Information Visualization (2002) 1, 166–181

www.palgrave-journals.com/ivs

The InfoSky visual explorer: exploiting hierarchical structure and document similarities

Keith Andrews[1]
Wolfgang Kienreich[2]
Vedran Sabol[3]
Jutta Becker[3]
Georg Droschl[2]
Frank Kappe[2]
Michael Granitzer[3]
Peter Auer[1]
Klaus Tochtermann[3]

[1]Graz University of Technology, Inffeldgasse 16c, A-8010 Graz, Austria; [2]Hyperwave R&D, Albrechtgasse 9, A-8010 Graz, Austria; [3]Know-Center, Inffeldgasse 16c, A-8010 Graz, Austria

Correspondence:
Dr Keith Andrews, Graz University of Technology, Inffeldgasse 16c, A-8010 Graz, Austria; E-mail: kandrews@iicm.edu

Received: 11 October 2002
Revised: 24 October 2002
Accepted: 24 October 2002

Abstract
InfoSky is a system enabling 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 with distinct shapes. 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 whole corpus. Collections of documents at a particular level in the hierarchy are visualised with bounding polygons using a modified weighted Voronoi diagram. Their area 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 tree browser is discussed.
Information Visualization (2002) **1,** 166–181. doi:10.1057/palgrave.ivs.9500023

Keywords: Information visualisation; navigation; document retrieval; search; hierarchical repositories; knowledge management

Introduction

The problem of interactively visualising very large, hierarchically structured document collections, as well as visualising the results of retrieval operations executed on such collections, has recently received much attention. 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. However, many current retrieval and visualisation tools operate on flat, unstructured repositories.

Tools for managing and exploring information and knowledge can often mean the difference between success and failure in the knowledge economy. Users want both to browse large structured information spaces and be able to search them based on explicit criteria. Today, typical desktop computers are capable of visualising millions of entities in real time. Interactive graphical representations of abstract information are becoming increasingly common and users are becoming familiar with such systems.

With this in mind, the following requirements for a next generation document repository visualisation tool 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 visualisation tool should allow the visualisation to display a maximum number of document properties and relationships without any need for user interaction. It should thus offer a means of locating documents without specifying a query, by simply browsing the information space. Furthermore, the tool should feature a number of additional information channels to which users can map document properties of their choice, again replacing explicit queries with navigation.

The InfoSky system, shown in Figure 1, addresses the above requirements. 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.

Section 2 presents the philosophy and interface of the InfoSky visual explorer. Section 3 discusses the implementation of InfoSky, in particular the details of two important algorithms. Section 4 describes user testing of the InfoSky prototype. Section 5 discusses current approaches to the visualisation and exploration of large document repositories. Section 6 describes possible next steps in the development of InfoSky.

InfoSky

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. 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.

The telescope metaphor

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 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 1 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 bottom-most 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.

