Santokh Singh, BSc

Dynamic Geometry Information Landscape Visualization using WebGL

MASTER'S THESIS

to achieve the university degree of

Master of Science

Master's degree programme: Computer Science

submitted to

Graz University of Technology

Supervisor

Sabol, Vedran, Dipl.-Ing. Dr.techn.

Interactive Systems and Data Science (ISDS)

Graz, September 2019

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis dissertation.

Date	Signature

Kürzfassung

Eine Informationslandschaft ist eine visuelle Darstellung für die Analyse großer Textrepositorien, die Zusammenhänge in multidimensionalen Daten durch räumliche Nähe in der Visualisierung darstellt. Die verwendete Landschaftsmetapher stellt die Themenverteilung im Datensatz als natürliche Umgebung in der Form einer geografischen Karte dar, wodurch diese intuitiv von der Öffentlichkeit verstanden werden kann. Diese Masterarbeit stellt eine web-basierte Anwendung vor, die eine interaktive Informationslandschaft Visualisierung unter Verwendung von WebGL- und HTML5-Technologien darstellt. Die implementierte visuelle Anwendung erlaubt dem Anwender, sich einen Überblick über die Daten zu verschaffen, aktuelle Zusammenhänge zu entdecken und die Verteilung der Themen im Datensatz zu verstehen. Darüber hinaus ermöglicht die Anwendung dem Benutzer, die Informationslandschaft interaktiv aus verschiedenen Perspektiven zu betrachten, und bietet mehrere Möglichkeiten, relevante Daten für eine detailliertere Analyse auszuwählen. Ein wesentliches Merkmal der Arbeit ist das GPU-beschleunigte, animierte Morphing der Landschaftsgeometrie, die den Benutzer dabei unterstützt, dynamische Änderungen in den Daten zu verfolgen und zu verstehen. Das bekannte Change Blindness Problem, das den Benutzer daran hindert, die Änderungen zu erkennen und zu verstehen, wird durch verschiedene aus der Literatur bekannte Techniken adressiert. Darüber hinaus werden neue Methoden, wie das Morphing der Landscape-Geometrie oder die Darstellung von Dokument-Traces, vorgeschlagen.

Keywords: Informationslandschaft, Textdokumente, webbasierte Anwendung, WebGL, GPU

Abstract

An information landscape is a visual representation for topical analysis of large text repositories, which conveys relatedness in multidimensional data through spatial proximity in the visualization. The employed landscape metaphor represents the topical distribution in the data set as a natural environment shown in the form of a geographic map; therefore, it is more likely to be intuitively understood by the public. This Master's Thesis presents a web-based application for rendering an interactive information landscape visualization using WebGL and HTML5 technologies. The implemented visual application allows the user to obtain an overview, discover topical relationships and understand the distribution of topics in the data set. Furthermore, the application enables the user to interactively explore the information landscape from different perspectives, and provides several ways to select data of interest for more detailed analysis. A major feature of this work is the GPU-accelerated, smoothly animated morphing of the landscape geometry to help the user follow and understand dynamic changes in the data. Addressing the well-known change blindness problem, which prevents the user to notice the changes, is addressed by various techniques known from the literature, as well as by introducing novel methods, such as the morphing information landscape or displaying document trails.

Keywords: information landscape, text documents, web application, WebGL, GPU

Acknowledgment

First I would like to thank my supervisor Vedran Sabol for his guidance and great support as well as for his great ideas and time, which helped me a lot to implement the information landscape way it is.

I also thankful to my two brothers Sucha and Devraj for believing in me and supporting me throughout my studies. Furthermore, I would like to thank one of my friends Stefan Schäfback, who was always there to provide me the technical support.

I would like to dedicate my thesis to my mother Dawarki Devi and to my father Pal. This thesis would not have been possible without their support and their belief in me.

Graz, September 17, 2019 Santokh Singh

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Large collections of text documents are typically unstructured and contain complex and multidimensional information. Additionally, the documents of a collection continuously go through dynamic changes such as the adding, removing and modification of respective documents. The major challenge with such collections of text documents is to make the dataset logically understandable and explorable for a user, for instance allowing the user to get an overview of the dataset by showing the relatedness and the topical distribution of the documents, letting the user explore the dataset of interest, and enabling the user to grasp the changes in datasets in an understandable way.

The information landscape is a visual metaphor that can address the mentioned challenges for complex, multidimensional and dynamically changing datasets such as text repositories. The approach is based on the well-known information landscape metaphor, which is intuitively understood by the public because it resembles the natural environment, i.e. the interpretation of natural landforms is part of people's biological makeup.

Information landscapes are commonly used for the visualization of large amount of data and are capable of conveying both topical relatedness and dynamic change. The spatial proximity in the visualization space shows the topical relatedness in the form of hills that represent groups or visual clusters of topically similar documents. Hills are labelled with the most significant keywords from the underlying documents.

When a collection of text documents changes over time, for example when documents are modified, new documents are added or old documents are removed, the overall topical structure and consequently the landscape topography change as well. To help the user to follow and understand the changes, the geometry of two different information landscapes are morphed using a smooth animation. In this process of morphing, the hills and islands grow or shrink to indicate the growth or decline of topic prevalence.Hills can also move closer or futher apart to convey converging or diverging topical clusters.

The aim of this master's thesis is to develop an interactive and scalable dynamic geometry information landscape using WebGL and HTM5 technologies. WebGL is a fully standardized JavaScript API based on OpenGL ES, which provides access to hardware accelerated 3D rendering of interactive 3D and 2D graphics within any compatible web browser without the use of any plug-ins.

1.1 Focus Points

The following aspects have been the focus while developing the information landscape:

- **GPU accelerated information landscape visualization**
  The creation of a information landscape which matches the realistic natural environment in order to make the landscape more intuitively understandable can be realized by using the height-based texture mapping method, where a water texture defines the lowest points, with sand, grass, rocks and snow textures as the height increases. GPUs (i.e. graphics processing units) are more efficient at manipulating computer graphics due to the highly parallel structure of their computations. The main focus is to utilize this aspect, first off to manipulate the geometry of the landscape using a shading language (i.e. vertex shader) to create a smooth 3D accelerated morphing animation of landscape geometry, and second to apply the height-based texture mapping method to each pixel of landscape using the shading language (i.e. fragment shader).
- **UI design**
  The goal of this section is to develop an interactive design for the information landscape that enables the user (i) to view the landscape from different perspectives, (ii) to select the documents individually, based on content as well as their positions in the landscape, (iii) to perform geometry morphing-related interactions, (iv) to explore, filter or navigate the documents and (v) to adjust the information landscape configuration, such as adjusting the brightness and opacity of the information landscape, changing the colour and size of the brush-based selection tool, changing the design of the information landscape, setting the timespan of animation of morphing and setting the type of height scaling (linear, square root, squared and logarithmic).

1.2 Structure of the Paper

The structure of this master's thesis is organized as follows. Chapter 2 describes the related scientific work concerning the most important aspects of the information landscape, including a general theoretical discussion of information landscapes as well as a discussion of algorithms for computing the landscape layout. Furthermore, the second chapter also gives an overview of landscape visualization on different systems and in different web browsers using 2D/3D rendering. Chapter 3 introduces geometry morphing of the information landscape in five separate sections. The first section introduces the pipeline for computing the landscape layout for textual data. The second section explains the general concept of morphing. The third section describes the visual design of a landscape, while the interactive design is covered in the next section. Chapter 4 illustrates the implementation of an information landscape. Chapter 5 provides the major case studies of the landscape. Chapter 6 presents the conclusions of the master's thesis and discusses possible future work in this direction.

Chapter 2

Related Work

This chapter gives an overview of the famous scientific work most relevant to the thesis. The chapter begins with a general theoretical discussion of the information landscape, which includes laying out the problems that the information landscape attempts to solve, describing the metaphor and morphing of the information landscape as well as discussing their advantages and drawbacks. Furthermore, this chapter covers various algorithms for computing the landscape such as principal component analysis (PCA), multi-dimensional scaling (MDS), force directed placement (FDP). Finally, the last section of this chapter discusses the visualization of information landscape on different systems.

2.1 Information Landscapes in General – Theoretical Discussion

2.1.1 Spatialization

Use of spatialized user interfaces has started to rise dramatically due to the arrival of virtual worlds and increasing computation power. Therefore, it is reasonable to discuss the meaning of spatialization in space in the context of visualization, user interfaces and interactions.

Spatialization [Kuhn W. & Blumenthal B., 1996] transforms physical space into spatial metaphors of abstract domains into user interfaces. The spatialization consists of cognitive and engineering aspects. All of these aspects arise from the pervasive role of space in human activities, including subconscious activities like perception, manipulation and motion in space, which involves little cognitive load and at same time offers great functionality.

From a cognitive domain point of view, space plays an important role for human reasoning, language and action [Jackendoff R. S., 1985]. Spatial metaphors form our understanding of abstract domains and this characteristic can be used to create a wide variety of user interface designs.

The engineering aspects of spatialization represent space and spatial operations in order to produce effective user interfaces. Appropriate means and tools are thus required to make the spatial metaphors visually perceivable.

2.1.2 Metaphor

The notion of a metaphor is often used in the context of information landscapes and spatialization.

According to the definition of [Kuhn W. & Blumenthal B., 1996] the notion metaphor has the following meaning: "The metaphors allow users to understand something in terms of something else, without requiring the users to believe that the two things are the same”. Vedran Sabol [Sablol V., 2012] defines the notion of metaphor as follows: "The metaphors visualize abstract information using a well-known equivalent for information which does have a natural representation in the real world. Metaphors are usually more intuitive than formalisms, because the user can infer the meaning through analogy".

2.1.3 Information Landscape

Information landscape employs a geographic landscape metaphor for relatedness analysis of large, complex, multidimensional collections of data, such as text documents. The relatedness of documents is conveyed through spatial proximity in the visualization [Krishnan M. et al., 2007].

Document relatedness is based on calculating the similarity of documents to each other and projecting their relationship into 2D or 3D space in such a way that similar items are positioned close together, while dissimilar items are placed far apart. Hills emerge where the density of related documents is high, indicating a topical cluster. Clusters that are topically similar are placed spatially close to each other, whereas dissimilar clusters are separated by larger empty spaces represented as oceans or valleys. Hills are labelled with the most significant keywords from the underlying documents, allowing the user to identify the area of interest and eliminate the outliers. The height of a hill indicates the number of documents [Sabol V. et al., 2010].

The visual representation of a landscape can be used to get an overall overview of an unfamiliar dataset, to understand the relationships as well as topical distribution of the dataset, to find similar documents, and identify outliers.

As a visual component, the information landscape must enable the user to interactively view the information landscape from different perspectives. It provides several ways to select the data of interest; typical interactions for viewing data from different perspectives and for item selection are described here:

- Zoom: The zoom operation increases the sense of proximity and displays the focused area in more detail, while in contrast zooming out decreases the level of proximity and displays the areas as smaller and possibly less detailed. The practical approach for this interaction is to add the ability to zoom in and out with the mouse scroll wheel.
- Panning: The panning operation is the action of dragging the whole area by keeping the zoom level the same in order to move the point of view and reveal the information which was previously outside the displayed screen area.
- Selection: This operation lets the user choose one or multiple displayed items so that further operation can be performed on the selected data.

2.1.4 Morphing Landscape

The previous section discussed relatedness analysis in large, complex, multidimensional datasets using the concept of the information landscape. However, the fact is that documents of a collection often go through dynamic changes, such as the adding, removing or modification of documents in a collection. The challenging part is getting the user to understand the changes using an information landscape.

The solution to this challenge is morphing, whereby morphing is a technique that constructs intermediate steps between two states by applying some data from each state to create a sequence of images, where the consequential state become less like first state and more like second state. When animated, the sequences appear to be in motion, such that first state is changed into second state thorough a seamless transition.

The fact is that when a collection of text documents changes over time, e.g. documents are modified, new documents are added or old documents are removed, the overall topical structure and consequently the landscape topography change as well. To help the user to follow and to understand the changes, the information landscape is typically morphed into a new information landscape involving dynamic changes using smooth animation sequence. In the process of morphing, the rising hills or clusters indicate the emergence of new topics while the shrinking of hills indicates the fading of topic prevalence. The movements of hills towards or away from each other express converging or diverging topical clusters [Sabol V. et al., 2010].

Morphing the landscape allows the user to discover the emergence of new topics and fading of others, while being able to understand the converging and diverging topical clusters. However, the user can run into change blindness phenomena (see section 2.1.5 below), which occurs when the user does not notice or remember major changes in a visual stimulus.

2.1.5 Change Blindness

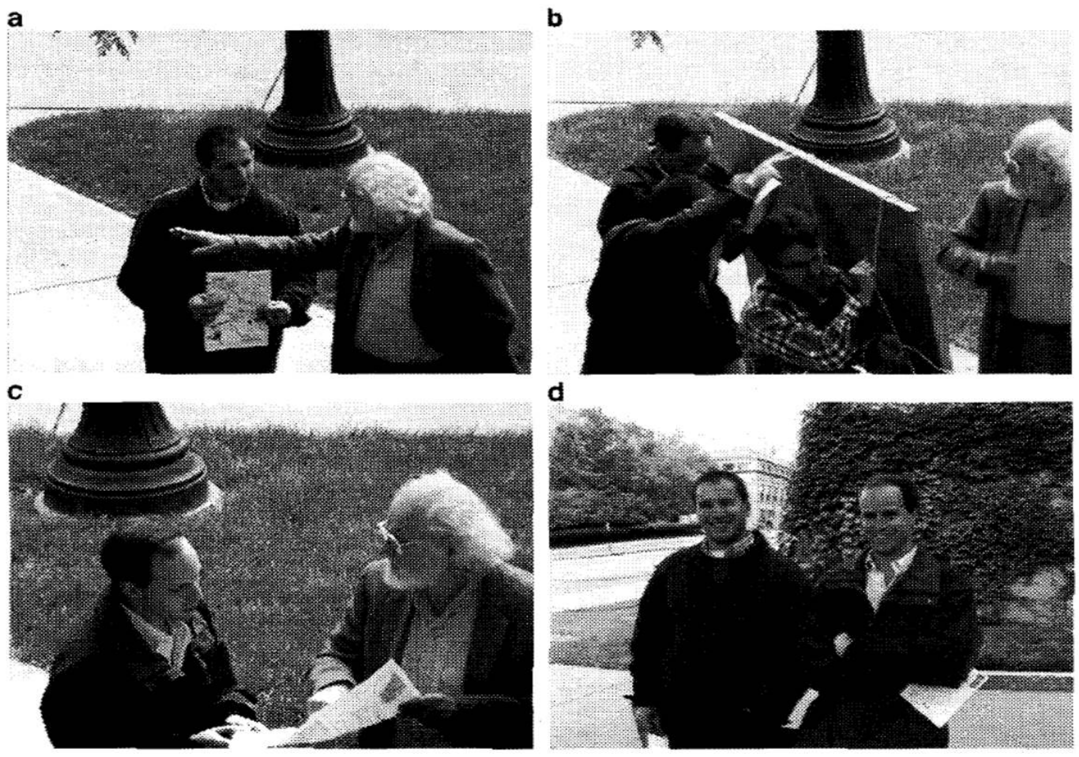


Figure 2.1: Frames from a video demonstrating the change blindness phenomena in the real world (taken from [Simons D. J. & Levin D. T., 1998])

Change blindness is a phenomenon of visual perception that occurs when the observer does not notice the changes in in a visual stimulus, with which the observer is actively engaged.([Simons D. J. & Levin D. T., 1997], ([Daniel J. Simons R. A. R., 2005])).

Simons and Levin [Simons D. J. & Levin D. T., 1998] carried out an An experiment to demonstrate change blindness phenomena in the real world. First, an experimenter started to have a conversation with a pedestrian. The initial experimenter was then replaced by a different experimenter during a brief interruption. Only half of the subjects realized that their conversational partner had changed. Figure 2.1 depicts the experiment as a sequence of events.

Users of the SPIRE ThemeView Time Slicer visualization, which also employs a geographic landscape metaphor, experienced the change blindness problem. They encountered change blindness when they were using the Time Slicer and found themselves unable to identify what had changed from one time period, or slice, to the next, or they were unable to remember what was different in the previous slice [Nowell L. et al., 2001]. The problem is illustrated in Figure 2.2, which shows the data from three different days. The basic problem was that for recognizing the changes from one slice to the next, the user had to memorize all the time slices.

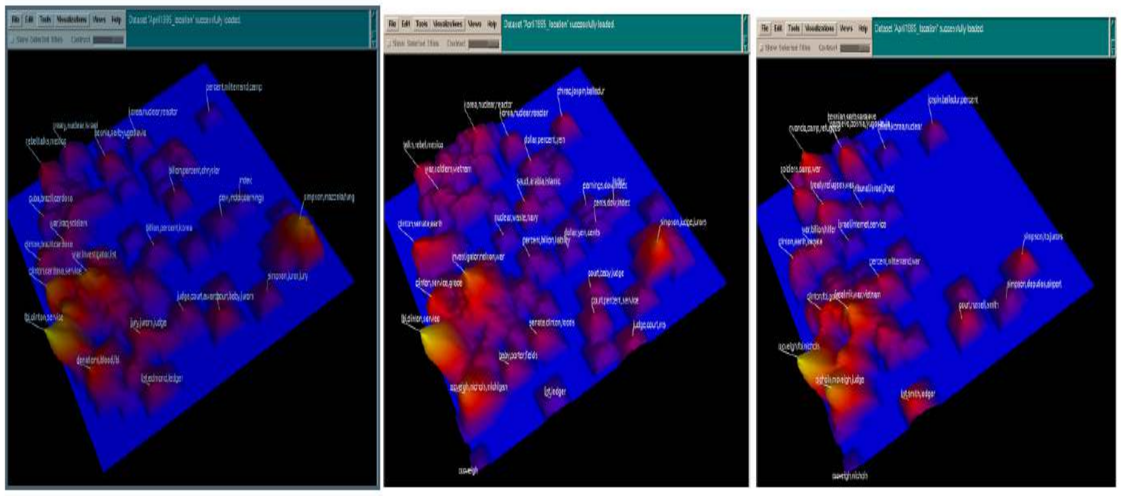


Figure 2.2: SPIRE ThemeView Time Slicer (taken from [Nowell L. et al., 2001]

Nowell proposed two criteria to solve the change blindness problem:

- The user should be able to see where the new landscape contours will be while the old contours are still visible

- At any point during the change, the user should be able to determine which landscape contours are emerging and which are vanishing

Furthermore, Nowell investigated three techniques for drawing attention to changes from one time slice to another:

- Morphing
- Cross fading
- Using a wireframe in combination with changes in colour and translucency.

The investigated technique of morphing had the following finding: Morphing did indeed make the areas of change noticeable, however it did not help the user to remember the changes because it was not possible for the user to tell what came before and what is going to appear next. Furthermore, the user could not determine which characteristics came from which slice (images).

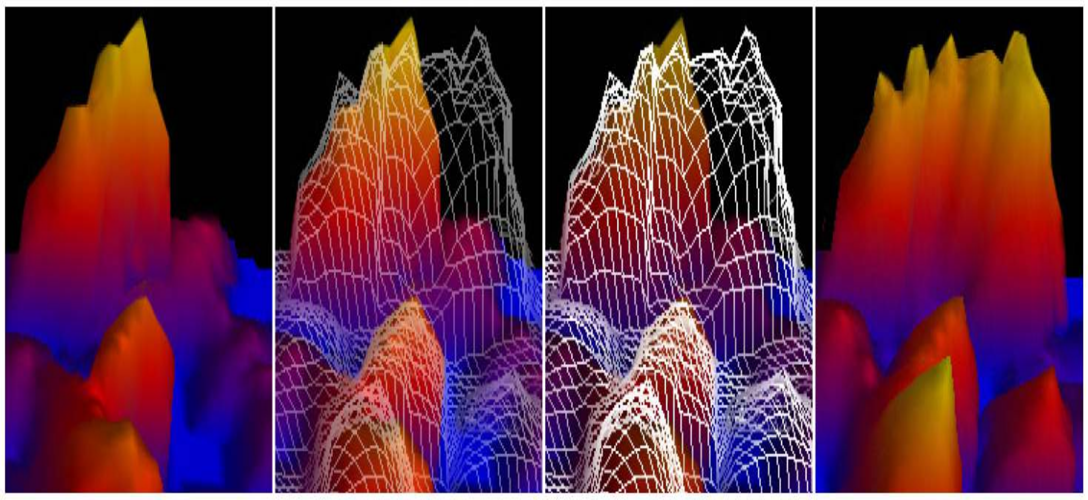


Figure 2.3: SPIRE moving from one time slice to another with a wireframe and variable translucency (taken from [Nowell L. et al., 2001]

Nowell recommends the third solution, i.e. ‘using a wireframe in combination with changes in colour and translucency’, shown in Figure 2.3 the authors explain the solution as follows: “The wireframe appears as soon as the transition from one image begins. At the same time, we begin reducing the opacity and colour saturation of the vanishing contours. The emerging contours, shown by the wireframe, begin to fill with translucent colour that gradually becomes brighter and less translucent. At the same time, the vanishing contours become progressively dimmer and more translucent. When the emerging image reaches opacity, the wireframe vanishes and the old image has completely faded away".

The advantage of the recommended technique is that the attention is drawn to the areas of changes by showing the vanishing (old) and emerging (new) changes in the form of images or motions at all times. Furthermore, by allowing the user having access to visual difference, the user can recognize what has changed.

