refer to the tree structure of files and directories in a file system, the information pyramids metaphor is of course applicable to any kind of hierarchical structure: hierarchical classification schemes, decision trees, or the hierarchical structure of a web site derived by traversing hyperlinks from a selected initial page.

Work is currently ongoing to define and implement a framework for hierarchical visualisation. The framework will provide multiple, synchronised visualisations of a hierarchy, of which information pyramids are one visualisation component, and search facilities. In information pyramids, future work includes the implementation of user-configurable mappings between an object’s metadata (such as age, owner, or size) and its visual representation. It is also intended to explore the placement of items (files) on a plateau into clusters according to similarity of their content using force-directed placement techniques (see Section 3.1.3). One can also imagine visualising relationships between documents (such as hyperlinks) by stringing arced wires between them.

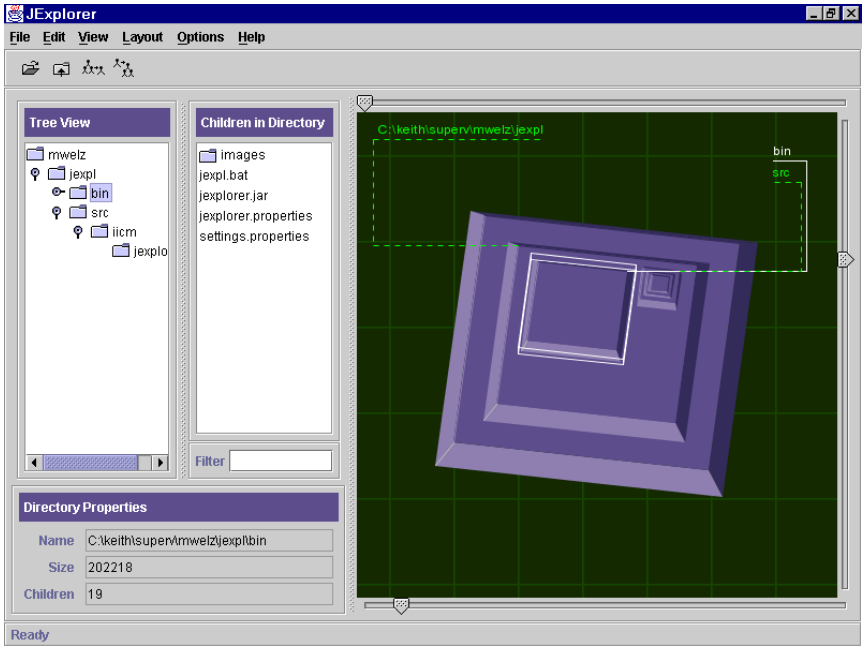


Figure 2.39: The Java Pyramids Explorer displaying the Java source tree of its own implementation. The tree view and pyramids view are synchronised.

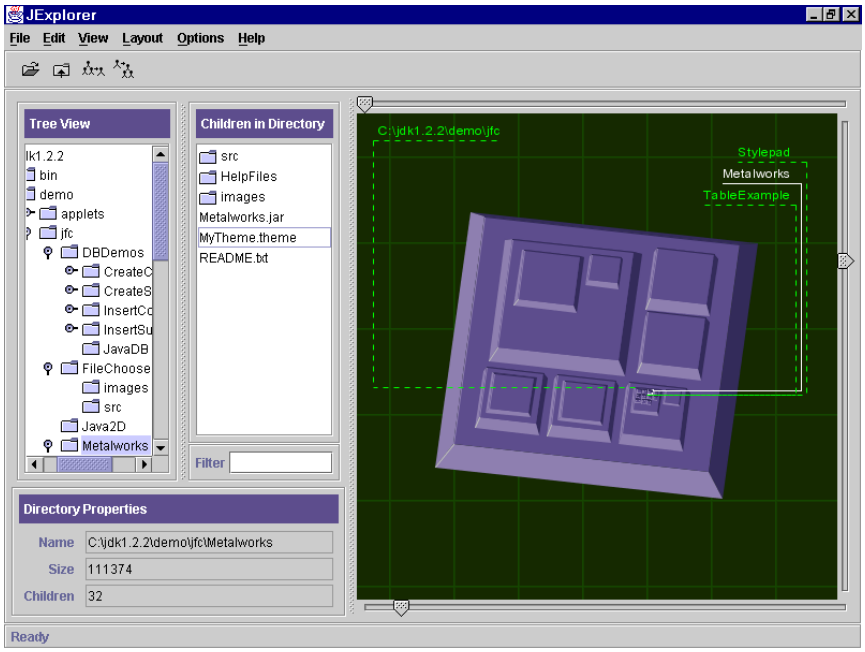


Figure 2.40: A view of the Java JDK 1.2.2 distribution in the Java Pyramids Explorer.

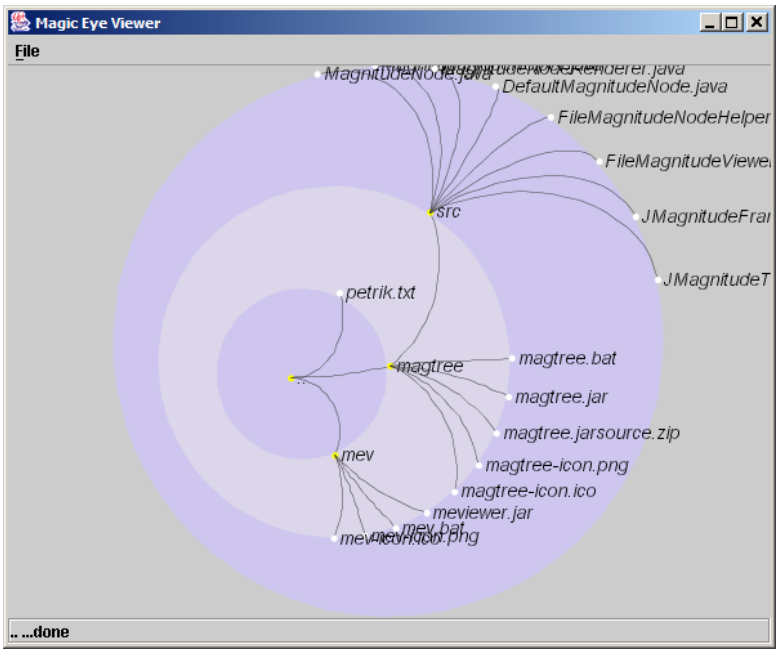


Figure 2.41: An initial implementation of a Magic Eye View at the IICM.

2.6 Magic Eye Viewer

In order to facilitate our own research into geometric projection of hierarchies, a version of a Magic Eye View was recently implemented at the IICM. The prototype shown in Figure 2.41 still needs further development and refinement, but already incorporates the basic algorithms of classic tree drawing in 2D and subsequent geometric projection onto a hemisphere.

2.7 Information Tunnel

The Information Tunnel is a prototype hierarchy browser along the lines of Bubble Trees [Boardman, 2000]. As can be seen in Figure 2.42, the idea is to present a miniature preview of each branch of the next level of the hierarchy to assist users in making educated choices. Clicking on a subtunnel starts an animated transition into that subdirectory. In Figure 2.43, the user has clicked on the subdirectory `keith`. Clicking on the central triangle moves back one level up the hierarchy.

The subtunnel miniatures are decorated with red and blue borders representing the relative size and number of files in that subdirectory. By default, subtunnels are displayed in alphabetical order, clockwise from inside to outside. This can be changed using the Options panel shown in Figure 2.44.

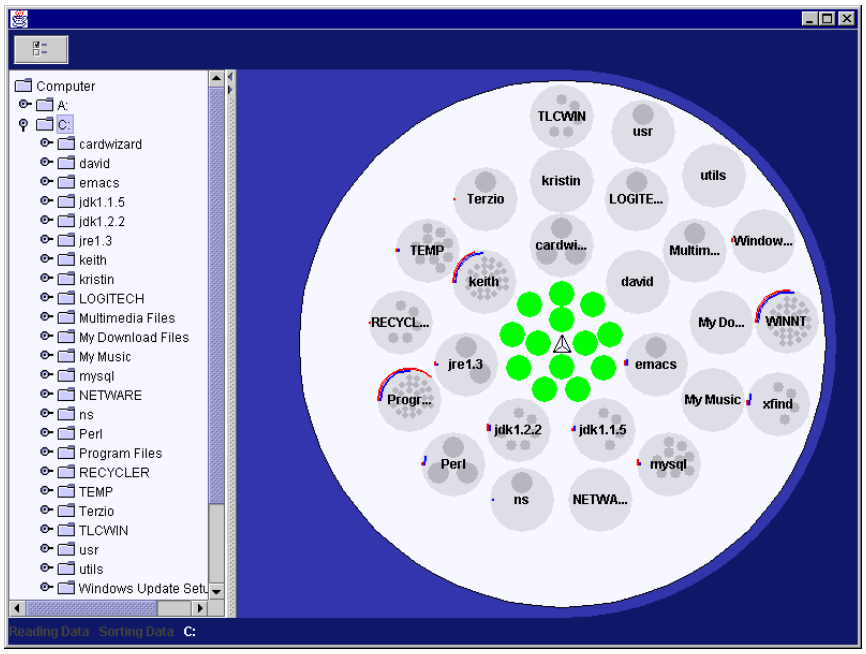


Figure 2.42: The Information Tunnel. Subtunnels lead off in various directions. Each subtunnel shows a preview of the contents of that subdirectory. Files and documents at this level are clustered around the centre.

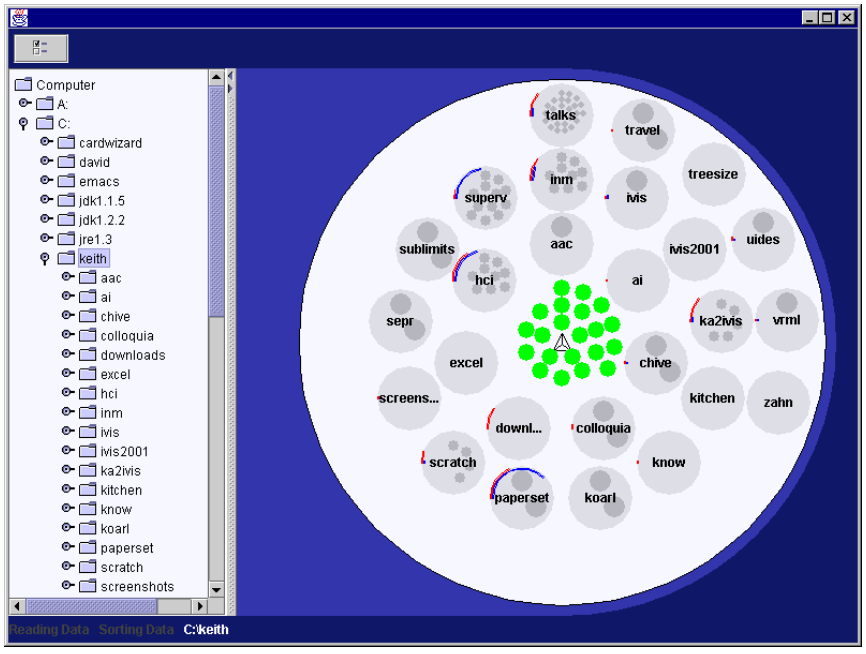


Figure 2.43: Information Tunnel. The borders of each subtunnel optionally display red and blue rings indicating the relative size in bytes of that subtunnel (red) and the relative number of files in the subtunnel (blue).

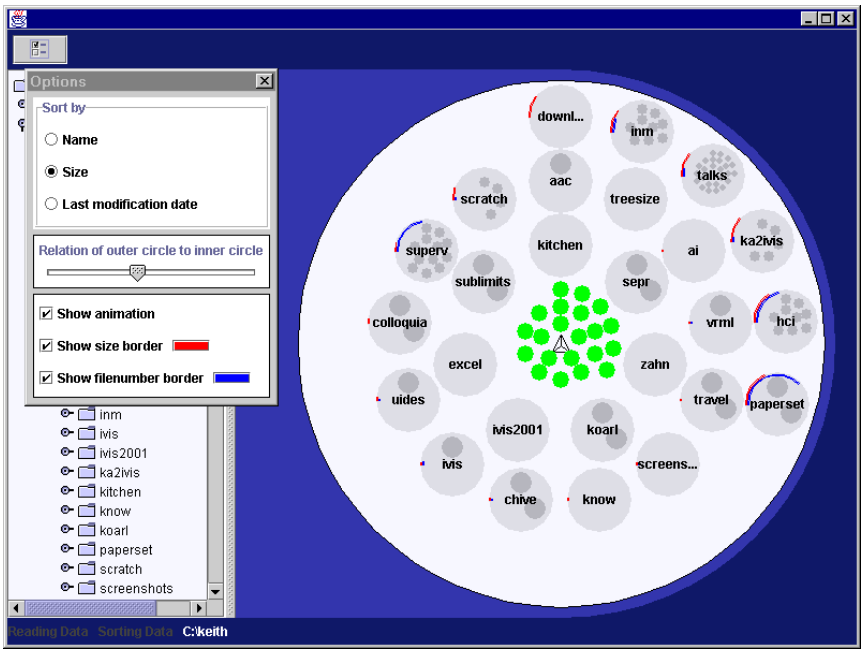


Figure 2.44: Information Tunnel. The transitions between display are smoothly animated. Here, the ordering of subtunnels has also been changed from alphabetical to increasing size (clockwise from inside to outside).

2.8 InfoSky Cobweb

The InfoSky [Kappe et al., 2002; Andrews et al., 2002] visual explorer is under development at the Know-Center [Know-Center, 2002] in cooperation with Hyperwave [Hyp, 2002] and the IICM, Graz University of Technology [IICM, 2002]. InfoSky integrates a traditional tree browser with a novel cobweb hierarchy visualisation based on Voronoi diagrams. Force-directed placement is used to position documents at each level of the hierarchy. The first prototype is shown in Figures 2.45 and 2.46. The dataset comprises approximately 109,000 news articles from the German daily newspaper, the Süddeutsche Zeitung, which have been manually classified into a hierarchy of some 6,900 collections and sub-collections upto 15 levels deep.

As can be seen in Figures 2.45 and 2.46, the hierarchical structure of the newspaper archive is accessible through either the lefthand tree browser or the righthand cobweb view. Individual documents are represented by stars; documents which are similar are placed near each other using a force-directed placement algorithm recursively at each level of the hierarchy.

Each collection has a collection centroid, a vector which represents the agglomeration of all the documents contained within the collection and all of its subcollections. To determine the layout of the cobweb, subcollection centroids are first positioned within their parent's polygon according to their similarity and then weighted Voronoi polygons are used to draw the boundaries between sibling subcollections. The space allocated to each collection is related to the total size of that collection. Full details of these algorithms are given in Section 5.3.

User testing showed that very small polygons in the initial prototype were very hard to dis-

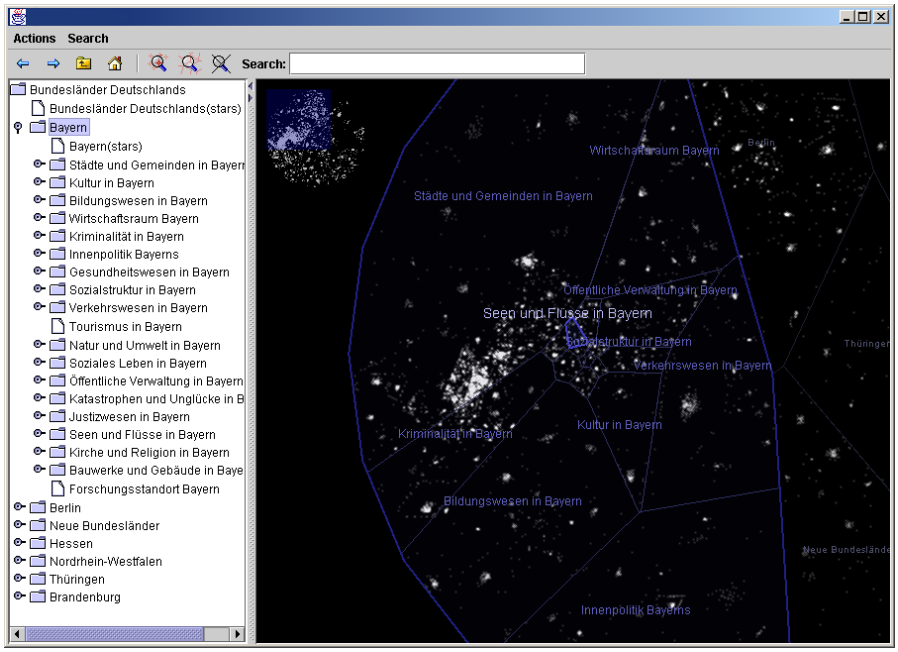


Figure 2.45: The InfoSky Cobweb visualisation of a hierarchy. On the left a traditional tree browser, on the right each child is given an area within its parent determined by a Voronoi diagram.

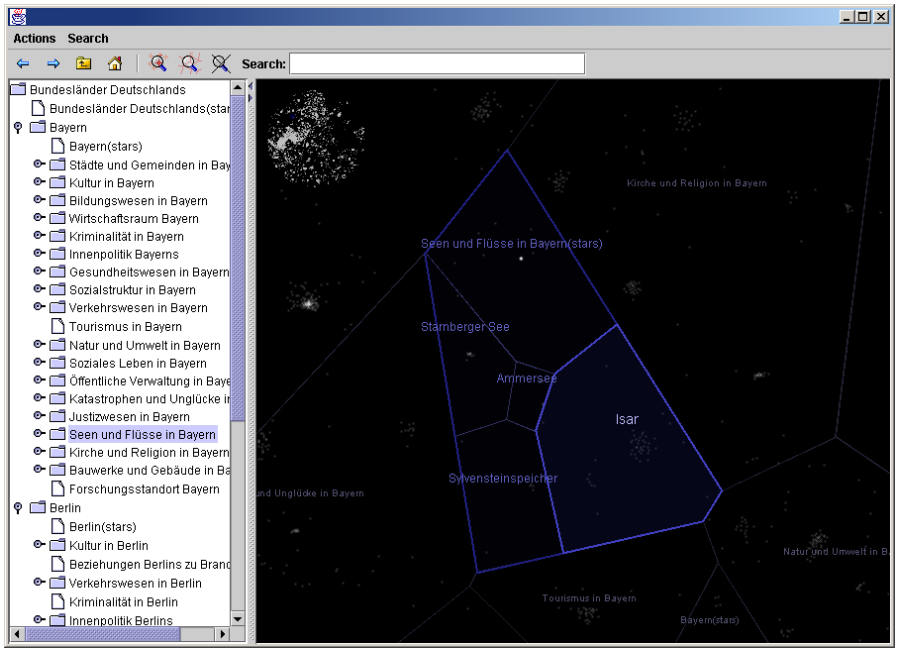


Figure 2.46: InfoSky Cobweb. Opening the child “Seen und Flüsse in Bayern” (Lakes and Rivers in Bavaria) reveals its inner structure.

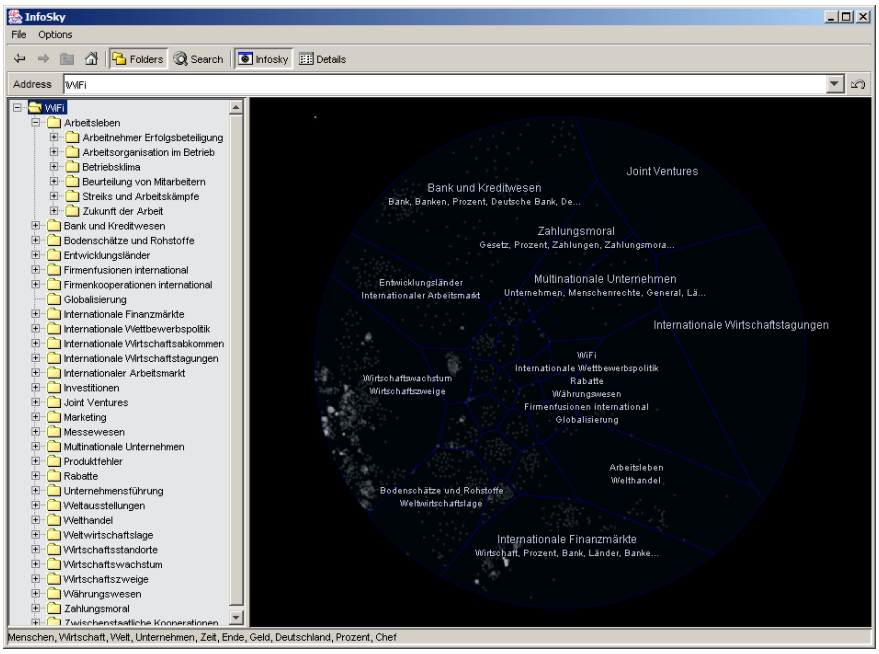


Figure 2.47: InfoSky Cobweb. The most recent version incorporates a minimum size for the Voronoi polygons to prevent the generation of very small polygons.

tinguish. Figure 2.47 shows the most recent version, where the Voronoi subdivision algorithm was modified to introduce a larger minimum size for the polygons.

Chapter 3

Visualising Networks

Networks are general graph structures, comprising sets of nodes connected by edges. Whereas a tree has order from top to bottom and no cycles are allowed, a graph has arbitrary connections between nodes. Examples of networks include web pages and the links between them, a map of the London Underground with stations connected by tube lines, or semantic networks of semantic concepts and the relations between them. The more complex such graphs become, the more helpful, and indeed necessary, techniques to visualise them become.

3.1 Related Work

The field of Graph Drawing has been exploring potential techniques for visualising networks for many years and three good textbooks are now available [di Battista et al., 1999; Kaufmann and Wagner, 2001; Sugiyama, 2002]. Herman et al. [2000] also provides a good overview of current research. Some of these methods are introduced here.

3.1.1 Using Hierarchical Visualisation Techniques

The first way to visualise general graphs is to reduce them to a tree and then use the techniques for visualising hierarchies presented in Chapter 2. Trees are simply a special case of general graphs: a tree is a connected acyclic graph with the property that there is a unique path from the root of the tree to each vertex. Several methods exist for extracting a spanning tree from a graph. The spanning tree can be visualised and any additional edges from the original graph added to the visualisation afterwards. An example of a system which does this is H3 [Munzner, 1997], which was discussed in Section 2.1.7.

3.1.2 Layered Methods

Layered methods are used for drawing directed graphs, which are graphs having a general flow or direction. As described in Kaufmann and Wagner [2001, Chapter 5], layered drawings are usually designed to meet the following aesthetic criteria:

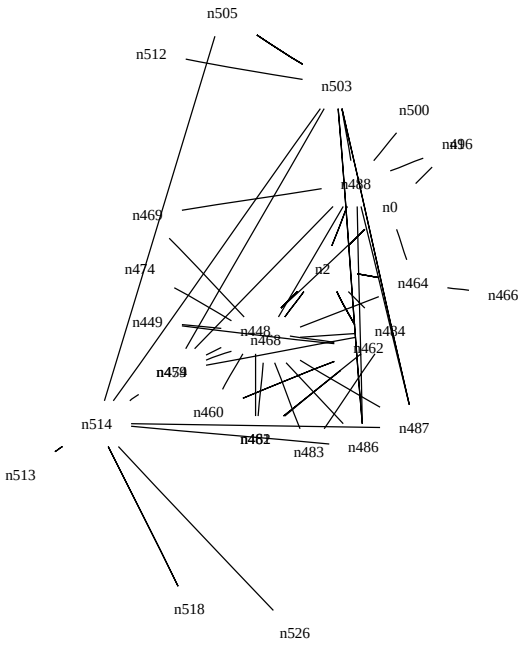


Figure 3.1: A graph with 30 nodes and 70 edges. This is the graph `ldbxtried.dot` from the AT&T Graphviz distribution [AT&T Labs - Research, 2002].

1. Edges pointing upward should be avoided.

2. Nodes should be evenly distributed and long edges should be avoided.

3. There should be as few edge crossings as possible.

4. Edges should be as straight and vertical as possible.

The approach is usually divided into four steps, initially proposed by Sugiyama et al. [1981], each addressing one of the criteria above:

1. *Cycle Removal*: As few of the edges as possible are reversed, so as to make the graph acyclic. At the end of the layout, these edges are changed back again.

2. *Layer Assignment*: The vertices are assigned to layers such that all edges point downwards. To ensure that edges do not extend over multiple layers, dummy nodes are inserted.

3. *Crossing Reduction*: For each layer, an ordering of the vertices is computed, such that the number of edge crossings is minimised.

4. *X-Coordinate Assignment and Edge Placement*: The horizontal positions of vertices within a layer are adjusted so that they do not overlap other real or dummy vertices. Finally, edges are drawn as polylines or curves.