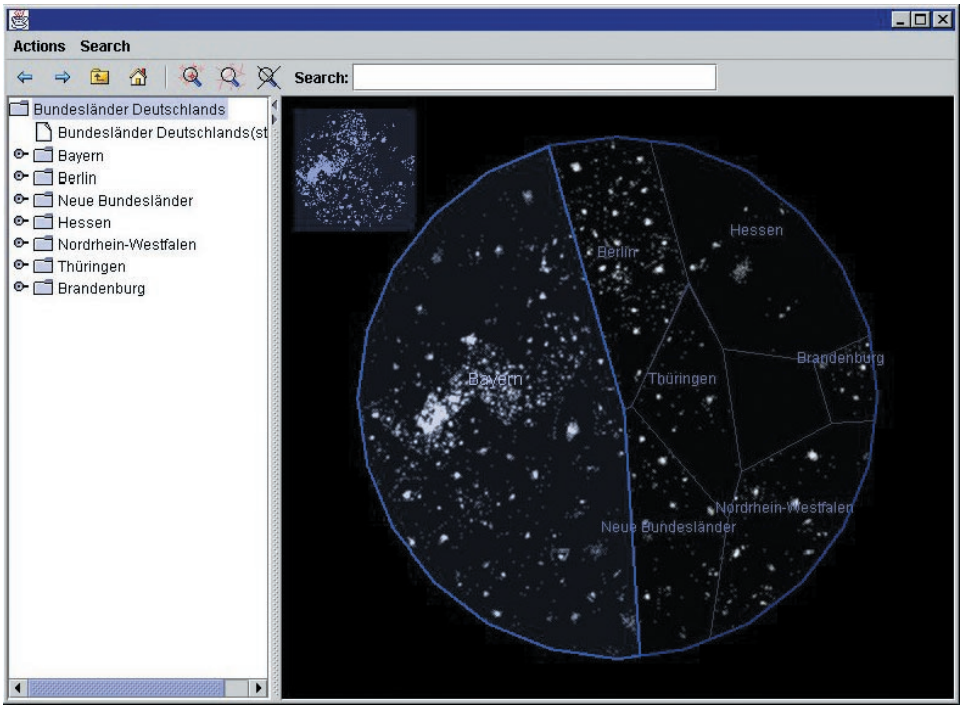

Figure 1 The user interface of the InfoSky visual explorer. A traditional tree browser is combined with a galaxy view. The galaxy here represents approximately 109,000 news articles from the German daily newspaper, the *Süddeutsche Zeitung*, which have been manually classified into a hierarchy of some 6,900 collections and sub-collections upto 15 levels deep.

User interface

As shown in Figure 1, 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.

Browsing

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 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,[1,2] 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 2 displays part of the galaxy whose global view is shown in Figure 1. All child collections of the currently selected collection 'München' (Munich) are displayed with 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 3 displays the galaxy after selecting the collection 'Kultur in München' by left-clicking it. The collection has been centred and magnification adjusted to make the collection fill the viewport. In Figure 3, 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.

Searching

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

InfoSky implementation

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

Figure 2 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.

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 Similarity placement.

(2) *Geometric transformation*: Geometric transformation is described in detail. (See Geometric transformation)

(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 subarea 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 (see Area partition).

Basing the layout on the underlying hierarchical structure of the repository has a major advantage in terms of performance. Similarity placement typically has a runtime 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).

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

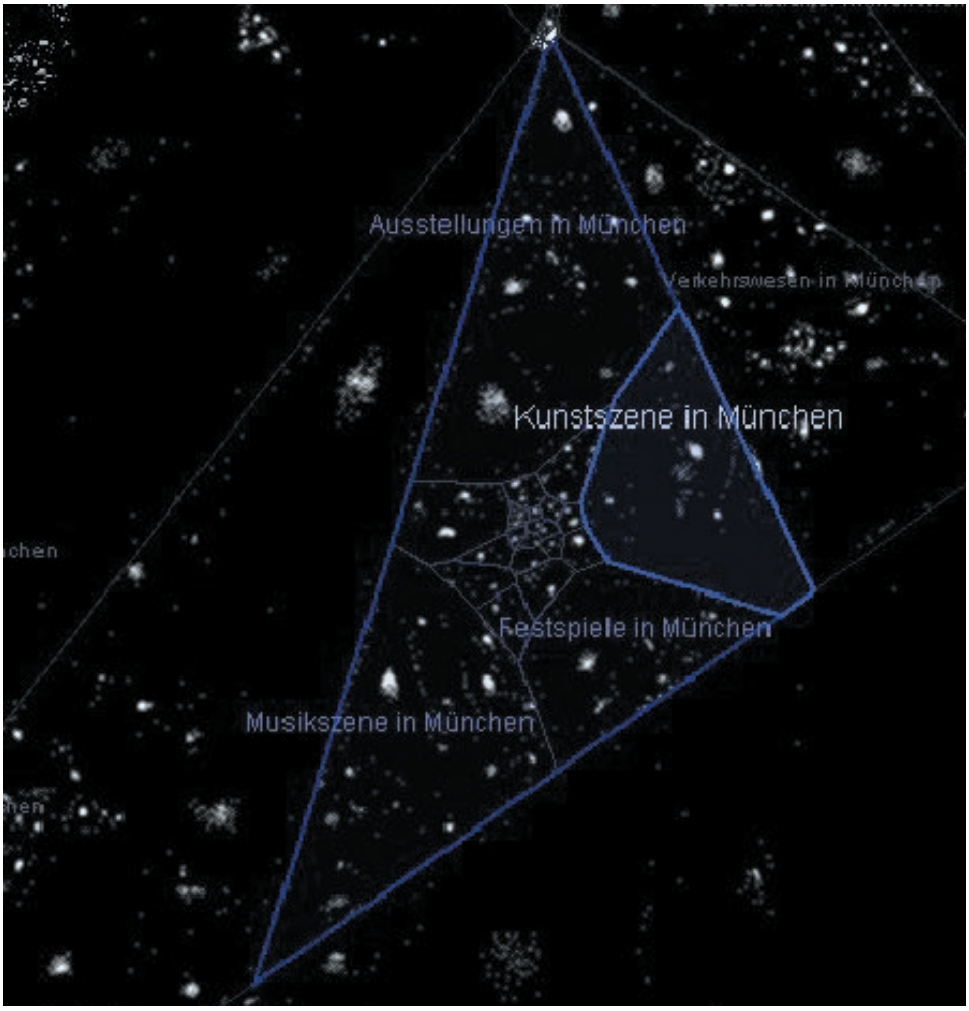

Figure 3 Clicking zooms in on and centres the collection 'Kultur in München' (Culture in Munich). The mouse is currently over the collection 'Kunstszene in München' (Arts Scene in Munich), highlighting it.