2.2 Computation of the Landscape Layout for Text Data

An information landscape layout requires positioning of documents in 2D or 3D space in such a way that similar documents are positioned closer together, while dissimilar documents are placed further apart. The real challenge is that the datasets are complex and multidimensional. The computation of the information landscape layout is a process of transforming the complex and multidimensional dataset into 2D or 3D space while preserving the original relationships as much as possible, by positioning similar documents close together and dissimilar ones far apart.

The transforming process is related to the idea of ordination, which in multipolarity analysis has the following definition: Ordination is a technique that corresponds to data clustering and is mainly used in exploratory data analysis. The objects of ordination orders are characterized by values on multiple variables (i.e. multivariate objects) so that similar objects are close to each other and dissimilar objects are far apart from each other [Davis J. C., 2002].

The ordination algorithms transform the data in the multidimensional space to a space of fewer dimensions, while still retaining as much information as possible.

This chapter covers two major ordination algorithms: the principal components analysis (PCA) and metric multidimensional scaling (MDS). Furthermore, this chapter introduces an iterative method known as a force directed placement algorithm (FDP), which is commonly used for computing information landscape layouts.

2.2.1 Principal Components Analysis

Principal component analysis or PCA [Jolliffe I., 2002] is a linear transform the data from multidimensional space to a new coordinate system in which the first coordinate (first principal component) holds the greatest possible variance in data, and the succeeding coordinates, while being uncorrelated and orthogonal to the preceding coordinates, have the largest possible variance.

The directions of principal components are determined by computing the eigenvector and eigenvalues of the covariance matrix and sorting them according to decreasing eigenvalue. The magnitude of the eigenvalues indicates the variance of the data along the eigenvector directions.

PCA is mostly used as a tool in exploratory data analysis, which summarizes the main characteristics of transformed data to fewer dimensions using visual methods. One of the major drawbacks is that the principal components are orthogonal, which mean that they overlap in space and are positioned at 90 degrees to each other. However, the assumption that informative dimensions are always orthogonal may not hold. Therefore, an alternative technique called independent component analysis (ICA) can solve this problem, because the vectors in ICA do not have to be orthogonal to each other. Figure 2.4 shows the major drawback of principal component analysis in comparison to independent component analysis.

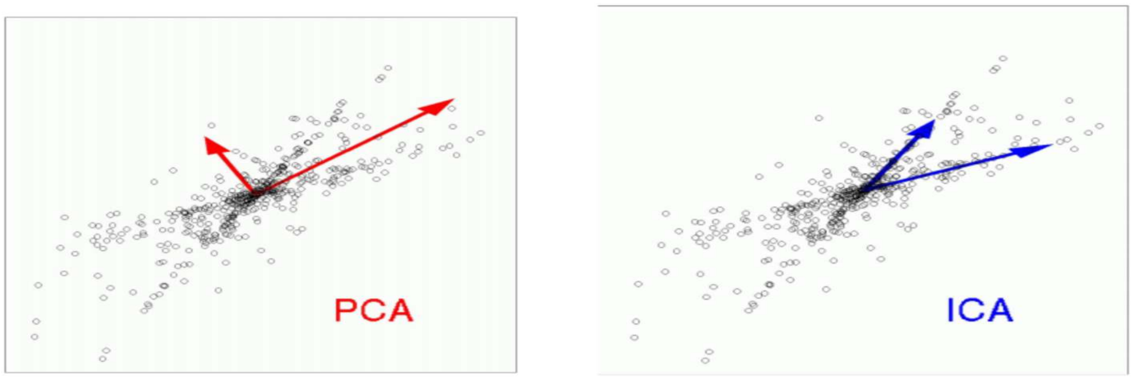


Figure 2.4: Drawback of principal components analysis in comparison to independent component analysis (Taken from [BLOHM D. G.,]

2.2.2 Metric Multidimensional Scaling

Multidimensional scaling (MDS) is a popular set of ordination techniques that is often used in exploratory data analysis for representing the data visually based on its dissimilarity information. The multidimensional scaling algorithm requires a pairwise-distance of a matrix of objects in a set as well as a chosen number of target space dimensions (n) as inputs to find the new representation of dataset, into an n-dimension space such that given pairwise distances between objects are preserved as well as possible. The pairwise distance of a matrix is calculated by applying similarity coefficients in the original high-dimensional space, such as Jaccard distance, cosine similarity or distance measures, such as Euclidean distance.

One of the multidimensional scaling methods is the classical method of multidimensional scaling (CMDS), which was introduced by Torgerson [Torgerson W. S., 1952]. The method is based on linear algebra arithmetic and transforms the distance matrix into a cross-product matrix to yield the eigendecomposition.

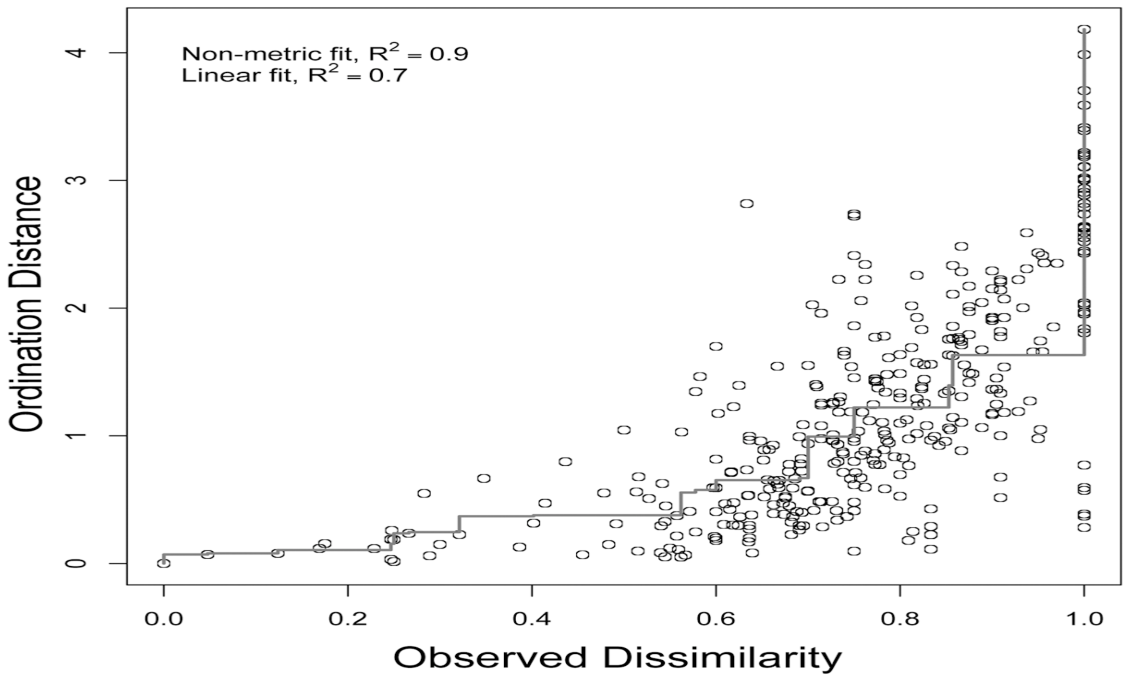


Figure 2.5: Shepard diagram showing relationships between NMDS ordination distance and original observed distance (taken from [Fraga R. et al., 2014])

The non-metric multidimensional scaling (NMDS) [Kruskal J., 1964] is a

Stress	Assessment of fit
Very Bad	0.20
Poor	0.10
Fair	0.05
Good	0.02
excellent	0.00

Table 2.1: Krusak guidelines for interpretation of the stress value

nonlinear method which looks for the best fit between the origin proximity of two objects and their pairwise distance in a low dimensional space based on a statistical model of least-squares analysis. NMDS constructs an initial configuration of objects in a chosen number of dimensions. The initial configuration is either selected randomly, or it is based on another ordination. In his work, Shepard ([Shepard R. N., 962a], [Shepard R. N., 962b]) showed a monotone relationship between the experimental dissimilarities or similarities and the distances in the configuration. Figure 2.5 shows the representation of a monotone regression by displaying the relationship between NMDS ordination distance and the original observed distance. Inspired by Shepard's procedure, Kruskal introduced the quality of the fit of the regression, called stress, which measures how well that configuration matches the data, whereby the lowest stress implies the best matches. The aim is therefore to find the solution with the lowest possible stress configuration. An iterative procedure is applied until an acceptable minimum stress value is found.

Kruskal provided some guidelines to interpret the stress value with respect to the quality of the fit of the solution, which are shown in the Table 2.1.

One of the drawbacks of MDS is that it is slow, particularly for large datasets. Another drawback is that MDS is a numerical optimization technique, so it can fail to find the true best solution because it can become stuck on local minima, i.e. solutions that are not the best solution but that are better than all nearby solutions.

2.2.3 Force Directed Placement (FDP)

The force directed placement algorithm (FDP) is a popular iterative method, which is used for creating pleasing graphic layouts based on the spring model as a physical system of rings and springs, in which the nodes are represented by steel rings and the edges are springs between them. The basic idea is to place vertices in some initial state and let the spring forces on the rings move the system to a minimal energy state [Eades P., 1984].

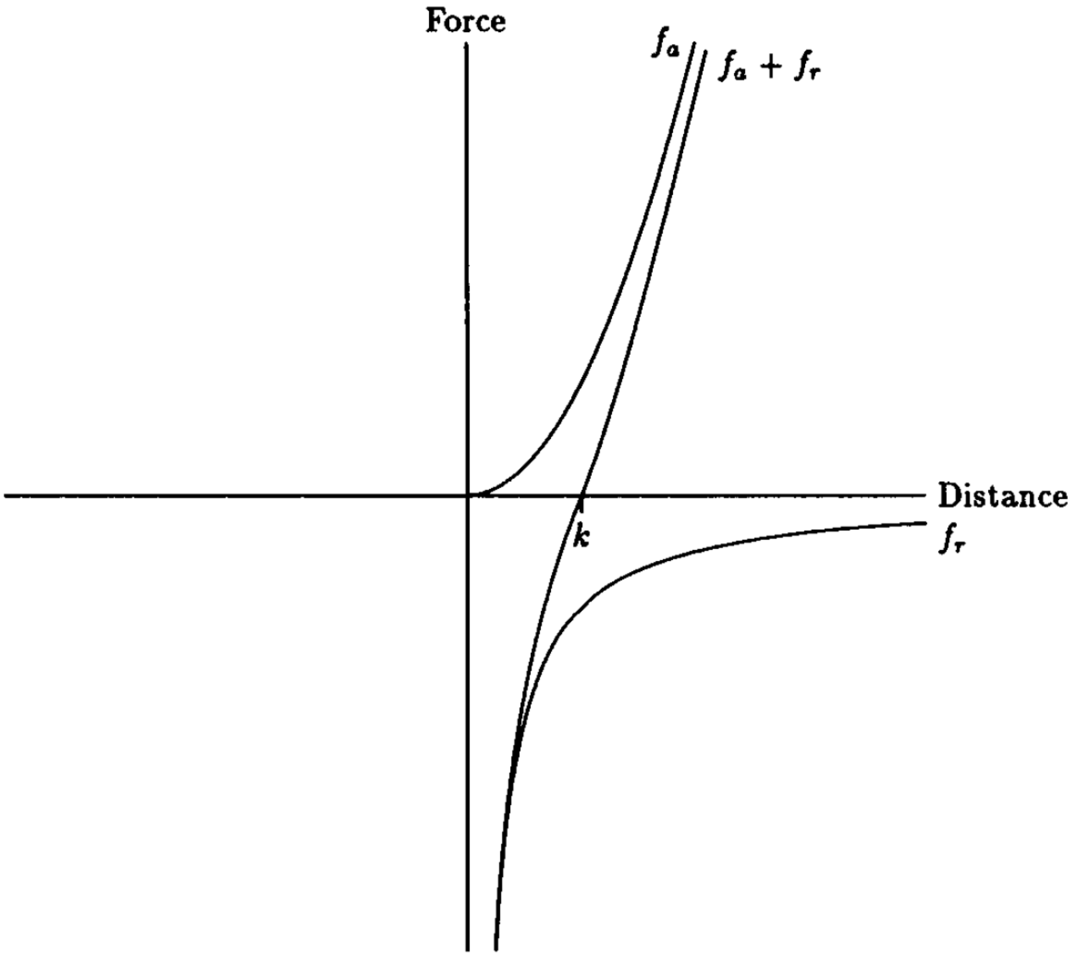


Figure 2.6: Illustration of attractive and repulsive forces and their sum versus distance (taken from [Fruchterman T. M. J. & Reingold E. M., 1991])

According to [Fruchterman T. M. J. & Reingold E. M., 1991] in their formulation of the algorithm, the vertices behave as atomic particles or celestial bodies, exerting attractive and repulsive forces on one another; the forces induce movement. The attractive forces are calculated only between neighbours while the repulsive forces are calculated between every pair, whereby the sum of the forces determines which direction a node should move. Figure 2.6 illustrates these forces and their sum versus distance. The nodes are moved iteratively until the nodes stop moving and the system reaches its minimal energy state.

One of the advantages of FDP is the projection of multidimensional relationships into 2D or 3D. Another advantage of FDP is that it is inherently incremental on a previously computed layout, which means that any modification of the dataset can be smoothly assimilated into the old layout without disrupting a whole landscape representation that the user has already familiarized themselves with. Also, incremental computing requires only a fraction of the full computation effort.

One of the drawbacks is that FDP is sensitive to becoming stuck at local minima and having high computational complexity, in particular when the involved dataset is too large. The main issue with the FDP algorithm is that it does not scale very well and it requires quadric time for repulsive forces.

Optimization of an algorithm can be achieved by applying a grid-variant. The algorithm divides the space into a grid of cells, and at each iteration each vertex is placed in its grid cell and repulsive forces are computed only for the nodes in the neighbouring cells.

2.3 Overview of Information Landscape Systems

2.3.1 Bead

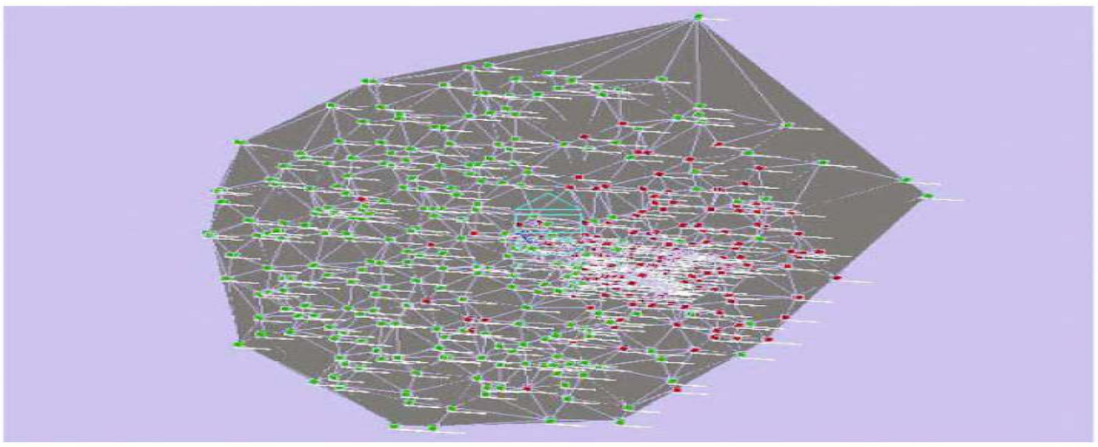


Figure 2.7: A view from far above an ‘island’ of documents constructed by Bead (taken from [Chalmers M., 1995])

The result of the initial Bead system is presented in Figure 2.7. The metric was based on co-concurrency of words and on a simulated annealing technique. In an iterative process, similar documents were pulled towards each other while dissimilar documents that were close together were pushed away from each other ([Chalmers M. & Chitson P., 1992], [Chalmers M., 1995]).

Initially, the work on the Bead system produced layouts of sets of documents using the metric of similarity or the metric of distances. The 3D point cloud on which Bead worked was remodelled to create a corpus of documents in the form of a landscape. The documents were represented by graphical objects and were placed in the space; the objects were meshed together with a polygon to make an island. Figure 2.8 illustrates the landscape of a Bead system.

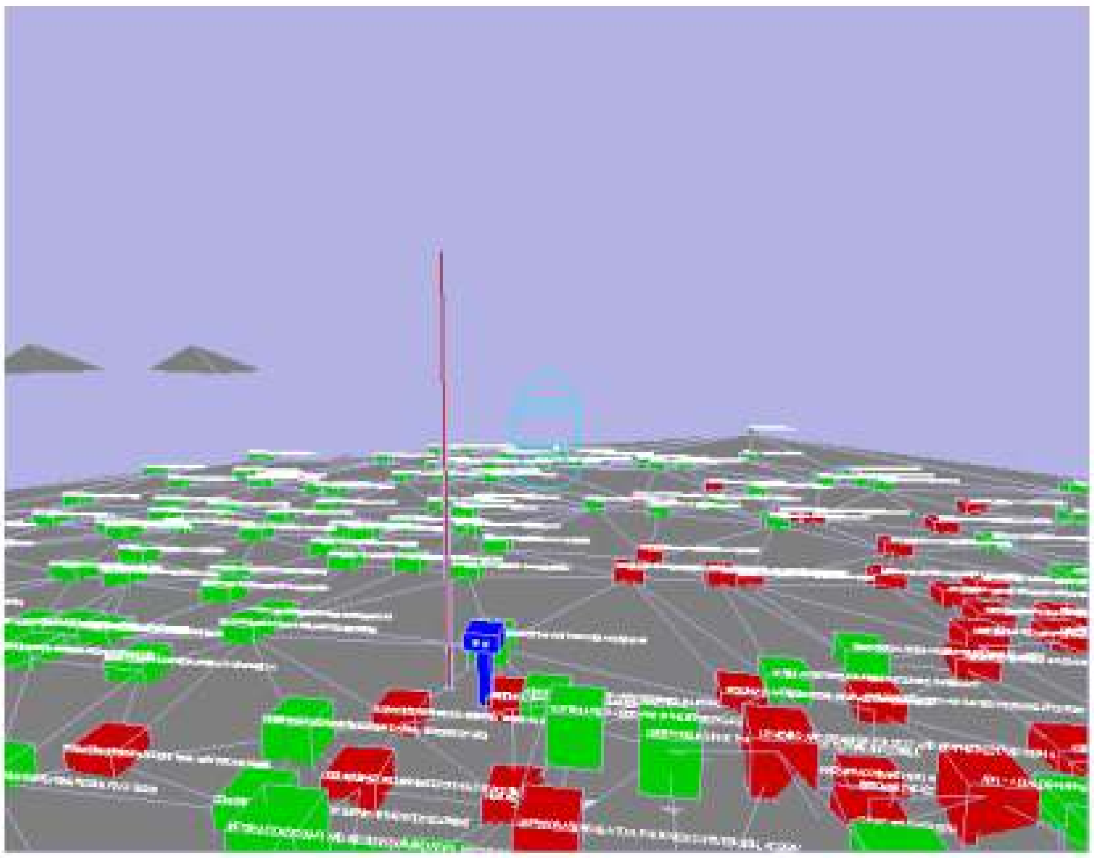


Figure 2.8: Dropping down closer to the landscape we can see individual documents as well as some artificial landmarks (the pole at the spatial origin, and some distant islands) (taken from [Chalmers M., 1995])

In these landscapes, similar documents were placed close to each other. A click of the mouse allowed the user to browse the individual documents in detail. The possibilities for searching and browsing were put in place to build up a concept of what themes and clusters there were, and insight is provided concerning where the various types of information could be found ([Chalmers M. & Chitson P., 1992], [Chalmers M., 1995])

The DIVE toolkit ([Fahlén L. E. et al., 1993]) was used for sharing a view of a landscape with another user by showing the same landscape on the network, so each user could see the other user moving around the landscape, and changes in colour due to the word search resulted.

2.3.2 SPIRE

SPIRE[1] [Thomas J. et al., 2001] is a software tool that was originally developed by the PNNL[2] as part of an assignment from the U.S. Army MRMC[3] to apply visualization concepts in a variety of areas of interest to the military. SPIRE provides innovative visual tools and approaches to automatically analyse large sets of textual documents such as technical reports, web data, newswire feeds and message traffic. SPIRE characterizes each document in n-dimensional vector space. The documents with n-dimensional feature vectors are clustered and projected from higher dimension into the lower dimension of 2D space. The projection in 2D is utilized to represent the data visually. The two visual interactive tools that spire offers are Galaxy visualization and ThemeView visualization.

Galaxy visualization uses the metaphor of the stars in the night sky, in which each star represents an individual document and each cluster of stars (galaxy) represents a cluster of topically related documents. Similar documents (stars) are placed near to each other in the galaxy (cluster), whereas dissimilar documents are separated by large space. The distance between the stars indicates their thematic similarity. The visualization provides an overview of the entire dataset. The figure 2.9 shows an example of galaxy visualization.