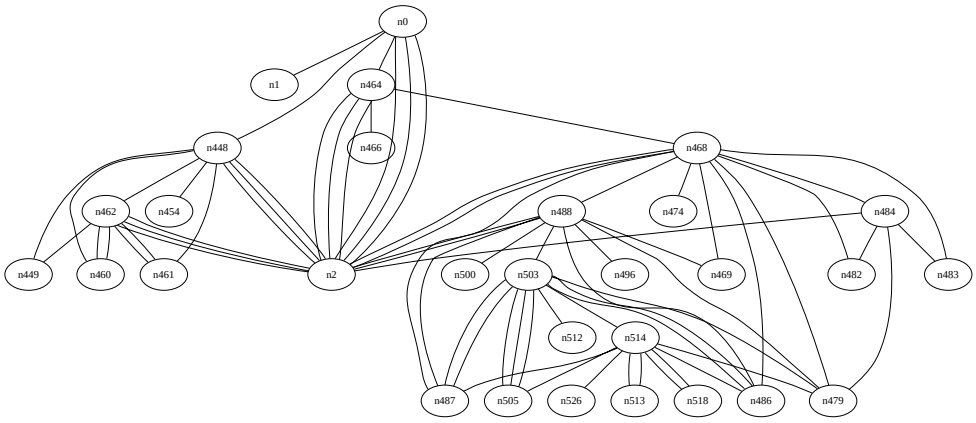


Figure 3.2: The final layout produced with the Graphviz layered algorithm and curved edges.

The graph shown in Figure 3.1 is the graph `ldbxtried.dot` from the AT&T Graphviz distribution [AT&T Labs - Research, 2002] and has 30 nodes and 70 edges. Using Graphviz's layered algorithm produces the layout shown in Figure 3.2.

3.1.3 Force-Directed Placement

These methods draw on physical analogies and are applicable to general graphs, without any prior knowledge of structural properties. As described in Kaufmann and Wagner [2001, Chapter 4], the methods generally consist of two components:

- A *model* consisting of physical objects and interactions between them.
- An *algorithm* which approximately computes an equilibrium configuration of the system.

In the original *spring embedder* model of Eades [1984], nodes are connected by imaginary springs rather than edges. An additional repulsive force, inversely proportional to the distance between objects, acts to prevent objects from colliding. At each iteration, every object is moved a certain amount in a certain direction, according to the net vector sum of all forces acting upon it. As iterations continue, the system approaches a stable state. A measure of the total stress remaining in the system is taken after every iteration, and the algorithm terminates once this is below a certain threshold. Numerous researchers have since published improvements to the original model including Fruchtermann and Reingold [1991] and Chalmers [1996b].

An example of a system using the spring model is HyperSpace (formerly Narcissus) [Wood et al., 1995; Hendley et al., 1995]. HyperSpace uses a spring model to organise web pages in a self-organising structure. The number of links between pages provides the attractive force and nodes repel each other at close distances. The size of each node is proportional to the total number of links pointing to and from it. As can be seen in Figure 3.3 certain nodes are much larger than others, reflecting their high connectivity and role as hubs.

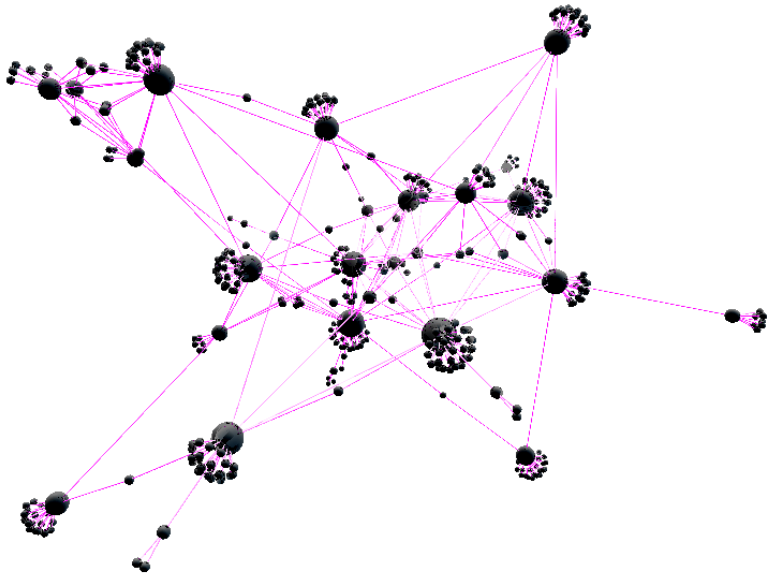


Figure 3.3: HyperSpace uses a spring model to lay out web pages and links between them. The size of each node is proportional to the total number of links pointing to and from it.

3.1.4 Energy-Based Placement

In the spring embedder model, a net force indicates which direction to move an object in order to reduce the forces acting upon it. This, in effect, is equivalent to minimising an implicit internal energy model of the system. *Energy-based placement* techniques attempt to minimise this residual energy directly. First, the potential energy of each spring is modelled. Then an objective function is defined which sums the potential energies over all springs. A numerical method such as Newton-Raphson is used to obtain a (local) minimum of the objective function, which then gives the positions of each node. This is essentially the same approach as used in *multidimensional scaling* [Cohen, 1997].

Simulated annealing is a related method for minimising an objective function. Given a candidate solution, a new solution is proposed by slightly modifying the current solution. If the new solution is better, it becomes the new candidate solution. If the the new solution is worse, it only becomes the new candidate solution with a certain probability depending on the *temperature* parameter. Convergence is enforced by slowly lowering the temperature to zero.

SemNet, published in 1988 [Fairchild et al., 1988; Wexelblat and Fairchild, 1991], was the first 3d information visualisation system and features a 3d spatial layout of a semantic network. Nodes are positioned in 3d space using a simulated annealing algorithm and connections between them are visualised as edges. Figure 3.4 shows a complete knowledge base of Prolog modules with nodes assigned to random positions. The relationships between them are hard to distinguish. Figure 3.5 shows the same knowledge base, but with the nodes positioned using a simulated annealing algorithm. The clustering and relationships are now much easier to discern.

Figure 3.4: SemNet. A complete knowledge base of Prolog modules with nodes assigned to random positions. [Image used with kind permission of Kim Fairchild.]

Figure 3.5: SemNet. The same knowledge base of Prolog modules after node positions have been determined by simulated annealing. The structure is more more readily apparent. [Image used with kind permission of Kim Fairchild.]

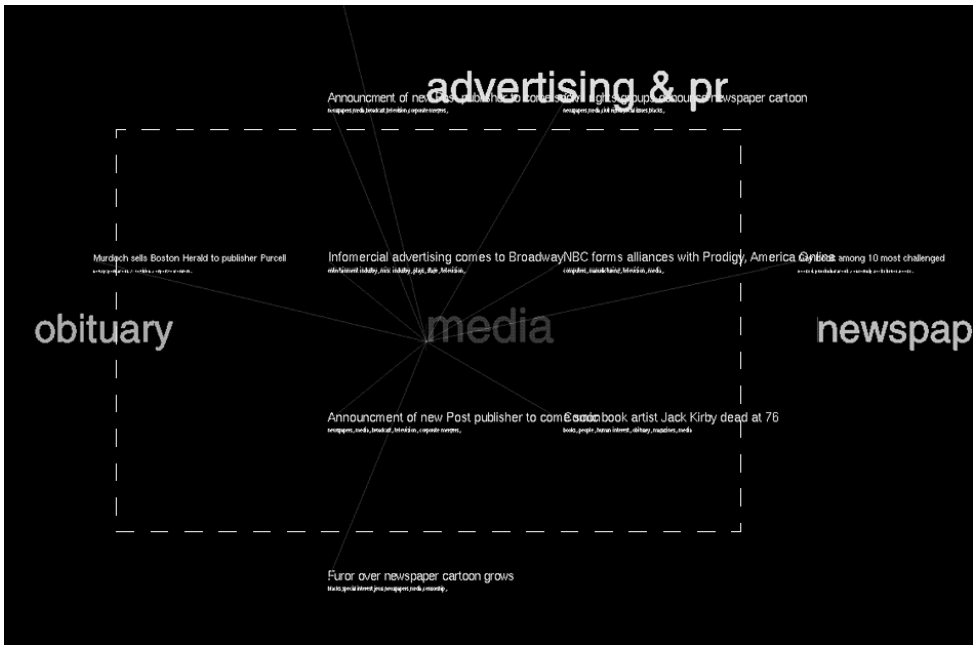


Figure 3.6: Galaxy of News. Fluid motion is provided between related concepts in a semantic network of newswire articles, but overall context is lost. [Image extracted from Proc. UIST '94. Copyright ©by the Association for Computing Machinery, Inc.]

3.1.5 Semantic Zooming

Semantic networks are networks of associations between concepts. Semantic networks are often too large to visualise all at once in a single visualisation. *Semantic zooming* techniques attempt to manage this complexity by visualising only a small part of the network at any one time, but providing for fluid motion between related concepts.

For example, Galaxy Of News [Rennison, 1994] constructs and then visualises a semantic associative relation network between related newswire articles. Semantic zooming is used to navigate the news article space. At first, a hierarchy of topical keywords from general to more specific is presented, which then lead into article headlines, and eventually to full news articles. This can be seen in Figure 3.6. However, the space is non-linear and changes as the user navigates, making it hard to maintain a sense of context and orientation.

The Brain [TheBrain.Com, 2002a,b; Harlan, 2000a,b] is a semantic zooming interface to pages on a web site. Figure 3.7 shows the brain-enabled site map at FoxSports.com. However the site map at FoxSports.com has since been taken down and replaced with a conventional site map.

ThinkMap [Plumb Design, 2002] uses a similar idea to navigate semantic networks. The Visual Thesaurus, shown in Figure 3.8, is a ThinkMap interface to the WordNet [Miller, 2002] thesaurus. Animated transitions take the user from one word to the next. The problem with all these interfaces is that, although they can be engaging and stimulating, it is very easy to become

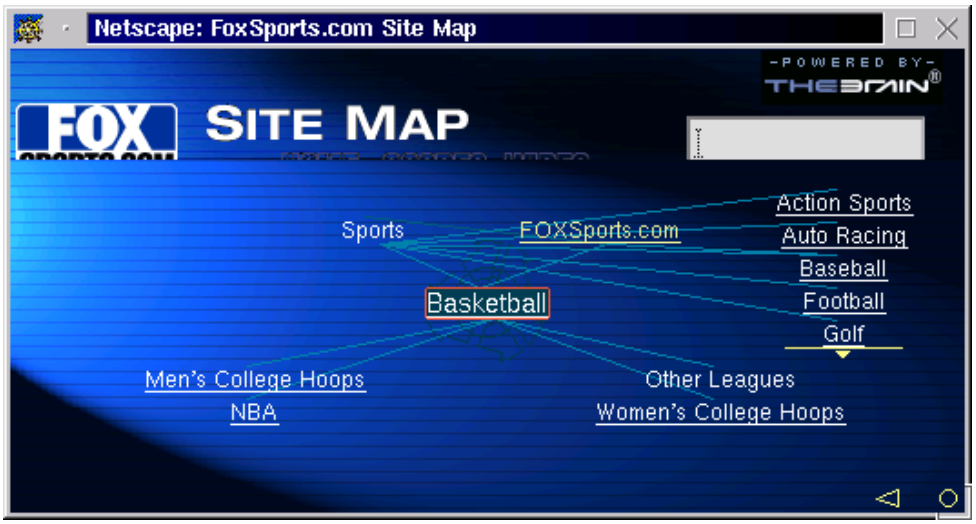


Figure 3.7: The Brain. Semantic zooming through concepts on the FoxSports.com web site. Semantic zooming has since been taken down and replaced with a conventional site map.

lost, since no overall context is provided.

3.2 Harmony Local Map

The Harmony Local Map [Schipflinger, 1998; Andrews, 1996] is part of the Harmony [Andrews, 1995; Andrews and Kappe, 1994] browser for Hyper-G [Andrews et al., 1995; Maurer, 1996]. Hyper-G, which has since been turned into a successful commercial product under the name Hyperwave [Hyp, 2002], uses a separate link database for hyperlink facilities, has typed links, maintains rich document metadata, and has fully integrated search facilities.

The Harmony Local Map is a dynamically generated map of the link neighbourhood of a selected document, similar the Web View of Intermedia [Haan et al., 1992; Utting and Yankelovich, 1989], but using a modified version of Eades and Sugiyama's [Eades and Sugiyama, 1990] graph layout algorithm for an aesthetically pleasing layout. Figure 3.9 shows part of the local map around the Unix "grep" manual page. Clicking on another document regenerates the local map around that document.

Unique features of the Harmony Local Map are its focus on a particular document in the middle of the layout and its display of both incoming links to and outgoing links from that document. To accomplish this, the Sugiyama algorithm was extended to include focusing the layout around a particular node. First of all, a layout of all documents linked by outgoing links is generated from left to right. Next, a layout of all remaining documents (those linked to the current document only by incoming links) is generated from right to left. These two layouts are positioned to the right and left of the focus node respectively. Finally, any links which cross from left to right or vice versa are incorporated into the layout. This produces a local map of the link neighbourhood of the focus node which resembles an hour-glass turned on its side.

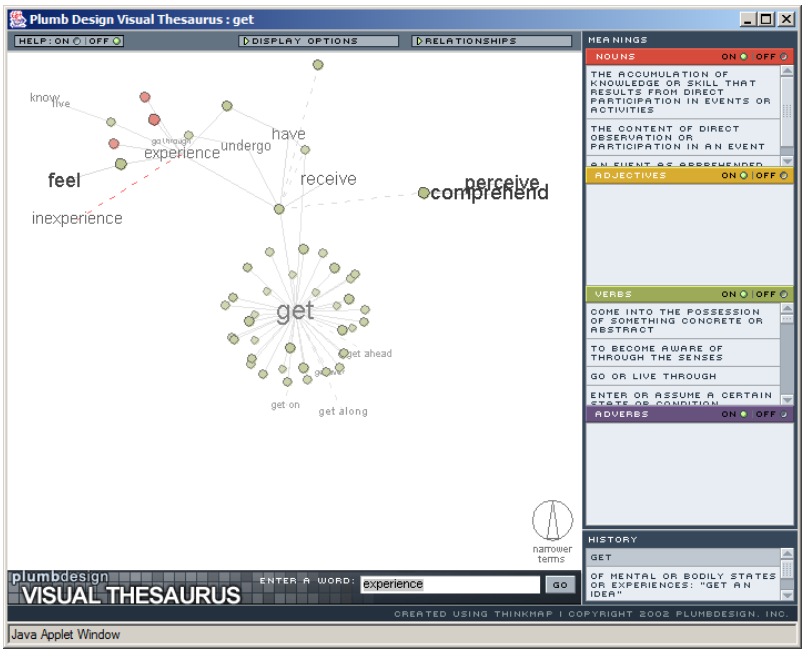


Figure 3.8: Plumb Design's Visual Thesaurus uses a ThinkMap interface to navigate concepts in the WordNet thesaurus.

Figure 3.9 shows a local map of Unix manual pages related to the "grep" manual page.

In Figure 3.10, the hour-glass shape of the layout of pages around the "ex" manual page can be seen. The Options panel allows the user the type and number of levels of incoming and outgoing links to display.

3.3 Harmony 3D Local Map

The Harmony 3D Local Map [Wolf, 1996; Andrews et al., 1996] extends the Harmony Information Landscape to display both hierarchical structure and hyperlink relationships in one integrated visualisation. The hierarchical structure of the server is laid out horizontally in the landscape as usual. The vertical dimension is then used to overlay associative hyperlink relationships on top of the structural landscape. Figure 3.11 shows links to and from the image document "Map of Graz". Documents which have links to a selected document are shown in the lower plane. Documents or collections to which the selected document contains links are shown in the upper plane.

This facility is not limited to standard referential hyperlinks, but can selectively show only inline images, model textures, annotations, etc. It can also provide a view of multiple parents, in cases where objects belong to more than one parent collection.

Sometimes, the user would like to know where the linked objects are actually located themselves in the collection hierarchy. A further control shows their location, by opening up paths to their position in the main landscape and drawing arcs to them, as can be seen in Figure 3.12.

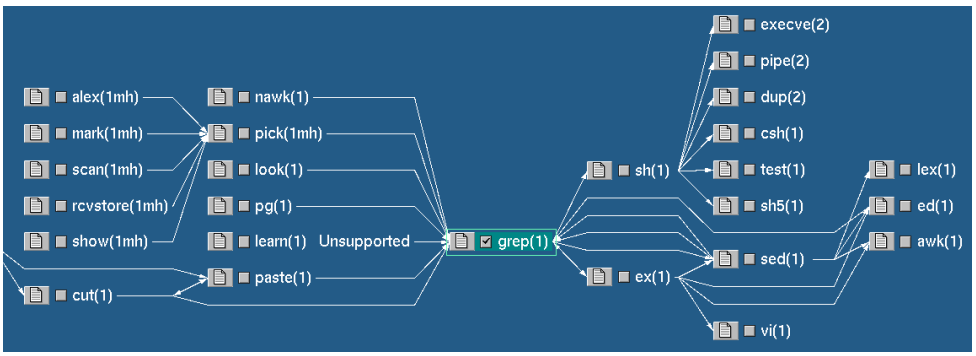


Figure 3.9: The Harmony Local Map uses modified Sugiyama algorithm to lay out a map of the link environment of hypermedia documents. In this example, Unix manual pages one and two links away from the "grep" manual page are visualised. The map resembles an hour-glass on its side, with incoming links to the left of the focus node and outgoing links to the right.

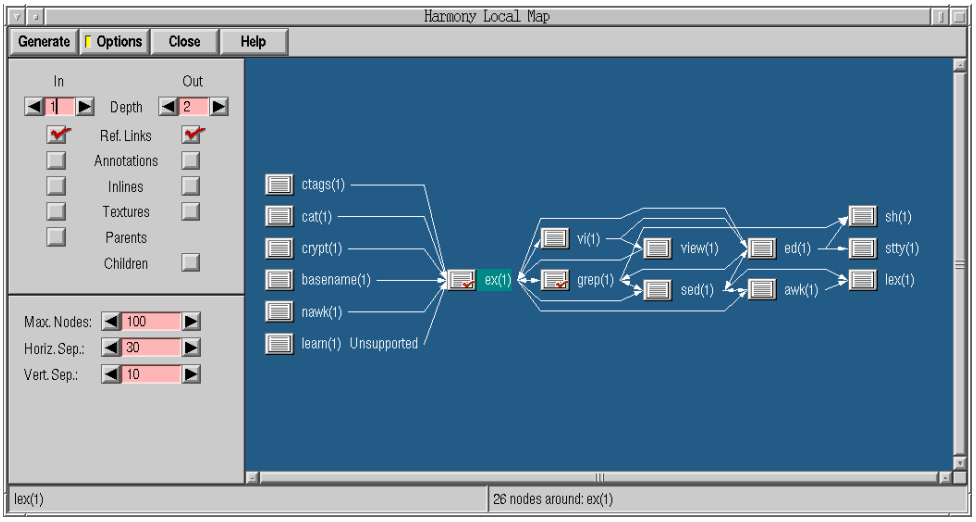


Figure 3.10: A local map of the "ex" man page. The type and number of levels of incoming and outgoing links can be set in the Options panel.

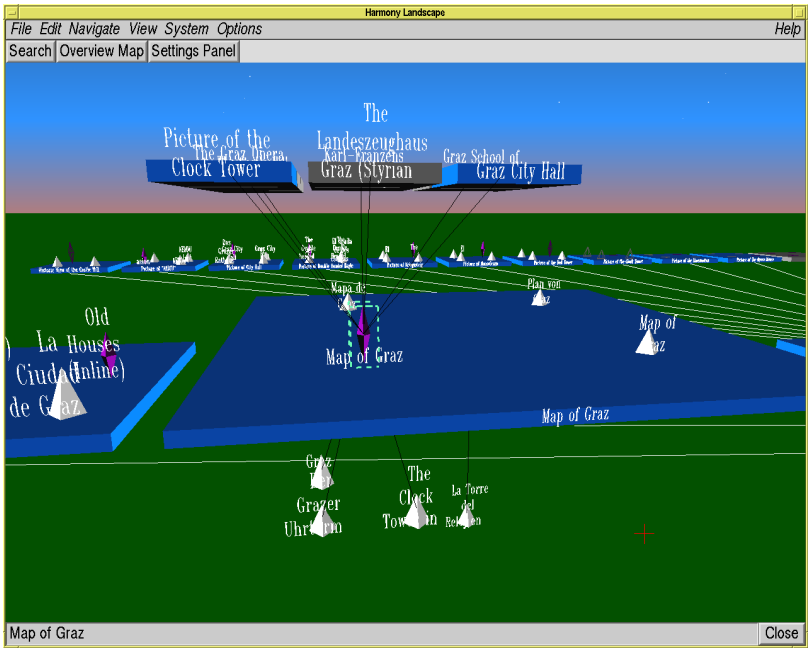


Figure 3.11: The Harmony 3D Local Map superimposes hyperlink relationships upon the structure map. Links to and from the image "Map of Graz" have been selected for display. Documents having links to this image are displayed on the lower plane. Documents or collections reachable from the image are displayed in the upper plane.

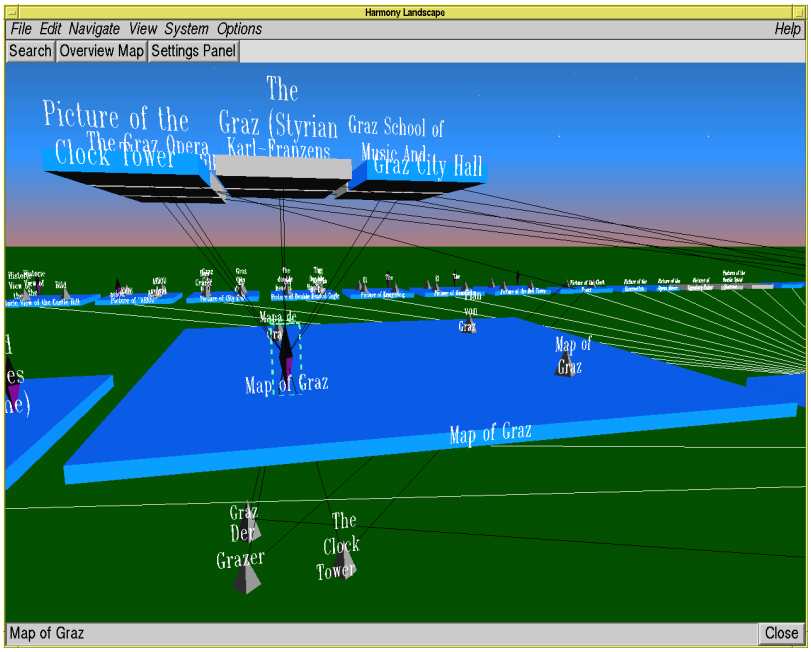


Figure 3.12: Harmony 3D Local Map. To find out where linked objects are actually located in the collection hierarchy, the user can display arcs linking objects to their locations in the landscape.

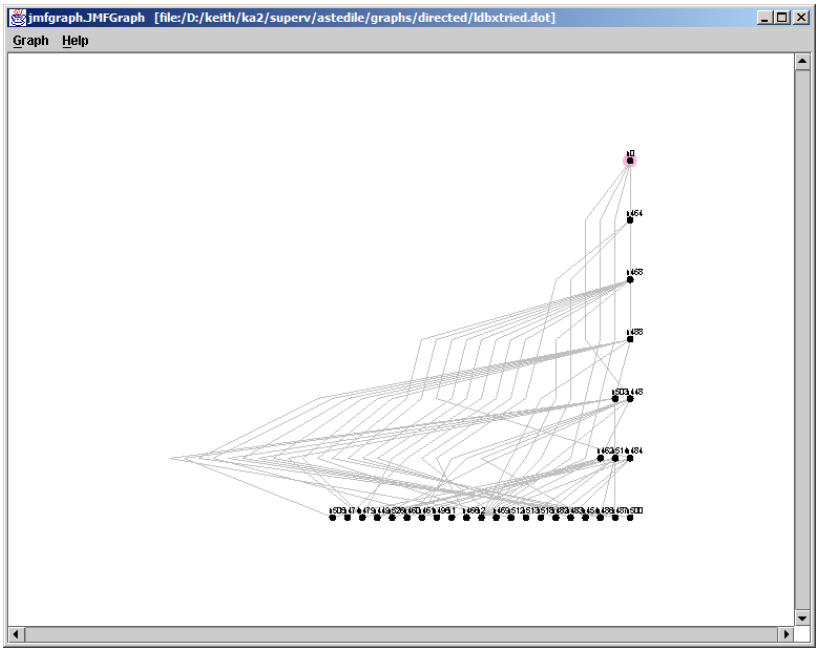


Figure 3.13: JMFGraph Layer assignment. In this top-down drawing, nodes are first assigned to layers according to their distance from the root. To ensure that edges do not cross multiple layers, dummy nodes are inserted.

Clicking on an arc flies the user gently down the path to the appropriate location. This corresponds to the location feedback feature in the Harmony Collection Browser.

3.4 Graph Layout in Java with JMFGraph

JMFGraph [Stedile, 2001] is a more recent implementation of a Sugiyama-style layered algorithm in Java. After removing any cycles by temporarily reversing some of the edges, the four steps of the unfocused standard layout algorithm are:

1. Layer assignment.
2. Crossing reduction.
3. X-coordinate assignment
4. Spacing and edge placement.

These steps are illustrated in Figures 3.13, 3.14, 3.15, and 3.16 respectively, the drawings being done from top to bottom.