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:

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

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*:

$$force(O_i, O_j) = dist(O_i, O_j) - distHD(O_i, O_j) \quad (2)$$

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:

$$distHD = 1 - sim(O_i, O_j) \quad (3)$$

where the similarity lies in [0, 1].

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

$$force(O_i, O_j) = dist(O_i, O_j) - (1 - sim(O_i, O_j)) \quad (4)$$

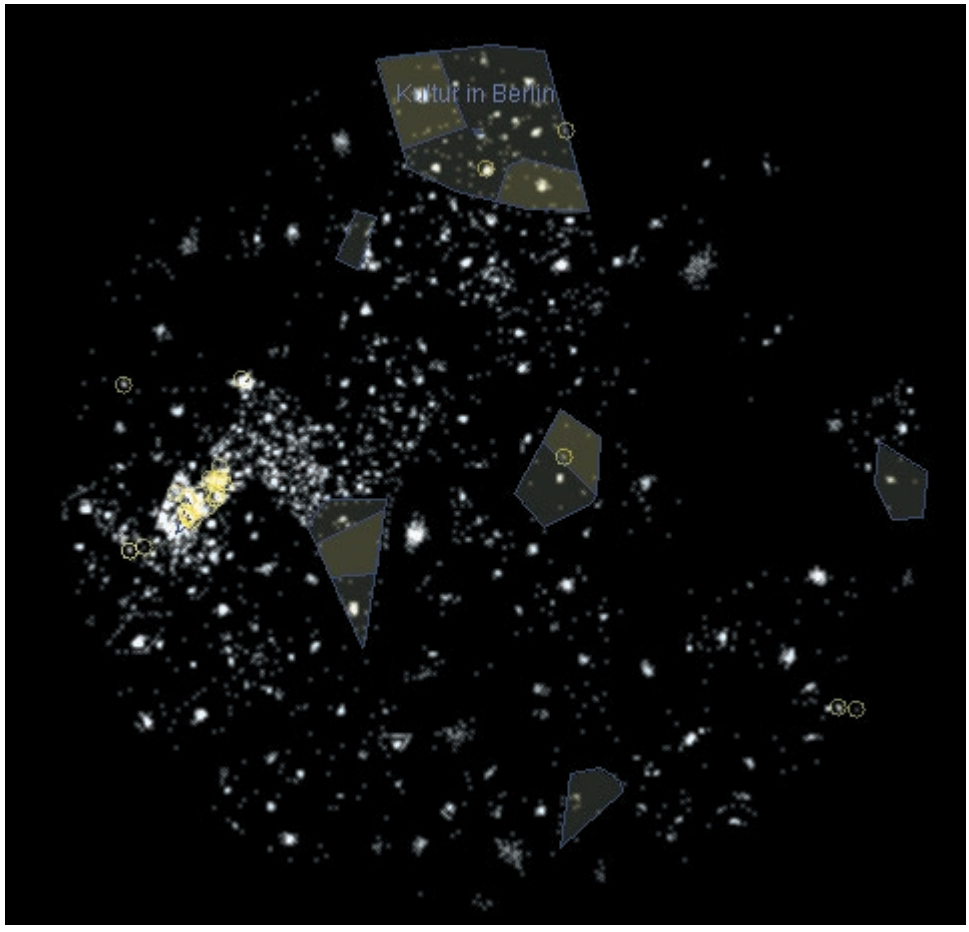

Figure 4 A query for the word 'kultur' (culture) displaying both matching collections and matching documents. Matching documents pulsate to further differentiate them.

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:

$$force(O_i, O_j) = sim(O_i, O_j)^d - \frac{w}{dist(O_i, O_j)^r} + grav \qquad (5)$$

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 Area partition). 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, and 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:

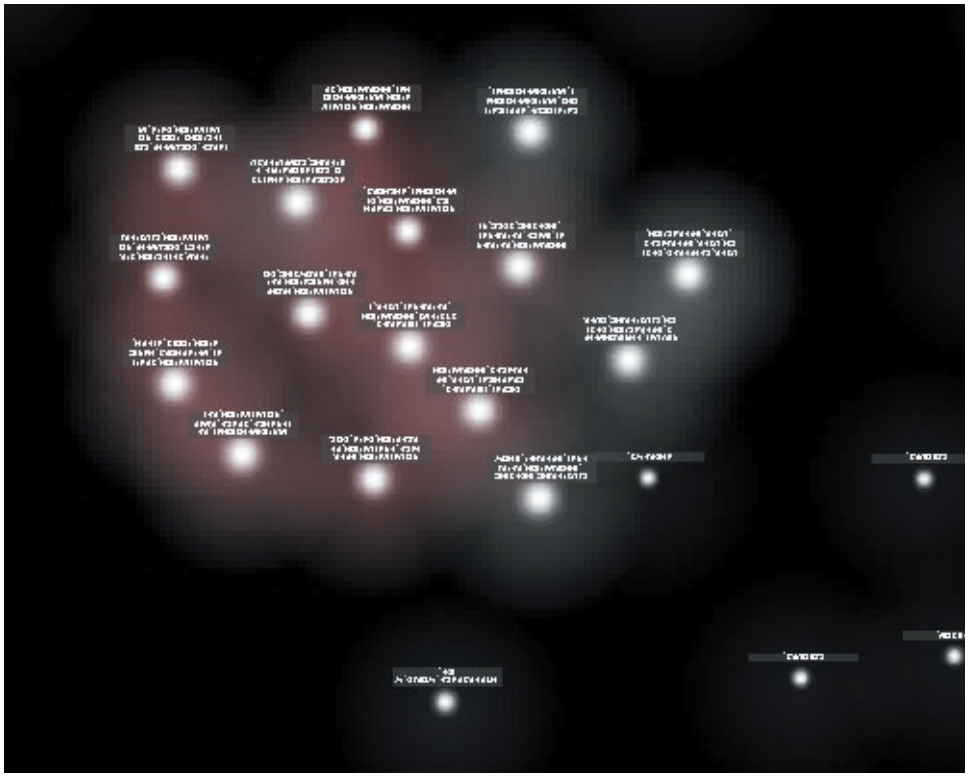

Figure 5 Stars displayed at high magnification with a halo representing their search result relevance value.

$$O_i.x = \frac{1}{|M|}\sum_{j \in M} force(O_i, O_j) * O_j.x + (1 - force(O_i, O_j)) * O_i.x \quad (6)$$

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[3] 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.

Geometric transformation

Given a layout of objects (subcollection centroids or documents) in a normalised space with co-ordinates 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.

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,[4] shown in Figure 6(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 InfoSky, the weight of each subcollection centroid is proportional to the total number of documents in that subcollection sub-tree.

For two points p and p_i shown in Figure 7, 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 7, 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, that it goes beyond p_i.

In Figure 6(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 8(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.

To avoid this behaviour, the force-directed placement algorithm was modified (as described in Similarity placement) 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 8(b) shows the same points and weights as Figure 8(a), but heavier points have been placed nearer the boundaries of the bounding polygon.

User testing

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 9. 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.

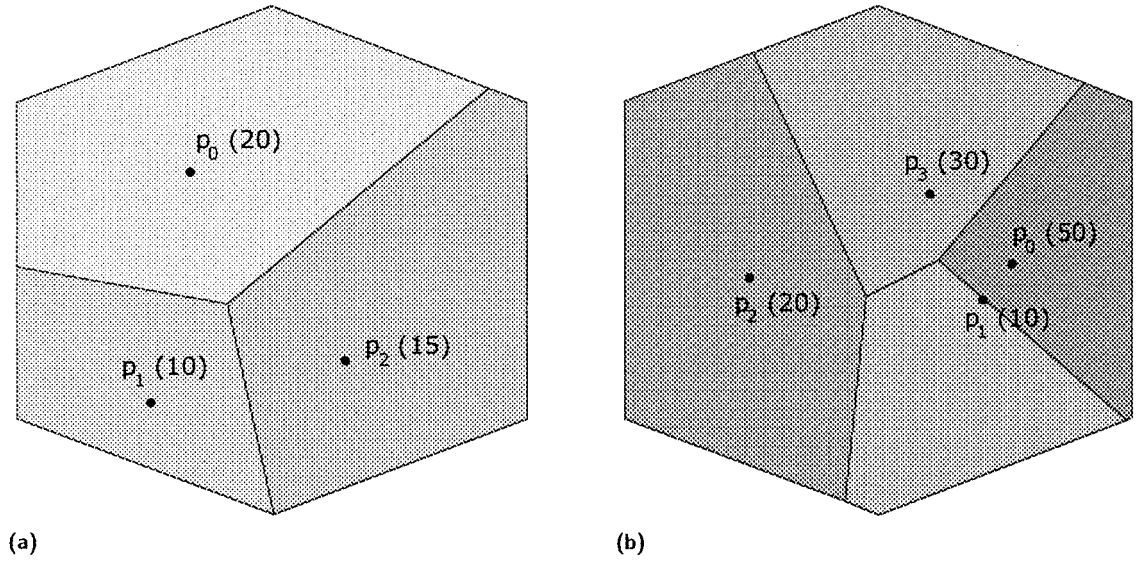

Figure 6 Additively weighted power Voronoi diagrams. (a) A standard additively weighted power Voronoi diagram. The distance between points is relatively large and weight differences are marginal. (b) If distances are very small, or weight differences are very large, the bisector between two points can overshoot, as is almost the case here with p_1. Scale factor f avoids this. The weight of each point is indicated in brackets.