The ThemeView visualization employs the geographic landscape metaphor as described in section (2.1.4) for conveying the main topics in a collection and providing an overall sense of how the topics and documents are related to each other. Hills and mountains emphasize the density of documents and topical terms symbolically representing clusters of related documents – the higher the peak, the higher the density of documents and topical terms. Colours also show the density of documents and topical terms. The tops of peaks with

[1]Spatial Paradigm for Information Retrieval and Exploration

[2]Pacific Northwest National Laboratory

[3]Medical Research and Material Command

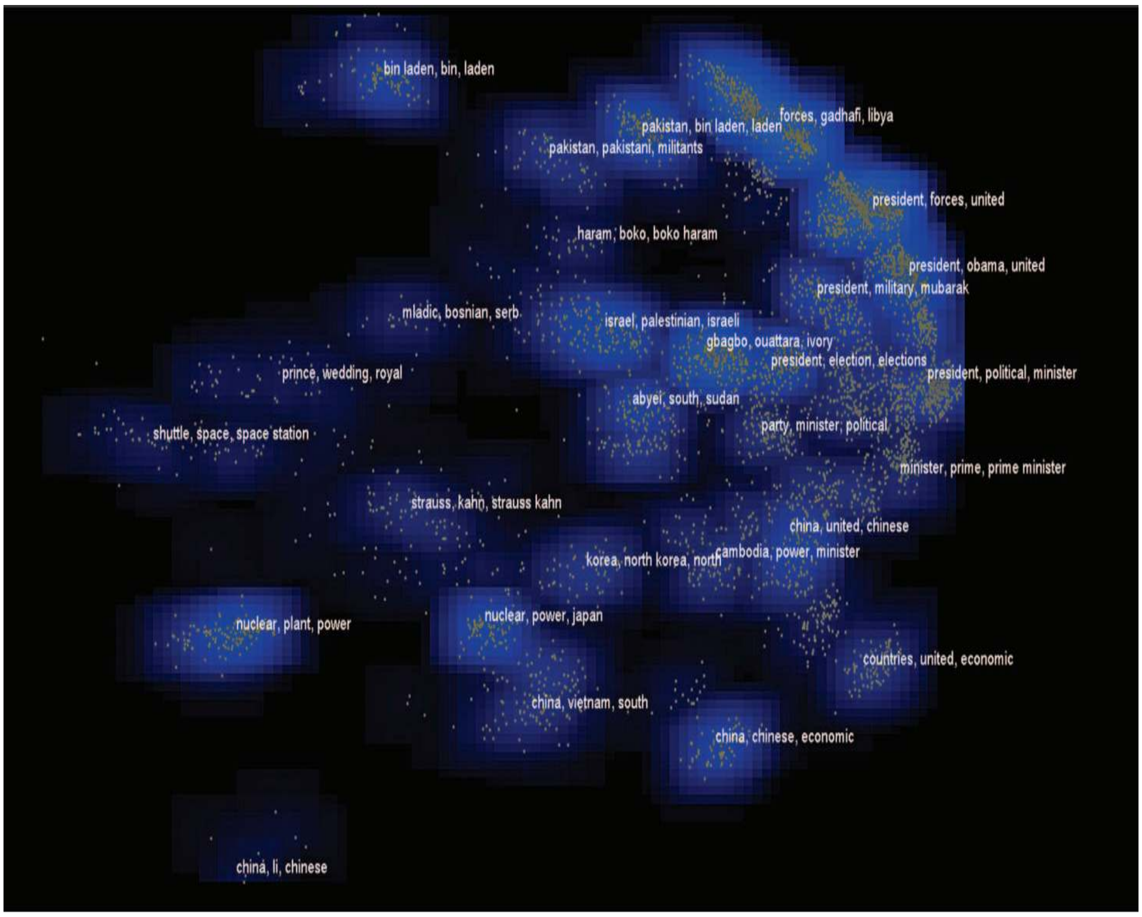


Figure 2.9: In galaxy visualization, the documents are represented by dots (stars), where each group of dots (stars) represents a cluster (galaxy) of similar documents (taken from [Endert A. et al., 2013])

a higher density of documents are represented with a bright yellow colour. A moderate density of documents is represented by a red colour, whereas the minimal density of documents is displayed in blue. Figure 2.2 shows a representation of ThemeView visualization.

As discussed in Section 2.1.4 the documents of a collection often continuously go through dynamic changes such as the adding, removing or modification of documents in a collection, which has to be taken into account when calculating the visualization and helping the user understand the changes through visualization (i.e. via a morphing information landscape). SPIRE provides ThemeView with a "Time Slicer", in which changes are shown from one slice and transitioning into the next slice. However, the users of SPIRE faced change blindness issues, when they were using the Time Slicer as they found

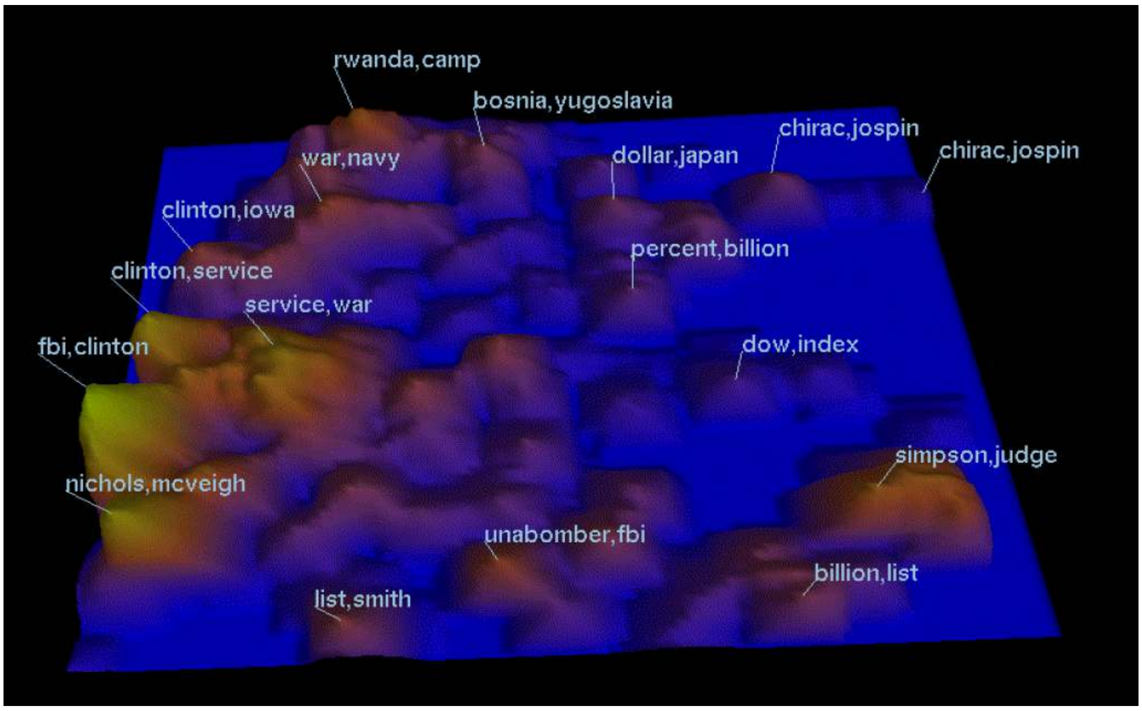


Figure 2.10: (taken from [Nowell L. et al., 2001]) Representation of ThemView visualization

themselves unable to identify what had changed. This topic has been discussed in section 2.1.5.

SPIRE can be deployed as a web browser applet and it can also be installed as a desktop application. Furthermore, it can handle real-time data, incorporating it very quickly into the equation ([vacommunity,], [SPIRE-PNNL,]).

2.3.3 VxInsight

VxInsight is a knowledge mining and visualization tool, which employs an intuitive landscape metaphor as described in Section 2.1.3. The system was built by projecting analysed and clustered collections of text abstract information into 2D space. VxInsight, which is similar to the SPIRE-Themeview offers a variety of interactive features to explore and manipulate the landscapes as well as ways to retrieve details on demand; this enables the user to get a quick and powerful analysis of the resulting landscapes [Davidson G. S. et al., 1998].

VxInsight was successfully applied in work with the analysis of patent databases [Boyack K. et al., 2000], and in the analysis of scientific and tech-

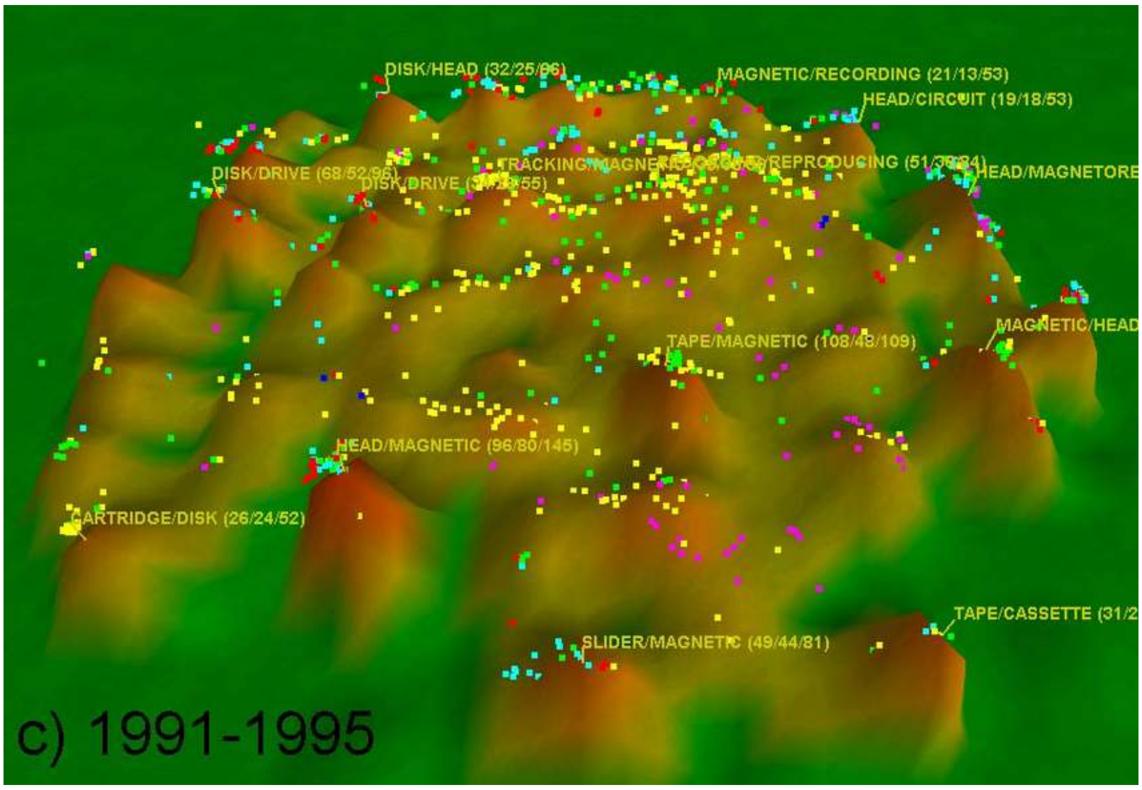

DISK/HEAD (32/25/96)
MAGNETIC/RECORDING (21/13/53)
HEAD/CIRCUIT (19/18/53)
DISK/DRIVE (68/52/96)
DISK/DRIVE
TRACKING/MAGNETIC/REPRODUCING
HEAD/MAGNETORE
MAGNETIC/HEAD
TAPE/MAGNETIC (108/48/109)
HEAD/MAGNETIC (96/80/145)
CARTRIDGE/DISK (26/24/52)
TAPE/CASSETTE
SLIDER/MAGNETIC (49/44/81)
c) 1991-1995

Figure 2.11: Landscapes of patent class 360 for four different five-year time periods (taken from [Boyack K. et al., 2000])

nological document sets [Boyack K. W. et al., 2002]. Figure 2.11 depicts the landscape representation of patent databases.

2.3.4 Galaxy of News

Galaxy of News ([Rennison E., 1994]) creates and visualizes an association network of relations between similar news articles. Galaxy of News illustrates the relationships between the articles in a three-dimensional information space using dynamically constructed pyramidal structuring, zooming and panning and animation. The aim of the tool is to provide the user a broad understand of a news base and interactively let the user refine the details of the information space for browsing and searching through large databases of news articles. The construction of information space is processed automatically based on relationships derived from the contents of the news articles [Krista Lagus S. K. & Kohonen T.,]. The Figure 2.12 shows the architecture of Galaxy of News. The Figure 2.13 shows the simplified version of associative relation network representation.

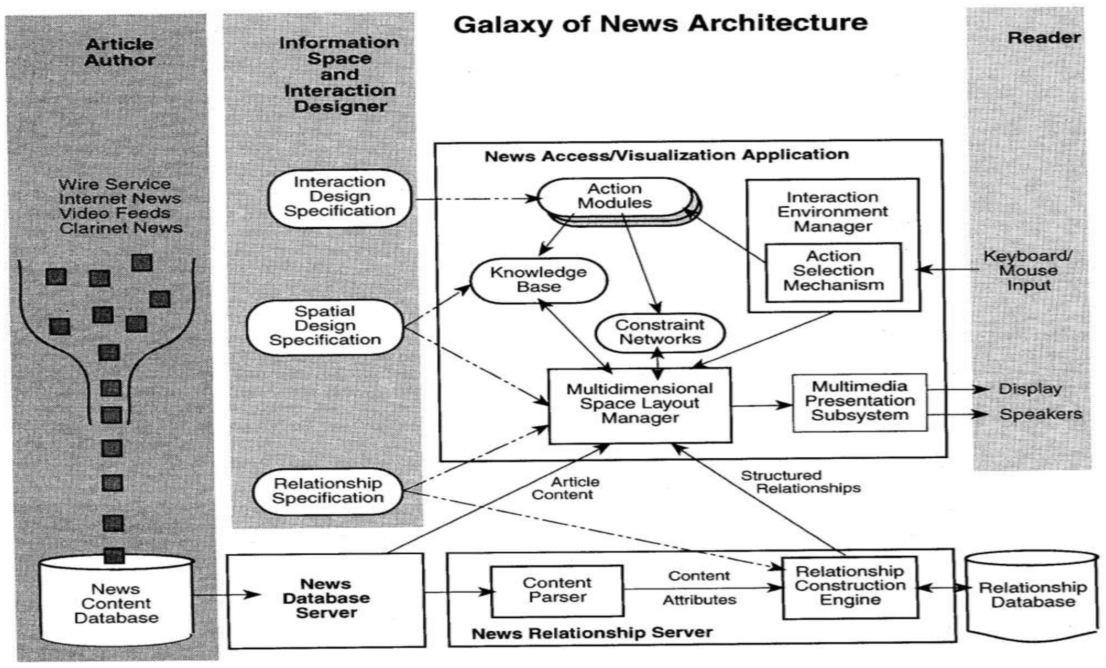

Galaxy of News Architecture
Article Author
Information Space and Interaction Designer
Reader
Wire Service
Internet News
Video Feeds
Clarinet News
News Content Database
News Database Server
Interaction Design Specification
Spatial Design Specification
Relationship Specification
News Access/Visualization Application
Action Modules
Knowledge Base
Constraint Networks
Interaction Environment Manager
Action Selection Mechanism
Multidimensional Space Layout Manager
Multimedia Presentation Subsystem
Keyboard/ Mouse Input
Display
Speakers
Article Content
Structured Relationships
Content Parser
Content
Attributes
Relationship Construction Engine
Relationship Database
News Relationship Server

Figure 2.12: Galaxy of News the Architecture (taken from [Rennison E., 1994])

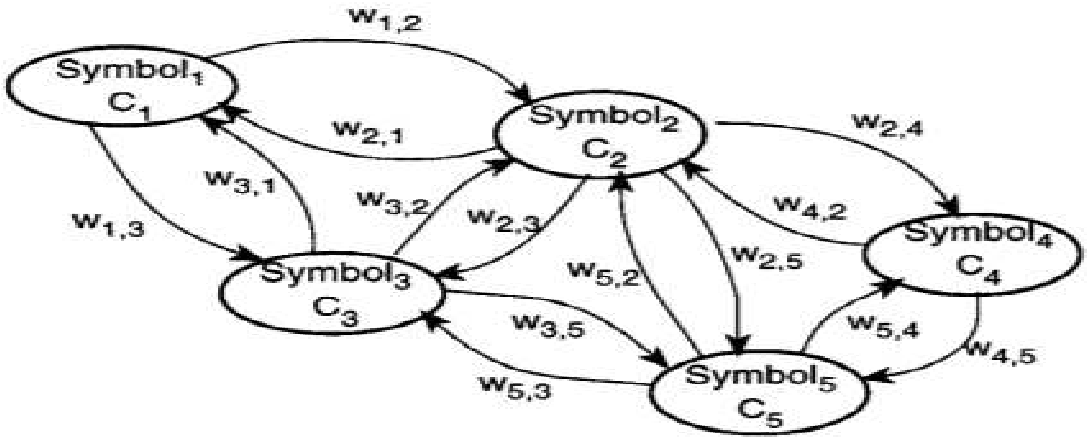
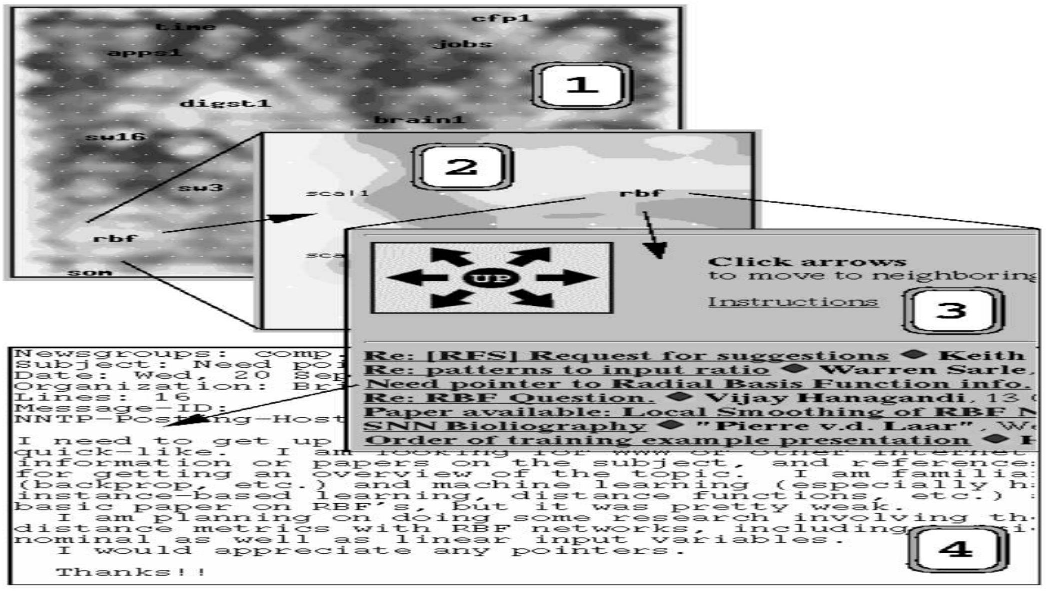


Figure 2.13: Simplified Associative Relation Network Representation(taken from [Rennison E., 1994])

2.3.5 WEBSOM

WEBSOM [Samuel Kaski K. L. T. K., 1998] is a method to retrieve full-text information as well as explore the large textual document collections. The document collections can be organized onto graphical map display for providing an overview of document collection and facilitating interactively browsing possibilities. Self-Organizing Map (SOM) [Kohonen T., 1997] algorithm is used to create a document map based on the statistical analysis of relationships between the words of documents. Documents of similar content are positioned close to each other on the document map, which forms a good basis for search and exploration.

A basic view of a web-based WEBSOM interface is shown in Figure 2.14, where in the descriptions of the screen captures are as follows: "(1) the whole map, (2) the zoomed map, (3) the map node, and (4) the document view, presented in the order of increasing detail. Moving between the levels or to neighboring areas on the same level is done by mouse clicks on the images or on the document links. Once an interesting area on the map has been found, exploring the related documents in the neighboring areas is simple" [Krista Lagus S. K. & Kohonen T.,].