Like the Harmony Local Map, JMFGraph also implements a focused mode, where a particular node in the graph is drawn in the middle of the layout in order to emphasise the link

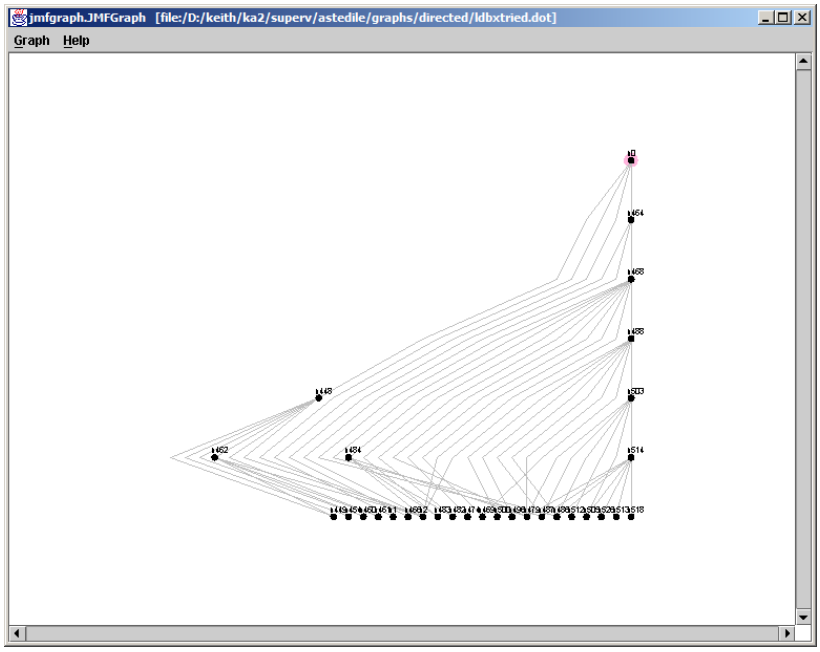


Figure 3.14: JMFGraph Crossing reduction. For each layer, an ordering of the vertices is computed such that the number of edge crossings is minimised.

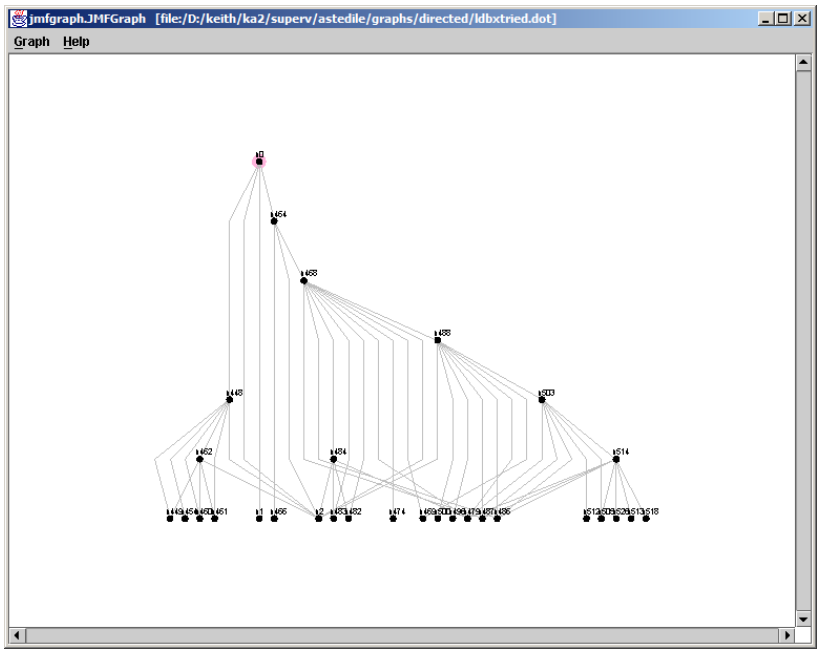


Figure 3.15: JMFGraph X-coordinate assignment. The horizontal positions of vertices within a layer are adjusted so that they do not overlap other real or dummy vertices.

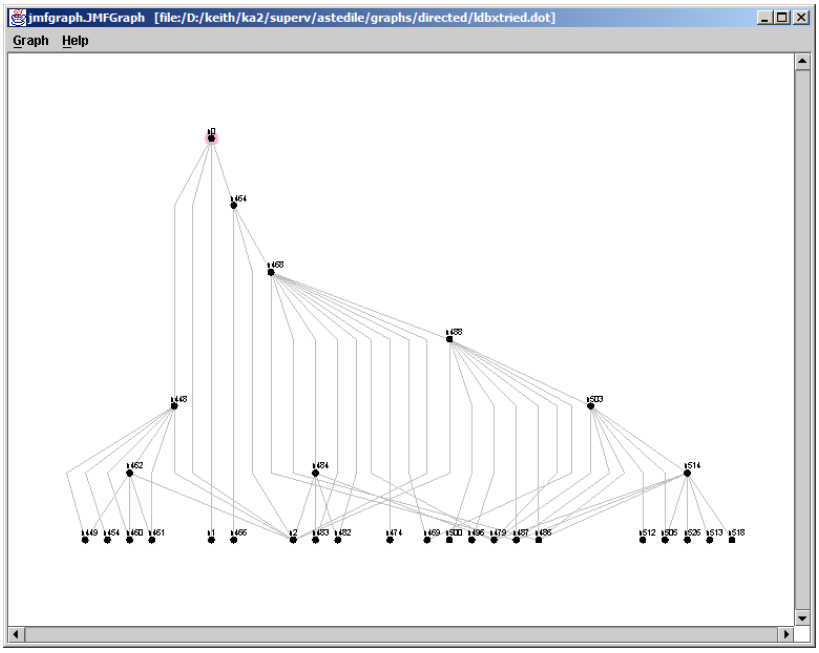


Figure 3.16: The JMFGraph final top-down layout. Some extra spacing is added to better distinguish nodes. The layout is approximately a mirror of that in Figure 3.2.

neighbourhood around that node. One of the difficulties with the Harmony Local Map hourglass layout is that adding paths which cross both halves of the layout is time-consuming and can sometimes produce poor layouts.

JMFGraph uses a new algorithm for focused node layout based on two sweep lines: a forward and a backward sweep line which alternately move forwards (backwards) from layer to layer away from the focus node. In essence, the algorithm makes a sweep from the centred focused node away to the borders of the drawing scope. The closer a node is to the focus node, the earlier it has the opportunity to take up a good position. See Stedile [2001] for full details of the algorithm.

Figure 3.17 shows the same graph as Figure 3.16 in unfocused standard layered layout but drawn from left to right. In Figure 3.18 focus mode has been turned on, the node in focus (node 483) is moved to the centre of the layout, and its link neighbourhood is laid out with the focus algorithm. In Figure 3.19 the user has clicked on node 503 to regenerate the map around that node.

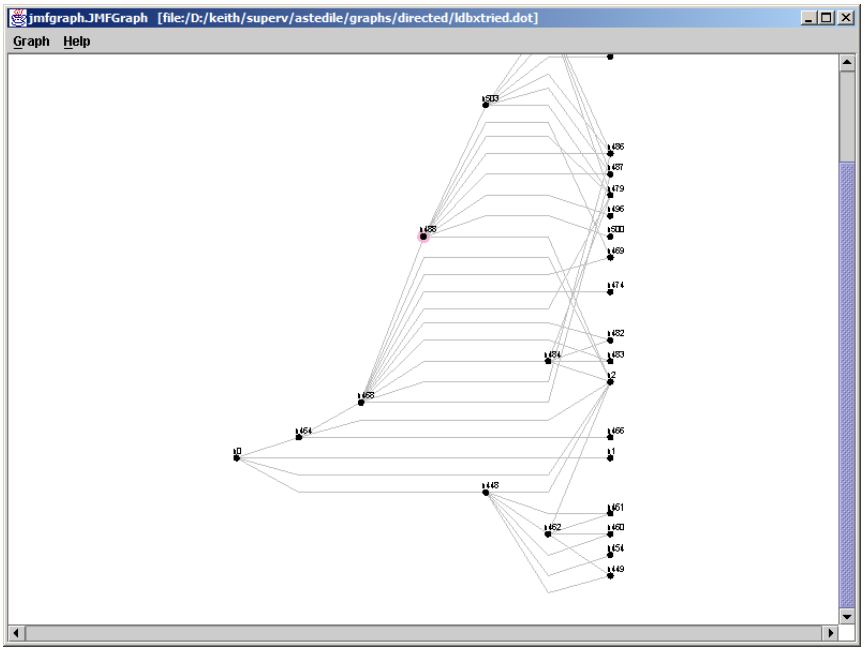


Figure 3.17: JMFGraph standard layout, but now drawn from left to right.

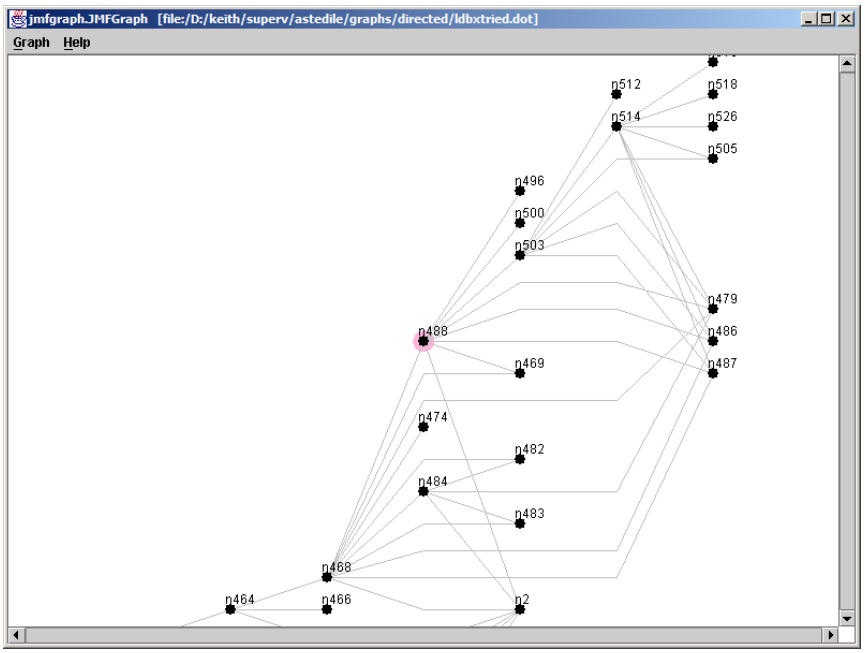


Figure 3.18: JMFGraph focused layout. The focus is on node 488.

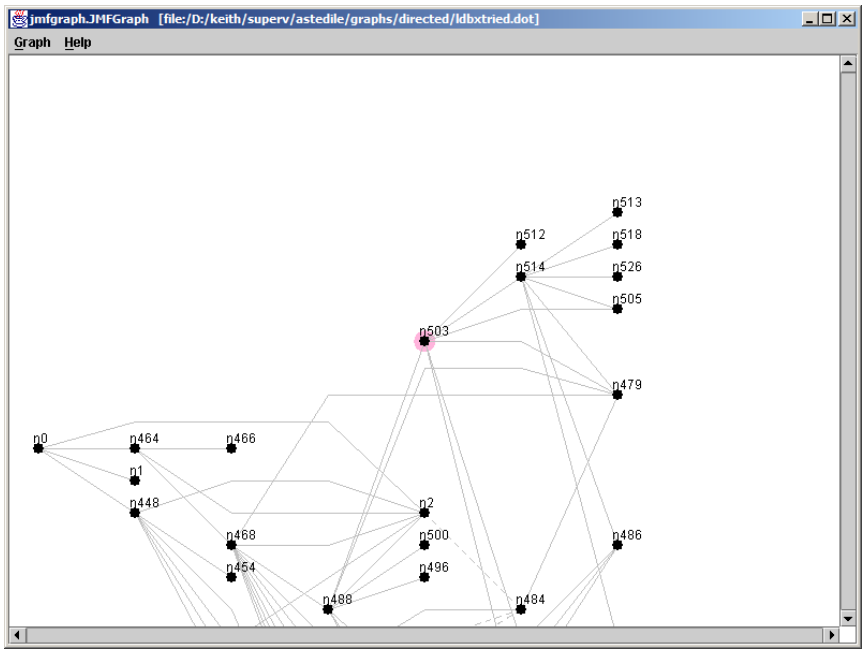


Figure 3.19: JMFGraph focused layout. The user has clicked on node 503, causing the layout to be regenerated around that node.

Chapter 4

Visualising Multidimensional Information

Files and documents often have associated metadata attributes such as author, title, modification date, size, type, and extension. These attributes form an n-dimensional space, within which each document is positioned. This n-dimensional space can then be mapped to a variety of 2d or 3d representations for exploration and viewing. Hence, rather than (or in addition to) displaying simple linear lists of documents, say in response to a search query, metadata attributes can be used to visualise sets of documents in an explorable multidimensional space.

4.1 Related Work

Michael Benedikt proposed a system whereby attributes of data could be mapped to *extrinsic* and *intrinsic* spatial dimensions [Benedikt, 1991]. Extrinsic dimensions indicate where an item is positioned in space (x, y, and z coordinates), and intrinsic dimensions determine the characteristics of the glyph used to represent the item (such as shape, colour, or size). Card and Mackinlay [1997] introduced a taxonomy for mappings from dimensions of the data to characteristics of the visual representation, based on the earlier work of Bertin [1983]. Some of the possibilities are discussed here.

4.1.1 Parallel Coordinates

In a parallel coordinate system, equidistant parallel vertical lines represent the axes of a multidimensional space, one vertical line for each dimension. Each object (with values along each dimension) is plotted as a polyline in parallel coordinates. Objects which are very similar will have polylines which follow each other.

Figure 4.1 shows a parallel coordinates plot of a fictitious dataset from Goel [1999]. Six dimensions (FirstName, Quiz1, Quiz2, Homework1, Homework2, Final) are represented by the six equidistant vertical lines. Eleven data points or objects representing eleven students are plotted as polylines, according to where their values lie along each dimension.

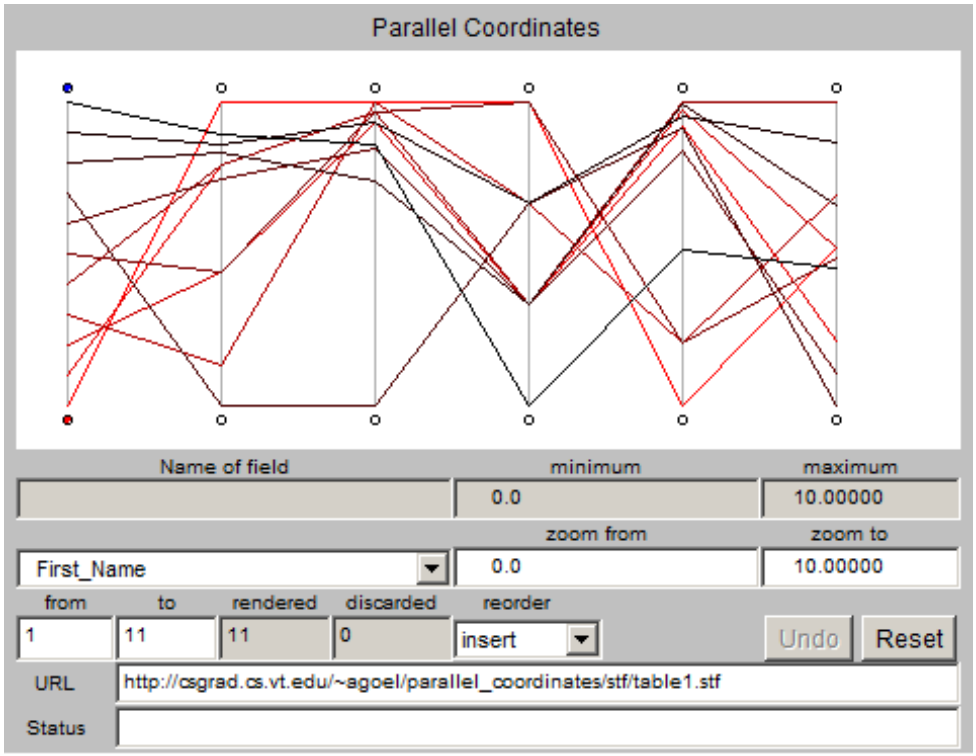


Figure 4.1: Parallel Coordinates. The six vertical lines represent (from left to right) the name and marks of students in five exams. The eleven polylines represent the data from eleven students. Polylines which mirror one another closely from vertical lines 2 to 6 indicate students achieving very similar marks.

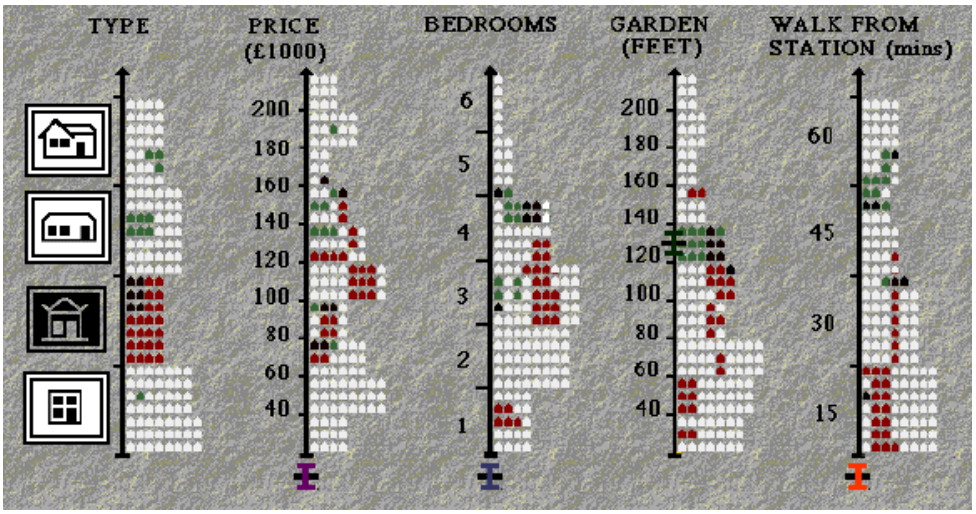


Figure 4.2: The Attribute Explorer. Each attribute is assigned to a scale with histograms showing the population spread running up one side. The effect of one attribute on the others can be explored by selecting values of interest on one scale and viewing where those items appear on the other attribute scales. [Image extracted from CHI 94 Electronic Proceedings. Copyright ©by the Association for Computing Machinery, Inc.]

4.1.2 Linked Histograms

The Attribute Explorer [Tweedie et al., 1994] provides direct manipulation of coupled views of histograms. Each attribute is assigned to a scale with histograms showing the population spread running up one side, as shown in Figure 4.2. Initially these display each item in the total population. The user can interact with the scales: using sliders for continuous attributes (e.g. price) and buttons for discrete attributes (e.g. type of house). The effect of one attribute on the others can be explored by selecting values of interest on one scale and viewing where those items appear on the other attribute scales.

4.1.3 Scatterplots

The FilmFinder [Ahlberg and Shneiderman, 1994a,b] combines a 2d scatterplot-like display (called a starfield display by the authors) with dynamic queries for rapid, incremental, and reversible exploration of a space of feature films. Dynamic queries [Ahlberg et al., 1992] interactively change the display to reflect changes made by the user to input sliders and ranges. Figure 4.3 shows the FilmFinder displaying a database of films from 1920 to 1995. The techniques espoused in the FilmFinder are now available commercially from Spotfire [Spotfire, 2000].

Envision[Nowell et al., 1996, 2002] plots the results of bibliographic searches in a two-dimensional scatterplot based on their metadata attributes. The mapping of particular attributes to visual representations such as the x-axis, y-axis, icon size, and icon shape, is controlled by

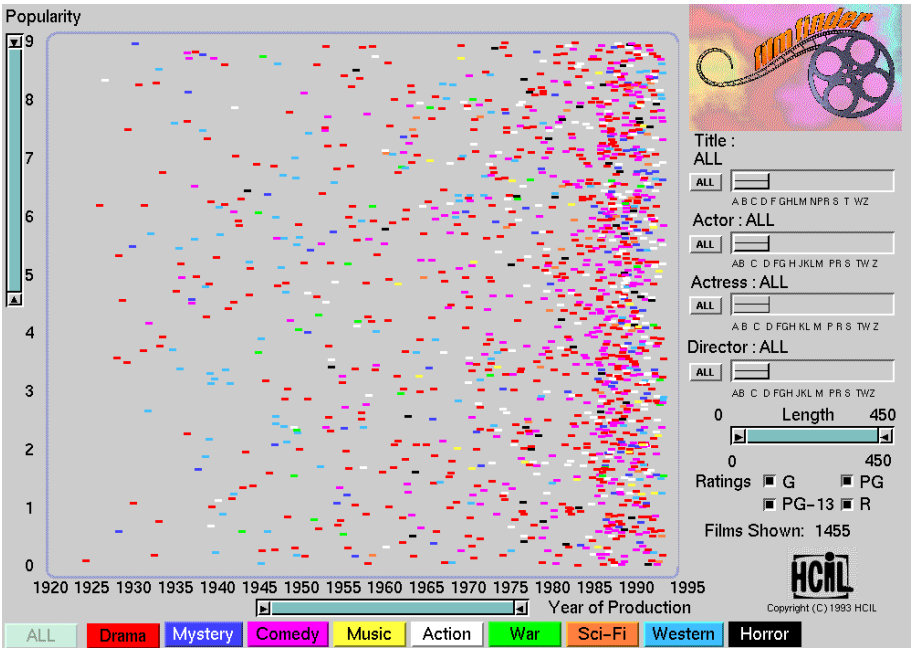


Figure 4.3: The FilmFinder, a starfield display (2d scatterplot) of metadata describing films. Dynamic queries provide for rapid filtering. [Image is used with permission of the University of Maryland. Copyright ©University of Maryland, all rights reserved.]

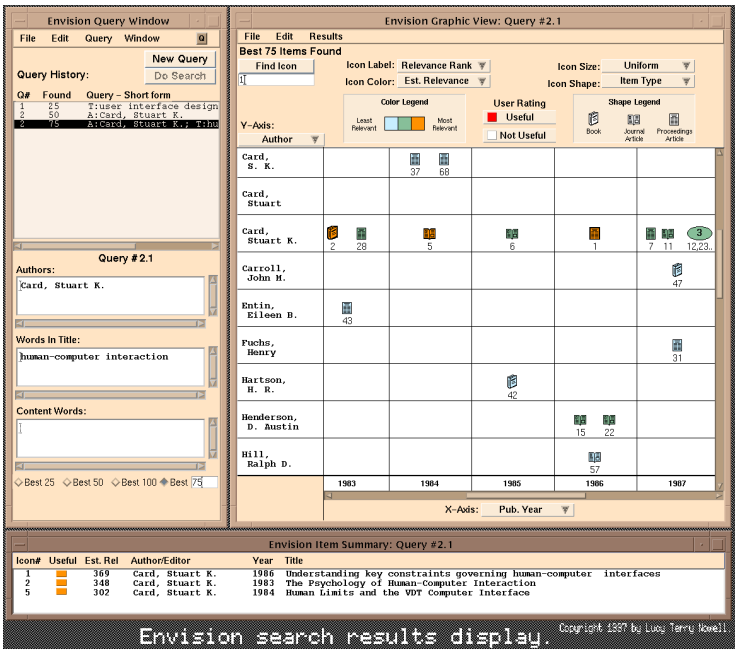


Figure 4.4: Envision visualises a set of search results, by mapping document attributes along two axes. Where too many documents would occupy a cell, an ellipse is used as a container object. [Image extracted from CHI 97 Electronic Proceedings. Copyright ©by the Association for Computing Machinery, Inc.]

drop-down menus, as can be seen in Figure 4.4. Where too many documents would occupy a cell, an ellipse is used as a container object. Documents matching multiple categories are displayed in each category.

These kinds of interface probably best lend themselves to the exploration of unfamiliar sets of documents, rather than directed navigation to a known document, where a Windows Explorer style tree browser or a search query is perhaps more efficient. At the IICM, three prototypes have been implemented to further explore these possibilities: a file attribute explorer, a search result explorer, and a file attribute explorer in 3d.

4.2 File Attribute Explorer

The IICM File Attribute Explorer [Weitlaner, 1999] scans a specified directory tree and visualises the files contained therein. The user interface is comprised of several elements, as shown in Figure 4.5. In the main scatterplot display, files are plotted according to any two of their metadata attributes (mapped to the x and y axes). Individual elements can be selected by left-clicking, a document's full set of metadata attributes is displayed by right-clicking.

If too many documents would be mapped to the same proximity on screen, a group icon is used to represent that subset of documents. For group icons, the size and colour of the group icon is determined (under user control) by the maximum, minimum, median, or average value

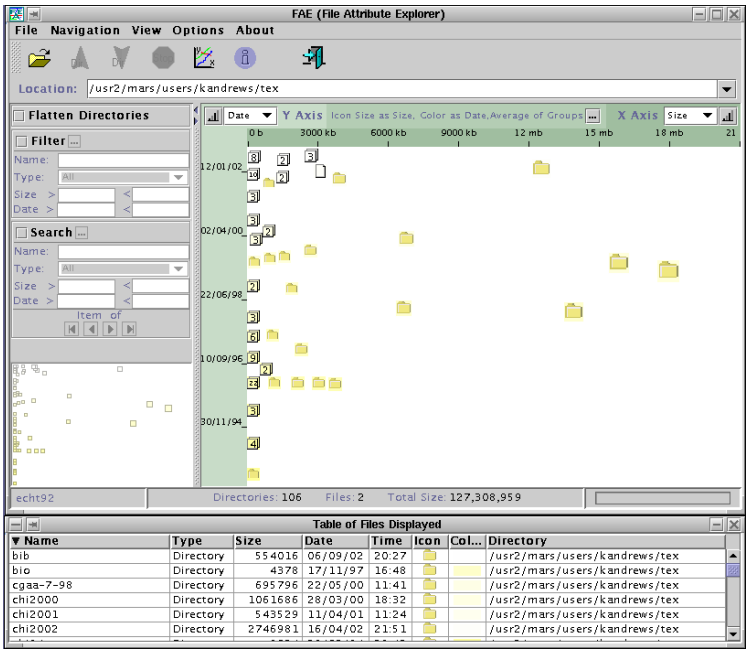


Figure 4.5: The File Attribute Explorer. The display shows 106 directories and 2 files at the top level of a directory called `tex`.

of the group's members. The group icon indicates the number of items it represents.