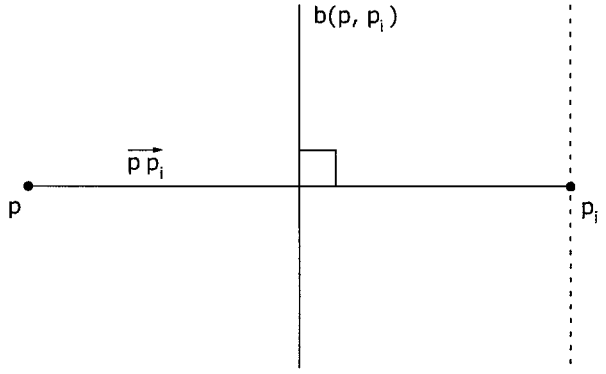

Figure 7 Geometric relationships in the additively weighted power diagram.

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 is shown in Figure 10. A pilot test with one user was run and some slight modifications were made. Table 1 shows the modified tasks as used in the main study in English translation, the study itself was done in German.

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 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 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.

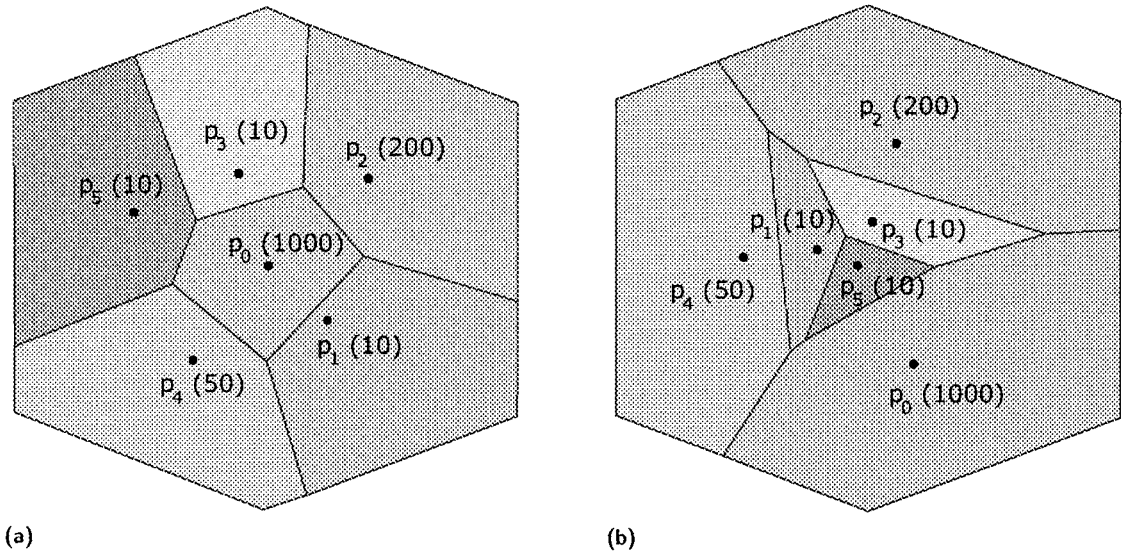

Figure 8 Ensuring that areas are related to weights. (a) The introduction of scale factor *f* can lead to solutions where areas are no longer related to weights. (b) Modifying the similarity placement algorithm so that heavier subcollection centroids tend towards the boundaries and lighter subcollection centroids tend towards the centre of the layout alleviates this problem.

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.