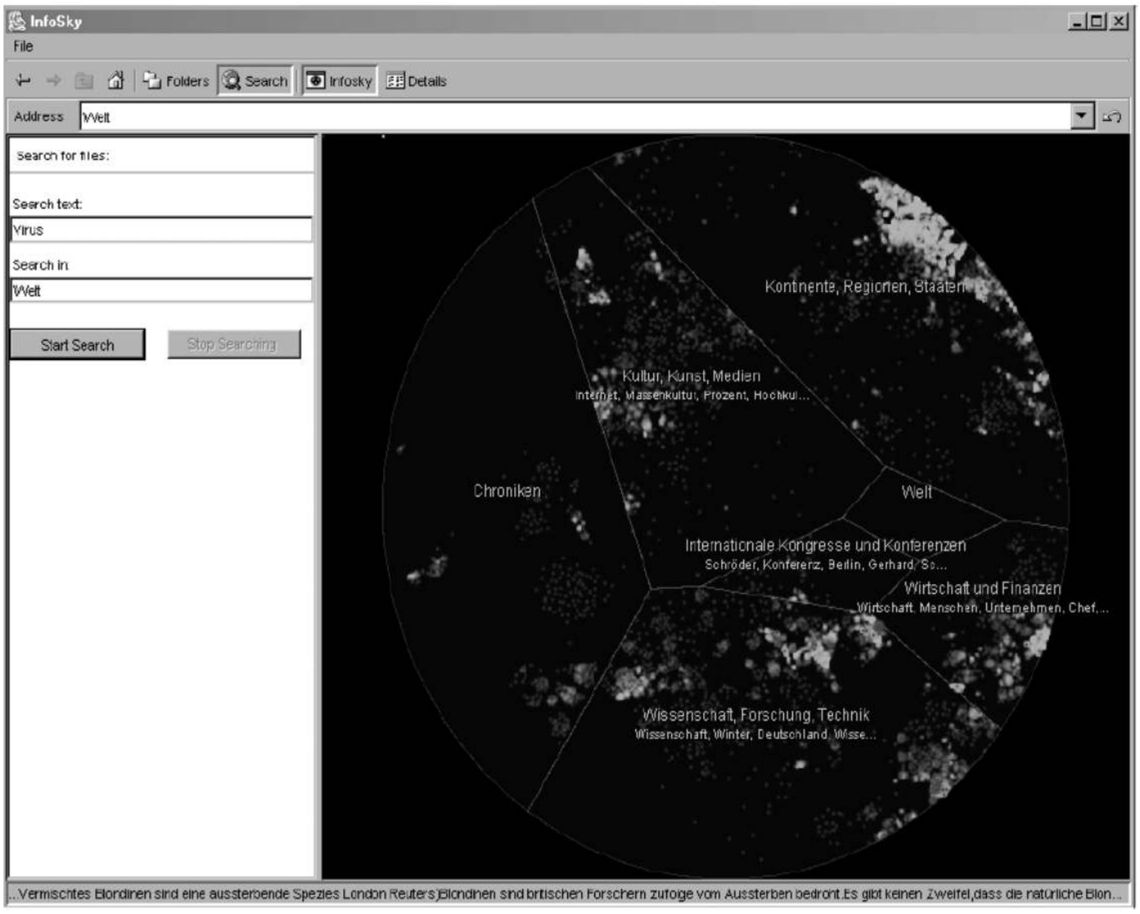
Re: [RFS] Request for suggestions ◆ Keith
Re: patterns to input ratio ◆ Warren Sarle,
Need pointer to Radial Basis Function info.
Re: RBF Question. ◆ Vijay Hanagandi, 13
Paper available: Local Smoothing of RBF N
SNN Bioliography ◆ "Pierre v.d. Laar", We
Order of training example presentation ◆ F

```
Newsgroups: comp.
Subject: Need poi
Date: Wed, 20 Sep
Organization: Br
Lines: 16
Message-ID:
NNTP-Posting-Host

I need to get up
quick-like.  I am looking for www or other internet
information or papers on the subject, and reference:
for getting an overview of the topic.  I am familia:
(backprop, etc.) and machine learning (especially h:
instance-based learning, distance functions, etc.) :
basic paper on RBF's, but it was pretty weak.
  I am planning on doing some research involving th
distance metrics with RBF networks, including
nominal as well as linear input variables.
  I would appreciate any pointers.

  Thanks!!
```

Figure 2.14: (taken from [Krista Lagus S. K. & Kohonen T.,]) WEBSOM user interface

2.3.6 Infosky

InfoSky ([Andrews K. et al., 2002], [Kienreich W. et al., 2003]) is a visualization technique that combines a traditional tree representation with a novel telescopic view of galaxies, helping the user to explore a large, hierarchically organized document collection. InfoSky employs a planar graphical representation, which is similar to a real telescope, and therefore it can be panned and zoomed to any magnification.


InfoSky
File
Folders
Search
Infosky
Details
Address
Welt
Search for files:
Search text:
Virus
Search in:
Welt
Start Search
Stop Searching
Kontinente, Regionen, Staaten
Kultur, Kunst, Medien
Internet, Massenkultur, Prozent, Hochkul...
Chroniken
Welt
Internationale Kongresse und Konferenzen
Schröder, Konferenz, Berlin, Gerhard, So...
Wirtschaft und Finanzen
Wirtschaft, Menschen, Unternehmen, Chef...
Wissenschaft, Forschung, Technik
Wissenschaft, Winter, Deutschland, Wisse...

...Vermischtes Blondinen sind eine aussterbende Spezies London Reuters)Blondinen sind britischen Forschern zufolge vom Aussterben bedroht.Es gibt keinen Zweifel,dass die natürliche Blon...

Figure 2.15: InfoSys the Visual Explorer (taken from [Kienreich W. et al., 2003])

In an InfoSky galaxy, documents are represented as stars, whereby within a collection, documents of similar content are displayed closer to each other than documents with little similarity to one another. In that way, clusters of stars are formed that represent similar documents. The collections are represented by areas (polygons) that surround other areas and stars, resembling the boundaries of constellations in the night sky. Again, collections with similar content are placed closer to each other. Access to both documents and collections may be restricted according to assigned user rights, therefore some areas (polygons) are displayed as black to a user due to insufficient access rights for documents, which resemble dark nebulae in real galaxies. The telescope is a metaphor for interacting with the visualization. The magnification of telescope is variable, therefore a user can point the telescope at specific objects and get an overview of the entire area, and there is also a "zoom" operation that allows focusing on small details. To make the navigation simple, a user has access to various tools so that the focused object can be displayed in the optimal size. An integrated history function allows a user to easily navigate back to previous views.

InfoSky follows the standard client-server model. The component on the server side takes the documents, creates galaxy geometry and stores it for a particular hierarchically structured document collection. The component on the client side visualizes a subset of a galaxy and makes it explorable for a specific user. Following this model, InfoSky is able to generate a galaxy representation of millions of documents within a few hours as well as visualize it in real time. The Figure 2.15 shows the user interface of InfoSky.

The generation of galaxy geometry is done recursively from top to bottom in multiple steps:

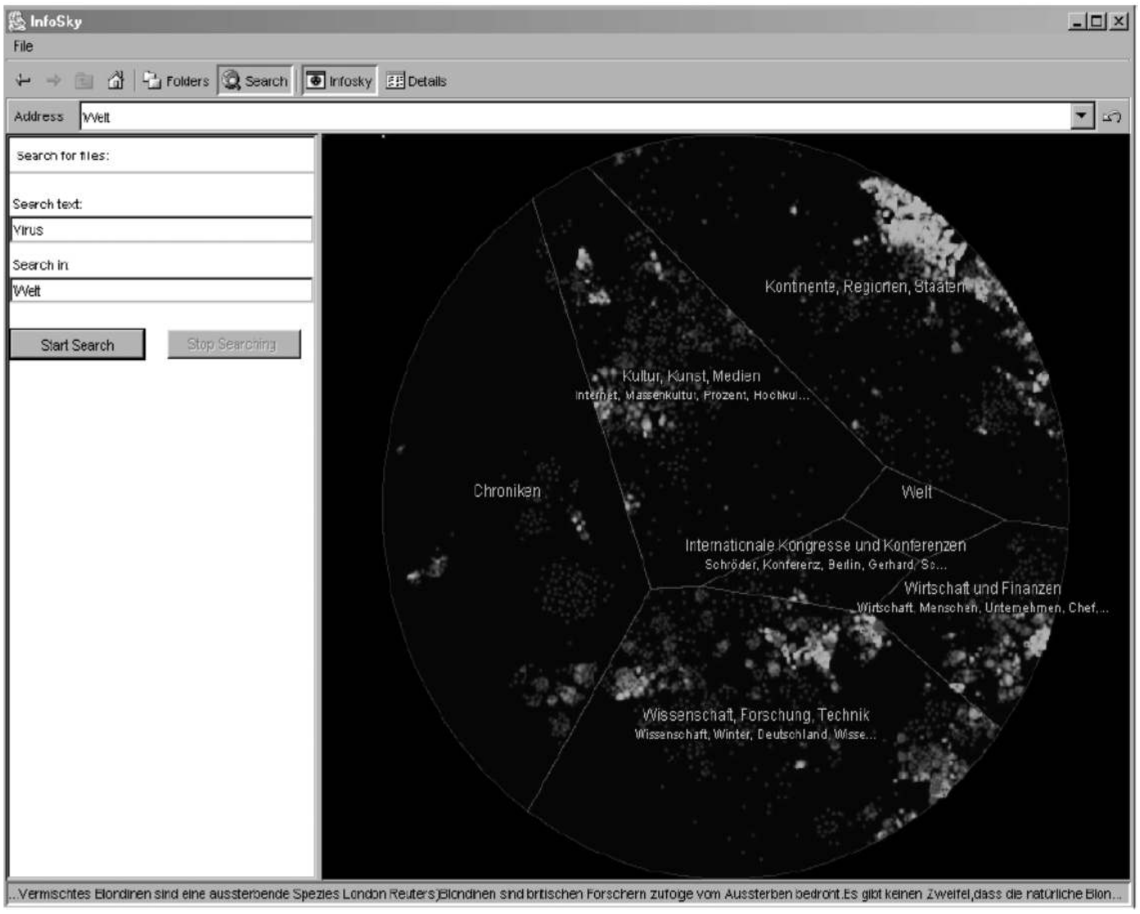
1. A similarity placement algorithm (FDP) is used to position the centroids of the sub-collections in a normalized 2D space at each level based on their similarity to each other.

2. A geometric transformation is used for transforming the layout space to a polygonal area of a parent collection.

3. For each sub-collection centroid, a polygonal area is calculated based on the total number of documents and collections in the sub-collection. A Voronoi diagram algorithm [Okabe A. et al., 2000] is used for partitioning the area of the parent collection.

4. Again, the similarity placement algorithm (FDP) is used to position documents of the collection as points, based on their similarity to vectors document as well as collection centroids.

Another feature of InfoSky is the possibility for users to search for documents and collections by issuing a query. The matching documents and collections are highlighted in shades of yellow and the users can then examine them in further detail (See Figure 2.16.


InfoSky
File
Folders
Search
Infosky
Details
Address
Welt
Search for files:
Search text:
Virus
Search in:
Welt
Start Search
Stop Searching
Kontinente, Regionen, Staaten
Kultur, Kunst, Medien
Internet, Massenkultur, Prozent, Hochkul...
Chroniken
Welt
Internationale Kongresse und Konferenzen
Schröder, Konferenz, Berlin, Gerhard, Sc...
Wirtschaft und Finanzen
Wirtschaft, Menschen, Unternehmen, Chef,...
Wissenschaft, Forschung, Technik
Wissenschaft, Winter, Deutschland, Wisse...

...Vermischtes Blondinen sind eine aussterbende Spezies London Reuters)Blondinen sind britischen Forschern zufolge vom Aussterben bedroht.Es gibt keinen Zweifel,dass die natürliche Blon...

Figure 2.16: Search results for "Virus" being displayed (red color as stars) (taken from [Kienreich W. et al., 2003])

One of the disadvantages of InfoSky is that it requires an externally defined hierarchy to generate the layout, which is not always available. One reasonable solution to that problem is to automatically generate a topic hierarchy [Muhr M. et al., 2010]. Sabol applied such an approach [**?**] by constructing a topical hierarchy using recursive clustering from a large document collection. The InfoSky projection algorithm is used to create an information landscape visualization from a hierarchy for exploring and navigating the data.

2.3.7 Dynamic Topography Information Landscape

This Section discusses an incremental and scalable approach for visualizing a a dynamic landscape as well as approaches to conveying changes. Furthermore, this section introduces algorithms for computing dynamic information landscapes. Additionally, this section presents an application on which the approach was successfully tested.

Incremental and scalable approach in general

Static landscape visualizations cannot convey changes. [Sabol V. et al., 2010] proposed the visualization of topical changes through information landscapes with dynamic topographies and morphing.

As discussed in Section 2.1.4, when a collection of text documents changes over time, e.g. documents are modified, new documents are added or old documents are removed, the overall topical structure and consequently the landscape topography change as well. Sabol [V. S. & Scharl A., 2008] describes the transition of the landscape topography from an old to a new landscape, taking the user perspective as well as the overall goal and performance in account. According to them, the transition of a landscape topography from an old to a new temporal configuration have to be incremental and adaptive such that no unnecessary changes are introduced in the topography. Furthermore, they state taht the parts of the topography which were unaffected by the modification or are not part of observation in a selected timeslot should remain as stable as possible with respect to their previous position and shape. In this way, the user will be able to understand the modified topography immediately due to the fact that the user has already become familiarized with preserved elements of the topography. Additionally, the authors argued that to make the user follow and understand the changes, the adaptive and incremental transitions should be smoothly animated. To achieve the above-described behaviour, a fast, incremental, scalable layout algorithm is required.

Algorithm for computing a dynamic topography information landscape

The force directed placement approach is one way to accomplish this; it fulfils the criteria of adaptive behaviour. The advantage of FDP is that FDP is inherently incremental for a previously computed layout, which means that any modification of a dataset can be smoothly assimilated into the old layout without disrupting a whole landscape representation that the user has already been familiarized with, and therefore FDP requires only a fraction of computation efforts for a full recomputation.

The incremental algorithm for computing dynamic information landscapes contains the following five consecutive steps [Sabol V. et al., 2010]:

1. Document clustering: Topically related documents are clustered into

groups using an incremental k-means algorithm.

2. Cluster positioning: A force-directed placement algorithm projects topical clusters into the 2D visualization space.

3. Document positioning: A similar method is used to project documents into the 2D space depending on their topical similarity to the clusters.

4. Map rendering: An elevation matrix is computed to generate the landscape image.

5. Label computation: The most descriptive keywords are computed and placed in the vacinity of peaks as labels.

Additionally, FDP scales very well in an optimized version [Sablol V., 2012], where it is applied results of recursive clustering (i.e. cluster hierarchy).

Incremental computation

The initial landscape is computed by applying the algorithm to all of the documents in a dataset. The computed document layout positions are stored for future use. When the dataset changes, the FDP algorithm is initialized with previously computed and stored layout positions before another computation is carried out. This crucial step is required because the FDP algorithm is very sensitive to the initial configuration – otherwise the FDP algorithm will most likely produce completely different results. The time complexity of the entire process is O(n), it the number of the clusters in constant, which is the case ith K-Means [Sabol V. et al., 2010].

Webrat

Webrat ([Sabol V. et al., 2002], [Granitzer M. et al., 2003]) was one of the first incremental and dynamic systems, which employed an animated information landscape supporting fully dynamic topography. The system visualized and refined the search result sets. The matched documents of a query were dynamically clustered on the fly and visualized as an information landscape. The thematic visual clusters were assembled, analysed and rendered in real time. The system provided various interaction possibilities for exploring the visualization, such as refining the queries by selecting from keywords of the visual clusters. Figure 2.17 gives an example of an information landscape using the Webrat system.

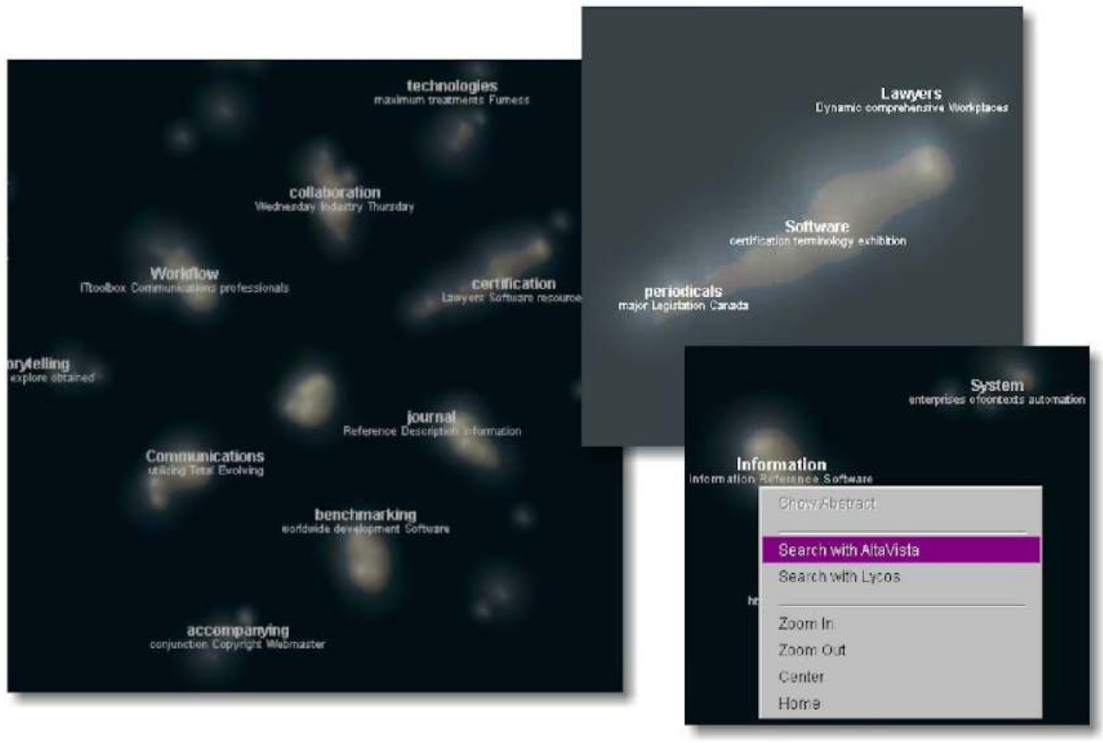


Figure 2.17: The Webrat interface: global view (left), zooming in (top right), context menu (bottom right) (taken from [Sabol V. et al., 2002])

Hierarchal Information Landscape

The concept of an hierarchical information landscape has been adapted InfoSky (See Section 2.3.6). Sabol [Sablol V., 2012] introduced hierarchical structures of a large "Flat" data set in the information landscape. It was designed to interactively explore the relatedness between the documents and clusters and as well as to navigate the information landscape according to a hierarchy of topical clusters represented by nested polygonal areas. Each polygonal area is labelled by the most descriptive terms, which summarizes the content of the underlying documents and guides the user to explore areas of interest. Incremental computation of the hierarchical layout is also possible, as it relies on K-Means and FDP Algorithms.

A representation of a hierarchical information landscape is shown in Figure 2.18.

The information landscape allows the users to zoom in, pan, rotate and tilt as well as manipulate the visual properties of the document. Furthermore, the

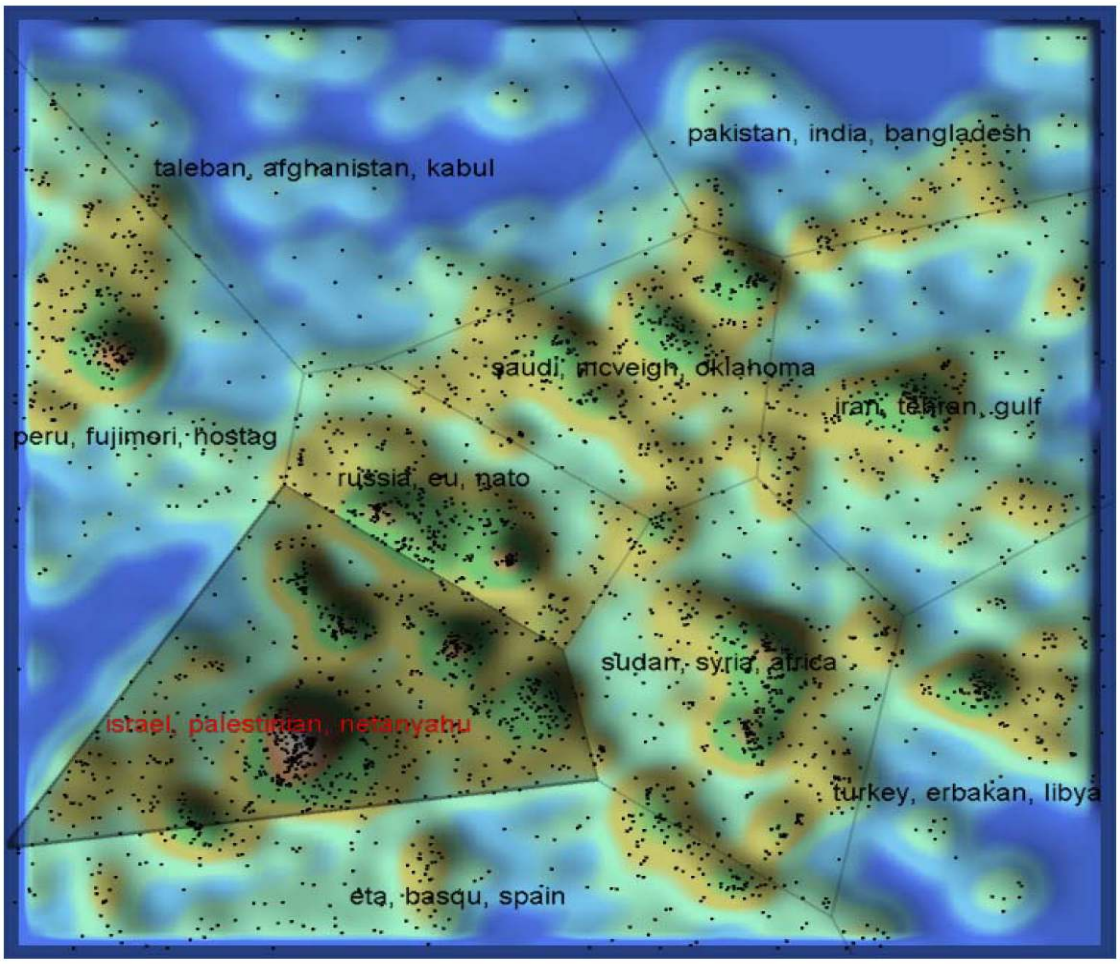


Figure 2.18: An information landscape (taken from [Sablol V., 2012] showing 3,000 documents on ”terrorism" and providing an overview of the whole data set in 10 topical clusters (polygonal areas); by clicking on the label (e.g. ”isreal, palestinian, netanyahu"), the corresponding cluster is zoomed in and the areas and labels of underlying clusters are shown (see Figure 2.19)

visualization allows the user to select documents individually or as clusters as well as based on arbitrary subsets using a lasso tool. By zooming in on a cluster, the corresponding sub-clusters are shown (See Figure 2.19), similarly to the InfoSky system (See Section2.3.6). The information landscape is implemented in Java, and for critical performance, 3D acceleration has been used through a (JOGL) library that allows visualization of more than a million items in real time on a standard PC with integrated graphics processing and 1GB of main memory.