Double-clicking on a directory steps down into that directory and maps its contents. By default, size is mapped to the x axis and modification date to the y axis. Sometimes it would be desirable to view the entire contents of a directory, contained at any level recursively within. The Flatten Directories button achieves this, as shown in Figure 4.6.

Further metadata attributes can be mapped to icon size and icon colour, allowing four dimensions of metadata to be visualised and explored simultaneously. By default icon size is mapped to file size and icon colour to modification date (newer files being white and older ones yellowish). Figure 4.7 shows the choices of mappings which are available to the user. The current implementation in Java only returns limited metadata about files in the file system, so the choices for mappings are limited. In other applications such as search result sets or document repositories or where supplementary information is available, further dimensions of attribute metadata can be made available.

Since it is possible to zoom in on specific areas of the display, an overview window is provided in the lower left corner to help maintain context and orientation. The overview window can also be used to navigate. A table of all current files is optionally displayed at the bottom of the display. The table can be sorted on various criteria and selections are synchronised with the other views.

The search and filter panels perform the corresponding functions. Figure 4.8 shows 25 directories of Powerpoint and other presentation materials. A search for `ppt` in the file name has revealed 22 matching files and the user can now step through them. Finally, on occasions where there are simply too many files and directories in the display, individual items can be displayed

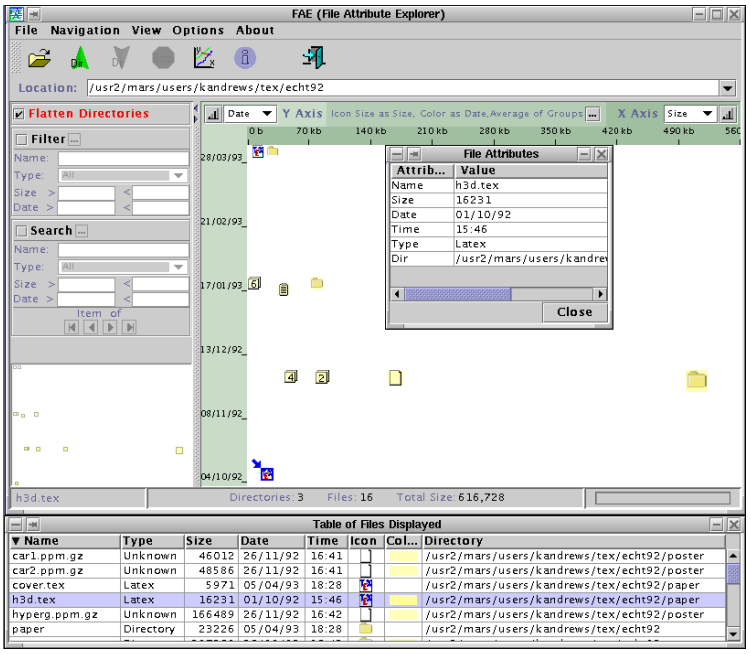


Figure 4.6: File Attribute Explorer. Subdirectory `echt92` contains only three subdirectories at the top level: `paper`, `poster`, and `trip`. However, flattening directories displays all files at or below this level revealing a total of 16 files.

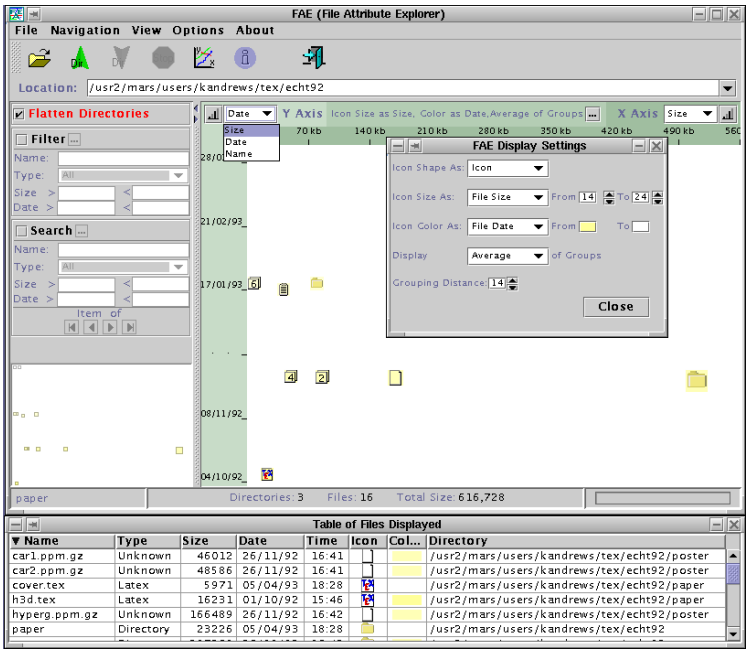


Figure 4.7: File Attribute Explorer. Size, date, and name can be mapped to either axis. Size, date, name, and depth in the hierarchy can be mapped to icon size or icon colour.

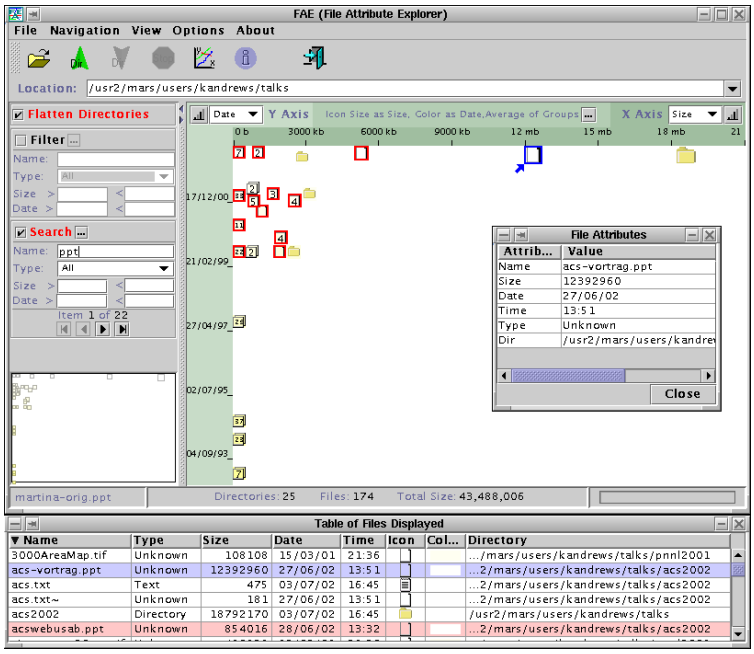


Figure 4.8: File Attribute Explorer. Directory `talks` contains 174 files concerning talks and presentations in 25 directories. A search for `ppt` in the file name reveals 22 matching files. The current selection is highlighted red, the other matching files in blue.

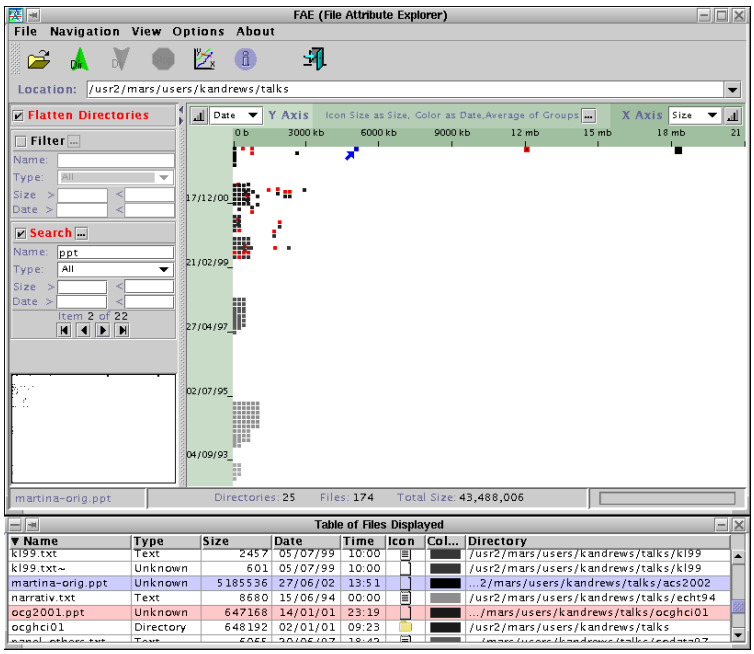


Figure 4.9: File Attribute Explorer. The scatterplot display has been switched from glyphs to points to accommodate more items at once.

as points rather than glyphs, as shown in Figure 4.9.

Informal user testing with students and colleagues showed that the File Attribute Explorer was useful for exploration of less familiar document collections, although traditional Windows Explorer style browsers were usually preferred for directed navigation to particular known files.

4.3 Search Result Explorer

The Search Result Explorer [Andrews et al., 2001] is an extension of the File Attribute Explorer for interactive exploration of search result sets, based on the rich metadata associated with each object. Documents returned by search engines typically have much more metadata available than files in a file system. In this case, the xFIND search engine [Gütl, 2000] maintains around 20 metadata fields for each document.

The Search Result Explorer plots a search result set in a scatterplot (starfield display). Documents are plotted according to two of their metadata attributes (corresponding to the x and y axes). Further metadata attributes can be mapped to icon size and icon colour, allowing four dimensions of metadata to be visualised and explored simultaneously. As in the File Attribute Explorer, if too many documents would be mapped to the same proximity, a group icon is used to represent that subset of documents. For group icons, the size and colour of the group icon is determined (under user control) by the maximum, minimum, median, or average value of the group's members. Since it is possible to zoom in on specific areas of the display, an overview window is provided in the lower left corner to help maintain context and orientation.

Figure 4.10 shows the first 210 (a user-configurable limit) of the 314 matching documents plotted by relevance on the y axis and document size on the x axis. The most relevant document is shown at the top of the plot. Here, the colour of each document icon is determined by the document's age, from yellow older documents to white recent documents. Relevance is mapped to icon size, providing a redundant encoding. More relevant documents are both larger and towards the top of the plot. The most relevant document has been selected and its metadata displayed. It is the document from 10th October 1998 entitled "Coordination as Distributed Search".

Most of the rich metadata attributes provided by xFIND can be mapped to either axis or to icon size or colour. In Figure 4.11, the user is considering which mapping to apply. Figure 4.12 shows the result of the change. The y axis now corresponds to the modification date of the document, and document relevance is mapped to both icon size and icon colour (more relevant are orange, less relevant are white). It can be seen at a glance, that the most relevant documents are about a year old and reasonably small.

Like the File Attribute Explorer, the Search Result Explorer provides two synchronised visualisations of a search result set: an interactive scatterplot and an interactive table, as shown in Figure 4.13. Initially, the Table of Documents is sorted on relevance, with the most relevant results at the top of the table, resembling a traditional linear list of search results. It can however be dynamically resorted on any metadata attribute, as shown in Figure 4.14.

In summary, the Search Result Explorer allows a user to explore a set of search results by various criteria, rather than simply browse a linear list of matching documents, as is the case in

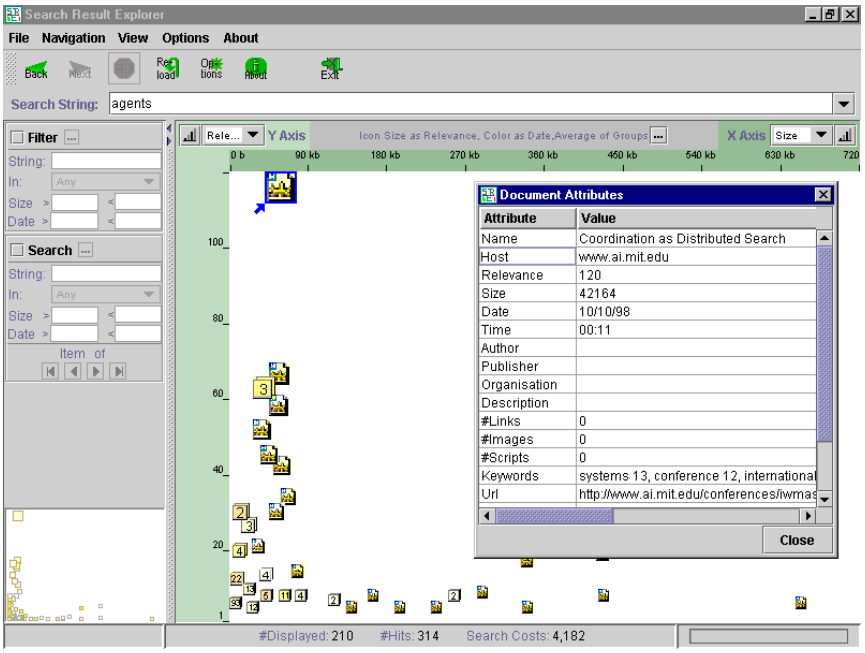


Figure 4.10: The Search Result Explorer plots search results along two axes. Here, document relevance is mapped to the y axis and document size to the x axis.

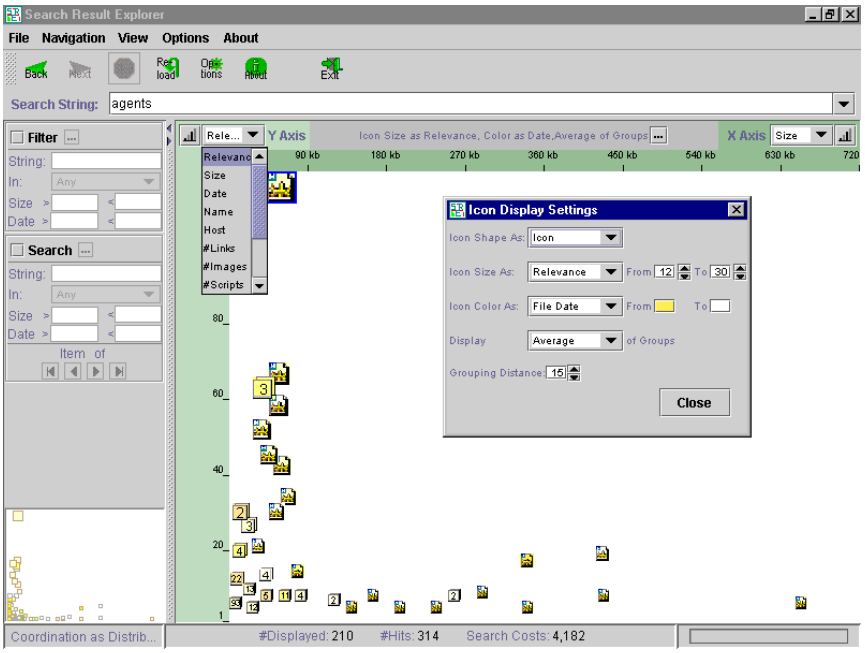


Figure 4.11: Search Result Explorer: Most of the rich metadata attributes provided by xFIND can be mapped to either axis, or to icon size or icon colour.

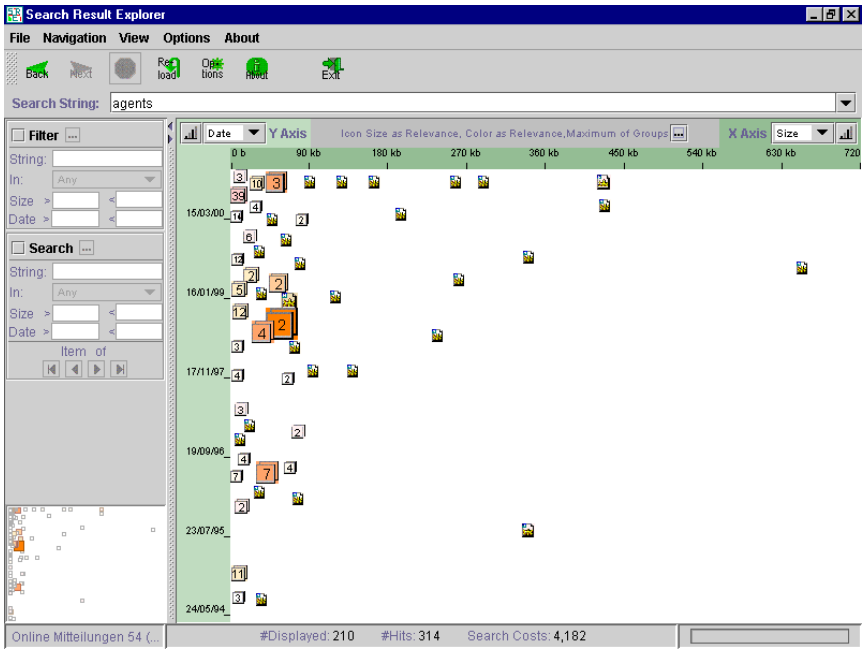


Figure 4.12: Search Result Explorer. An alternative view: the modification date has been mapped to the y axis. More relevant documents are now both larger and more orange.

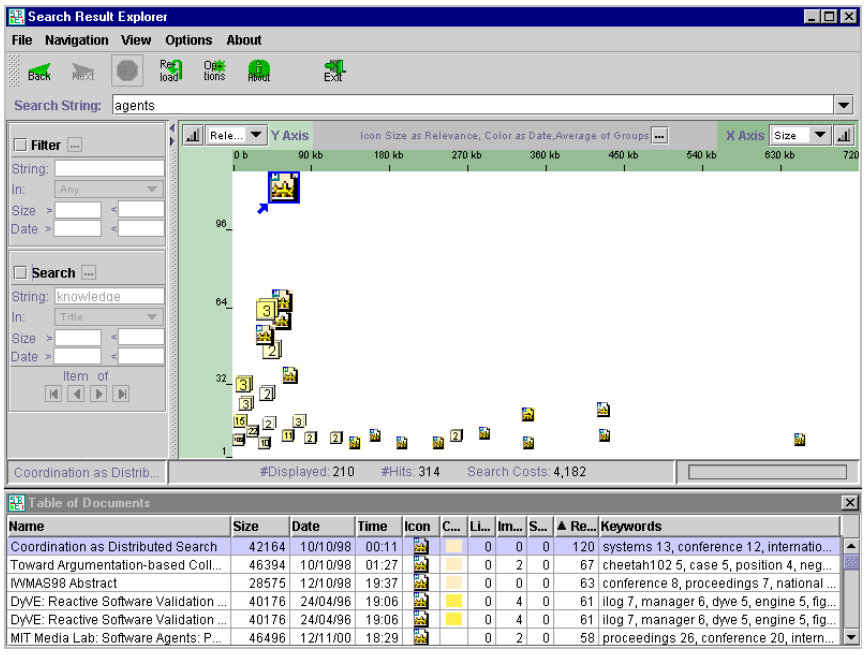
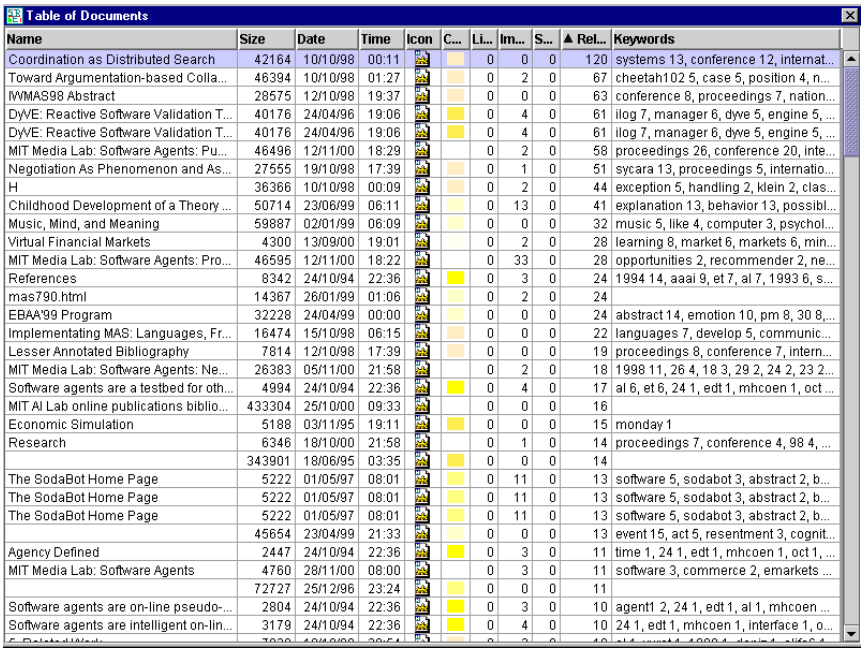


Figure 4.13: The Search Result Explorer provides two linked visualisations of search result sets: an interactive scatterplot and an interactive table.



Name	Size	Date	Time	Icon	C...	Li...	Im...	S...	▲ Rel...	Keywords
Coordination as Distributed Search	42164	10/10/98	00:11			0	0	0	120	systems 13, conference 12, internat...
Toward Argumentation-based Colla...	46394	10/10/98	01:27			0	2	0	67	cheetah102 5, case 5, position 4, n...
IWMAS98 Abstract	28575	12/10/98	19:37			0	0	0	63	conference 8, proceedings 7, nation...
DyVE: Reactive Software Validation T...	40176	24/04/96	19:06			0	4	0	61	ilog 7, manager 6, dyve 5, engine 5, ...
DyVE: Reactive Software Validation T...	40176	24/04/96	19:06			0	4	0	61	ilog 7, manager 6, dyve 5, engine 5, ...
MIT Media Lab: Software Agents: Pu...	46496	12/11/00	18:29			0	2	0	58	proceedings 26, conference 20, inte...
Negotiation As Phenomenon and As...	27555	19/10/98	17:39			0	1	0	51	sycara 13, proceedings 5, internatio...
H	36366	10/10/98	00:09			0	2	0	44	exception 5, handling 2, klein 2, clas...
Childhood Development of a Theory ...	50714	23/06/99	06:11			0	13	0	41	explanation 13, behavior 13, possibl...
Music, Mind, and Meaning	59887	02/01/99	06:09			0	0	0	32	music 5, like 4, computer 3, psychol...
Virtual Financial Markets	4300	13/09/00	19:01			0	2	0	28	learning 8, market 6, markets 6, min...
MIT Media Lab: Software Agents: Pro...	46595	12/11/00	18:22			0	33	0	28	opportunities 2, recommender 2, ne...
References	8342	24/10/94	22:36			0	3	0	24	1994 14, aaai 9, et 7, al 7, 1993 6, s...
mas790.html	14367	26/01/99	01:06			0	2	0	24	
EBAA'99 Program	32228	24/04/99	00:00			0	0	0	24	abstract 14, emotion 10, pm 8, 30 8,...
Implementating MAS: Languages, Fr...	16474	15/10/98	06:15			0	0	0	22	languages 7, develop 5, communic...
Lesser Annotated Bibliography	7814	12/10/98	17:39			0	0	0	19	proceedings 8, conference 7, intern...
MIT Media Lab: Software Agents: Ne...	26383	05/11/00	21:58			0	2	0	18	1998 11, 26 4, 18 3, 29 2, 24 2, 23 2...
Software agents are a testbed for oth...	4994	24/10/94	22:36			0	4	0	17	al 6, et 6, 24 1, edt 1, mhcoen 1, oct ...
MIT AI Lab online publications biblio...	433304	25/10/00	09:33			0	0	0	16	
Economic Simulation	5188	03/11/95	19:11			0	0	0	15	monday 1
Research	6346	18/10/00	21:58			0	1	0	14	proceedings 7, conference 4, 98 4, ...
	343901	18/06/95	03:35			0	0	0	14	
The SodaBot Home Page	5222	01/05/97	08:01			0	11	0	13	software 5, sodabot 3, abstract 2, b...
The SodaBot Home Page	5222	01/05/97	08:01			0	11	0	13	software 5, sodabot 3, abstract 2, b...
The SodaBot Home Page	5222	01/05/97	08:01			0	11	0	13	software 5, sodabot 3, abstract 2, b...
	45654	23/04/99	21:33			0	0	0	13	event 15, act 5, resentment 3, cognit...
Agency Defined	2447	24/10/94	22:36			0	3	0	11	time 1, 24 1, edt 1, mhcoen 1, oct 1, ...
MIT Media Lab: Software Agents	4760	28/11/00	08:00			0	3	0	11	software 3, commerce 2, emarkets ...
	72727	25/12/96	23:24			0	0	0	11	
Software agents are on-line pseudo-...	2804	24/10/94	22:36			0	3	0	10	agent1 2, 24 1, edt 1, al 1, mhcoen ...
Software agents are intelligent on-lin...	3179	24/10/94	22:36			0	4	0	10	24 1, edt 1, mhcoen 1, interface 1, o...

Figure 4.14: The Search Result Explorer's Table of Documents is an interactive, sortable table of search results.

most search interfaces today.