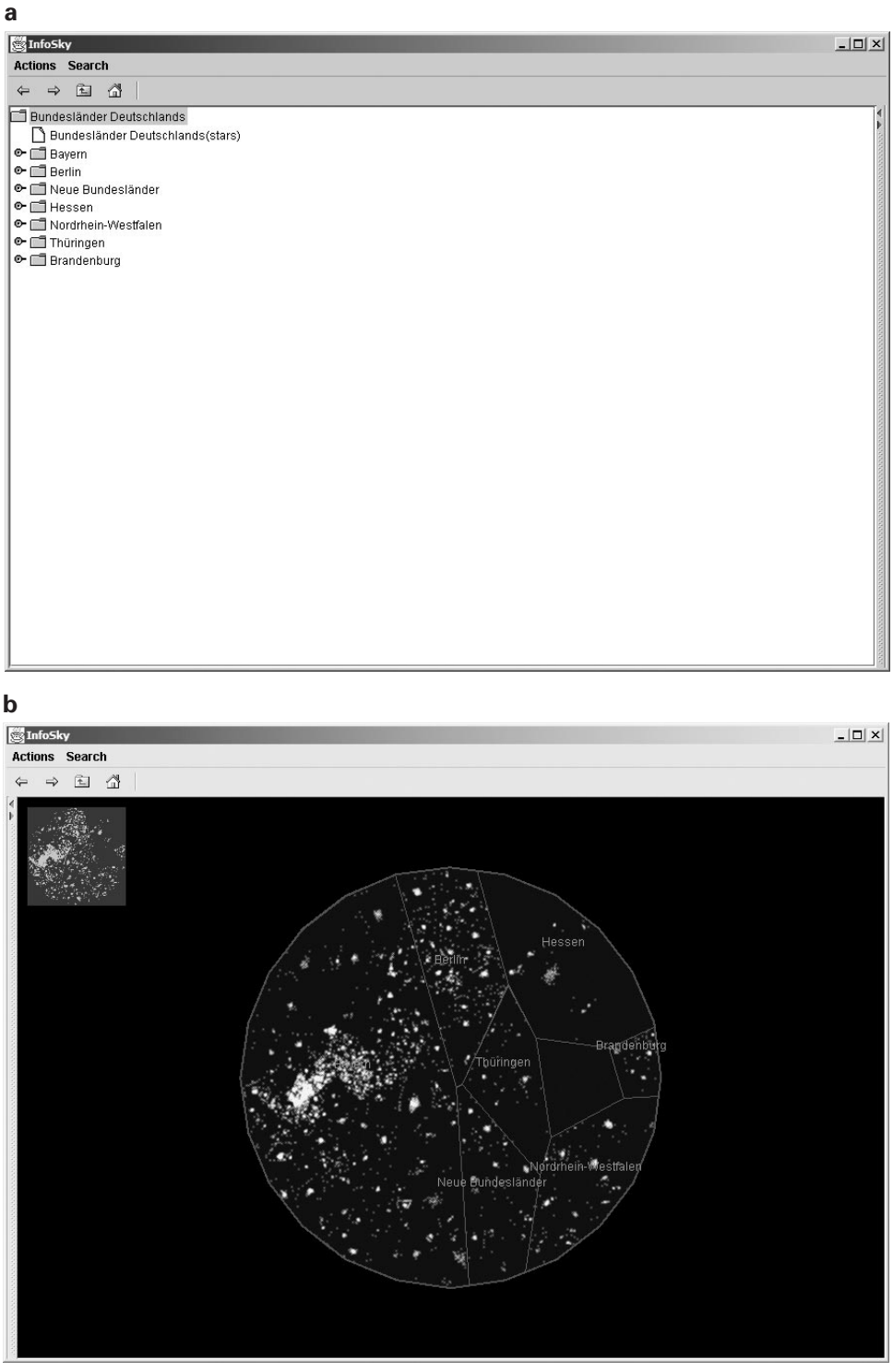

Figure 9 The two browsers as used in the study. (a) The tree view browser, test condition TV. (b) The telescope browser, test condition TS.

Related work

Published work on the visualisation of large document repositories tends to gravitate either toward information retrieval (IR) based approaches utilising inter-document similarity measures within flat repositories, or toward the visual exploration of hierarchically organised structures. Only recently have some first steps been taken towards integrating these two approaches.

Figure 10 The environment used for user testing.

Table 1 Sets A and B of five equivalent tasks. The test was carried out in German, the tasks are given here in English translation

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.

Approaches based on inter-document similarity

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

The Bead system[5] is a typical example of a thematic landscape. The information space is arranged based on inter-document similarity forming a 2.1D landscape. Users can navigate freely around the information landscape. Search result sets are displayed *in situ* within the landscape. In contrast to InfoSky, Bead operates on flat document repositories and does not take advantage of hierarchical structure.

Galaxy Of News[6] constructs and then visualises an associative relation network between related news articles. 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. Unlike InfoSky, the space is non-linear and changes as the user navigates, making it hard to maintain a sense of orientation.

SPIRE[7,8] operates on flat, unstructured document collections and provides two visualisations. SPIRE's Galaxies visualise documents as stars in a galaxy, where documents which are close in high dimensional space are also close in the two-dimensional galaxy view. This is similar to the approach taken by InfoSky to lay out documents at any particular level of the collection hierarchy. SPIRE does not exploit any inherent hierarchical structure. SPIRE's ThemeView (formerly Themescape)

builds on the galaxy view by aggregating frequently occurring topical keywords from neighbouring documents and displaying the main themes in a thematic landscape. Documents matching particular search criteria can be grouped and colour-coded in the galaxy display.