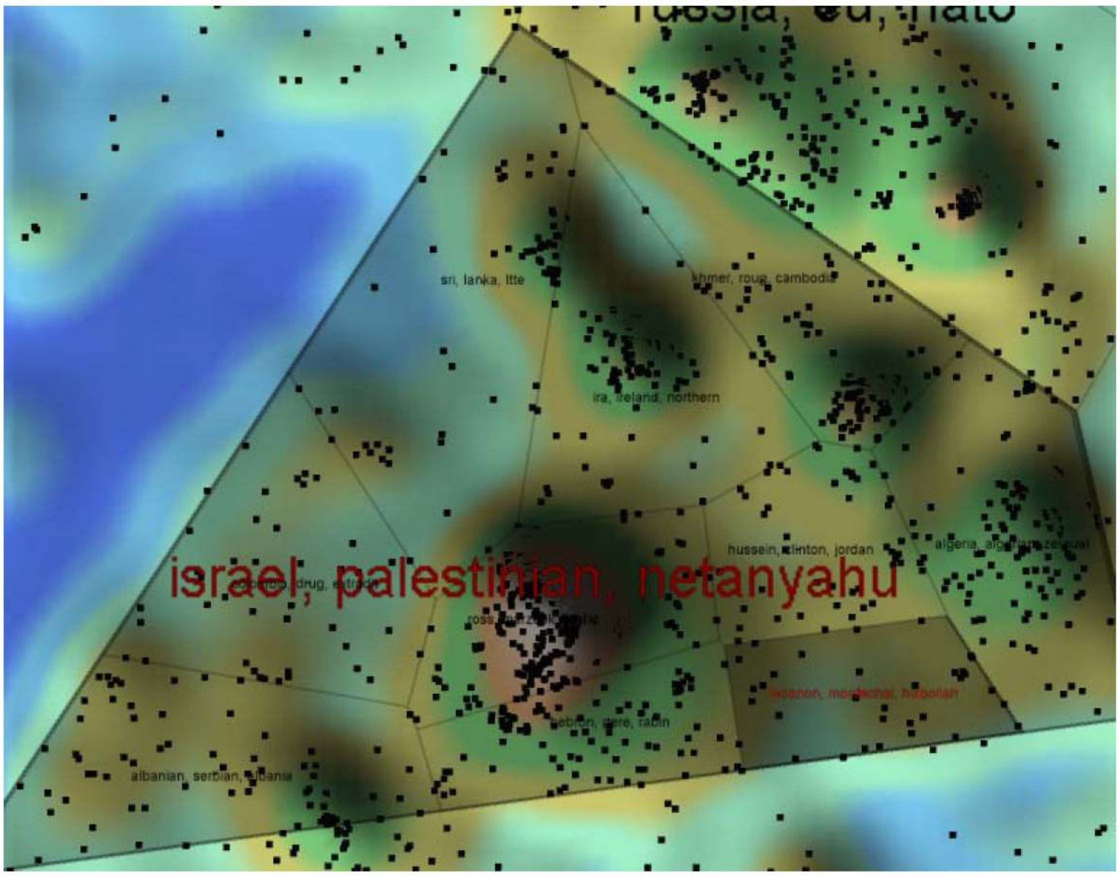


Figure 2.19: A user can narrow the potential topic of interest by showing a sub-cluster of a zoomed-in cluster (e.g. ”isreal, palestinian, netanyahu"). Sub-Clusters are shown as nested voronoi areas within the area of their parent clusters (taken from [Sablol V., 2012])

The hierarchical information landscape component was one visualization of the VisTools Framework from KnowMiner ([Klieber W. et al.,]). KnowMiner is a knowledge discovery framework for automatically extracting knowledge from large, text document collections.

Reading through dynamic landscape metaphor

[Ulbrich E. et al., 2015] built an HTML5-based implementation of a dynamic information landscape for presenting interactive metaphors, inspired by map reading and visual transitions, which enhanced the landscape representation for the analysis of topical changes in dynamic text repositories. Figure 2.20 illustrates the user interface with multiple views representing the testing en-

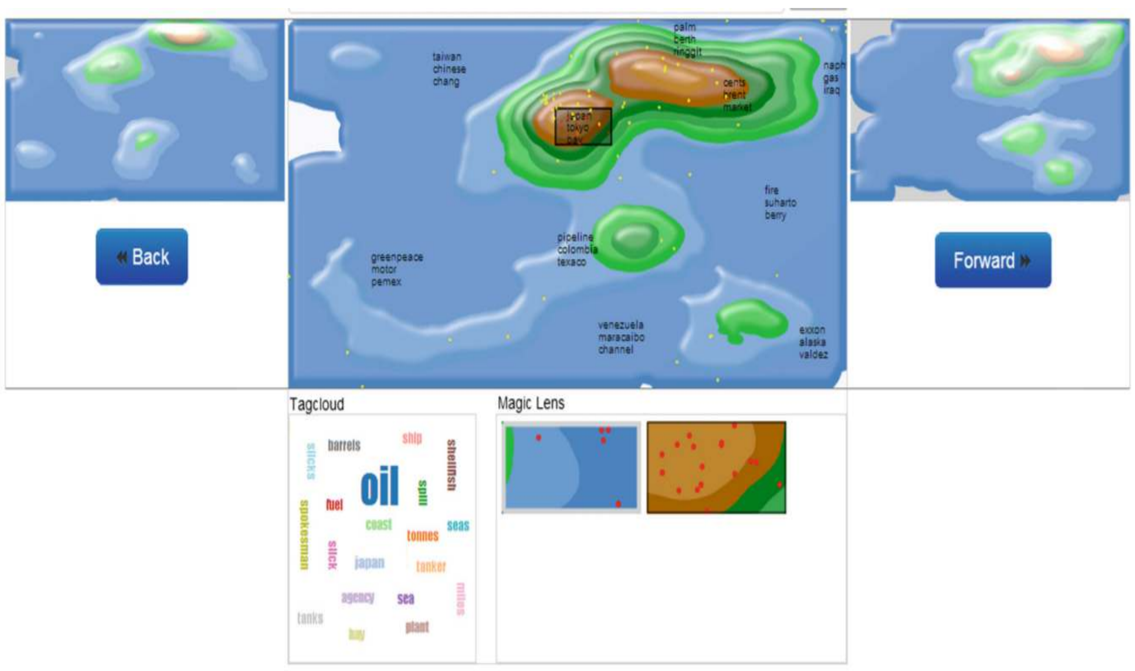


Figure 2.20: (taken from [Ulbrich E. et al., 2015]) The user interface with multiple views

vironment.

The interactive metaphors for topic analysis included interactive morphological transitions, multiple views, trails and traces. Their study concentrated on building and testing hypotheses using map-reading metaphors. The change blindness phenomena, which occurs when the user does not notice major changes as describe in section (2.1.5) was not part of their study. The information landscape has been integrated into a web browser using scalabe vector graphics (SVG).

One of the drawbacks of rendering information landscape in SVG in comparison to WebGL is that two additional steps are required (i) extracting of topic mountains by cutting these into contour levels and (ii) computing of mappings between the corresponding contour lines for pairs of consecutive landscapes in order to apply a smooth morphing in a later stage. In contrast to SVG, WebGL requires only elevation maps.The second, and major, SVG drawback is performance.

Chapter 3

Morphing Information Landscape

This chapter introduces morphing landscapes, which still roughly match the realistic natural environment in order to make the information landscape intuitively understandable. The first section introduces all the required features and aspects for developing an interactive information landscape using WebGL and HTML5. The second section discusses the pipeline for processing a complex and multidimensional dataset consisting of text documents and showing in 3D space so that the information landscape can be visualized. The third section gives an overview of the visual design of a morphing landscape. The fourth section covers the overall interaction design and the UI of the application.

3.1 General Concept

The concept is based on applying an information landscape for analysis of real-word dynamic data, i.e. text documents of a collection, which typically contain complex and multidimensional information. The main goal is to make the collections of text documents logically understandable and explorable for a user by providing an overview of the dataset, showing the relatedness and topical distribution of datasets as well as visually showing how the topography of documents of one period of time is changed to another span of time. This is achieved by using the morphing concept of an information landscape for allowing the user to grasp the dynamic changes in datasets in an understandable way.

As was already discussed in Sections 2.1.3 and 2.1.4, an information landscape is a commonly used method of conveying topical relatedness and dynamic changes through spatial proximity in large, multidimensional and dynamically changing datasets, such as a collection of text documents.

This section discusses all the features and aspects for developing an interactive information landscape using WebGL and HTML5, which includes the features for computing the layout of the information landscape, for creating intuitively understandable visual design as well as for designing the interactions by taking into account the following aspects: (i) view of the information landscape from different perspectives, (ii) selection of documents, (iii) interaction related to morphing, (iv) exploration, filtering and navigation of documents and (v) adjusting the configuration parameters of the information landscape as desired.

3.1.1 Features for computing the layout of information landscape

The layout of an information landscape first has to be computed by transforming the complex and multidimensional text documents into 2D or 3D spatial representations while preserving the original relationships in such a way that similar items are positioned more closely together, while dissimilar items are placed further apart. Hills emerge where the density of related documents is high, indicating a topical cluster. Areas of higher elevation are labelled with the most significant keywords from the underlying documents.

By changing the collection of text documents (e.g. adding new documents) the overall topical structure and consequently the landscape topography is changed as well. To help the user to follow and understand the changes, the geometry of two landscapes are morphed using a smooth animation. In this process of morphing, the hills and islands grow or shrink to indicate the growth or decline of topic prevalence, and hills move towards or apart from each other indicate topical convergence and divergence.The text-processing pipeline developed by the Know-Center is used to compute the layout of the landscape. The development of the text-processing pipeline is not part of this master's thesis – additional adaption had to be integrated to generate morphing data from periodically organized documents of collection.

For building the use case scenarios for this master's thesis, 489 available papers that have been published by the Know-Center were organized temporally in a separate folder. Furthermore, the pipeline takes the folder that contains all the temporarily organized data in a folder as an input parameter as well as a folder as an output parameter. The pipeline runs all the steps as described in Sections (2.1.3 and 2.1.4) and generates the morphing data as a JSON file for every period. Every JSON file contains the following extracted information: (i) the location of documents, (ii) the height map for modelling the information landscape, (iii) the peak-specific information (labels, locations), (iv) a list of keywords including the frequency in the specific document. JSON Files are loaded into the application in order to build an interactive information landscape.

3.1.2 Visual Design Features

The first aspect that has to be considered is the designing and the creating of an information landscape that roughly matches the real natural environment in order to make the landscape more intuitively understandable. The height map from the JSON file should be used to model the information landscape. It is reasonable to involve the human interpretation of landscapes in the generation of a visual design of the landscape. A human sees a landscape as different sets of layers, where a water texture defines the lowest points, with sand, grass, rocks and snow textures as the height increases – such a representation is technically called height-based texture mapping. WebGL provides the possibility to run custom shading code on graphics hardware (GPU). The efficiency of GPUs (i.e. graphics processing units) in manipulating computer graphics, due to the highly parallel structure of their computations, should be utilized in applying the height-based texture mapping method to each pixel of landscape using the shading language (i.e. the fragment shader).

The peaks of related information from the JSON file (labels, locations) should be used to place the labels above the peaks as legible and recognizable text in the 3D environment of the information landscape. Furthermore, the labels should always be readable, regardless the viewpoint of the user.

The documents should be placed in the 3D environment of the information landscape according to the extracted information in the JSON file.

As discussed in Section 2.1.4, seamless and smooth animation is used to help the user to follow and understand the changes by morphing the information landscape into a new information landscape involving dynamic changes. While generating the visual design of morphing, the fact that humans interpret the height as different layers should be kept in mind. Furthermore, the efficiency of the GPUs should again be utilized to manipulate the geometry of the landscape using a shading language (i.e. a vertex shader) to create a smooth 3D accelerated morphing animation of landscape geometry. As discussed earlier, the concept of morphing can elicit a change blindness problem, which makes the user unable to identify what has changed. To counteract this, one of the recommended methods from Novell (e.g. ‘using a wireframe in combination with changes in colour and translucency') should be adapted in the implementation in order to overcome such blindness phenomena as well as to provide the users with better orientation in terms of emerging and vanishing topics. Furthermore, the animation should include the movement of documents, indicated by drawing a line between the old position and the new position for a document-based analysis of emergence of convergence and divergence of topics.

3.1.3 Interaction Design Features

The interaction design for the information landscape should be developed by grouping the interactions into the following five categories:

- Perspective category: Enables the users to interactively view the information landscape from different perspectives, i.e. (i) rotate and tilt the information landscape in any direction, (ii) pan the information landscape to reveal the information which was previously outside the displayed screen area, (iii) zoom in and out to view the information landscape in more or less detail.

- Selection category: Enables the users to select the documents individually, based on content and as well as based on their positions in the landscape, by using either the brush-based selection tool or the rectangle-based selection tool. For selecting the documents based on their positions in the landscape, two selection tools should be available. The brush-based selection tool should be implemented in WebGL by utilizing the efficiency of graphics processing units. The visual design of the brush should imitate the effect of a torchlight or flashlight, i.e. a circle that changes in size and shape based on height. Furthermore, it makes sense to represent the selected data of interest in a logical arrangement of keywords – a so-called tag cloud – which displays the keywords in different sizes based on their prevalence in the selected content. As a navigation tool, this concept helps the user to easily search the content by selectiong keywords.

- Morphing category: Enables the users to load the temporal morphing data processed by the text processing pipeline in the application and represent it as executable buttons for triggering the morphing in the form of smooth animation.

- Data exploration, filtering and navigation category: Provides the users with possibilities to explore the selected data as well as filter and navigate through the data.

- Settings category: Provides the users with the following basic possibilities to adjust the information landscape according to their desired setup: (i) adjust the brightness and opacity of the information landscape, (ii) change the colour and size of the brush-based selection tool, (iii) change the design of the information landscape, (iv) set the timespan of animation of morphing and (v) set the height scaling (linear, square root, squared and logarithmic). The selected option should be applied to the information landscape with immediate effect. In cases in which morphing is running and the user has applied one of the above-mentioned options, the morphing should not be hindered.

3.2 Computation of the Landscape Layout

As discussed in chapter 2.2, the landscape layout requires the positioning of documents in 2D or 3D space in such a way that similar documents are positioned closer together, while dissimilar documents are placed further apart. The real challenge is that the datasets are complex and multidimensional. This section describes the text-processing pipeline for computing information landscape layouts as well as a process for computing information landscape layouts with dynamically changing datasets. The general approach presents all the steps needed to compute the layout of a landscape, beginning with the process of transforming the complex and multidimensional text dataset into 2D space while preserving the original relationships as much as possible, followed by computation of a height map and labels.

3.2.1 Text Processing Pipeline

The figure 3.1 depicts the overall the text processing workflow in following six steps:

1. NLP feature engineering: A vectorization process is applied to a collection of text documents to turn them into numerical feature vectors containing terms and their frequencies. Before the vectorization process is applied to a collection of text documents, a data cleaning process is carried out on the collection of text documents, which removes features (i.e. stopwords) from the text of the document that have little or no significance.

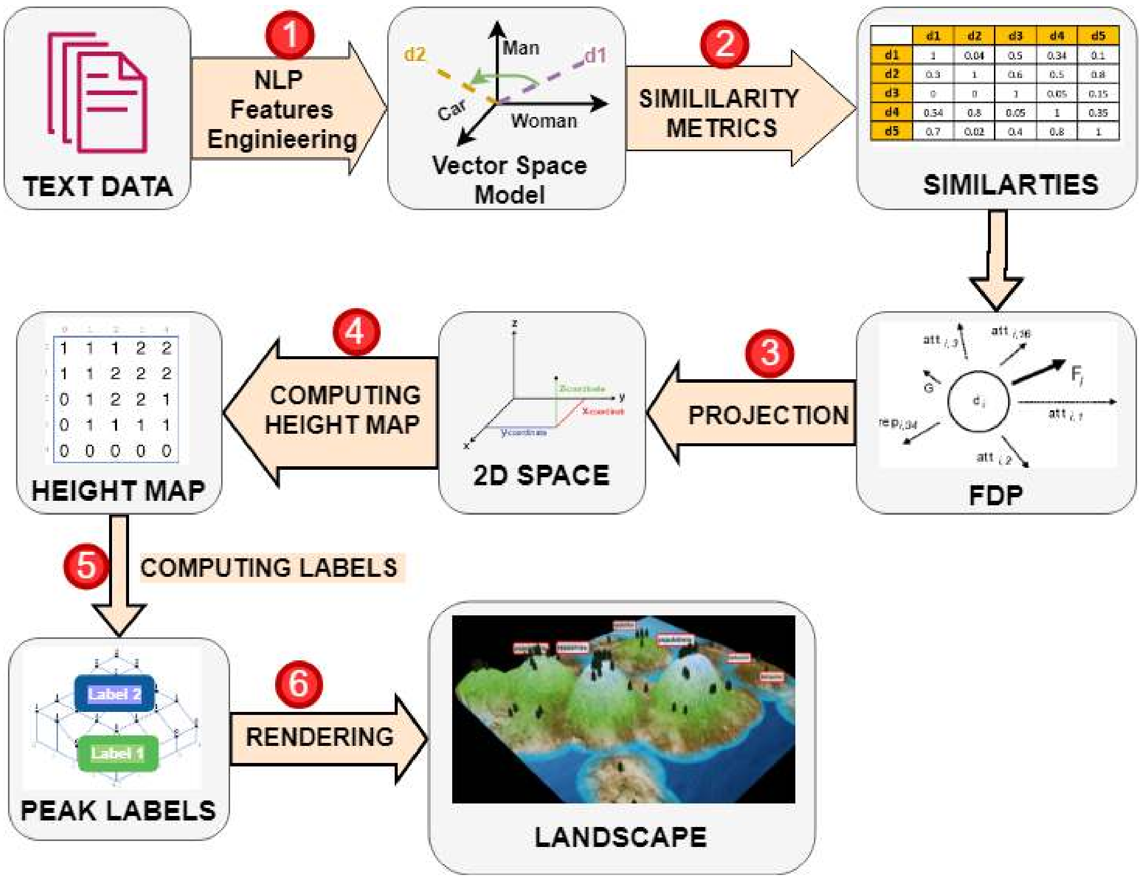


Figure 3.1: Representation of workflow for (i) projecting complex, high dimensional documents into 2D space, (ii) computing a height map and labels

2. Similarity metrics: The cosine similarity calculates the similarity of two document vectors by measuring the cosine of the angle between two vectors. The result of this process is a similarity matrix in which the similarity of each pair of documents is in the range of 0 to 1 – a value closer to 0 indicates less similarity, and for a value closer to 1 there is greater similarity.

3. Projection: A force-directed algorithm (see Section 2.2.3) is used to project a multidimensional dataset into 2D space in an iterative process. The algorithm requires a similarity matrix and the initial position of the documents for computing.

4. Computing the height map: In this process a height map is computed, which represents the information landscape model. As mentioned in Section 2.1.3, hills emerge where the density of related documents is high.

Thus, the factor of how dense the documents are positioned together is a value that is used to calculate the height map.

5. Computing labels: The computation of labels consists of two steps. In the first step the locations, heights and related documents for each peak are identified. In the next step, the most significant and descriptive labels are calculated from the list of documents in relation to a peak.

After completion of the six-step workflow, the following extracted information should be known: (i) the location of documents, (ii) the height map for modelling the information landscape, (iii) the peak-specific information (labels, locations, a list of documents related to each peak), (iv) a list of keywords along with their frequency in each document. All of the extracted information is used to visualize the interactive landscape using WebGL and a tag cloud.

3.2.2 Morphing Layout Generation

As discussed in Section 2.1.4, when a collection of text documents changes the overall topical structure, the landscape topography consequently also changes.

An incremental and adaptive method is used to convey the changes through spatial proximity in the visualization by applying a transition of the landscape topography from an old to a new landscape. The FDP approach, as described in Section (2.2.3), fulfils the criteria of incremental and adaptive behaviour. One of the advantages of FDP is that it is inherently incremental for a previously computed layout, and therefore any modification of a dataset can be smoothly assimilated into the old layout without disrupting a whole landscape. Figure 3.2 illustrates the overall workflow.

The initial landscape is computed normally, as described in Section 3.2.1, whereby the computed layout positions of the documents are stored for future use. When the dataset changes (e.g. a document is added, removed or modified) the layout of the information landscape is computed following all the steps described in the previous section, except a change in the FDP algorithm before computation is applied. The FDP algorithm is initialized with the previously computed and stored layout positions. This step is required because the FDP algorithm is very sensitive to the initial configuration; otherwise, the FDP algorithm will most likely produce completely different layouts.

To allow the user to follow and understand the changes, the initial information landscape is seamlessly morphed into a new information landscape containing dynamic changes using smooth animation.