4.4 3D File Attribute Explorer

The 3D File Attribute Explorer [Köhler, 2002] plots three extrinsic dimensions from the metadata in a three-dimensional display. Figure 4.15 shows the contents of directory `talks`. The x, y, and z axes are mapped to name, extension, and date respectively. The size of the file and its extension are mapped to the size and colour of the glyph respectively. An overview map in the top right corner maintains an overall sense of orientation. The 3D File Attribute Explorer always shows a flattened view of the directory structure, i.e. all files belonging to a directory and all its subdirectories are included in the display.

To accommodate a larger number of files and reduce clutter, the overall size of file glyphs can be scaled down, as shown in Figure 4.16. Users can also select which colours to map to which aspects of the data.

Informal user testing showed the visual appeal of the 3d display, but the prototype quickly ran into performance problems when displaying more than a few hundred files. Initially, more complex 3d glyphs were used to represent each document, but these were then simplified to straightforward cones to improve graphics performance.

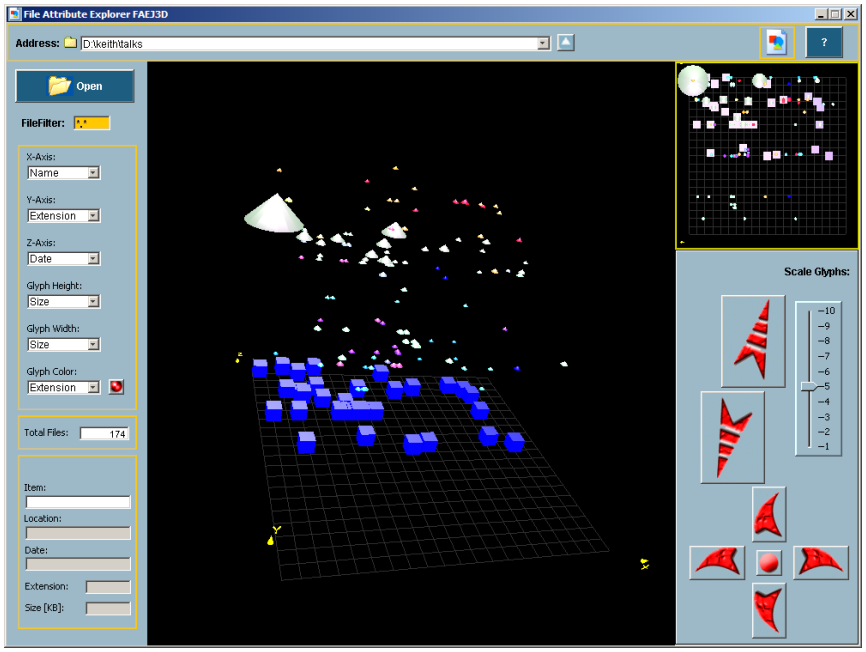


Figure 4.15: The 3D File Attribute Explorer visualises three extrinsic dimensions. Here, 174 files in directory `talks` are being displayed. The blue blocks represent directories.

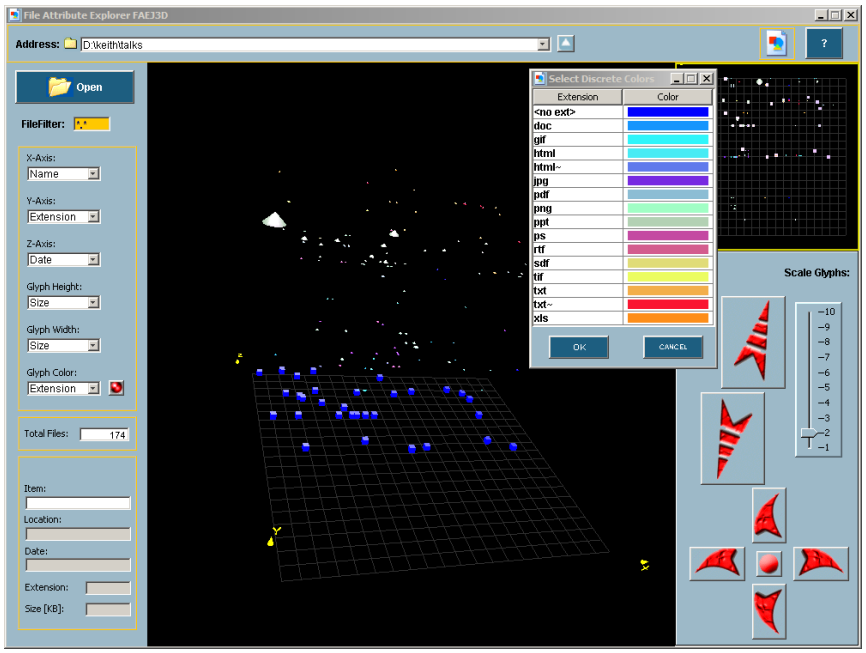


Figure 4.16: To reduce clutter, the overall size of file glyphs can be scaled down. Discrete colours can be chosen for specific file extensions and colour scales for continuous variables.

Chapter 5

Visualising Text Corpora

Text documents in a large corpus can be characterised in terms of their content. Content-based approaches from information visualisation have been used to visualise, for example, how documents relate to one another and how many documents are available on particular topics.

5.1 Related Work

The *vector space model* [Salton et al., 1975] is often used to characterise text documents in a collection. Each document is represented by an n-dimensional vector, where n is total number of unique words or terms used in the collection as a whole. Usually, a *stop list* is used to exclude very common words such as "and" and "the", since they are present in most documents and are not helpful for distinguishing between documents. However, n is still often very large, typically several thousands. A document's term vector places it as a point in the n-dimensional term space, whereby each component in the vector indicates the weighting of that term in the document, often as value between 0 and 1. The similarity between two documents is sometimes expressed as the Euclidean distance between them, but more usually as the "angle" between them by taking a scalar product (the cosine similarity metric).

A number of systems employ methods for mapping documents from the high-dimensional term space to a lower dimensional display space, whilst preserving the high-dimensional distances as far as possible.

5.1.1 Graph Drawing Methods

If documents are considered to be nodes in a graph, then methods for drawing graphs such as force-directed and energy-based placement, presented in Sections 3.1.3 and 3.1.4, can be used to visualise the corpus.

The Bead system [Chalmers, 1993, 1996a,b] uses force-directed placement to arrange documents in landscape, as shown in Figure 5.1. Users can navigate freely around the information landscape. Searches can be made and search result sets are displayed in situ in the landscape.

SPIRE [Hetzler et al., 1998; Wise, 1999; Thomas et al., 2001] provides two visualisations.

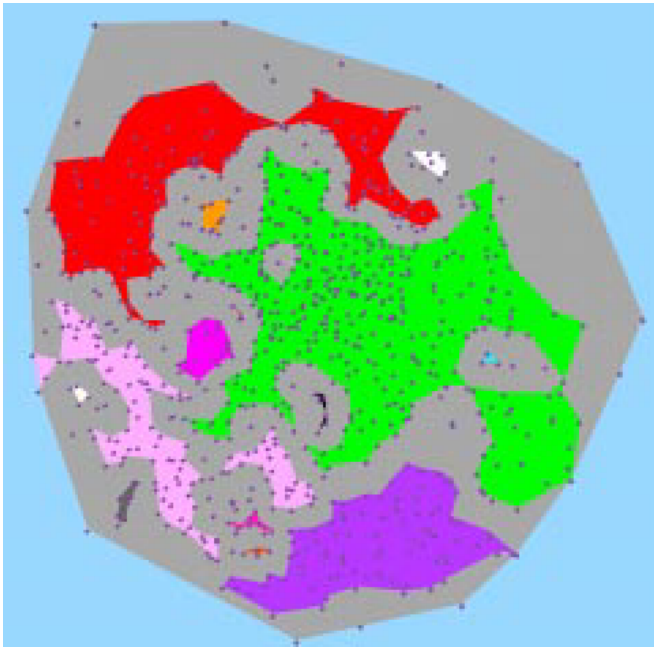


Figure 5.1: Bead displaying a layout of 831 bibliography entries. [Image extracted from UIST 96 Proceedings. Copyright ©by the Association for Computing Machinery, Inc.]

SPIRE's GalaxyView uses multi-dimensional scaling to visualise documents as stars in a galaxy. SPIRE's ThemeView (formerly Themescape) builds on the galaxy view by aggregating frequently occurring topical keywords from neighbouring documents as a height field, resulting in a display of the main themes in a thematic landscape. Both of these visualisations can be seen in Figure 5.2. Documents matching particular search criteria can be grouped and colour-coded.

VxInsight [Davidson et al., 1998; Meyers et al., 1999; Hendrickson, 1999] uses a combination of force-directed and energy-based placement. Figure5.3 shows VxInsight displaying a set of 1,231 bibliographic records of articles from the physical sciences portion of the Science Citation Index Expanded. The layout is generated using similarities based on direct citation and co-citation links between articles. VxInsight is now sold as a product by VisWave [VisWave, 2002]. Mountain peaks are labelled with the names of the journals most present in that area. The user can zoom in and inspect individual articles.

WebMap's InternetMap [WebMap, 2002; Iron et al., 2001] visualises hierarchically categorised web sites. Each site is represented by a pixel, sites belonging to multiple categories are represented by separate pixels in each category. Within a category, sites are arranged according to their similarity (the exact algorithm is not disclosed). Each category is visualised as a multi-faceted shape, enclosing the sites within that category. A topographical contour map representing an aggregation of ratings of some kind for each site is underlaid, similar to SPIRE's ThemeView. Searches can be made and the results are displayed in context; matches at lower levels in the hierarchy are propagated up to the current level on display.

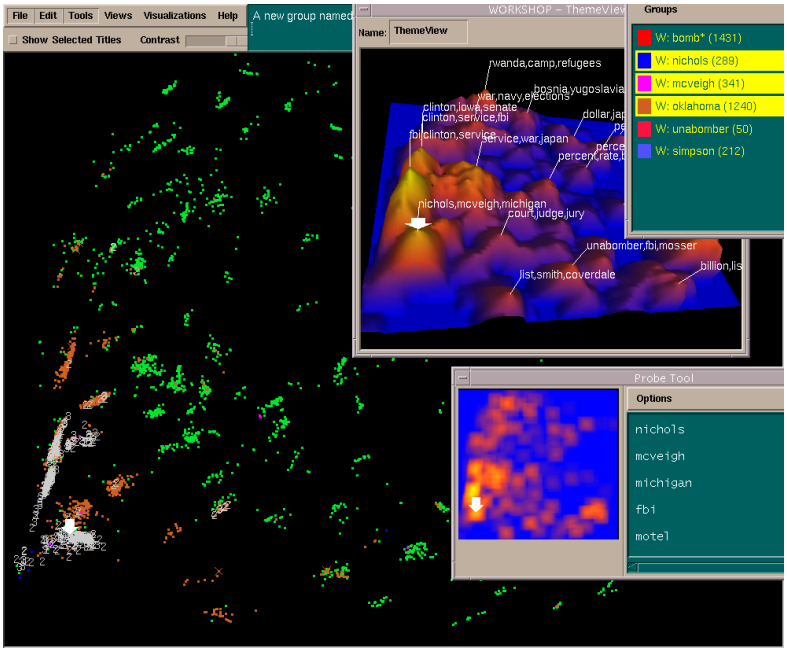


Figure 5.2: SPIRE. The GalaxyView (below) shows Reuters newswire reports in the seven days following the Oklahoma City bombing. The ThemeView (upper right) aggregates the most frequently occurring terms in regions of the galaxy to provide a thematic overview. The Probe Tool (bottom right) lets an analyst drill down into a region and see the most frequent terms. [Image used with kind permission of Pacific Northwest National Labs.]

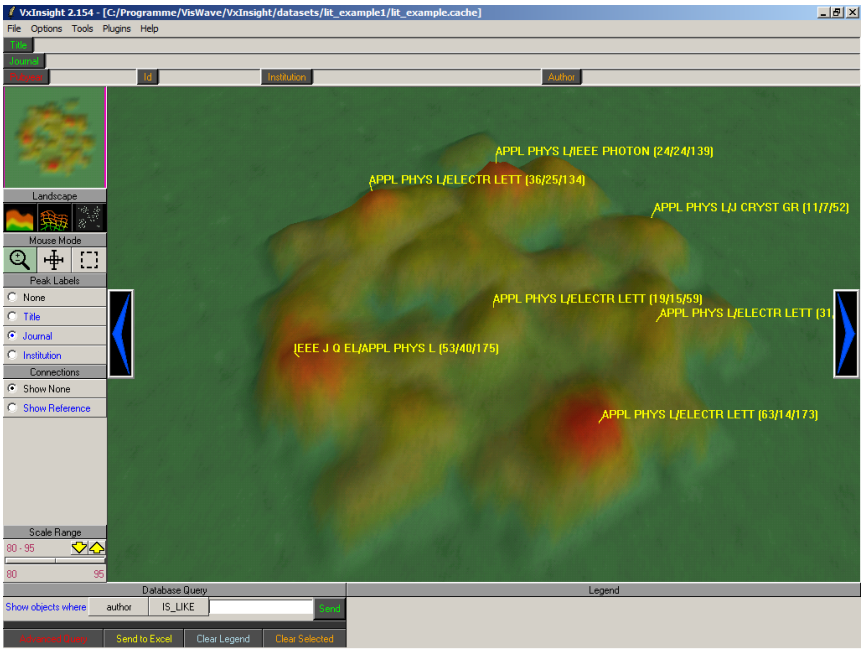


Figure 5.3: VxInsight displaying a set of 1,231 bibliographic records of articles from physics journals. The mountain peaks are labelled with the names of the journals whose articles are congregated in that area.

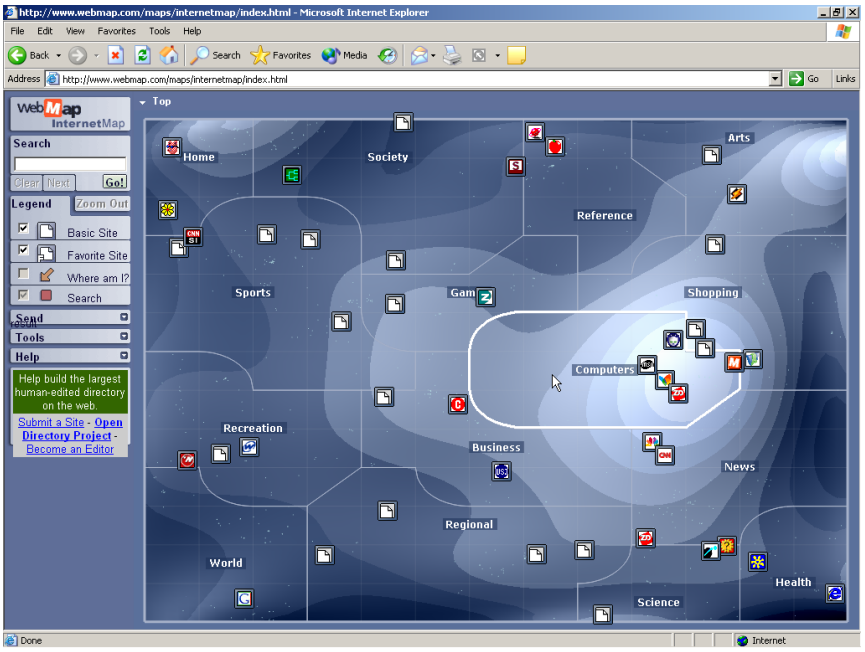


Figure 5.4: WebMap. Within each hierarchical category, web sites are arranged according to their similarity and represented by small white dots. Special web sites are identified with coloured icons.

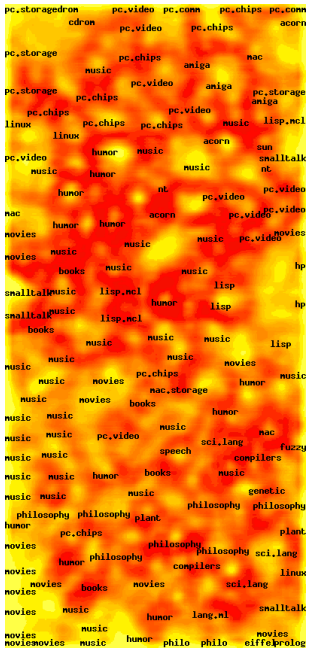


Figure 5.5: WEBSOM. A self-organising map of over a million postings from 83 Usenet newsgroups. Yellow areas represent denser clusters of articles and red areas are empty space (ravines) between the clusters.

5.1.2 Self-Organising Maps

A completely different approach is taken by WEBSOM [WEBSOM, 2000; Kohonen et al., 2000] and other systems based on neural networks. Self-organising maps [Kohonen, 2000] use neural networks to thematically organise and visualise very large document collections. Figure 5.5 shows over a million postings from 83 Usenet newsgroups. Yellow areas show denser clusters of articles and red areas are empty space (ravines) between the clusters. The main problem with self-organising maps, however, is that the underlying neural networks generally have to undergo extensive training in order to achieve good results.

5.2 VisIslands

The IICM's VisIslands [Andrews et al., 2001; Sabol, 2001] system supports dynamic thematic clustering of search result sets, in a manner similar to SPIRE's ThemeView [Wise, 1999]. Like the Search Result Explorer described in Section4.3, it utilises the xFIND search engine [Gütl, 2000].

The search result set is first pre-clustered using either hierarchical agglomerative clustering (for smaller result sets, since run time complexity $O(n^2 logn)$) or k-means clustering (for larger result sets in time $O(n)$). Anil K. Jain and Flynn [1999] gives a good overview of clustering algorithms. The cluster centroids are then distributed randomly in the viewing rectangle. Next, documents belonging to each cluster, as determined by the initial pre-clustering, are placed in

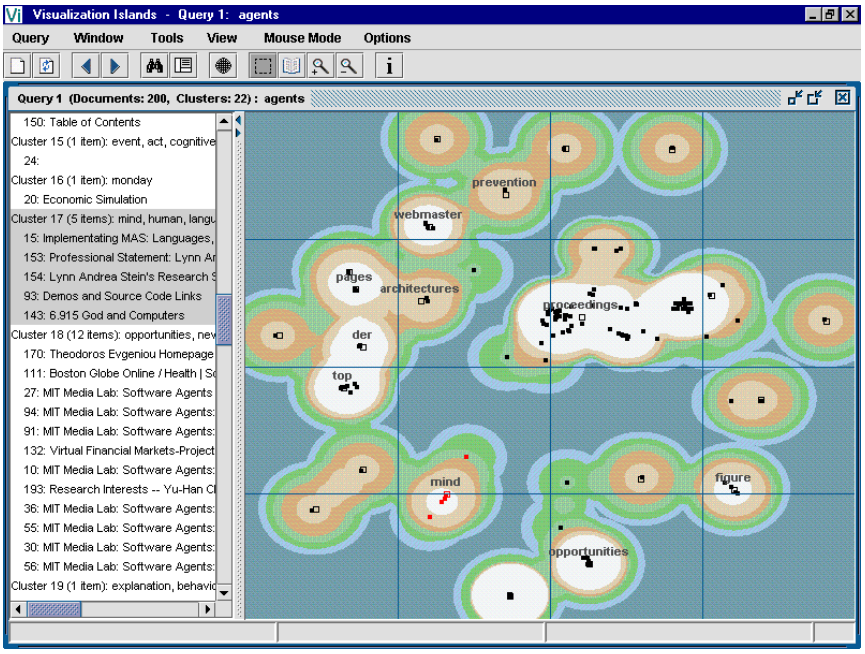


Figure 5.6: VisIslands. 200 documents matching the query “agents”. Pre-clustering using hierarchical agglomerative clustering has identified the 22 clusters shown in the left-hand panel. Cluster 17 has been selected.

a ring around each centroid. This arrangement is fine-tuned using a sampling linear iteration force-directed placement algorithm derived from Chalmers [1996b]. Documents similar to one another are attracted towards each other. After a certain cut-off point, the arrangement has stabilised, and each document contributes its weight to the height field of the grids within which it lies. Dense areas of many documents have corresponding peaks. The overall result resembles a contour map of islands.

Figure 5.6 shows the islands visualisation for the first 200 documents matching the query “agents”. Pre-clustering has identified 22 clusters and Cluster 17 concerning “mind, human, language” has been selected by the user. Note the corresponding visual cluster of red documents in the islands display. Figure 5.7 shows the metadata associated with the centroid of Cluster 17.

In Figure 5.8 the user focuses on Cluster 22, containing 108 documents. Note that fine tuning with force-directed placement has attracted one document, which pre-clustering assigned to Cluster 22, over towards the “webmaster” and “architectures” clusters. Zooming in on Cluster 22, Figure 5.9 shows that, in fact, many of the documents assigned to Cluster 22 on pre-clustering, should perhaps have been assigned to a separate cluster called “erläuterungen” (explanations).

VisIslands improves upon a simple linear list of search results by presenting the result set in topical clusters. Typically, users will be interested in just one or two of the clusters, and can visually determine which of the search results are of interest in the current context. Informal, formative testing showed that users liked this kind of representation more than simple linear lists of matching documents.

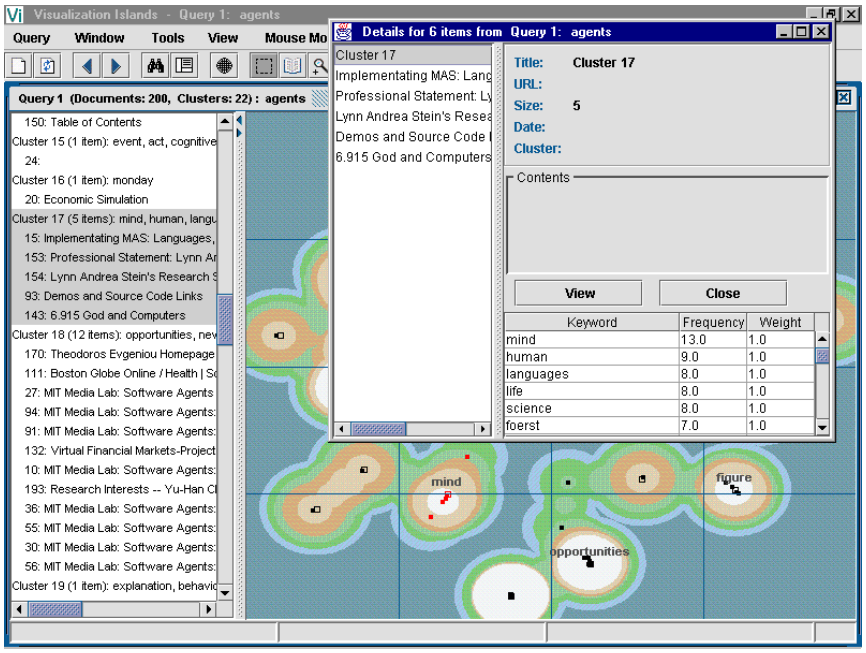


Figure 5.7: VisIslands. The metadata associated with Cluster 17 is displayed. Its most frequent terms include “mind”, “human”, and “language”.

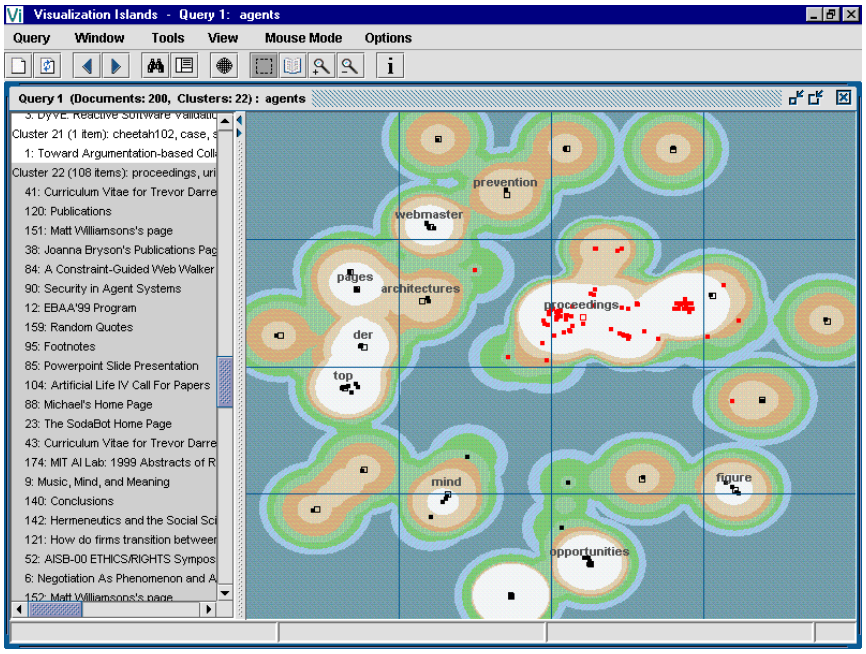


Figure 5.8: VisIslands. Cluster 22 deals with a variety of topics including “proceedings” and “conference”.