Earlier work at the IICM on VisIslands[9,10] used standard clustering techniques to cluster document sets returned in response to a search query on the fly. The clusters were used for more efficient similarity placement, by first placing cluster centroids, and then placing documents around them.

WEBSOM[11] and other systems employ self-organising maps to thematically organise and visualise very large document collections. However, the underlying neural networks have to undergo extensive training in order to achieve good results.

Approaches based on hierarchical structure

Systems focusing on the visualisation and navigation of large hierarchical structures tend to optimise the use of screen (pixel) real estate by either (or both) geometric transformations or zooming and panning interactions.

The Hyperbolic Browser[12] is a two-dimensional tree browser, which utilises hyperbolic geometry to always display the entire hierarchy on the display. The tree is laid out using hyperbolic axes (which are infinite) and then mapped to a two-dimensional unit disc for display. Areas in the centre of the disc are in focus and clearly visible, areas toward the edge of the disc become infinitely small and can no longer be seen. The H3 browser[13] makes even better use of screen space by using 3D, at the cost of some occlusion. However, neither of these systems make explicit use of document content and subcollection similarities.

Cone Trees[14] lay out hierarchies in three dimensions. Each node in the hierarchy is the apex of a cone. The root of the hierarchy is placed near the top of the three-dimensional display space and its children are evenly spaced along its base. The next layer of nodes is drawn below the first layer, recursively until the whole hierarchy is drawn. Nodes are usually given a label or name. However, since horizontal labels do not fit well in vertical cone trees, horizontal cone trees (also called cam trees) extending from left to right are often used. Cone trees suffer from problems of occlusion as hierarchies become broad and branches become hidden behind their siblings, interactivity has to be employed to rotate hidden branches. Again, the hierarchical structure alone determines the shape of the visualisation, inter-document or inter-collection similarities have no bearing.

The File System Navigator (FSN)[15,16] uses a landscape metaphor to lay out a file system in three dimensions. Directories are represented as rectangular pedestals, successive subdirectories spread out in ranks back towards the horizon. Lines connecting the pedestals show the structure of the hierarchy and are traversable. Individual files are represented by boxes arranged atop each pedestal, the height of a box indicates the size of a file, while its colour represents its age. FSN has an overview window which always shows the current location in the context of the entire structure. Searches can be performed and matches are displayed in context, with the possibility to step sequentially through them. The layout, however, is determined purely by the structure of the hierarchy.

CyberGeo Maps[17,18] 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 far away they are from the home page. To avoid collisions, a simple hash function based on directory names and random element distributes the stars in their

Table 2 Ordering of tasks and browsers for each test user

Test user	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 3 The timings in seconds for eight test users for each of five tasks in the TV and TS conditions

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

orbits. The metaphor is similar to that used in InfoSky and the visual display is similar, but the underlying layout is very different. In CyberGeo Maps levels of the hierarchy form concentric rings around the root, in InfoSky Voronoi diagrams are used recursively. In CyberGeo Maps the proximity of stars is not related to their similarity.

Integrated approaches

Information Pyramids[19] use a three-dimensional landscape to visualise a hierarchy. Full usage of the third dimension is made by visualising both the content and structural information in three dimensions. Children are arranged on top of their parents in a recursive fashion. The 3D Explorer application uses information pyramids to visualise a file system. Directories are represented by plateaus and files by boxes. Various metrics can be mapped to size and colour. The general impression is that of pyramids growing upwards as the hierarchy grows deeper. The entire hierarchy is always displayed. Elements deep in the hierarchy might be too small to be visible, but the user can freely navigate and smoothly zoom to reveal such sub-structures. The placement of plateaus and documents at each level can be influenced by similarities between them, for example similar subdirectories could be placed near one another. Whereas Information Pyramids uses recursive placement of rectangles at each level of the hierarchy, InfoSky uses recursive partition of polygons with Voronoi diagrams.

WebMap's InternetMap[20,21] 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. Each category is visualised as a multi-faceted shape, enclosing the sites within that category. Within a category, sites with similar content are geometrically close. However, there is no correspondence between the local view at each level and the global view. A topographical contour map representing a rating or value of some kind for each site is underlaid. 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.

Discussion and future work

The initial calculation of cluster geometry for the approximately 109,000 documents shown in Figure 1 takes about 4 hours in the current prototype on a standard desktop PC (Pentium III, 850 MHz, 256 MB, 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.[22] 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. It is also intended to provide interaction profiles on a per-user basis, supporting personalised browsing and search.

Concluding remarks

This paper presented the InfoSky, a first prototype system for the interactive visualisation and exploration of large, hierarchically structured, document repositories. 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, we hope that further improvements will make it comparable.