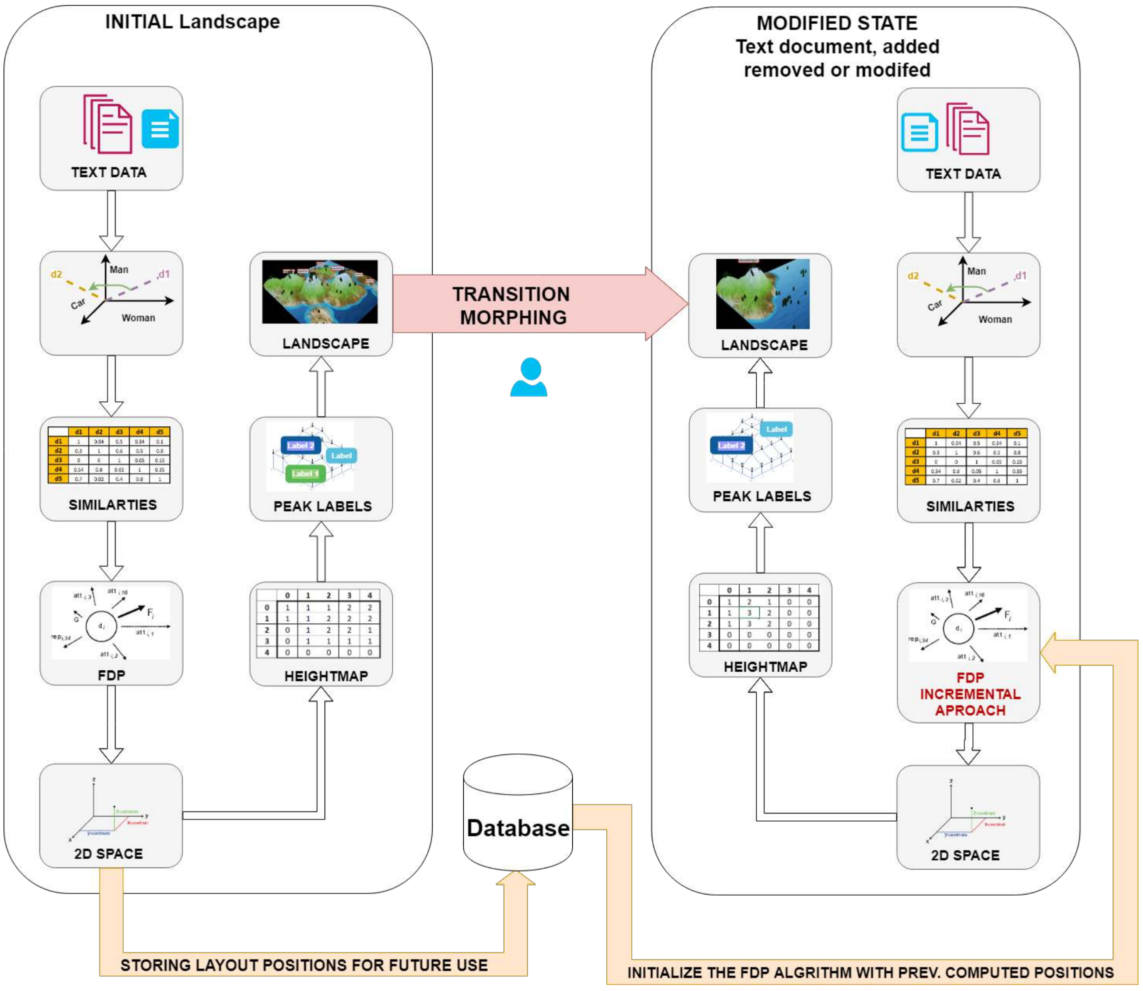


Figure 3.2: Representation of a workflow for producing a landscape layout with dynamic changes by applying an incremental method

3.3 Visual Design

The information landscape employs a geographic landscape metaphor. Representing the data as visualizations of natural landforms that imitate the natural environment, it will be easier and more intuitive for a user to interpret the visualization. The more the information landscape resembles the natural environment, the more useful the visualization. Thus, the goal is to create a landscape in a way that roughly looks like what a real natural environment might look like. This section consists of five subsections. The first subsection explains the general approach to generating the visual design of an information landscape that roughly matches the real natural environment. The second and third subsections cover the general and visual representation of documents and peak labels. The fourth section introduces the visual design of using a brush for selecting documents. The last section describes the visual concepts of morphing.

3.3.1 Visual Design of Landscape

A human user interprets the landscape using different layers as follow: the bottom layer is seen as a sea, the next upper layer is seen as a motion of a wave hitting the coast, the layers in the middle part viewed as sand followed by a grass layer. The top layers are rocks followed by snow.

Representing the information landscape as a visualization in layers is actually a technique called height-based texture mapping. A texture shown in Figure 3.3 is chosen depending on the height at that part of the information landscape.

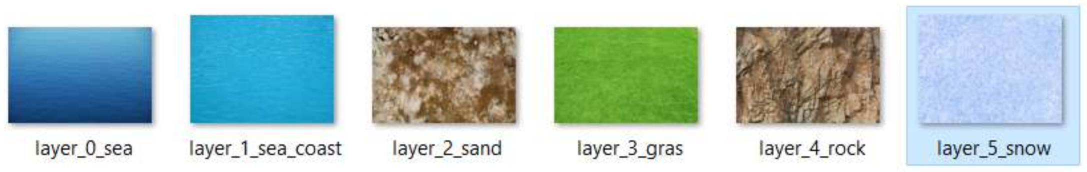


Figure 3.3: The layer set in height-mapped texture mapping

The efficient and highly parallel structure of computations carried out by graphic processing units are used to apply the height-based texture mapping method to each pixel of landscape using the shading language (i.e. the fragment shader).

The first representation of an information landscape applying height-based texture method in a fragment shader is displayed in Figure 3.4, which shows the water texture as the lowest points, with sand, grass, rock and snow textures as the height increases. What is noticeable is that information landscape does not look very realistic due to the fact that no blending is applied between the layers. To produce a landscape with a more realistic look, a simple blending method using linear interpolation is applied between the two textures over a small area around the height layer boundary. The result with applied blending can be seen in Figure 3.5. Full details of the exact implementation are provided in Section 4.3.

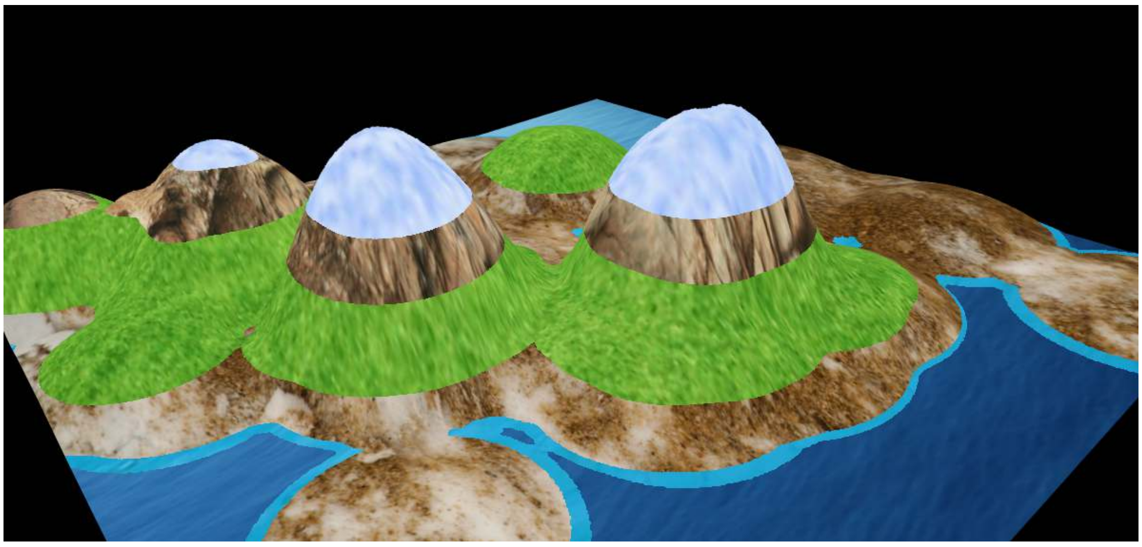


Figure 3.4: Representation of an information landscape using the height-mapped texture mapping method

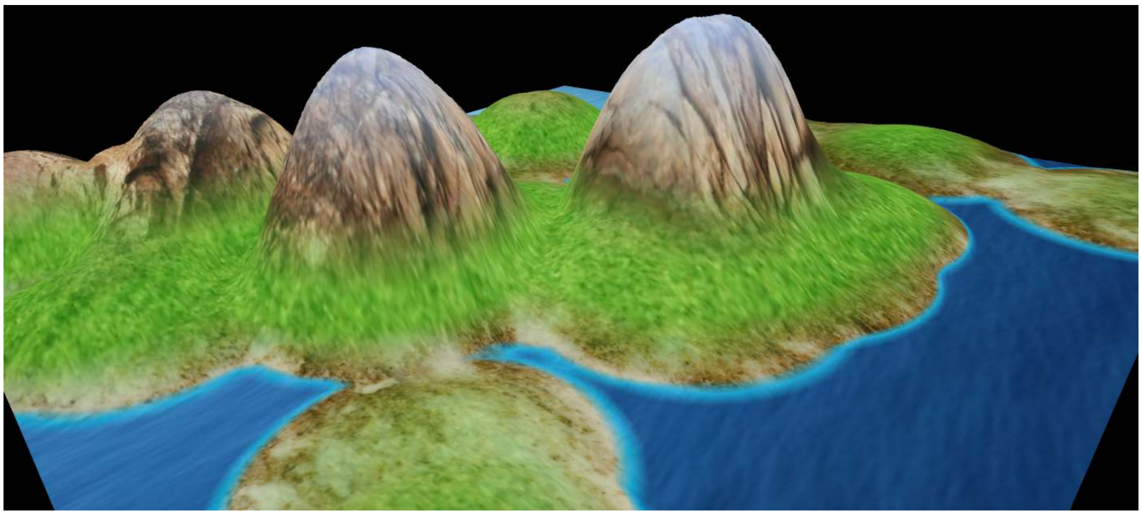


Figure 3.5: Representation of an information landscape with blending

3.3.2 Visual Concept of Documents

Two concepts have been considered for generating the visual representation of documents in the landscape. The first intention was to imitate the natural environment and represent the documents as trees in the information landscape. One of the drawbacks of this concept is that the trees will overlap due the natural way that the projection places similar documents together, such that the original shape of the tree will not be recognizable. Instead, the documents are represented as dots. Another aspect is that the visual representation of document should be easily scalable and adaptable based on the needs of the application or the type of information that the document is representing (See Figure 3.6).

3.3.3 Visual Concept of Labels

The labels describe the underliving documents of a cluster (peak). The labels are placed above the peak as legible and recognizable text. It is not enough to only have one label describing all the underlining documents (i.e. in a cluster), therefore it is reasonable to have at least two labels for describing each cluster, and it also would be sensible to place the labels stacked over each other in the 3D environment, instead of placing them in one line due to the issue of overlaps with other peak labels. Furthermore, an orientation line between the peak and stacked labels should be rendered such that users can immediately see which stacked labels belong to which peak.

Another aspect that has to be considered in the generation of visual label design is making the labels reactive, so whenever the user tilts or rotates the landscape, the labels adapt their orientation accordingly to always being readable (See Figure 3.6).

3.3.4 Visual Brush Concept

The general concept behind adding a brush is to interactively select the documents from the visual representation, whereby the formation of the visual design of the brush in a 3D environment requires the adaptation of an additional dimension of height. The visual design of the brush imitates the effect of a torchlight (i.e. a flashlight) as a circle with the ability to adapt its form to the height. Once activated, the brush follows the mouse and adapts in shape according to the elevation. Users have the possibility to set the size of the brush in the settings or through the use of a combination of keys on the keyboard. Furthermore, it would make sense to provide the possibility for changing the colour of the brush in the settings to make the brush stand out from the selected design of the information landscape. 3.6 illustrates the functionality of the brush as a selection tool. Section 4.3 describes the full implementation in detail.

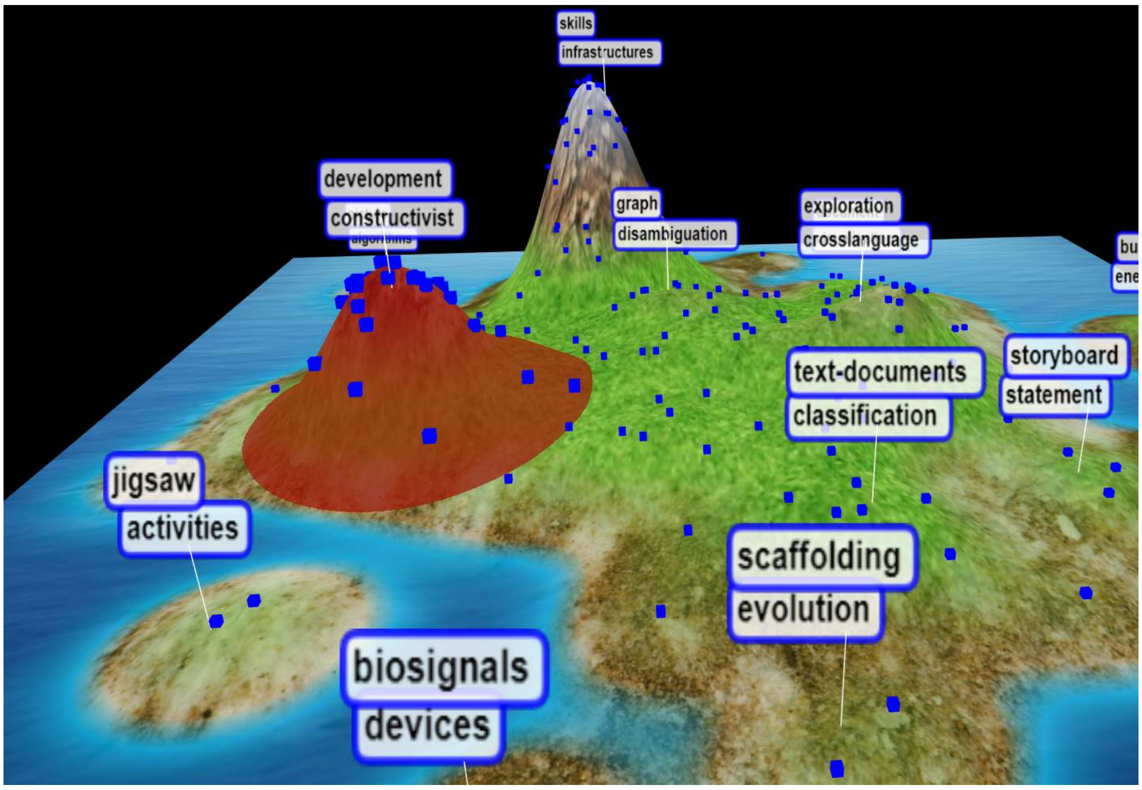


Figure 3.6: Information landscape with following rendered items: (i) brush as selection tool, (ii) documents displayed as cubes and (iii) stacked labels with orientation lines between peaks and labels.

3.3.5 Visual Concept of Morphing

This section discusses the overall concept of morphing by considering the following four concepts: (i) top-down and bottom-up approaches that make the visual design of morphing more understandable and easy to follow, (ii) introduction of an adapted method for a solution to the blindness problem and for better orientation for emerging and vanishing topics, (iii) utilization of opacity in the formation of visual itmes and (iv) analysis of the documents present in both information landscapes.

Top-down and bottom-up approach

As described in Section 3.3.1, a person interprets the height of a landscape using different sets of layers as textures. While generating the visual design of morphing, this aspect has to be considered when attempting to make the transition understandable, easy to follow and clearly visible. The morphing of a fading topic, which may cause islands and hills to disappear, will simulate the top-down approach in which the top layers of texture will disappear first and then the lower layers. In contrast, the morphing of new topics, which causes new islands to appear, will follow the bottom-up approach in which the bottom layers of texture will appear first then the upper layers.

As described in Section 3.3.1, a shading language (i.e. a fragment shader) is a method used to apply height-based texture mapping to each pixel of landscape, which means that the code of the fragment shader does not require any other extension for morphing, however the height of every pixel has to be updated accordingly during the process of morphing before the fragment shader is executed. Therefore, the shading language (i.e. the vertex shader) is used to manipulate the geometry of the landscape. Depending on the structure of the GPU, the vertex shader may be executed before the fragment shader. How often the height of every pixel has to be updated depends on the duration of morphing and the frame rate. Figure 3.7 simulates the fading of a topic as a sequence.

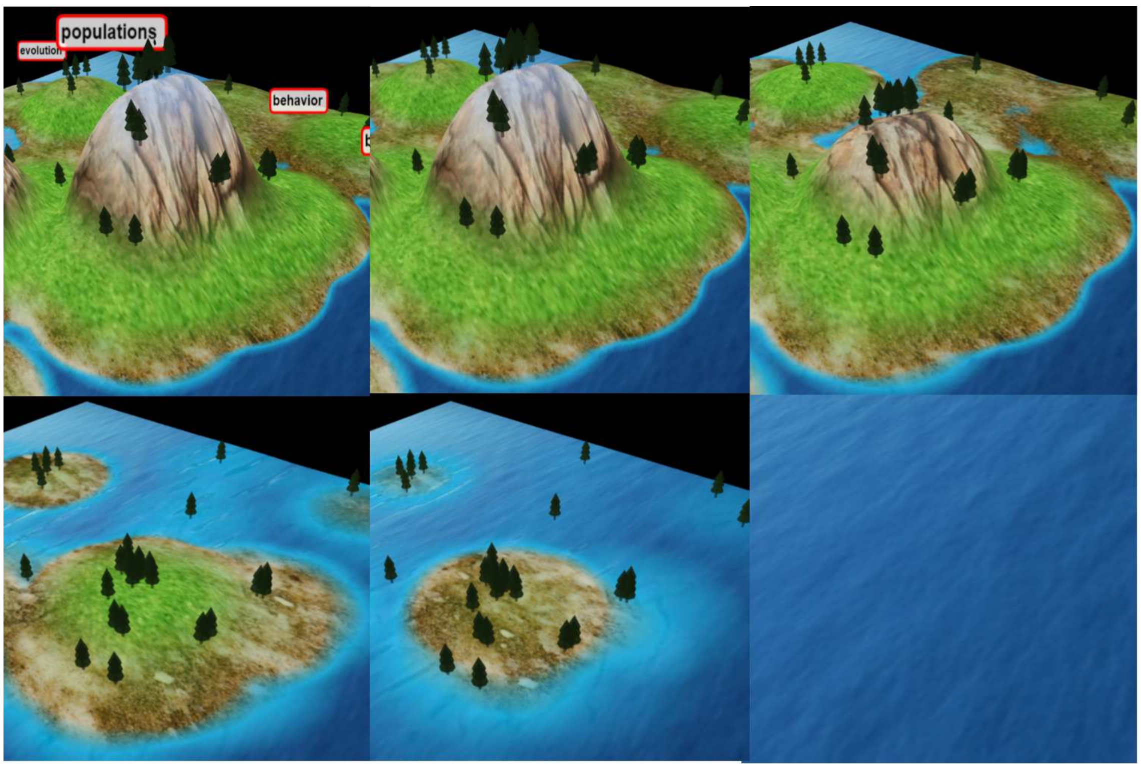


Figure 3.7: Simulation of a fading topic, which causes islands and hills to disappear

Adapted method to solve change blindness problem

Another aspect that has to be considered is that the concept of morphing can run into a change blindness problem, which makes users unable to identify the changes.

As discussed in the Section 2.1.5 Nowell proposed two criteria to solve the change blindness problem: (i) the user should be able to see where the new landscape contours will be while the old contours are still visible, and (ii) at any point during the change, the user should be able to determine which landscape contours are emerging and which are vanishing. One of the recommended solutions for change blindness was the use of a wireframe in combination with changes in colour and translucency.

A similarly method should be implemented to overcome the blindness phenomena as well as to give improve user orientation in terms of emerging and vanishing topics. The basic idea of morphing is to move the height values (z-properties) of the old landscape towards the height values of the new landscape in a seamless and smooth animation. As soon as the animation of morphing begins, a mesh containing the height map of the new landscape styled as a wireframe is added to the scene, indicating beforehand where the topics are going to emerge. The height values (z-properties) of the landscape will move towards the height values of the mesh landscape in a seamless and smooth animation in order to show the emergence of new topics as well as the fading of topics. As soon the animation ends, the added mesh wireframe will be removed from the scene, and at same time same new mesh containing the height map of the old landscape, in the form of a wireframe, is added to the scene to indicate to the users what topics have vanished.

Utilization of opacity in the design of visual concepts

A new landscape will most probably have new documents and new labels, and therefore there will be an orientation line between the label and peak. Their emergence can be shown by adding them to the scene and increasing their opacity from an initial value of zero toward an opacity value of one. Similarly, documents, labels and orientation lines that are not part of the new landscape can be shown by decreasing their opacity from initial value of one toward an opacity value of zero, i.e. they fade away. After the animation, all fading items will be entirely removed from the scene.

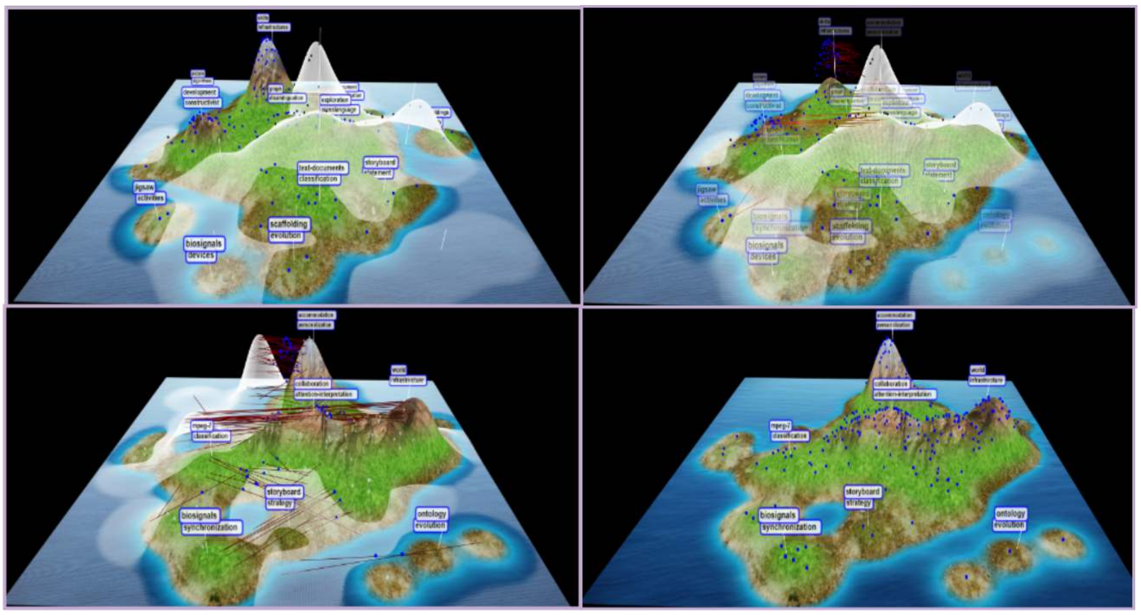


Figure 3.8: Morphing animation: (top-left) beginning phase where a new landscape is shown as a wireframe, (top-right) on-going animation phase, (bottom-left) after the animation phase, the old landscape briefly shown as a wireframe, and (bottom-right) new landscape after completion of the morphing process

Analysis of documents present in both information landscapes

The basic idea behind this concept is to show the users how the documents that are part of the old as well as of the new landscape contributed in the formation of new topics as well as in the convergence and divergence of topics. Therefore, the animation should include the movement of documents, indicated by drawing a tracing line between the old position and the new position.