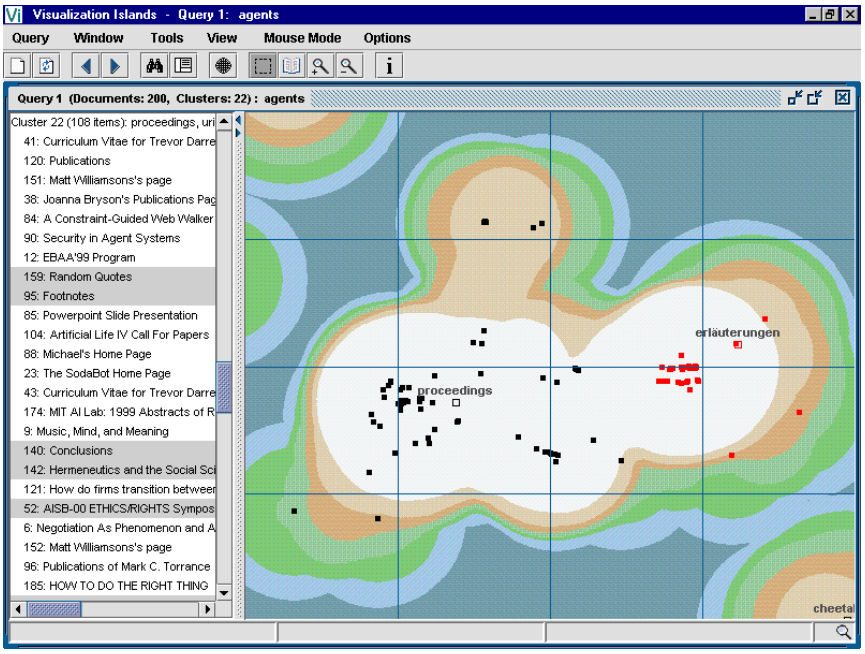


Figure 5.9: VisIslands. After zooming in on Cluster 22, the group of documents on the right-hand peak has been manually selected.

5.3 InfoSky

Due to the ever-increasing number of documents stored within corporate intranets as well as on the world wide web, hierarchical structures for organising documents into collections are beginning to replace flat repositories. Many current retrieval and visualisation tools, however, operate on flat, unstructured repositories.

To address the need for a next generation document repository visualisation tool for large, hierarchically organised document collections, the following requirements were formulated:

1. *Scalability*. Visualise very large (hundreds of thousands, if not millions of entities), hierarchically structured document repositories.

2. *Hierarchy plus similarity*. Represent both the hierarchical organisation of documents and inter-document similarity within a single, consistent visualisation.

3. *Focus plus context*. Integrate both a global and a local view of the information space into one seamless visualisation.

4. *Stability*. Use a stable metaphor which promotes visual recall and recognition of features. The visualisation should remain largely unchanged at a global level even if changes occur to the underlying document repository on a local level.

5. *Unified frame of reference*. Support a single, consistent view of the document space for all users, regardless of the access rights of each individual user, thus providing a common

frame of reference for all parties.

6. *Exploration*. Provide simple, intuitive facilities to browse and search the repository.

The InfoSky [Kappe et al., 2002; Andrews et al., 2002] visual explorer, shown in Figure 5.10, was designed to meet these requirements. InfoSky is a joint development of the Know-Center [Know-Center, 2002], Hyperwave [Hyp, 2002] and the IICM, Graz University of Technology [IICM, 2002].

InfoSky enables users to explore large, hierarchically structured document collections. Similar to a real-world telescope, InfoSky employs a planar graphical representation with variable magnification. Documents of similar content are placed close to each other and are visualised as stars, forming clusters featuring distinct shapes, which are easy to recall. For greater performance, the hierarchical structure is exploited and force-directed placement is applied recursively at each level on much fewer objects, rather than on the corpus as a whole.

Collections of documents at a particular level in the hierarchy are visualised with bounding polygons using a modified weighted Voronoi diagram. The area of each Voronoi polygon is related to the number of documents contained. Textual labels are displayed dynamically during navigation, adjusting to the visualisation content. Navigation is animated and provides a seamless zooming transition between summary and detail view.

Users can map metadata such as document size or age to attributes of the visualisation such as colour and luminance. Queries can be made and matching documents or collections are highlighted. Formative usability testing is ongoing; a small baseline experiment comparing the telescope browser to a traditional tree browser has been carried out.

5.3.1 The InfoSky User Interface

The InfoSky visual explorer employs the metaphor of a zooming galaxy of stars, organised hierarchically into clusters. InfoSky assumes that documents are already organised in a hierarchy of collections and sub-collections, called the *collection hierarchy*. Both documents and collections can be members of more than one parent collection, but cycles are explicitly disallowed. This structure is otherwise known as a directed acyclic graph. The collection hierarchy might, for example, be a classification scheme or taxonomy, manually maintained by editorial staff, but the collection hierarchy could also be created or generated (semi-)automatically.

Documents are assumed to have significant textual content, which can be extracted if necessary with specialised tools. Documents are typically text, PDF, HTML, or Word documents, but may also include spreadsheets and many other formats.

Access to both documents and collections is restricted according to assigned user rights. Depending on their access rights, certain users may not be able to “see” particular documents or collections. Meta-information present in the repository, such as author and modification date, can be processed and visualised by the InfoSky system, but the core visualisation is generated from actual document content.

InfoSky combines both a traditional tree browser and a new telescope view of a galaxy. In the galaxy, documents are visualised as stars, with similar documents forming clusters of stars. Collections are visualised as polygons bounding clusters and stars, resembling the boundaries

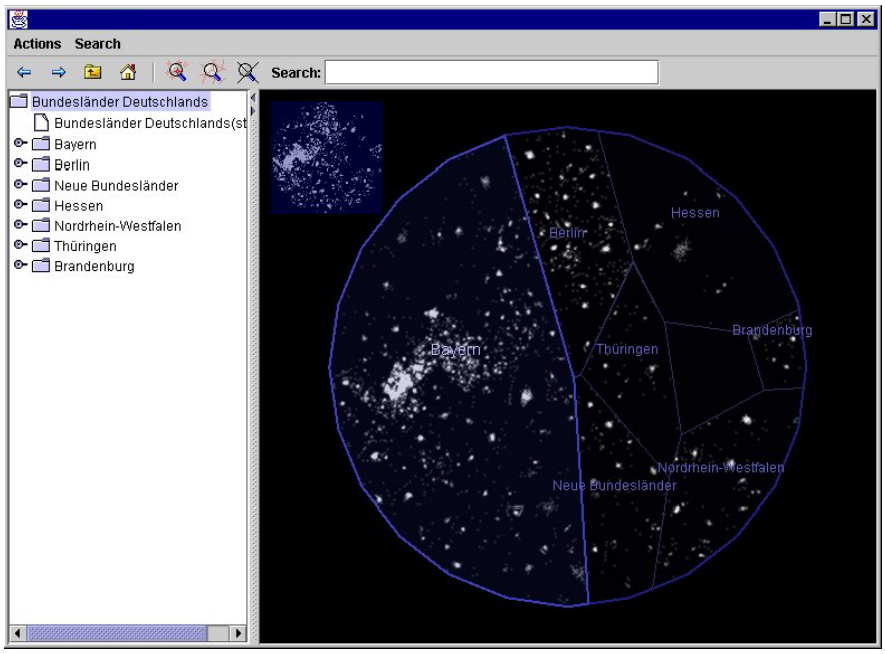


Figure 5.10: The InfoSky visual explorer. A traditional tree browser is combined with a galaxy view. The galaxy here represents approximately 109,000 news articles from the German daily newspaper, the Süddeutsche Zeitung, which have been manually classified into a hierarchy of some 6,900 collections and sub-collections upto 15 levels deep.

of constellations in the night sky. Collections featuring similar content are placed close to each other, as far as the hierarchical structure allows. Empty areas remain where documents are hidden due to access right restrictions, and resemble dark nebulae found quite frequently within real galaxies.

The telescope is used as a metaphor for interaction with the visualisation. Users can pan the view point within the visualised galaxy, like an astronomer can point a telescope at any point of the sky. Magnification can be increased to reveal details of clusters and stars, or reduced to display the galaxy as a whole. Several facilities support users in operating this virtual telescope. Simple interactions cause the system to automatically shift focus to an object of interest and magnify it to optimal viewing size. When changing the magnification or position manually, constellation boundaries are automatically displayed and hidden to avoid display cluttering. Finally, history and bookmark functions allow easy recall of previously visited "galactic coordinates".

The right hand side of Figure 5.10 shows a galaxy view in InfoSky. This galaxy is derived from a collection of approximately 109,000 German language news articles from the German daily newspaper, the Süddeutsche Zeitung. The articles have been classified thematically by the newspaper's editorial staff into around 6,900 collections and sub-collections upto 15 levels deep.

Constellation boundaries and labels are shown for the topmost level of the hierarchy. The galaxy itself is complete in the sense that it displays all the stars it contains, down to the bottommost level of the hierarchy. At this level of magnification, individual stars are not discernible. The clusters forming the galaxy consist of thousands of stars which, in accordance with the telescope metaphor, can only be resolved individually at a higher magnification.

As shown in Figure 5.10, the InfoSky user interface consists of four major elements:

- A control panel featuring pull-down menus and a button toolbar, which is located in the upper part of the window. This panel controls options and features such as history, bookmarks, and search.
- A traditional tree view control featuring collapsible and expandable folders. This element is located in the left part of the window and is synchronised with the telescope visualisation.
- The telescope view displaying the galaxy at the currently selected location and magnification, on the right side of the window.
- A small overview chart of the whole galaxy, which always indicates the current position and magnification of the telescope, is embedded in the upper left corner of the telescope view.

5.3.2 Navigation in InfoSky

Interactive exploration (navigation) of the galaxy is achieved through a combination of browsing and searching capabilities. Selection of a region of interest (a collection or document) causes

Figure 5.11: InfoSky. The collection “München” (Munich) and its surroundings. The mouse is hovering over the collection “Kultur in München” (Culture in Munich) and hence it is highlighted.

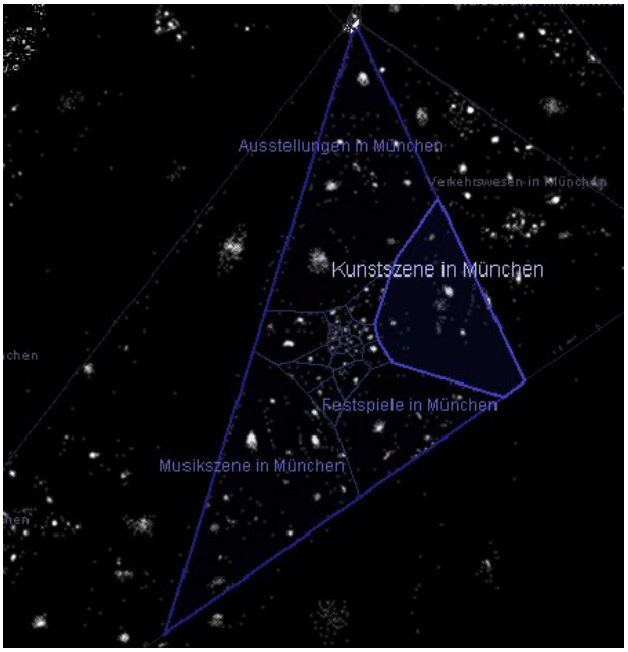


Figure 5.12: InfoSky. Clicking zooms in on and centres the collection “Kultur in München” (Culture in Munich). The mouse is currently over the collection “Kunstszene in München” (Arts Scene in Munich), highlighting it.

that region to be smoothly *auto-centred*: the viewport and magnification are adjusted so that the region of interest is displayed in full. In addition, the user can freely change the current view by changing the magnification (*zooming*) and sliding the viewport around at the current magnification (*panning*). While zooming and panning, collections are *auto-selected* based on magnification and position: the maximum level of the hierarchy fitting completely inside the viewport is determined and the collection at that level nearest to the centre of the viewport is selected.

To ensure the widest possible audience, only a keyboard and mouse are used for navigation. In the current prototype, the following navigational facilities are provided (note that these can easily be changed and extended):

- *Selecting a collection*: Left-clicking a collection selects the collection and auto-centres it.
- *Selecting a document*: Left-clicking an individual star selects the corresponding document and auto-centres it.
- *Zooming in on a document*: Left-clicking again an already selected star further increases magnification to display the star and its immediate neighbourhood, and individual star labels are displayed.
- *Selecting the parent collection*: Right-clicking at any location selects the parent collection of the currently selected star or collection. The viewport is zoomed out to display the parent collection.
- *Continuous hierarchical zoom*: Holding down the left mouse button at any location activates continuous hierarchical zoom. After a brief delay, ever deeper sub-levels in the hierarchy located under the mouse pointer are successively opened up and highlighted. Releasing the left mouse button at any point selects the currently highlighted collection. Moving the mouse pointer outside the currently highlighted collection at any point resets the process to the initial hierarchy level.
- *Panning*: Dragging with the left mouse button pans the viewport. Collections are auto-selected based on magnification and position.
- *Zooming*: Dragging with the right mouse button changes magnification. Collections are auto-selected based on magnification and position.

The many features supporting interaction are very important for intuitive navigation. In particular, for users who are familiar with it, continuous hierarchical zoom represents a significant advance over conventional step-by-step browsing of a hierarchy. Similar to related work on zooming interfaces [Bederson and Hollan, 1994; Bederson, 2002], continuous hierarchical zoom allows users to bypass upper levels of the hierarchy and quickly move to a known position within the galaxy. Without continuous zoom, users must explicitly select the correct parent collection at each hierarchy level, until the desired collection is reached, resulting in a greatly increased number of interactions as in a conventional tree browser.

Figure 5.11 displays part of the galaxy whose global view is shown in Figure 5.10. All child collections of the currently selected collection “München” (Munich) are displayed with

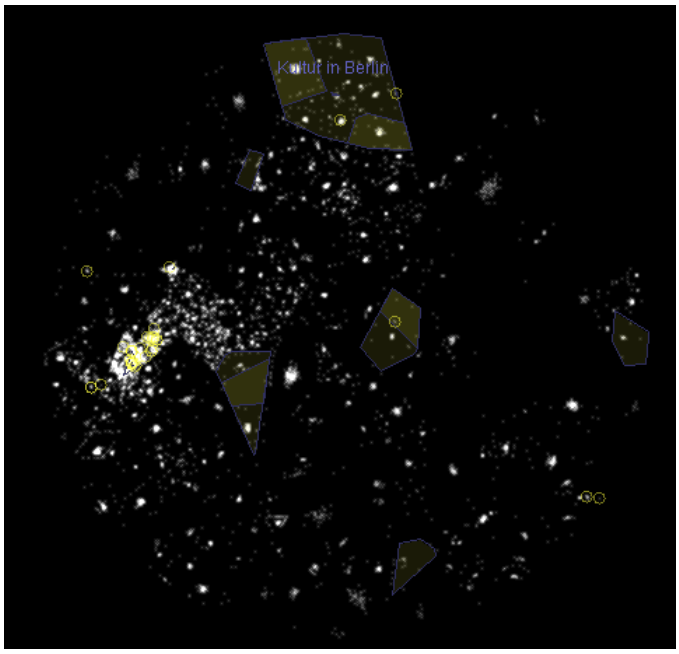


Figure 5.13: InfoSky. A query for the word "kultur" (culture) displaying both matching collections and matching documents. Matching documents pulsate to further differentiate them.

boundaries and labels (as far as overlapping can be avoided). The collection over which the mouse is currently positioned "Kultur in München" (Culture in Munich) is highlighted. Left-clicking the highlighted collection selects it.

Figure 5.12 displays the galaxy after selecting the collection "Kultur in München" (Culture in Munich) by left-clicking it. The collection has been centred and magnification adjusted to make the collection fill the viewport. In Figure 5.12, the level of magnification reached allows some clusters to be resolved into individual stars representing documents, enabling users to select those individual documents directly. Furthermore, the clusters reveal distinctive shapes and structures, making them easier to remember and recall.

Users can search for documents and collections contained in the corpus by issuing a query. Matching documents and collections are highlighted and can be examined in further detail. In the current prototype, searching for documents and searching for collections are provided as separate, orthogonal operations. A query is formulated in the search box, and either or both of the buttons "Matching Collections" or "Matching Documents" are activated:

- *Search for collections*: Collections at all levels of the hierarchy which match the query are highlighted in shades of yellow.
- *Search for documents*: Matching documents at all levels of the hierarchy are given pulsating yellow halos.

Figure 5.13 shows both matching collections and matching documents corresponding to the query "kultur" (culture). For experienced analysts, a further query mode allows the results of

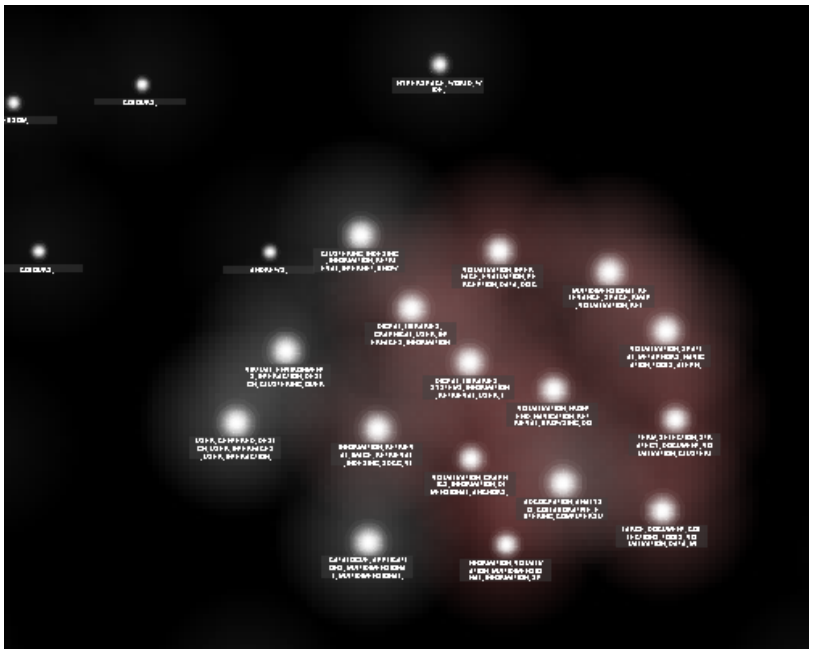


Figure 5.14: InfoSky. Stars displayed at high magnification with a halo representing their search result relevance value.

individual queries to be assigned to an individual colour channel and overlays created to express the combined results of several queries, as shown in Figure 5.14.

5.3.3 The Implementation of InfoSky

InfoSky is implemented as a client-server system. On the server side, galaxy geometry is created and stored for a particular hierarchically structured document corpus. On the client side, the subset of the galaxy visible to a particular user is visualised and made explorable to the user. Java was chosen as the development platform for both client and server, because of its platform-independence and geometric libraries. Together, these components are able to generate a galaxy representation from millions of documents within a few hours, and to visualise the galaxy in real time on a standard desktop computer.

The galactic geometry is generated from the underlying repository recursively from top to bottom in several steps:

1. First, at each level, the centroids of any subcollections are positioned in a normalised 2D plane according to their similarity with each other using a similarity placement algorithm. The similarities to their parent's sibling collection centroids are used as static influence factors to ensure that similar neighbouring subcollections across collection boundaries tend towards each other (they are not allowed to actually cross the boundary). The centroid of a synthetic subcollection called "Stars", which holds the documents at that level of the hierarchy, is also positioned.

2. The layout in normalised 2D space is transformed to the polygonal area of the parent collection using a simple geometric transformation.

3. Next, a polygonal area is calculated around each subcollection centroid, whose size is related to the total number of documents and collections contained in that subcollection (at all lower levels). This polygonal partition of the parent collection's area is done with a modified Voronoi diagram.

4. Finally, documents contained in the collection at this level are positioned using the similarity placement algorithm as points within the synthetic "Stars" collection, according to their inter-document similarity and their similarity to the subcollection centroids at this level, which are used as static influence factors.

Three algorithms are particularly prominent:

1. *Similarity placement*: Similarity placement is used to position both subcollection centroids within their parent collection and to position documents within the synthetic Stars collection. Similarity placement is realised using an optimised force-directed placement algorithm and is described in detail in Section 5.3.4.

2. *Geometric transformation*:

   Geometric transformation is described in detail in Section 5.3.5.

3. *Area partition*: The centroids of subcollections are used to partition the polygon representing the parent collection into polygonal sub-areas. The size of each sub-area is related to the total number of documents contained within the corresponding sub-collection. Area partition is accomplished using modified, weighted Voronoi diagrams and is described in detail in Section 5.3.6.

Basing the layout on the underlying hierarchical structure of the repository has a major advantage in terms of performance. Similarity placement typically has a run-time complexity approaching $O(n^2)$, where n is the number of objects being positioned. However, since similarity placement is only used on one level of the hierarchy at a time, the value of n is generally quite small (the number of sub-collection centroids plus the number of documents at that level).

5.3.4 Similarity Placement

Force-directed placement (FDP) is an iterative method for mapping a set of high-dimensional vectors to a low-dimensional space, whilst preserving their high-dimensional relations as far as possible. The algorithm calculates force vectors from the similarities between documents and collection centroids. These forces, and additional, custom-defined forces, influence the position of the objects at each iteration in the placement algorithm.

High dimensional vector representation allows comparison of a pair of objects by computing a similarity metric between them. InfoSky uses a cosine similarity metric. If O_i and O_j are objects to be compared, H is the dimensionality of the high-dimensional space, and $x_{i,q}$ is the

q'th component of the term vector representing object O_i, the cosine similarity metric is given by:

$$sim(O_i, O_j) = \frac{\sum_{k=1}^{H} (x_{i,k} x_{j,k})}{\sqrt{\sum_{k=1}^{H} x_{i,k}^2 \sum_{k=1}^{H} x_{j,k}^2}}$$

and has a value in $[0, 1]$. In InfoSky, the objects placed by force-directed placement are either individual documents or the centroids of collections and subcollections. Layout is performed on a 2d plane with infinite space in each direction (upto the limits of real number precision). The layout's bounding box is determined by keeping a running track of the maximum and minimum coordinate in each direction as layout proceeds.

Objects are initially placed randomly in 2d space in the interval $[[-1, -1], [1, 1]]$ and forces are computed between them. A traditional spring force model of force-directed placement would express the force between two objects O_i and O_j at each iteration in terms of the difference between their Euclidean separation in the current (2d) layout, $dist$, and their Euclidean separation in high-dimensional space, $distHD$:

$$force(O_i, O_j) = dist(O_i, O_j) - distHD(O_i, O_j)$$

where $distHD$ is usually normalised to $[0, 1]$.

Since InfoSky uses a cosine similarity metric, the high-dimensional distance is expressed as the inverse of the similarity:

$$distHD = 1 - sim(O_i, O_j)$$

where the similarity lies in $[0, 1]$.

The first version of InfoSky used the following simple spring model:

$$force(O_i, O_j) = dist(O_i, O_j) - (1 - sim(O_i, O_j))$$

This simple model attempts to reflect high-dimensional distances in low-dimensional space and should therefore in theory reproduce high-dimensional relations most faithfully. However, the model's simplicity does not allow any fine control over the final layout. In our experience the largest drawback of the standard spring model is the fact that very similar objects may be placed extremely close to one another causing occlusions in the final display.

InfoSky now uses a modified force model which allows finer control of the layout and produces visually more appealing layouts. The force acting between two objects has three components: an attractive component proportional to the similarity between the objects, a repulsive component inversely proportional to their current 2D distance, and a weak gravitational component:

$$force(O_i, O_j) = sim(O_i, O_j)^d - \frac{w}{dist(O_i, O_j)^r} + grav$$

The first component pulls similar objects together. It was found that adjusting discriminator $d > 0$ (with a standard value of 1) to the characteristics of the similarity matrix can significantly improve the separation of the layout. The second component pushes objects apart. Exponent r controls the compactness of very dense clusters, preventing objects from coming too close.

Factor w is the weight of the object and influences the area each object occupies. In the case of collection centroids, w is proportional to the total number of documents contained in that collection subtree. Objects with larger weight also tend to be pushed towards the boundaries of the layout (this is important for generating well-proportioned Voronoi polygons, see Section 5.3.6). The third component is a weak but constant gravitational force which provides overall cohesion to the layout by ensuring that even very dissimilar objects attract once they become very distant.

The new coordinates of an object are calculated by letting it interact with other objects from the set, subsequently averaging the results over all interactions. For example, assuming that object O_i interacts with a set M of other objects, its new x coordinate $O_i.x$ is given by:

$$O_i.x = \frac{1}{|M|} \sum_{j \in M} force(O_i, O_j) * O_j.x + (1 - force(O_i, O_j)) * O_i.x$$

The new y coordinate is calculated analogously.

At each iteration, a new position is computed for every object, and the iteration continues until a termination condition is satisfied. The commonly used termination condition of mechanical stress is computationally intensive. It was therefore replaced with a more lightweight, adaptive condition which can be summarised as: the execution terminates when object positions stabilise sufficiently, or when a maximum number of iterations is reached.

For a set of N objects, to calculate the influence of every object on every other object, each object would have to interact with $|M| = N - 1$ others, resulting in quadratic time complexity for each iteration. However, if $|M|$ can be held constant, a linear (per iteration) execution time can be achieved. Chalmers [1996b] describes a stochastic sampling algorithm, where each object maintains two small sets of constant size. The *random set* is refilled with random elements every iteration, and the *neighbour set* maintains a list of similar, neighbouring objects. In each iteration, members of the neighbour set are compared to the new samples in the random set, and are replaced by any elements which are more similar.

For performance reasons the implemented algorithm does not use velocities or viscosity. As a result of random sampling, a certain amount of jitter is introduced. This jitter causes a small inaccuracy of the computed object positions, but proves to be useful in avoiding local minima (falling into a local minimum is a well-known drawback of FDP algorithms). Generally speaking, the sampling algorithm introduces little computing overhead and requires the same number or fewer iterations than the non-sampling version to reach stable layouts.