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

Acknowledgments

We would like to thank our colleagues at the Know-Center, Hyperwave, and Graz University of Technology for their feedback and suggestions. The Know-Center is a Compe-

tence Center funded within the Austrian K plus Competence Centers Program (www.kplus.at) under the auspices of the Austrian Ministry of Transport, Innovation and Technology.

References

1 Bederson B, Hollan J. Pad++: A zooming graphical interface for exploring alternative interface physics. In: *Proc. UIST'94*. (Marina del Rey, California), 1994; 17–26. ACM.
2 Bederson B. *Jazz,* 2002; http://www.cs.umd.edu/hcil/jazz/.
3 Chalmers M. A linear iteration time layout algorithm for visualising high-dimensional data. In: *Proc. Visualization'96* (San Francisco, California), 1996; 127–132. IEEE Computer Society. http://www.dcs.gla.ac.uk/~matthew/papers/vis96.pdf.
4 Okabe A, Boots B, Sugihara K, Nok Chiu S. *Spatial Tesselations: Concepts and Applications of Voronoi Diagrams* 2nd edn. Wiley: ????, 2000; ISBN 0471986356.
5 Chalmers M. Using a landscape metaphor to represent a corpus of documents. In: *Spatial Information Theory, Proc. COSIT'93* (Boston, Massachusetts), 1993 377–390; Springer LNCS 716. http://www.dcs.gla.ac.uk/~matthew/papers/ecsit93.pdf.
6 Rennison E. Galaxy of news: An approach to visualizing and understanding expansive news landscapes. In: *Proc. UIST'94* (Marina del Rey, California), 1994; 3–12; ACM. http://www. acm.org/pubs/citations/proceedings/uist/192426/p3-rennison/.
7 Thomas J, Cowley P, Kuchar O, Nowell L, Thomson J, Wong PC. Discovering knowledge through visual analysis. *Journal of Universal Computer Science*; **7(6)**: 517–529 2001; http://www.jucs.org/jucs_7_6/discovering_knowledge_through_visual.
8 Wise JA. The ecological approach to text visualization. *Journal of the American Society for Information Science*; **50(9)**: 814–835. 1999; http://www.vistg.net/hat/Wise_draft/Ch5.Wise.html.
9 Andrews K, Gütl C, Moser J, Sabol V, Lackner W. Search result visualisation with xfind. In: *Proc. UIDIS* 2001; (Zurich, Switzerland), IEEE Computer Society Press: 2001; 50–58.
10 Sabol V. Visualisation islands: Interactive visualisation and clustering of search result sets. Master's thesis, Graz University of Technology, Austria, 2001; ftp://ftp.iicm.edu/pub/theses/vsabol.pdf.
11 WEBSOM Websom - self-organizing maps for internet exploration. Helsinki University of Technology, 2000; http://websom.hut.fi/websom/.
12 Lamping J, Rao R, Pirolli P. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In: *Proc. CHI'95* (Denver, Colorado), 1995; 401–408. ACM. http://www.acm.org/sigchi/chi95/Electronic/documnts/papers/jl_bdy.htm.
13 Munzner T. H3: Laying out large directed graphs in 3d hyperbolic space. In: *Proc. IEEE InfoVis'97* (Phoenix, Arizona), 1997 210. IEEE Computer Society. http://graphics.stanford.edu/papers/h3/.
14 Robertson GG, Mackinlay JD, Card SK. Cone trees: Animated 3D visualizations of hierarchical information. In: *Proc. CHI'91* (New Orleans, Louisiana), 1991; 189–194. ACM.
15 Tesler JD, Strasnick SL. Fsn: The 3d file system navigator. Silicon Graphics, Inc., 1992; ftp://ftp.sgi.com/sgi/fsn.
16 Strasnick SL, Tesler JD. Method and apparatus for displaying data within a three-dimensional information landscape. US Patent 5528735 , Silicon Graphics, Inc., 1996; Filed 23rd March 1993, issued 18th June 1996.
17 Holmquist LE, Fagrell H, Busso R. 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.
18 Skog T, Holmquist LE. 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.
19 Andrews K, Wolte J, Pichler M. Information pyramids: A new approach to visualising large hierarchies. In: *IEEE Visualization '97, Late Breaking Hot Topics Proc, Phoenix, Arizona,* 1997; 49–52. ftp://ftp.iicm.edu/pub/papers/vis97.pdf.
20 WebMap. WebMap. 2002; http://www.webmap.com/.
21 Iron M, Neustedt R, Ranen O. Method of graphically presenting network infor-mation. US Patent Application 2001; 0035885A1 , WebMap, November 2001. Filed 20th March 2001.
22 Hyperwave. Hyperwave, 2002; http://www.hyperwave.com/.