Figure 3.8 simulates the morphing animation by taking into account the above-mentioned four concepts, with tracing lines visible on bottom-left.

The full description of morphing implementation is given in Section 4.3.6.

3.4 Interaction Design

The interaction design concept consists of five categories: (i) perspective category, (ii) selection category, (iii) morphing category, (iv) data exploration, filtering and navigation category and (v) setting category.

3.4.1 Perspective Category

The perspective category enables the user to view the landscape from different perspectives and offers the following possibilities for interaction:

- Zoom: The view in the information landscape can be zoomed in to view the focused area in more detail, while in contrast the view of the landscape can be zoomed out to display more area, possibly in less detail, in order to provide a view of a bigger section of the landscape and more general information. A user can zoom in and out using the mouse scroll wheel.
- Panning: The user can perform the panning operation by clicking on the right mouse button, holding it and dragging the whole information landscape in the desired orientation in order to move the point of view and reveal the information which was previously outside the displayed screen area. Another way to perform the panning operation is by using the arrow buttons on the keyboard. When performing the panning operation, the level of zoom remains unchanged.
- Rotation and tilting: The information landscape can be tilted in any direction. The operation can be used to rotate the landscape too. The user keeps the control-button pressed and drags the area in the direction in which the landscape is to be tilted or rotated. The effect of rotation or tilting is based on the distance covered from the start of dragging and to end of dragging.

3.4.2 Selection Category

The selection category enables the user to choose one or multiple displayed items so that further operations can be performed on the selected data. By hovering the mouse over the document, a tool tip is shown to provide the user with the following details: (i) title of document, (ii) author of the document and (iii) the name of the conference. The tool tip fades away once the user moves the mouse out of the document. There are four ways to select one or multiple documents.

- The user can select a document just by clicking on it, and the document can be deselected by clicking on it once again.
- The second method of selection for documents is to make use of the rectangle selection tool. To do this, users have to click the shift button first,

hold it down and click the right mouse button, then hold and drag the mouse to the position of the diagonally opposite corner. Users will immediately be shown the documents within the selection tool as highlighted. Once the user releases the mouse, the documents within the selection tool will be selected and the rectangular selection tool will vanish.

- The third way to interactively select the documents from the information landscape is by using the brush selection tool. By clicking the “brush” button in the menu bar, the brush-based selection tool will be activated. Once the brush tool has been activated, it will follow the mouse and adapt its shape according to the elevation. The documents that are within the brush will be highlighted, and by clicking on the brush documents within the brush, they will be selected. Similarly, by clicking once again on the brush, a deselecting of documents will be performed. Users have the possibility to set the size of the brush by using the slider in the settings menu or by using the mouse wheel while holding down the shift key. The brush-based selection tool can be deactivated by clicking on “brush” in the menu bar or by pressing the escape key on the keyboard.
- The fourth way to select documents is with a click on a label of a peak, and the documents that contain the label will be selected. By clicking on the labels again, the associated documents will be deselected. This functionality can be disabled in the menu bar. The labels are designed to be reactive by adapting their respective positions and being always readable for the users whenever the landscape is rotated, panned or tilted.

All the documents can be removed from the selection by clicking in an empty space of the information landscape.

3.4.3 Morphing Category

By clicking on the button “open” in the menu bar, a pop-up dialog appears to enable users to select the temporal morphing data, that should be loaded in the application. After confirming the dialog window, the morphing data are read. The morphing data of a period is represented by a set of actions in the form of an executable button. By clicking on the executable button, the current state of the information landscape will be morphed into a new state, belonging to a period associated with the executable button, using smooth animation.

3.4.4 Data Exploration, Filtering and Navigation Category

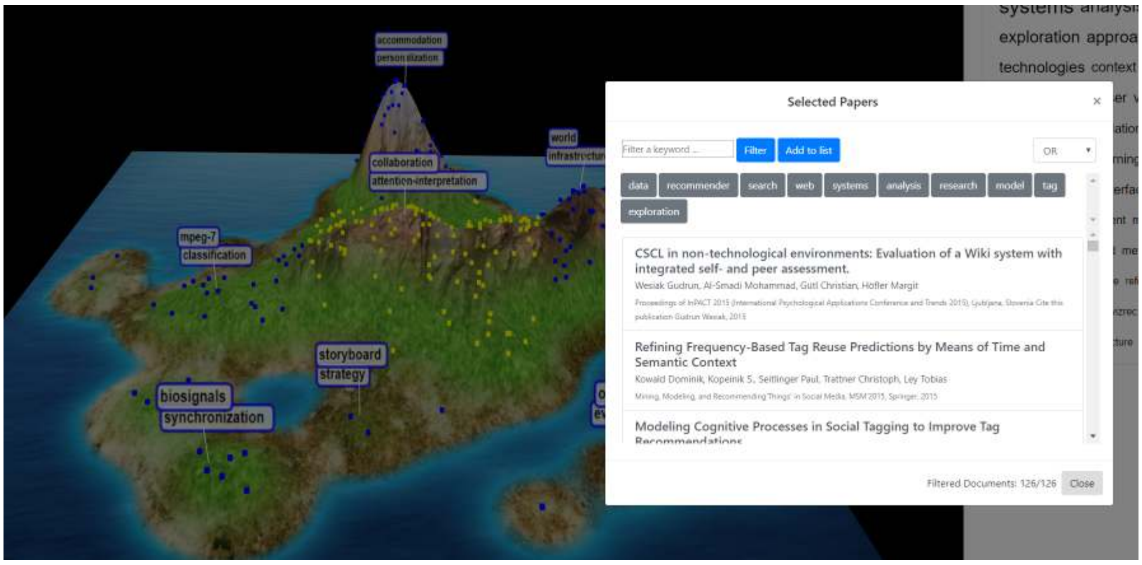
A user can explore the data by selecting the document as described in Section (3.4.2). The tag cloud is used to provide an overview of the selected documents by displaying the keywords in different sizes based on their relevancy in the selected content. Furthermore, an overview of the whole dataset is provided when nothing has been selected. Additionally, by hovering the mouse over a keyword in the tag cloud, the documents that contain the keyword are highlighted in the landscape.

Another way to explore and filter the documents is to select the documents first and then click the right mouse button in the landscape. A draggable pop-up window is displayed which contains the following displayed elements (see Figure 3.9): (i) the selected documents as a list, (ii) most relevant keywords of the selected documents, (iii) a selection box containing two operators (OR and AND) and (iv) a search field to filter the documents. A user has the possibility to filter the documents by selecting multiple keywords. Depending on which operator is selected in the selection box, the documents that contain all the selected keywords or any of the selected keywords are highlighted in the landscape. Additionally, the search field includes automatic suggestion functionality. As soon as the user types in the search field, the keywords that are parts of the selected corpus (documents) are suggested to the users. Once the user selects one of the suggested keywords, the corresponding documents that contain the selected keyword will be filtered.

3.4.5 Setting a Category

The following settings are provided to the users to adjust the information landscape according to their desired setup.

- Changing the size of the brush-selection tool using the slider as well as changing the colour of the brush by clicking on the colour picker tool and selecting a colour from the palette.
- Setting the timespan for the smooth animation of morphing using the slider
- Adjusting the brightness and opacity of information landscape using sliders
- Changing the design of the information landscape by selecting the design from a selection box, available design are: island as default, earthy and greeny.



CSCL in non-technological environments: Evaluation of a Wiki system with integrated self- and peer assessment.
Wesiak Gudrun, Al-Smadi Mohammad, Gütl Christian, Höfler Margit
Proceedings of InPACT 2015 (International Psychological Applications Conference and Trends 2015), Ljubljana, Slovenia Cite this publication Gudrun Wesiak, 2015

Refining Frequency-Based Tag Reuse Predictions by Means of Time and Semantic Context
Kowald Dominik, Kopeinik S., Seitlinger Paul, Trattner Christoph, Ley Tobias
Mining, Modeling, and Recommending 'Things' in Social Media, MSM 2015, Springer, 2015

Modeling Cognitive Processes in Social Tagging to Improve Tag Recommendations

Filtered Documents: 126/126

Figure 3.9: A draggable pop-up window with the following elements displayed (i) the selected documents as a list, (ii) the most relevant keywords of the selected documents, (iii) a selection box containing two operators (OR and AND) and (iv) a search field to filter the documents

- Setting the type of density of height scaling from a selection box, wherein the following types of scales are provided to users: linear, square root, squared and logarithmic
- Enabling or disabling the selection of labels in the information landscape by selecting or deselecting a checkbox

Chapter 4

Implementation

This chapter presents the implementation of the interactive scalable information landscape visualization in detail. The first section provides an overview of the used programming languages and technologies, including their general purpose, advantages and drawbacks. The second section introduces the software architecture of the application. The last section presents the details of the core parts of the implementation.

4.1 Programming Languages and Technologies

HTML5 and CSS3 have been used to structure and design the layout of the information landscape on a website. To render the information landscape as an interactive 3D and 2D image within a browser, the JavaScript WebGL API has been used. Furthermore, the shading languages of the GPU (vertex shader and fragment shader) have been employed to utilize the efficient parallel computing of GPUs for creating information landscapes that mirror a realistic natural environment.

The following scripting languages and JavaScript libraries have been used:

- ECMASCRIPT(ES6) [1], a scripting language specification standardized by Ecma International
- Jquery library [2]

[1] `https://www.ecma-international.org/`
[2] `https://jquery.com`

- D3.js visualzation library[3]
- Three.js 3D rendering library [4]

This section moreover provides a short overview of the technologies used, which includes WebGL, GPU and Three.js library.

4.1.1 WebGL

WebGL is a fully standardized JavaScript API based on OpenGL ES, which provides access to hardware-accelerated 3D rendering of interactive 3D and 2D graphics within any compatible web browser, moreover without the use of any plug-ins. The advantages of WebGL in terms of information landscapes are the following:

- Requires only an height map to render an information landscape, in contrast to a SVG solution as described in Section 2.1.3, which requires two additional steps for properly rendering information landscapes and applying smooth morphing animation.
- Offers various interactions like zoom, rotation and tilt for viewing data from different perspectives in 3D.
- Provides possibilities to run custom written shader code on graphics hardware in order to create landscapes that match the realistic natural environment as well as smooth 3D accelerated morphing animation of landscape geometry
- Allows applying textures to 3D objects

4.1.2 GPU

GPUs (i.e. graphics processing units) are more efficient at manipulating computer graphics due to the highly parallel structure of their computations. Figure 4.1 shows the basic structure of a GPU. Shaders are pieces of code that run on the GPU and also reside within the unit. Two of these shaders are the vertex shader and the fragment shader. The vertex shader manipulates the received attribute data and provides the output to the fragment shader, while the fragment shader assigns a colour to every fragment. The following variables are available for shaders:

[3]`https://d3js.org`
[4]`https://threejs.org`

- Uniforms: Uniforms variables are sent to vertex shaders as well as to fragment shaders. The values of uniform variables stays the same across the entire frame.
- Attributes: Attributes are values that are provided per vertex and are only available for the vertex shader.
- Varyings: Varying variables are declared and manipulated in the vertex shader; they can be shared with the fragment shader, whereby the type and name of the varying variables should be the same in both shaders
- Textures: Textures are only received by fragment shaders

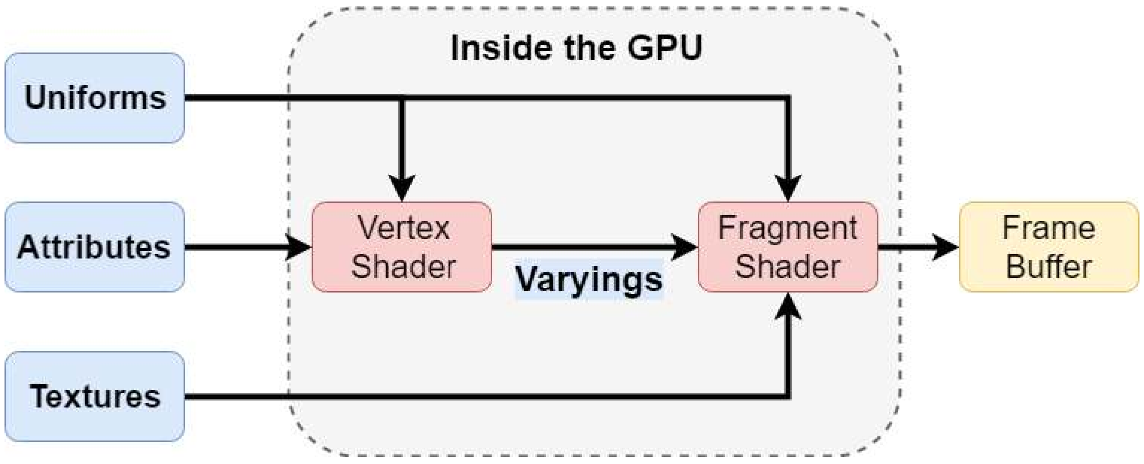


Figure 4.1: Structure of a GPU

4.1.3 Three.js

Three.js is a popular JavaScript framework that uses WebGL for creating and displaying interactive animated 3D computer graphics within the web browser. Every application that is based on the three.js library must have following basic components:

- Scene: a container for storing and keeping track of all the rendered objects
- Camera: defines the view based on its position when the scene is rendered

- Renderer: takes the camera angle into account when displaying the scene in the browser

The optional component of three.js called OrbitControls allows the use of the mouse to easily move, pan and zoom around the scene.

The reasons why Three.js library was chosen is because the library is easy to use, it is very stable, it has a large community, it allows one to run custom shading code on hardware graphics processors and it has various builds in terms of features, especially for constructing interactions.

In three.js application, a global function window.requestAnimationFrame is used to perform any updates of the scene before a new frame is rendered. The exact definition of the function window.requestAnimationFrame is as follows: "The window.requestAnimationFrame method tells the browser that you wish to perform an animation and requests that the browser call a specified function to update an animation before the next repaint. The method takes a callback as an argument to be invoked before the repaint"[5].

4.2 Software Architecture of the Application

The overall structure of the workflow can be divided into three parts: (i) initialization stage, (ii) rendering stage and (iii) updating stage

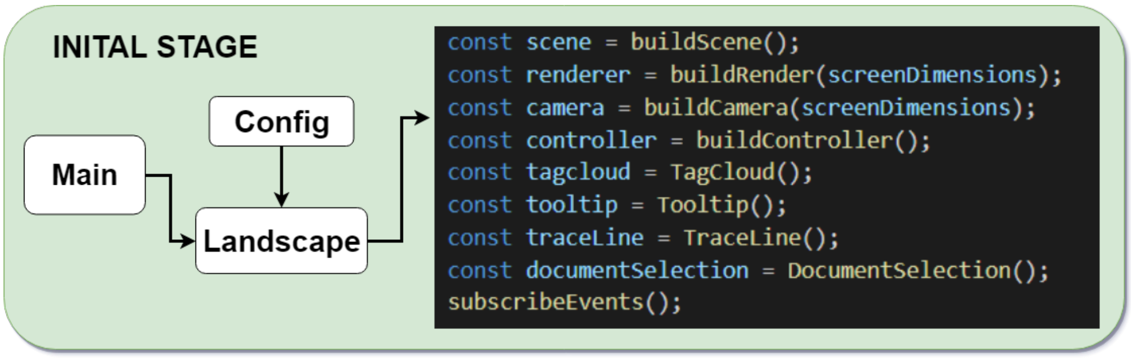
4.2.1 Initialization Stage

In the initialization stage, the application initializes the module landscape, the purpose of which is to handle the whole application. The module landscape creates all the required basic components of three.js with initial configurations, which includes scene, renderer, camera and controller, whereby the component controller is used as an optional component for supporting the formation in terms of functionality, e.g. moving, panning and zooming around the scene. Furthermore, the user interface is set up for handling the events related to the settings. Additionally, five others modules are created in the initial stage: (i) the "TagCloud" to give an overview of the selected documents, (ii) the "ToolTip" to show the basic information of a selected document in a tool tip, (iii) the "TraceLine" to help the users understand how the documents in the initial state of the information landscape have moved to another position in the next state of the information landscape, which is shown by drawing

[5]https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame

tracing lines between the two positions during the morphing phase, (iv) the "DocumentSelection" to display the selected documents in a pop-up window as a list, thus providing for exploration, filtering and navigation, (v) the "RectangleSelection" to enable the free rectangle-based selection tool.

After the initialization stage, the application is set up to render the information landscape with the extracted data. Figure 4.2 illustrates all the processing sequences of the initial stage.



```
const scene = buildScene();
const renderer = buildRender(screenDimensions);
const camera = buildCamera(screenDimensions);
const controller = buildController();
const tagcloud = TagCloud();
const tooltip = Tooltip();
const traceLine = TraceLine();
const documentSelection = DocumentSelection();
subscribeEvents();
```

Figure 4.2: Software architecture of the application with initialization stage representation

4.2.2 Rendering stage

As described in Section 3.2.1, the text-processing pipeline and layout computation delivers the following data: (i) the height map, (ii) the location of documents, (iii) the peak-related information (i.e. locations, labels), (iv) a list of keywords in the document, including their frequency. The rendering stage converts the extracted data into 3D objects, positions them according to their computed locations and adds them into the scene, whereby the 3D objects, which model the information landscape, are linked with shader code in order to execute their processing on the graphics hardware (GPU). Furthermore, the list of keywords and their frequency is used to build an interactive tag cloud that gives an overview of the selected dataset, whereas an overview of the entire dataset is given when nothing has been selected.

4.2.3 Updating stage

Section 4.1.3 described the global method window.requestAnimationFrame. It takes a callback as an argument to be invoked before a new frame is rendered.

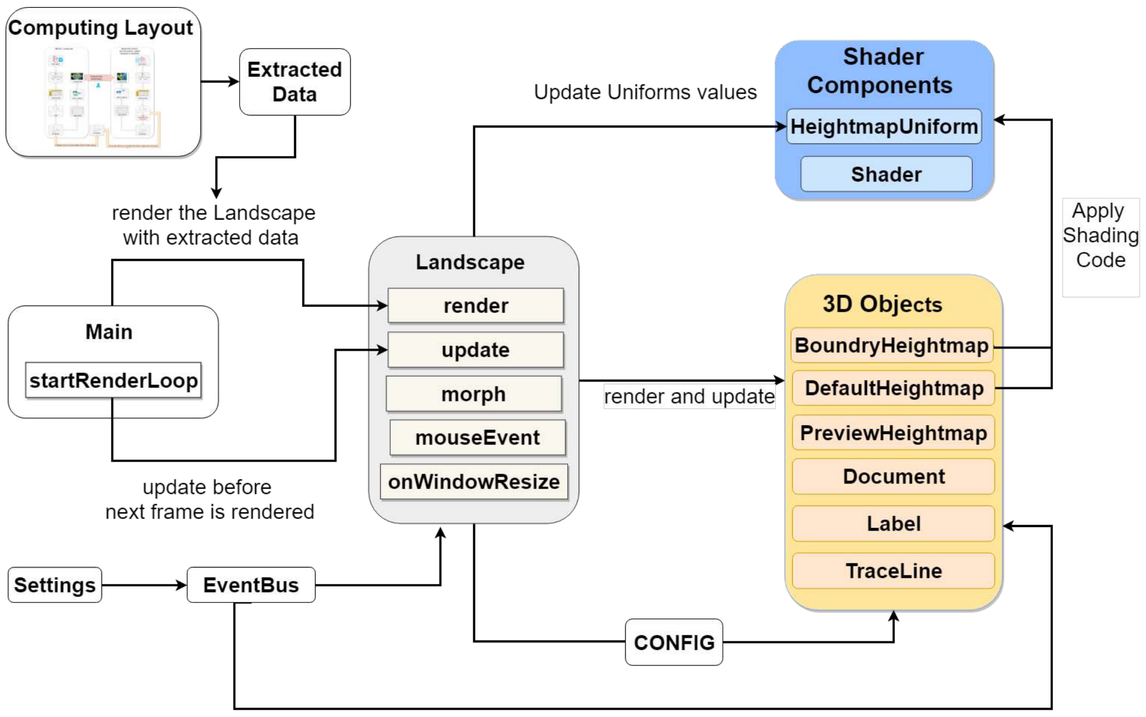


Figure 4.3: The overall software architecture of the application

In this application, the startRenderLoop method of the main module is used as callback argument for the requestinAnimationFrame method, which calls the update method of the landscape module to update all the changes before a new frame is rendered. How often the starRenderLoop is to be called depends on the frame rate of the browser and the computer, but typically it is 60fps. Synced by the GPU, the animation should be very smooth. Figure 4.3 depicts the overall workflow of the updating stage.