Once a layout has been calculated with the sampling algorithm, one or more iterations are performed with the non-sampling $O(N^2)$ per iteration algorithm. This step was found to almost eliminate the layout inaccuracy introduced by sampling. So as not to compromise time complexity, the number of iterations of the non-sampling algorithm is carefully limited, such that the total number of object interactions is smaller than the number of object interactions just performed by the sampling algorithm.

Finally, the object positions are transformed using the layout bounding box to a normalised space with coordinates in $[-1, 1]$ in each direction.

5.3.5 Geometric Transformation

Given a layout of objects (subcollection centroids or documents) in a normalised space with coordinates in $[-1, 1]$ in each direction, the task now is to transform each object's position to the polygonal bounds of the objects' parent collection using a simple geometric transformation.

Assume there exists a set P of n points p_i, with $p_i.x \in [-1, 1]$ and $p_i.y \in [-1, 1]$. Let there be a convex polygon A with m bounding edges a_j which form a closed path. To inscribe P into A, i.e. to transform all p_i such that they are contained within A, but the original proportions are preserved as far as possible, the following algorithm is used:

1. Calculate the centroid of the polygon, c.
2. For each point p_i:
   (a) Construct ray r_i running from centroid c through point p_i.
   (b) Determine which bounding edge a_j intersects r_i.
   (c) Calculate the intersection point q_i of r_i with edge a_j.
   (d) Calculate the resulting point p'_i, by projecting p_i onto the ray $(c - q_i)$.

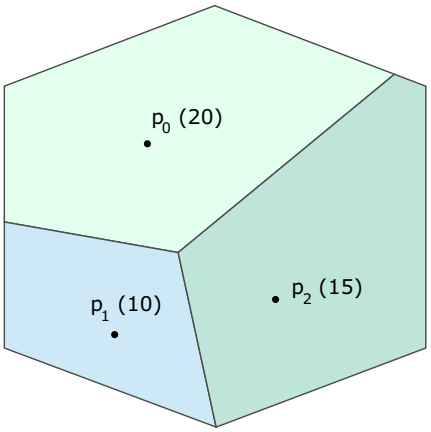
In other words, each point is moved radially towards the polygon centroid, whereby the distance of the new resulting point from the centroid is proportional to the distance of the original point from the layout origin.

5.3.6 Area Partition

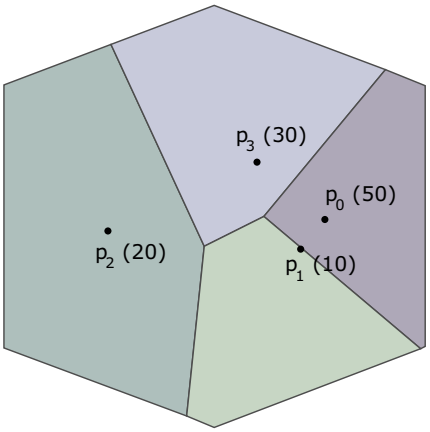
Considering one level of the repository hierarchy, there are N points p_i of known weight w_i, representing the centroids of the subcollections of the current collection. These points have been positioned within a given polygonal area A representing the area of the current collection (or an initial starting area for the top-level collection). The problem is now to find a partition of area A into N sub-areas A_i which satisfy:

- $p_i \in A_i$
- A_i is convex
- $A_i \sim w_i$
- A_i is larger in area than a certain minimum value

The calculation of the area subdivision is accomplished using a modified version of the additively weighted power Voronoi diagram [Okabe et al., 2000, pg. 128], shown in Figure 5.15(a). The area of each polygon is related to the weight of each point. Point p_0 with a weight of 20 has a larger area than p_2 with a weight of 15, and they are both larger than the area of p_1 with a weight of 10. In the InfoSky, the weight of each subcollection centroid is proportional to the total number of documents in that subcollection sub-tree.



(a) A standard additively weighted power Voronoi diagram. The distance between points is relatively large and weight differences are marginal.



(b) If distances are very small, or weight differences are very large, the bisector between two points can overshoot, as is almost the case here with p_1. Scale factor f avoids this.

Figure 5.15: Additively weighted power Voronoi diagrams. The weight of each point is indicated in brackets.

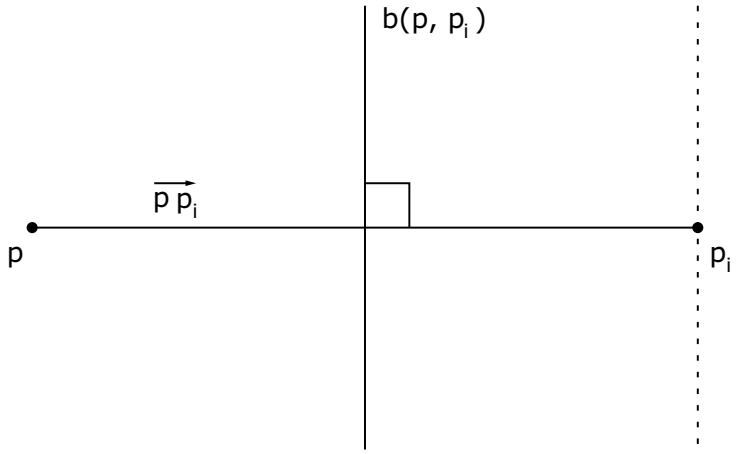
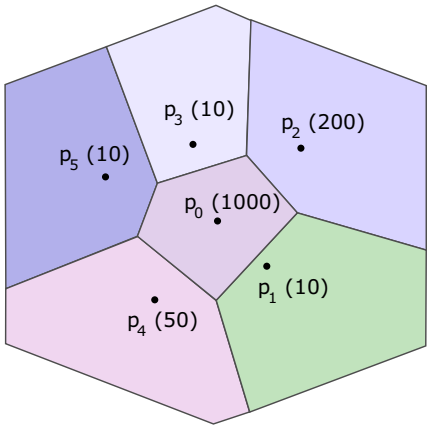
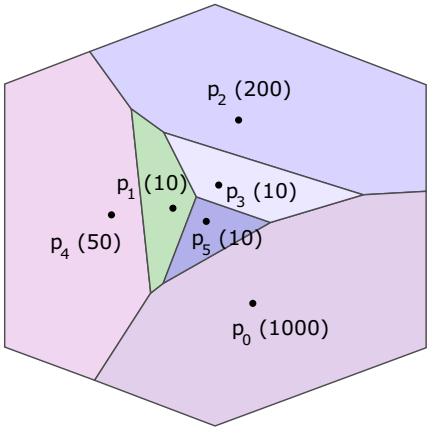


Figure 5.16: Geometric relationships in the additively weighted power diagram.



(a) The introduction of scale factor f can lead to solutions where areas are no longer related to weights.



(b) Modifying the similarity placement algorithm so that heavier subcollection centroids tend towards the boundaries and lighter subcollection centroids tend towards the centre of the layout alleviates this problem.

Figure 5.17: Ensuring that areas are related to weights.

For two points p and p_i shown in Figure 5.16, the additively weighted power distance given by:

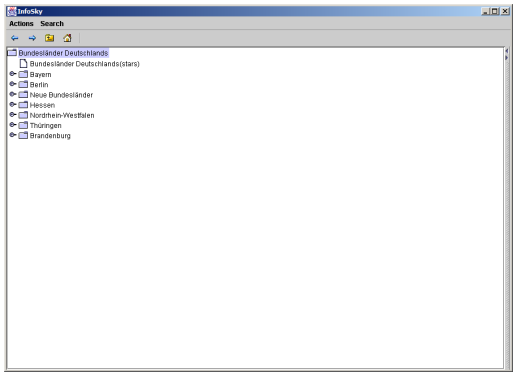
$$d_{pw}(p, p_i; w_i) = ||\vec{p} - \vec{p_i}||^2 - w_i$$

is used to determine the position of the bisector $b(p, p_i)$ perpendicular to $\overline{pp_i}$ which forms an edge of the polygon around p.

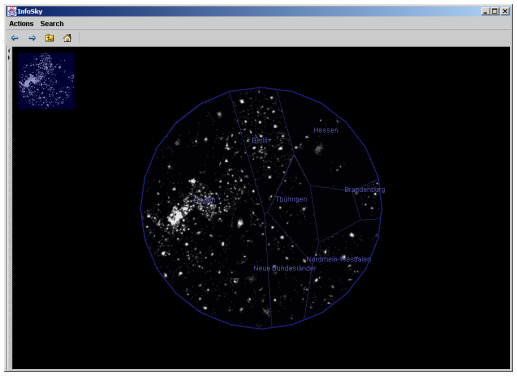
However, the additively weighted power distance has the property that if the weight difference between two points is very large and the points are close to one another, the lighter point can lie on the wrong side of the bisector and hence outside its own area. For example, in Figure 5.16, if the weight of p_i is much less than p and the distance between them is short, bisector b can move so far to the right of p_i, that it goes beyond p_i.

In Figure 5.15(b), points p_0 and p_1 are close together and their weight difference is relatively large, pushing the bisector towards and almost beyond p_1. For InfoSky, each collection centroid must lie within its own polygonal area. To ensure that each centroid p_i lies within its own area A_i, each w_i is scaled down by a global factor f such that all bisectors $B(p_i, p_j)$ fall between p_i and p_j. Factor f is defined as the maximum scale factor which can be uniformly applied to all weights without causing a bisector to overrun.

Unfortunately, since the outer polygon boundaries are fixed and only the inner boundaries (bisectors) can slide, the introduction of scale factor f leads to cases where an area A_i is no longer related to its weight w_i. Such cases occur when relatively light points are placed close to the outer border of the boundary polygon, or are placed in between a number of other points. In Figure 5.17(a), point p_0 has a weight of 1000, but its area is smaller than points p_1, p_3, and p_5 having only weights of 10.



(a) The tree view browser, test condition TV.



(b) The telescope browser, test condition TS.

Figure 5.18: The two browsers as used in the study.

To avoid this behaviour, the force-directed placement algorithm was modified (as described in Section 5.3.4) so that heavier objects (subcollection centroids) tend to move towards the boundaries and lighter objects tend to move towards the centre of a layout. The resulting solution expresses weights through their corresponding areas if geometrically possible, and satisfies all the conditions above. Figure 5.17(b) shows the same points and weights as Figure 5.17(a), but heavier points have been placed nearer the boundaries of the bounding polygon.

5.3.7 InfoSky User Study

Some initial formative thinking aloud testing of the entire InfoSky prototype was done with colleagues at the Know-Center. It was then decided to run a first formal experiment to establish a baseline comparison between the InfoSky telescope browser and the InfoSky tree browser. The two browsers used are shown in Figure 5.18. Users were only allowed to use one or the other in isolation. Both browsers were used in full screen mode, and the search box was removed (the search menu item should have been removed as well, since some users wanted to use it during the test).

Due to users' much greater familiarity with Windows Explorer style tree browsers and the early development nature of the telescope browser prototype, it was expected that users would probably be more efficient initially using the tree browser alone than the prototype telescope browser alone, but running a study would provide a feeling for how much of a difference there was. Note that this initial study did not test the power of a combination of tree browser, telescope browser, and search functionality – that will be tested at a later date.

The dataset from the Süddeutsche Zeitung was taken and two sets of tasks were formulated (five pairs of equivalent tasks). The tasks were designed to be equivalent between the two sets in the sense that their solutions lay at the same level of the hierarchy and involved inspecting approximately the same number of choices at each level. The test environment set up as shown in Figure 5.19. A pilot test with one user was run and some slight modifications were made. Table 5.1 shows the modified tasks as used in the main study in English translation, the study itself was done in German.

Figure 5.19: The environment used for user testing.

Task	*Wording*
A1	Find the collection Weimar in Thüringen.
A2	Find the collection Elections in North Rhine-Westphalia.
A3	Which state has more entries on the topic of Transportation: Bavaria or Berlin? [Bavaria]
A4	How many documents are available on the topic of The Relationship of Brandenburg to Berlin? [12]
A5	Find a document about the Voting Out of the Red-Green Coalition in Hessen.
B1	Find the collection Cologne in North Rhine-Westphalia.
B2	Find the collection Elections in Thüringen.
B3	Which city has more entries: Essen in North Rhine-Westphalia or Bad Reichenhall in Bavaria? [Bad Reichenhall]
B4	How many documents are available on the topic of The Governments of Thüringen? [6]
B5	Find a document about the Election of Sepp Niedermaier as Mayor of Bad Tölz in Bavaria.

Table 5.1: Sets A and B of five equivalent tasks. The test was carried out in German, the tasks are given here in English translation.

Test Person	*Ordering*			
TP1	TS	A	TV	B
TP2	TS	A	TV	B
TP3	TS	B	TV	A
TP4	TS	B	TV	A
TP5	TV	A	TS	B
TP6	TV	A	TS	B
TP7	TV	B	TS	A
TP8	TV	B	TS	A

Table 5.2: Ordering of tasks and browsers for each test person.

Browser/Task	*TP1*	*TP2*	*TP3*	*TP4*	*TP5*	*TP6*	*TP7*	*TP8*	*Av*	*Diff*
TV1	7	12	2	5	11	63	15	7	15.25	
TS1	21	24	10	40	9	7	22	33	20.75	5.50
TV2	8	6	4	9	7	6	8	14	7.75	
TS2	15	9	11	22	3	15	11	7	11.63	3.88
TV3	69	40	42	84	13	34	23	137	55.25	
TS3	189	114	35	121	32	71	74	22	82.25	27.00
TV4	8	8	7	13	32	41	6	14	16.13	
TS4	32	94	35	16	84	48	39	21	46.13	30.00
TV5	32	37	75	85	55	43	32	18	47.13	
TS5	148	72	52	143	50	57	36	194	94.00	46.88

Table 5.3: The timings in seconds for eight test users for each of five tasks in the TV and TS conditions.

Eight employees of Hyperwave were recruited for the study and divided randomly into four groups of two. Four users began with the telescope browser (condition TS), then used the tree view (condition TV). The other four users began with TV then used TS. Within these conditions two users started with task set A, the other two with task set B. This experimental design is shown in Table 5.2. Before using the telescope browser, users were given two minutes of brief training on the browser's features. At the end of each test, an interview was conducted with the test user to gain additional feedback. The entire session was videotaped.

In the TS condition users were presented with the InfoSky telescope browser at full screen, in the TV condition with the InfoSky tree view browser at full screen. Users were not given access to both browsers simultaneously, nor were they allowed to use the InfoSky search function. The intention of the study was purely to compare the telescope browser to a traditional tree view browser.

The results of the study are summarised in Table 5.3. Timings were determined by analysing the videotape of each session and noting the time in seconds from the time the facilitator read the last word of the task to the time the task was completed.

On average, the tree browser performed better than the prototype telescope browser for each of the tasks tested. The overall difference between tree browser and telescope browser was significant at $p < 0.05$ (paired samples t-test, 39 degrees of freedom, $t = 3.038$). The reasons for the slower performance of the telescope browser appear to be two-fold. Firstly, users typically have spent many, many hours using a traditional explorer-like tree browser and are very familiar with its metaphor and controls. They were not familiar with the telescope browser and two minutes of training could not make up the deficit.

Secondly, whereas the tree view component has already undergone many iterations of development, the telescope browser is a prototype at a fairly early stage of development and has numerous bugs and issues affecting its usability. Some of the problems in the current implementation of the telescope browser which emerged during the study were:

1. The Voronoi polygons in the centre of each collection were far too small for many test users and a minimum size should probably be set.

2. When near the bottom of the hierarchy, where collections contained many documents, users were confused by the "jumping around" of document titles. The prototype displayed the titles of those documents which were "near" to the cursor. Users, however, consistently wanted to use the cursor to visually step through what they perceived to be a list of documents, but moving the cursor caused document titles to appear and disappear seemingly at random.

3. When more than a handful of document titles were displayed, the telescope display became cluttered.

Problems 2 and 3 might be alleviated by displaying a scrolling, linear list of documents once users reach a level of the hierarchy where they want to inspect document titles.

During the study, one problem each with the data set, the task wording, and the test system became apparent:

- The synthetic collection "Stars" containing documents at a particular level of the collection hierarchy was confusing to users.

- In tasks 3 and 4, which involved counting or estimating numbers of documents, users were sometimes unsure if only documents at that level were meant, or all documents at and beneath that level (which was actually our intention). During the test, the facilitator accepted either approach.

- Although the search box had been removed from the tested version of the browsers, the search menu item was still visible. Three users attempted to use the search functionality through the menu, but were then asked not to by the facilitator.

However, these problems applied equally to both tree browser and telescope browser and are not thought to have biased the study.

When interviewed after the test, users indicated that they were very familiar with a tree browser and liked being able to use the mouse cursor as a visual aid when scanning lists. They liked the overview which the telescope browser provided and could imagine using it for exploring a corpus of documents. This study did not include a task asking users to find similar or related documents or subcollections, something which the telescope metaphor should support quite well. Users further indicated that a combination of both browsers and search functionality could be very powerful. This is something which InfoSky provides, but was not tested in this study.

As development and formative (thinking aloud) testing is ongoing, these issues will hopefully start to be resolved, and a second experiment to measure performance of an improved telescope browser can then be carried out.

5.3.8 Discussion

The initial calculation of cluster geometry for the approximately 109,000 documents shown in Figure 5.10 takes about 4 hours in the current prototype on a standard desktop PC (Pentium III, 850MHz, 256MB, JDK 1.3.1). Work is ongoing to revise and extend both the server and the client side of InfoSky. The server components will be integrated into a commercial knowledge management system, the Hyperwave eKnowledge Infrastructure [Hyp, 2002]. This will make the InfoSky visual explorer available to a substantial number of users, accessing large, real-world document repositories within a cooperative environment. As a consequence, extensive feedback and usage testing will be possible and further small and larger scale usability studies are planned.

Advances in the rendering technology available for standard desktop computers will permit constant improvement of the visualisation client. The implementation of the similarity placement and area partition algorithms could further be improved.

Using its telescope and galaxy metaphor, the InfoSky system addresses several key requirements for such systems. In particular, InfoSky:

- uses Voronoi diagrams and force-directed placement to integrate hierarchical structure and document similarity into a single, consistent visualisation.
- exploits the hierarchical structure to reduce the number of objects being compared during similarity placement.
- is scalable to very large, hierarchically structured document repositories.
- solves the problem in other hierarchical subdivision algorithms that objects on either side of a collection boundary may be close in proximity but quite dissimilar in content.
- allows users to both browse the collection hierarchy and view search results in context.
- integrates both a global and a local view of the information space into one seamless visualisation.
- provides a unified view to multiple users with potentially differing access rights.

The field of information visualisation has evolved rapidly over the past decade, but practical applications of information visualisation techniques are still not widespread. Although the baseline user study showed the current telescope browser on its own to be less efficient than a tree view browser on its own, it is hoped that further improvements will make it comparable.

The complete InfoSky armoury of synchronised tree browser, telescope browser, and search in context has not yet been tested against other methods of exploring large hierarchical document collections. Nor have tasks involving finding related or similar documents or subcollections been tested, something the telescope metaphor should be well-suited to. As development proceeds, it is hoped that the InfoSky prototype will constitute a step towards practical, user-oriented, visual exploration of large, hierarchically structured document repositories.

Chapter 6

Concluding Remarks

This thesis presented an overview of my work in the field of information visualisation over the past seven years. My work has spanned much of the field. It began with hypermedia visualisation in the early 1990s, included lengthy forays into hierarchies, some jousting with multidimensional metadata visualisation, and continues most recently with text corpus and vector space visualisation. It has certainly been interesting, sometimes frustrating, but in the end always illuminating.

In the future, opportunities many lie in combining individual information visualisation techniques into integrated hybrid packages. Countless application domains could benefit from the introduction of information visualisation packages. The field of bioinformatics is an especially promising melting pot for techniques from both traditional scientific visualisation and information visualisation. Much work remains to be done in evaluating and comparing techniques by way of usability studies.

New techniques await discovery, exciting application domains fill the radar, and usability studies are long overdue . . . may this thesis be a foundation for the road ahead!

Appendix A

Information Visualisation Resources

A.1 Books

- Card, Mackinlay, Shneiderman; *Readings in Information Visualization : Using Vision to Think*; Morgan Kaufman, Jan. 1999. ISBN 1558605339
- Bob Spence; *Information Visualization*; Addison-Wesley, Dec. 2000. ISBN 0201596261
- Colin Ware; *Information Visualization: Perception for Design*; Morgan Kaufmann, Jan. 2000. ISBN 1558605118
- Chaomei Chen; *Information Visualisation and Virtual Environments*; Springer, Nov. 1999. ISBN 1852331364.
- Fayyad et al; *Information Visualization in Data Mining and Knowledge Discovery*; Morgan Kaufmann, Aug. 2001. ISBN 1558606890.
- Jacques Bertin; *Semiology of Graphics : Diagrams, Networks, Maps*; Univ. of Wisconsin Press, 1983. ISBN 0299090604 [Out of print]
- Jacques Bertin; *Sémiologie graphique Les diagrammes - Les réseaux - Les cartes*; Reissue of the 1973 edition, Editions EHESS, 1999. In French. ISBN 2713212774
- Robert Harris; *Information Graphics: A Comprehensive Illustrated Reference*; Oxford University Press, Feb. 2000. ISBN 0195135326
- Richard Saul Wurman; *Information Architects*; Graphis Press, 1996. ISBN 3857094583
- Edward Tufte; *Visual Explanations*; Graphics Press, 1997. ISBN 0961392126
- Edward Tufte; *The Visual Display of Quantitative Information*; Graphics Press, 1992. ISBN 096139210X
- Edward Tufte; *Envisioning Information*; Graphics Press, 1990. ISBN 0961392118

A.2 Journals

- Information Visualization, Palgrave Macmillan. http://www.palgrave-journals.com/ivs/
- IEEE Computer Graphics and Applications (CG&A). http://www.computer.org/cga/
- IEEE Transactions on Visualization and Computer Graphics (TVCG). http://www.computer.org/tvcg/

A.3 Articles

- Ben Shneiderman; *The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations*; Proc. 1996 IEEE Symposium on Visual Languages (VL'96), Boulder, Colorado, Sept. 1996. [Chapter 15 of [Shneiderman, 1997]]
- Herman et al; *Graph Visualisation and Navigation in Information Visualisation: A Survey*; IEEE Transactions on Visualization and Computer Graphics, Vol. 6, No. 1, Jan.–Mar. 2000. http://www.cwi.nl/InfoVisu/Survey/FinalFromIEEE.pdf
- Gershon, Eick, Card; *Information Visualization*; ACM Interactions, March/April 1998.
- Nahum Gershon and Steve Eick; *Visualization's New Tack: Making Sense of Information*; IEEE Spectrum, Nov. 1995.

A.4 Conferences

- IEEE Symposium on Information Visualization (InfoVis). Since 1995. The main conference in the field, very focused. http://www.infovis.org/
- International Conference on Information Visualisation, London. Since 1997. Much broader in scope and less focused than InfoVis. http://www.graphicslink.demon.co.uk/IV02/
- There are usually some information visualisation papers at the CHI, AVI, and UIST conferences.
- Knowledge and Information Visualisation 2003 (KIV 2003), Graz, Austria. http://www.know-center.at/en/conference/i-know03/workshop.htm

A.5 Online Resources

- Gary Ng; *Information Visualization Resources*; May 2002. http://www.cs.man.ac.uk/~ngg/InfoViz/

- Martin Dodge; *Cyber-Geography Research* April 2002. http://www.cybergeography.org/
- Michael Reed and Dan Heller; *OLIVE: On-line Library of Information Visualization Environments*; University of Maryland, Nov. 1997. http://www.otal.umd.edu/Olive/
- Peter Young; *Three Dimensional Information Visualisation*; University of Durham, Nov. 1996. http://vrg.dur.ac.uk/misc/PeterYoung/pages/work/documents/

A.6 Companies

Suppliers of information visualisation toolkits and components:

- Inxight http://www.inxight.com
- Spotfire http://www.spotfire.com/
- Maya Viz http://www.mayaviz.com/
- OmniViz http://www.omniviz.com/
- VisWave http://www.viswave.com/
- VisIntuit http://www.visintuit.com/
- Oculus http://www.oculusinfo.com/
- Visual Insights http://www.visualinsights.com/
- Tom Sawyer Software http://www.tomsawyer.com/

Bibliography

Ahlberg, C. and Shneiderman, B. (1994a). *Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays*. In Proc. CHI'94, pages 313–317, Boston, Massachusetts (1994a). ACM. `ftp://ftp.cs.umd.edu/pub/hcil/Reports-Abstracts-Bibliography/3131html/3131.html`. 57

Ahlberg, C. and Shneiderman, B. (1994b). *Visual Information Seeking using the FilmFinder*. In CHI'94 Video Program. ACM. 57

Ahlberg, C., Williams, C. and Shneiderman, B. (1992). *Dynamic Queries for Information Exploration: An Implementation and Evaluation*. In Proc. CHI'92, pages 619–626, Monterey, California (1992). ACM. 57

Andrews, K. (1995). *Visualising Cyberspace: Information Visualisation in the Harmony Internet Browser*. In Proc. First IEEE Symposium on Information Visualization (InfoVis'95), pages 97–104, Atlanta, Georgia (1995). `ftp://ftp.iicm.edu/pub/papers/ivis95.pdf`. 18, 45

Andrews, K. (1996). *Browsing, Building, and Beholding Cyberspace: New Approaches to the Navigation, Construction, and Visualisation of Hypermedia on the Internet*. PhD thesis, Graz University of Technology, Austria. `http://www.iicm.edu/keith-phd`. 19, 45

Andrews, K. (1998). *Visualizing Rich, Structured Hypermedia*. IEEE Computer Graphics and Applications, 18(4):40–42. `http://www.computer.org/cga/cg1998/g4040abs.htm`. 23

Andrews, K. (1999). *Visualising Cyberspace: Information Visualisation in the Harmony Internet Browser*. In Card, S. K., Mackinlay, J. D. and Shneiderman, B., editors, Readings in Information Visualization, pages 493–502. Morgan Kaufmann. 19

Andrews, K. (2002). *Visual Exploration of Large Hierarchies with Information Pyramids*. In Proc. Sixth International Conference on Information Visualisation (IV02), pages 793–798, London, England (2002). IEEE Computer Society Press. 27

Andrews, K., Gütl, C., Moser, J., Sabol, V. and Lackner, W. (2001). *Search Result Visualisation with xFIND*. In Proc. UIDIS 2001, pages 50–58, Zurich, Switzerland (2001). IEEE Computer Society Press. 63, 73

Andrews, K. and Heidegger, H. (1998). *Information Slices: Visualising and Exploring Large Hierarchies using Cascading, Semi-Circular Discs*. In Late Breaking Hot Topic Paper, IEEE

Symposium on Information Visualization (InfoVis'98), pages 9–12, Research Triangle Park, North Carolina (1998). ftp://ftp.iicm.edu/pub/papers/ivis98.pdf. 23

Andrews, K. and Kappe, F. (1994). *Soaring Through Hyperspace: A Snapshot of Hyper-G and its Harmony Client*. In Herzner, W. and Kappe, F., editors, Proc. Eurographics Symposium on Multimedia/Hypermedia in Open Distributed Environments, pages 181–191, Graz, Austria (1994). Springer. ftp://ftp.iicm.edu/pub/papers/egmm94.pdf. 18, 45

Andrews, K., Kappe, F. and Maurer, H. (1995). *Serving Information to the Web with Hyper-G*. Computer Networks and ISDN Systems, 27(6):919–926. Proc. Third International World-Wide Web Conference, WWW'95, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/proceedings/papers/105/hgw3.html. 18, 45

Andrews, K., Kienreich, W., Sabol, V., Becker, J., Droschl, G., Kappe, F., Granitzer, M., Auer, P. and Tochtermann, K. (2002). *The InfoSky Visual Explorer: Exploiting Hierarchical Structure and Document Similarities*. Information Visualization, 1(3/4). To appear. 36, 77

Andrews, K., Pichler, M. and Wolf, P. (1996). *Towards Rich Information Landscapes for Visualising Structured Web Spaces*. In Proc. 2nd IEEE Symposium on Information Visualization (InfoVis'96), pages 62–63, San Francisco, California (1996). ftp://ftp.iicm.edu/pub/papers/ivis96.pdf. 19, 46

Andrews, K., Wolte, J. and Pichler, M. (1997). *Information Pyramids: A New Approach to Visualising Large Hierarchies*. In IEEE Visualization'97, Late Breaking Hot Topics Proc., pages 49–52, Phoenix, Arizona (1997). ftp://ftp.iicm.edu/pub/papers/vis97.pdf. 27

Anil K. Jain, M. N. M. and Flynn, P. J. (1999). *Data Clustering: A Review*. ACM Computing Surveys, 31(3):264–323. http://www.acm.org/pubs/citations/journals/surveys/1999-31-3/p264-jain/. 73

AT&T Labs - Research (2002). Graphviz. http://www.research.att.com/sw/tools/graphviz/. 40, 41

Beaudoin, L., Parent, M.-A. and Vroomen, L. C. (1996). *Cheops: A Compact Explorer for Complex Hierarchies*. In Proc. Visualization'96, pages 87–92, San Francisco, California (1996). IEEE Computer Society. http://www.crim.ca/hci/cheops/paper.html. 16

Bederson, B. (2002). *Jazz*. http://www.cs.umd.edu/hcil/jazz/. 81

Bederson, B. and Hollan, J. (1994). *Pad++: A Zooming Graphical Interface for Exploring Alternative Interface Physics*. In Proc. UIST'94, pages 17–26, Marina del Rey, CA (1994). ACM. 81

Benedikt, M. (1991). *Cyberspace: Some proposals*. In Benedikt, M., editor, Cyberspace: First Steps. MIT Press. 55

Bertin, J. (1983). *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press. ISBN 0299090604. 55

Boardman, R. (2000). *Bubble Trees: Visualization of Hierarchical Information Trees*. In CHI 2000 Extended Abstracts, The Hague, The Netherlands (2000). http://www.iis.ee.ic.ac.uk/~rick/research/pubs/bubbletree-chi2000.pdf. 34

Buchheim, C., Jünger, M. and Leipert, S. (2002). *Improving Walker's Algorithm to Run in Linear Time*. In Proc. Graph Drawing 2002, Irvine, California (2002). Springer LNCS. http://www.zaik.uni-koeln.de/~paper/index.html?show=zaik2002-431. 6, 16

Burger, T. (1999). Magic Eye View: Eine neue Fokus + Kontext Technik zur Darstellung von Graphen. In german, University of Rostock, Institute of Computer Graphics. http://wwwicg.informatik.uni-rostock.de/Diplomarbeiten/1999/Thomas_Buerger/. 16

Card, S. K. and Mackinlay, J. D. (1997). *The Structure of the Information Visualization Design Space*. In Proc. IEEE InfoVis'97, pages 92–99, Phoenix, Arizona (1997). IEEE Computer Society. 55

Chalmers, M. (1993). *Using a Landscape Metaphor to Represent a Corpus of Documents*. In Spatial Information Theory, Proc. COSIT'93, pages 377–390, Boston, Massachusetts (1993). Springer LNCS 716. http://www.dcs.gla.ac.uk/~matthew/papers/ecsit93.pdf. 69

Chalmers, M. (1996a). *Adding Imageability Features to Information Displays*. In Proc. UIST'96, Seattle, Washington (1996a). ACM. http://www.dcs.gla.ac.uk/~matthew/papers/uist96.pdf. 69

Chalmers, M. (1996b). *A Linear Iteration Time Layout Algorithm for Visualising High-Dimensional Data*. In Proc. Visualization'96, pages 127–132, San Francisco, California (1996b). IEEE Computer Society. http://www.dcs.gla.ac.uk/~matthew/papers/vis96.pdf. 41, 69, 74, 86

Cohen, J. D. (1997). *Drawing Graphs to Convey Proximity: An Incremental Arrangement Method*. ACM Transactions on Computer-Human Interaction, 4(3):197–229. http://www.acm.org/pubs/citations/journals/tochi/1997-4-3/p197-cohen/. 42

CRIM (2002). CRIM's Hierarchical Engine for OPen Search. http://www.crim.ca/hci/cheops/. 16

Davidson, G. S., Hendrickson, B., Johnson, D. K., Meyers, C. E. and Wylie, B. N. (1998). *Knowledge Mining with VxInsight: Discovery Through Interaction*. Journal of Intelligent Information Systems, 11(3):259–285. http://www.cs.sandia.gov/projects/VxInsight/pubs/jiis98_prepub.pdf. 70

di Battista, G., Eades, P., Tamassia, R. and Tollis, I. G. (1999). *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, New Jersey. ISBN 0133016153. 39

Dodge, M. (2002). *An Atlas of Cyberspaces*. http://www.cybergeography.org/atlas/atlas.html. 5

Dykstra, P. (1991). XDU. Army Research Laboratory. ftp://ftp.arl.mil/pub/xdu-3.0.tar.Z. 8

Eades, P. (1984). *A Heuristic for Graph Drawing*. Congressus Numerantium, 42:149–160. 41

Eades, P. and Sugiyama, K. (1990). *How to Draw a Directed Graph*. Journal of Information Processing, 13(4):424–437. 45

Eyl, M. (1995). The Harmony Information Landscape: Interactive, Three-Dimensional Navigation Through an Information Space. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/meyl.pdf. 19

Fairchild, K. M., Poltrock, S. E. and Furnas, G. W. (1988). *SemNet: Three-Dimensional Representations of Large Knowledge Bases*. In Guindon, R., editor, Cognitive Science and its Applications for Human-Computer Interaction, pages 201–233. Lawrence Erlbaum, Hillsdale, New Jersey. 42

Fruchtermann, T. M. J. and Reingold, E. M. (1991). *Graph Drawing by Force-Directed Placement*. Software: Practice and Experience, 21(11):1129–1164. 41

Goel, A. (1999). *Parallel Coordinates Visualization Applet*, Virginia Tech. http://csgrad.cs.vt.edu/~agoel/parallel_coordinates/. 55

Goerzen, J. (2002). Gopher. http://gopher.quux.org:70/. 16

Gütl, C. (2000). xFIND: Extended Framework for Information Discovery. IICM, Graz University of Technology. http://xfind.iicm.edu/. 63, 73

Haan, B. J., Kahn, P., Riley, V. A., Coombs, J. H. and Meyrowitz, N. K. (1992). *IRIS Hypermedia Services*. Communications of the ACM, 35(1):36–51. 45

Harlan, H. M. (2000a). Method and apparatus for displaying a thought network from a thought's perspective. US Patent 6037944 , Natrificial. Filed 7th Nov. 1996, issued 14th March 2000. 44

Harlan, H. M. (2000b). Method and apparatus for displaying a thought network from a thought's perspective. US Patent 6031537 , Natrificial. Filed 14th July 1997, issued 29th Feb. 2000. 44

HCIL (2002). Treemap 3. University of Maryland. http://www.cs.umd.edu/hcil/treemap3/. 8

Hendley, B. J., Drew, N. S., Wood, A. M. and Beale, R. (1995). *Narcissus: Visualising Information*. In Proc. IEEE InfoVis'95, pages 90–96, Atlanta, Georgia (1995). IEEE Computer Society. 41

Hendrickson, B. A. (1999). Method of data mining including determining multidimensional coordinates of each item using a predetermined scalar similarity value for each item pair. US Patent 5930784 , Sandia Corporation. Filed 21st August 1997, issued 27th July 1999. 70

Herman, I., Melancon, G. and Marshall, M. S. (2000). *Graph Visualization and Navigation in Information Visualization: A Survey*. IEEE Transactions on Visualization and Computer Graphics, 6(1):24–43. http://www.cwi.nl/InfoVisu/Survey/StarGraphVisuInInfoVis.html. 5, 39

Hetzler, B., Harris, W. M., Havre, S. and Whitney, P. (1998). *Visualizing the Full Spectrum of Document Relationships*. In Proc. Fifth International ISKO Conference, pages 168–175, Lille, France (1998). ERGON Verlag. http://www.pnl.gov/infoviz/isko.pdf. 69

Holmquist, L. E., Fagrell, H. and Busso, R. (1998). *Navigating Cyberspace with CyberGeo Maps*. In Proc. of Information Systems Research Seminar in Scandinavia (IRIS 21), Saeby, Denmark (1998). http://www.viktoria.informatik.gu.se/groups/play/publications/1998/navigating.pdf. 10

(2002). Hyperwave. http://www.hyperwave.com/. 18, 36, 45, 77, 94

Hyun, Y. (2002). Walrus. http://www.caida.org/tools/visualization/walrus/. 14

Iacovou, N. and McCahill, M. P. (1995). *GODOT: GopherVR Organized Directories of Titles*. In Proc. of CIKM '95 Workshop on New Paradigms in Information Visualization and Manipulation, Baltimore, Maryland (1995). ACM. http://www.cs.umbc.edu/~cikm/1995/npiv/. 16

IICM (2002). *Institute for Information Processing and Computer-Supported New Media*, Graz University of Technology, Austria. http://www.iicm.edu/. 36, 77

Inxight (2002). Inxight Star Tree Product Demos. http://www.inxight.com/products/core/star_tree/demos.php. 14

Iron, M., Neustedt, R. and Ranen, O. (2001). Method of Graphically Presenting Network Information. US Patent Application 20010035885A1 , WebMap. Filed 20th March 2001. 70

Jerding, D. F. and Stasko, J. T. (1988). *The Information Mural: A Technique for Displaying and Navigating Large Information Spaces*. IEEE Transactions on Visualization and Computer Graphics, 4(3):257–271. http://www.cc.gatech.edu/gvu/softviz/infoviz/information_mural.html. 2

Johnson, B. and Shneiderman, B. (1991). *Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures*. In Proc. IEEE Visualization '91, pages 284–291, San Diego, California (1991). IEEE Computer Society. http://www.cs.umd.edu/hcil/treemaps/. 8

Kappe, F., Droschl, G., Kienreich, W., Sabol, V., Becker, J., Andrews, K., Granitzer, M., Tochtermann, K. and Auer, P. (2002). *InfoSky: Eine neue Technologie zur Erforschung großer, hierarchischer Wissensräume*. In KnowTech 2002, Munich, Germany (2002). In German. 36, 77

Kaufmann, M. and Wagner, D., editors (2001). *Drawing Graphs: Methods and Models*. Springer LNCS Tutorial 2025. http://link.springer.de/link/service/series/0558/tocs/t2025.htm, ISBN 3540420622. 39, 41

Köhler, H. (2002). FAEJ3D: A File Attribute Explorer in Java3D. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/hkoehler.pdf. 66

Kleiberg, E., van de Wetering, H. and van Wijk, J. (2001). *Botanical Visualization of Huge Hierarchies*. In Proc. IEEE InfoVis 2001, pages 87–94, San Diego, California (2001). IEEE Computer Society. http://www.win.tue.nl/~vanwijk/botatree.pdf. 18

Know-Center (2002). *The Styrian Competence Center for Knowledge Management*, Graz, Austria. http://know-center.at/. 36, 77

Kohonen, T. (2000). *Self-Organizing Maps*. Springer, third edition. ISBN 3540679219. 73

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V. and Saarela, A. (2000). *Self Organization of a Massive Document Collection*. IEEE Transactions on Neural Networks, 11(3):574–585. 73

Kreuseler, M., Lopez, N. and Schuhmann, H. (2000). *A Scalable Framework for Information Visualization*. In Proc. IEEE InfoVis 2000, pages 27–36, Salt Lake City, Utah (2000). IEEE Computer Society. http://www.informatik.uni-rostock.de/~mkreusel/SInVis/infovis.html. 16

Kreuseler, M. and Schumann, H. (1999). *Information Visualization using a New Focus+Context Technique in Combination with Dynamic Clustering of Information Space*. In Proceedings of the 1999 Workshop on New Paradigms in Information Visualization and Manipulation (NPIV'99), pages 1–5. ACM Press. 16

Lamping, J. and Rao, R. (1994). *Laying out and Visualizing Large Trees Using a Hyperbolic Space*. In Proc. UIST'94, pages 13–14, Marina del Rey, California (1994). ACM. 14

Lamping, J., Rao, R. and Pirolli, P. (1995). *A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies*. In Proc. CHI'95, pages 401–408, Denver, Colorado (1995). ACM. http://www.acm.org/sigchi/chi95/Electronic/documnts/papers/jl_bdy.htm. 14

Lamping, J. O. and Rao, R. B. (1997). Displaying Node-Link Structure with Region of Greater Spacings and Peripheral Branches. US Patent 5619632 , Xerox Corporation. Filed 14th Sept. 1994, issued 8th April 1997. 14

Maurer, H., editor (1996). *HyperWave: The Next Generation Web Solution*. Addison-Wesley. http://www.iicm.edu/hwbook, ISBN 0201403463. 18, 45

McCahill, M. P. and Erickson, T. (1995). *Design for a 3D Spatial User Interface for Internet Gopher*. In Proc. of ED-MEDIA 95, pages 39–44, Graz, Austria (1995). AACE. http://www.iicm.edu/edmedia_papers_ps. 16

Meyers, C. E., Davidson, G. S., Johnson, D. K., Hendrickson, B. A. and Wylie, B. N. (1999). Method of data mining including determining multidimensional coordinates of each item using a predetermined scalar similarity value for each item pair. US Patent 5987470 , Sandia Corporation. Filed 21st August 1997, issued 16th Nov. 1999. 70

Miller, G. A. (2002). *WordNet*, Princeton University. http://www.cogsci.princeton.edu/~wn/. 44

Munzner, T. (1997). *H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space*. In Proc. IEEE InfoVis'97, pages 2–10, Phoenix, Arizona (1997). IEEE Computer Society. http://graphics.stanford.edu/papers/h3/. 14, 39

Munzner, T. (2002). Software. http://www.cs.ubc.ca/~tmm/. 14

Munzner, T. and Burchard, P. (1995). *Visualizing the Structure of the World Wide Web in 3D Hyperbolic Space*. In Proceedings of the 1995 Symposium on the Virtual Reality Modeling Language (VRML'95), pages 33–38. ACM Press. http://graphics.stanford.edu/papers/h3/. 14

Nation, D. (2002). WebTOC. University of Maryland. http://davenation.com/java2/. 6

Nation, D. A., Plaisant, C., Marchionini, G. and Komlodi, A. (1997). *Visualizing websites using a hierarchical table of contents browser: WebTOC*. In Proc. 3rd Conference on Human Factors and the Web, Denver, Colorado (1997). US WEST. ftp://ftp.cs.umd.edu/pub/hcil/Demos/WebTOC/Paper/WebTOC.html. 6

Nowell, L. T., France, R. K., Hix, D., Heath, L. S. and Fox, E. A. (1996). *Visualizing Search Results: Some Alternatives to Query-Document Similarity*. In Proc. SIGIR'96, pages 67–75, Zurich, Switzerland (1996). ACM. 57

Nowell, L. T., Schulman, R. and Hix, D. (2002). *Graphical Encoding for Information Visualization: An Empirical Study*. In Proc. IEEE InfoVis 2002, pages 43–50, Boston, Massachusetts (2002). IEEE Computer Society. 57

Okabe, A., Boots, B., Sugihara, K. and Chiu, S. N. (2000). *Spatial Tesselations: Concepts and Applications of Voronoi Diagrams*. Wiley, second edition. ISBN 0471986356. 87

Plumb Design (2002). ThinkMap. http://www.plumbdesign.com/products/thinkmap. 44

Reingold, E. M. and Tilford, J. S. (1981). *Tidier Drawing of Trees*. IEEE Transactions on Software Engineering, 7(2):223–228. 6, 16

Rennison, E. (1994). *Galaxy of News: An Approach to Visualizing and Understanding Expansive News Landscapes*. In Proc. UIST'94, pages 3–12, Marina del Rey, California (1994). ACM. http://www.acm.org/pubs/citations/proceedings/uist/192426/p3-rennison/. 44

Robertson, G. G., Mackinlay, J. and Card, S. K. (1994). Display of Hierarchical Three-Dimensional Structures with Rotating Substructures. US Patent 5295243 , Xerox Corporation. Filed 29th Dec. 1989, issued 15th March 1994. 12

Robertson, G. G., Mackinlay, J. D. and Card, S. K. (1991). *Cone Trees: Animated 3D Visualizations of Hierarchical Information.* In Proc. CHI'91, pages 189–194, New Orleans, Louisiana (1991). ACM. 2, 12

Sabol, V. (2001). Visualisation Islands: Interactive Visualisation and Clustering of Search Result Sets. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/vsabol.pdf. 73

Salton, G., Wong, A. and Yang, C. S. (1975). *A Vector Space Model for Automatic Indexing*. Communications of the ACM, 18(11):613 ff. 69

Schipflinger, J. (1998). The Design and Implementation of the Harmony Session Manager. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/jschipf.pdf. 18, 45

Shneiderman, B. (1996). *The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations*. In Proc. 1996 IEEE Symposium on Visual Languages, pages 336–343, Boulder, Colorado (1996). IEEE Computer Society. ftp://ftp.cs.umd.edu/pub/papers/papers/3665/3665.ps.Z. 2, 3

Shneiderman, B. (1997). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, MA, third edition. http://www.aw.com/cseng/titles/0-201-69497-2/website/, ISBN 0201694972. 100

Shneiderman, B. and Wattenberg, M. (2001). *Ordered Treemap Layouts*. In Proc. IEEE InfoVis 2001, pages 73–78, San Diego, California (2001). IEEE Computer Society. http://www.computer.org/proceedings/infovis/1342/13420073abs.htm. 10

Skog, T. and Holmquist, L. E. (2000). *Continuous Visualization of Web Site Activity in a Public Place*. In Student Poster, CHI 2000 Extended Abstracts, The Hague, The Netherlands (2000). http://www.viktoria.informatik.gu.se/groups/play/publications/2000/WebAware.pdf. 10

Smith, A. and Clark, D. (2001). Building a Customized Tree View. IBM developerWorks Tutorial. http://www.ibm.com/developerworks/training/. 8

Spotfire (2000). Spotfire. http://www.spotfire.com/. 57

Stasko, J., Catrambone, R., Guzdial, M. and McDonald, K. (2000). *Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations*. International Journal of Human-Computer Studies, 53(5):663–694. http://www.cc.gatech.edu/gvu/ii/sunburst/. 10

Stasko, J. and Zhang, E. (2000). *Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations*. In Proc. IEEE InfoVis 2000, pages 57–65, Salt Lake City, Utah (2000). IEEE Computer Society. http://www.cc.gatech.edu/gvu/ii/sunburst/. 10, 27

Stedile, A. (2001). JMFGraph A Modular Framework for Drawing Graphs in Java. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/astedile.pdf. 49, 51

Strasnick, S. L. and Tesler, J. D. (1996a). Method and Apparatus for Displaying Data within a Three-Dimensional Information Landscape. US Patent 5528735 , Silicon Graphics, Inc. Filed 23rd March 1993, issued 18th June 1996. 12

Strasnick, S. L. and Tesler, J. D. (1996b). Method and Apparatus for Navigation within Three-Dimensional Information Landscape. US Patent 5555354 , Silicon Graphics, Inc. Filed 23rd March 1993, issued 10th Sept. 1996. 12

Sugiyama, K. (2002). *Graph Drawing and Applications for Software and Knowledge Engineers*. World Scientific. http://www.wspc.com/books/compsci/4902.html, ISBN 9810248792. 39

Sugiyama, K., Tagawa, S. and Toda, M. (1981). *Methods for Visual Understanding of Hierachical System Structures*. IEEE Transactions on Systems, Man, and Cybernetics, SMC-11 (2):109–125. 40

Tesler, J. D. and Strasnick, S. L. (1992). *FSN: The 3D File System Navigator*, Silicon Graphics, Inc. ftp://ftp.sgi.com/sgi/fsn. 12, 19

TheBrain.Com (2002a). TheBrain. http://www.thebrain.com. 44

TheBrain.Com (2002b). WebBrain. http://www.webbrain.com. 44

Thomas, J., Cowley, P., Kuchar, O., Nowell, L., Thomson, J. and Wong, P. C. (2001). *Discovering Knowledge Through Visual Analysis*. Journal of Universal Computer Science, 7(6):517–529. http://www.jucs.org/jucs_7_6/discovering_knowledge_through_visual. 69

Tweedie, L., Spence, B., Williams, D. and Bhogal, R. (1994). *The Attribute Explorer*. In CHI'94 Video Program. ACM. 57

Utting, K. and Yankelovich, N. (1989). *Context and Orientation in Hypermedia Networks*. ACM Transactions on Information Systems, 7(1):58–84. 45

VisWave (2002). *VxInsight*. http://www.viswave.com/. 70

Walker, II, J. Q. (1990). *A Node-Positioning Algorithm for General Trees*. Software: Practice and Experience, 20(7):685–705. 6

Wattenberg, M. (1999). *Visualizing the Stock Market*. In CHI 99 Extended Abstracts, pages 188–189, Pittsburgh, Philadelphia (1999). ACM. http://www.smartmoney.com/marketmap/. 10

WebMap (2002). WebMap. http://www.webmap.com/. 70

WEBSOM (2000). WEBSOM - Self-Organizing Maps for Internet Exploration. Helsinki University of Technology. http://websom.hut.fi/websom/. 73

Weitlaner, E. (1999). Metadata Visualisation: Visual Exploration of File Systems and Search Result Sets based on Metadata Attributes. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/eweit.pdf. 59

Welz, M. (1999). The Java Pyramids Explorer: A Portable, Graphical Hierarchy Browser. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/mwelz.pdf. 32

Wexelblat, A. D. and Fairchild, K. M. (1991). Method and system for generating dynamic, interactive visual representations of information structures within a computer. US Patent 5021976 , Microelectronics and Computer Technology Corporation (MCC). Filed 14th Nov. 1988, issued 4th June 1991. 42

Wise, J. A. (1999). *The Ecological Approach to Text Visualization*. Journal of the American Society for Information Science, 50(9):814–835. http://www.vistg.net/hat/Wise_draft/Ch5.Wise.html. 69, 73

Wolf, P. (1996). Three-Dimensional Information Visualisation: The Harmony Information Landscape. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/pwolf.pdf. 19, 46

Wolte, J. (1998). Information Pyramids: Compactly Visualising Large Hierarchies. Master's thesis, Graz University of Technology, Austria. ftp://ftp.iicm.edu/pub/theses/jwolte.pdf. 27

Wood, A. M., Drew, N. S., Beale, R. and Hendley, B. J. (1995). *HyperSpace: Web Browsing with Visualisation*. In Proc. WWW3, pages 21–25, Darmstadt, Germany (1995). http://www.igd.fhg.de/www/www95/proceedings/posters/35/index.html. 41

Young, P. (1996). *Three Dimensional Information Visualisation*, University of Durham. http://vrg.dur.ac.uk/misc/PeterYoung/pages/work/documents/lit-survey/lit-surv.ps.gz. 5