4.3 Core Details of the Implementation

This section describes the core parts of the implementation in detail. It begins by introducing all the steps that are required to create a 3D object successfully using the three.js library. Then the basic concept of creating information landscapes in 3D using the three.js library is provided. Furthermore, the most relevant shader-based implementations are described in detail, which includes the setup of information landscapes for shaders, the scaling and morphing of information landscapes and the implementation of the height-based texture mapping method. Moreover, the details about selecting data of interest and realizing perspective-based interactions are given in the final part of this section.

4.3.1 Steps for creating 3D objects using three.js

To create a 3D object in the scene of a three.js library, the following four steps are required:

- Create the geometry of the 3D object
- Create a material to define the look of the 3D object
- Combine the geometry and material into a mesh
- Add the mesh to the scene

4.3.2 Creating 3D information landscapes using three.js

The three.js library provides some geometries that result in a two-dimensional mesh. PlaneGeometry is one of them; it can be used to create a two-dimensional rectangle. The creation of PlaneGeometry requires four parameters: the first parameter is the width, the second parameter is the height, and third and the fourth parameters are used to divide the width and the height into segments. The size of the segments has to be equal to the size of the extracted height map.

To make the information landscape appear as 3D, two PlaneGeometries are created. In one PlaneGeometry, the z-property of the vertices are filled with computed height map values, whereas in another PlaneGeometry, z-properties of the boundaries (the first row, the first column, the last row and the last column) are filled with corresponding values from the height map, whileother z-properties of vertices are filled with zeros. With the exception of the geometries, every other configuration must be kept same for both PlaneGeometries.

Once both PlaneGeometries are added as meshes in the scene, the meshes appear as a combined 3D information landscape in the final result. The figure 4.4 below explains the approach visually.

PlaneBufferGeometry is a low memory alternative for PlaneGeometry. It is much faster and requires less memory. The geometry is more flexible and allows one to perform certain operations with greater ease. The PlaneBufferGeometry is ideal for running the related shader code on the GPU.

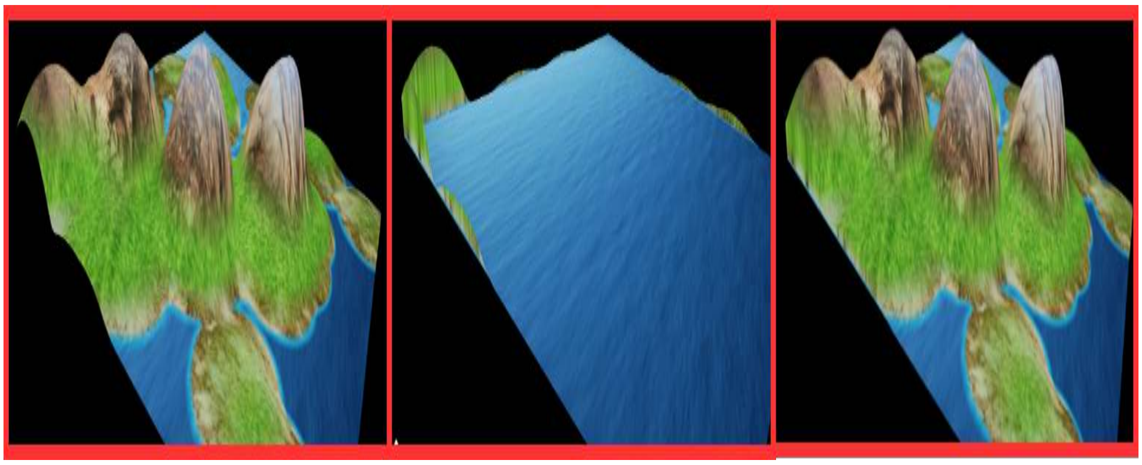


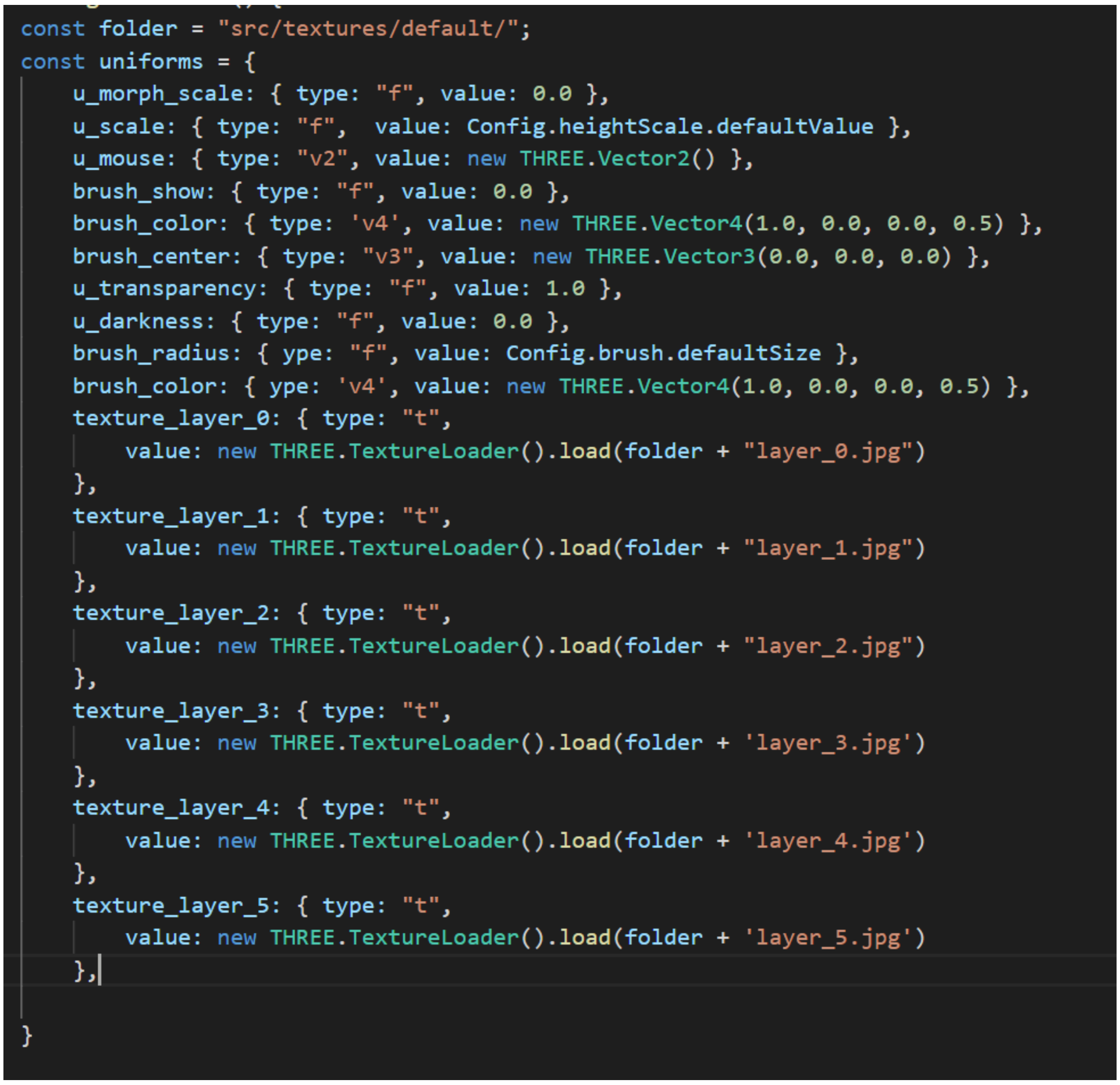
Figure 4.4: The first picture (left) shows the result of PlaneGeometry containing the whole computed height map, the second picture (middle) shows the PlaneGeometry containing computed height map values on the boundaries, the last picture (right) displays both PlanGeometries combined.

4.3.3 Information landscape setup for shaders

The height map contains a data set that is not normalized; it has to be normalized between 0 and 1 in order to ease the process of scaling the whole information landscape between 0 and the optimal maximum height. The following procedure is required to set up the information landscape as a mesh for shaders: Figure 4.5 shows all the uniforms (i.e. uniform values) that are passed to the shader.

- The first step is to create a geometry of the information landscape. The three.js library provides a suitable geometry object called PlaneBufferGeometry – the data structure more directly maps to how data is used in the shader program (i.e. a vertex shader). A BufferAttribute of a position called by PlaneBufferGeometry is created with z-properties that are filled with normalized height map values. Additionally, with the same size, a BufferAttribute object named displacement is created with zeros values and attached to the PlaneBufferGeometry – this is used for morphing the landscapes with vertex shaders. The BufferAttribue object stores the data for an attribute and can be used directly in the shader code.

- In the next step, material that defines the look of an information landscape is created. The three.js library provides a custom shader called ShaderMaterial, which runs on the GPU. ShaderMaterial requires the



```
const folder = "src/textures/default/";
const uniforms = {
    u_morph_scale: { type: "f", value: 0.0 },
    u_scale: { type: "f",  value: Config.heightScale.defaultValue },
    u_mouse: { type: "v2", value: new THREE.Vector2() },
    brush_show: { type: "f", value: 0.0 },
    brush_color: { type: 'v4', value: new THREE.Vector4(1.0, 0.0, 0.0, 0.5) },
    brush_center: { type: "v3", value: new THREE.Vector3(0.0, 0.0, 0.0) },
    u_transparency: { type: "f", value: 1.0 },
    u_darkness: { type: "f", value: 0.0 },
    brush_radius: { ype: "f", value: Config.brush.defaultSize },
    brush_color: { ype: 'v4', value: new THREE.Vector4(1.0, 0.0, 0.0, 0.5) },
    texture_layer_0: { type: "t",
        value: new THREE.TextureLoader().load(folder + "layer_0.jpg")
    },
    texture_layer_1: { type: "t",
        value: new THREE.TextureLoader().load(folder + "layer_1.jpg")
    },
    texture_layer_2: { type: "t",
        value: new THREE.TextureLoader().load(folder + "layer_2.jpg")
    },
    texture_layer_3: { type: "t",
        value: new THREE.TextureLoader().load(folder + 'layer_3.jpg')
    },
    texture_layer_4: { type: "t",
        value: new THREE.TextureLoader().load(folder + 'layer_4.jpg')
    },
    texture_layer_5: { type: "t",
        value: new THREE.TextureLoader().load(folder + 'layer_5.jpg')
    },

}
```

Figure 4.5: Representation of all uniforms passed to the shader

uniform values, the code of vertex shader and the code of fragement shader as inputs. All the uniform values that are relevant to both the vertex shader and the fragment shader are passed once in ShaderMaterial. The uniform values are used directly in the shader code. Hence, if a uniform value changes, so does the overall calculation in the shader. This property can be used to include user input and perform specific behaviour in the shader.

- Next, the PlaneBufferGeometry and the ShaderMaterial are combined into a mesh.

- In the last step, the combined mesh is added into a scene.

The DefaultHeightmap and BoundryHeightmap modules implement the above-mentioned procedure to create two meshes containing different geometries that incorporate the previously defined criteria for giving the landscape a 3D look. The DefaulHeightmap creates a mesh containing the entire extracted height map, while the BoundryHeightmap module creates a mesh containing only the boundary values from the extracted height map.

Both meshes share the same uniform values stored in the HeighmapUniform module, therefore a change in a uniform value will affect the both meshes in the shader code. Nonetheless, the shaders of both meshes will work independently even if they share the same piece of code.

4.3.4 Vertex shader: scale the information landscape

A range slider is used to provide the user with the option to set the maximum height of the information landscape. As the user moves the slider with the mouse, the information landscape is rendered with a new scaled geometry between 0 and the maximum height value. The following actions are executed once the range of the mouse slider changes:

- The uniform value "u_scale", which is stored in HeightmapUniform and is linked with the shader code, is updated with the new maximum height.
- A signal is sent so that the height of all documents and all labels can be scaled accordingly.

The GPU hardware runs multiple instances of a vertex shader simultaneously, and the code in a vertex shader is executed once per vertex. As described in the previous section, two attributes named position and displacement are passed to the vertex shader. They are used along with the uniform value "u_scale" to perform scaling by executing the following steps:

- In the vertex shader, a varying variable named "vPosition" is created, which is shared with the fragment shader such that the height-based texture mapping method can be applied to each pixel by taking the new scaled height into consideration.
- The attribute position, which contains the extracted height map in z-property, is assigned to the varying variable "vPosition".

The z-property of variable "vPosition" is multiplied by uniform value "u_scale" and the result is stored in z-property of "vPosition". This snippet of code includes all the steps for scaling.

```
attribute float displacement;
uniform float u_scale;
varying vec3 vPosition;

void main( void ) {
   vPosition = position;
   vPosition.z = vPosition.z*u_scale;
   .
   .
   gl_Position = projectionMatrix * modelViewMatrix * vec4(vPosition,1.0);
}
```

4.3.5 Fragment shader: implementation of height-based texture mapping

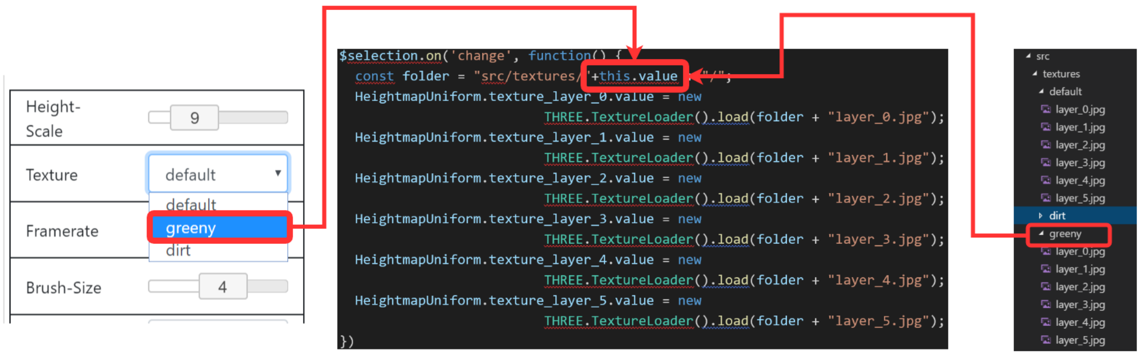
As discussed in Section 3.3.1, the human interpretation of a landscape in the form of a visual landscape design has to be considered to make the information landscape more intuitively understandable. A human sees a landscape as a set of varied layers, where a water texture defines the lowest points, with sand, grass, rock and snow textures as the height increases.

As previously described, a varying variable named "vPosition" is created in the vertex shader, which is shared with the fragment shader such that the height-based texture mapping method can be applied to each pixel.

To implement the height-based texture mapping, the varying variable has to be declared in the fragment shader with the same name and type as was used in the vertex shader. Furthermore, all the textures have to be loaded and have to be made available in the fragment shader. Figure 4.5 shows how the textures are loaded as uniform values. In the fragment shader, the uniform sampler2D are declared in order to reference the textures. Moreover, a build in function texture2D is used to retrieve the colour composition of each referenced texture. A function called mixNextLayer has been implemented inside the fragment shader code, which takes five parameters and returns a colour based on the height of the pixel. The first two parameters are the colours of the layers (textures), whereby the second parameter is the colour of the next layer (texture), the third parameter is the height of the pixel ("vPosition"), the fourth and fifth parameters are the minimum and maximum height levels, between which the two colours (textures) are linearly interpolated. The function mixNextLayer works as follow:

- If the height of the pixel is lower than or equal to the minimum height level of the blending, then the first parameter is returned as colour.
- Otherwise, if the height of the pixel is greater than or equal to the maximum height level of the blending, then the second parameter is returned as colour.
- If, however the height of the pixel is between the minimum and maximum level of the blending, then two colours are linearly interpolated between the minimum and maximum level of the blending.

Inside the fragment shader code, the functionality of the function mixNextLayer is utilized so that the return value of the function is used as the first parameter of next mixNextLayer function and the second parameter is the colour of the next level above.



Height-Scale 9

Texture default

Framerate

Brush-Size 4

```
$selection.on('change', function() {
  const folder = "src/textures/"+this.value +"/";
  HeightmapUniform.texture_layer_0.value = new
                            THREE.TextureLoader().load(folder + "layer_0.jpg");
  HeightmapUniform.texture_layer_1.value = new
                            THREE.TextureLoader().load(folder + "layer_1.jpg");
  HeightmapUniform.texture_layer_2.value = new
                            THREE.TextureLoader().load(folder + "layer_2.jpg");
  HeightmapUniform.texture_layer_3.value = new
                            THREE.TextureLoader().load(folder + "layer_3.jpg");
  HeightmapUniform.texture_layer_4.value = new
                            THREE.TextureLoader().load(folder + "layer_4.jpg");
  HeightmapUniform.texture_layer_5.value = new
                            THREE.TextureLoader().load(folder + "layer_5.jpg");
})
```

```
src
  textures
    default
      layer_0.jpg
      layer_1.jpg
      layer_2.jpg
      layer_3.jpg
      layer_4.jpg
      layer_5.jpg
    dirt
    greeny
      layer_0.jpg
      layer_1.jpg
      layer_2.jpg
      layer_3.jpg
      layer_4.jpg
      layer_5.jpg
```

Figure 4.6: Process of applying a new design to the information landscape

Additionally, the user is provided with the option to select the design of the landscape from a selection box. The overall design is based on rendering the information landscape with a different set of textures, meaning that the textures of the new design have to be reloaded using the uniform values. Figure 4.6 illustrates the overall workflow. First off, the user selects a new design from a selection box, and consequently a corresponding event function is called that uses the name of the new design to build the path to the container. The container holds all the textures of the news design. The corresponding uniforms (i.e. uniform values) are updated with the textures of the new design.

Once the user sets the design in the selection box, the information landscape will immediately be rendered smoothly in the browser with the new design. Even if the morphing is running and the user changes the design of the information landscape, the morphing process is not hindered at all and new design is applied with immediate effect such that rest of the morphing process will be continued with the new design.

Furthermore, for scaling the landscape or for building the morphing animation, nothing extraordinary has to be added in the fragment shader because the implemented height-based texture mapping method assigns the colour to a pixel based on the height. The vertex shader, which manipulates the height, is executed before the fragment shader

4.3.6 Vertex shader: morphing the information landscape

The basic idea of morphing is to move the height values (z-properties) of the old landscape towards the height values of the new landscape in a seamless and smooth animation. The process of generating a new information landscape, which involves dynamic changes, is discussed in Section 2.1.4. To make the animation smooth, the animation is rendered for every frame over a timespan of 10 seconds. The timespan can be changed using the slider in the settings menu.

All the steps of the animation are executed inside the ‘update’ method of the landscape because ‘update’ is executed before a frame is rendered. Before the animation starts or the first frame is rendered, the ‘morph’ method of the landscape module is called on to execute the following steps:

- The new documents, new labels and the orientation lines between the labels of the new information landscape are added to the scene and their opacity is set to zero.
- A mesh containing a height map of the new information landscape is presented as a wireframe and added to the scene.
- The ‘morph’ method of the DefaultHeightmap module and the BoundryHeightmap module is called to calculate the distances between the height values of the new landscape and the height values of the old landscape by subtracting the height values of the old landscape from the height values of the new landscape. The distances are stored in a BufferAttribue object named displacement, which is directly linked with the vertex shader.
- A uniform value "u_morph_scale", which is stored in HeightmapUniform and is linked with shader code, is updated to zero.

For every frame of the animation, the following steps are executed:

- The opacity of new documents, new labels and orientation lines of the new landscape changes incrementally towards one.
- The opacity of documents, labels and orientation lines, which are not part of old landscape, changes towards zero.
- • The uniform “u_morph_scale” is updated with a new scale as shown in the snippet of code below.

```
let ratio = 1 / Config.frameRate / 5;
HeightmapUniform.u_morph_scale.value = HeightmapUniform.u_morph_scale.value +
     ratio;
```

After the animation is finished, the following steps are executed:

- The uniform value “u_morph_scale” is set to zero.
- The documents, labels and orientation lines that are not part of the new landscape are removed from the scene.
- The mesh containing the height map of new information landscape that is displayed as a wireframe is removed from the scene, however the mesh containing the height map from the old information landscape is given as a wireframe and is added to the scene.

The extended code snippet of the vertex shader is shown below:

```
attribute float displacement;
uniform float u_scale;
uniform float u_morph_scale;
varying vec3 vPosition;

void main( void ) {
  vPosition = position;
  vPosition.z=vPosition.z*u_scale+(displacement*u_morph_scale*u_scale);
   .
   .
  gl_Position = projectionMatrix * modelViewMatrix*vec4(vPosition,1.0);
}
```

The equation “(displacement*u_morph_scale*u_scale)” sets how much the z-property of the landscape changes towards the z-property of new landscape in terms of elapsed frames of the animation process.

4.3.7 Fragement shader: implementation of brush selection tool

As described in Section 3.3.3, the general concept behind brushing is to interactively select the documents from the visual representation. The implementation of the brush selection tool is based on the possible ways in which the user can interact with the brush. The following interaction possibilities are provided to the user:

- Activate and deactivate the brush selection tool
- Select the size of the brush
- Choose a colour for the brush

Once activated, the brush follows the mouse and selects the documents that are placed inside the brush. The visual design of brush in a 3D environment imitates the effect of torchlight, i.e. a circle with the ability to adapt its shape to an additional dimension of height. To achieve the desired visual design, the fragment shader is utilized to render every pixel of the brush.

As previously described, the uniforms (i.e. uniform values) can be used to include user input in the shader code in order to perform certain behaviour. The uniforms are set as follows, depending on user actions.

- If the user activates the brush for a selection, the corresponding uniform value is set to 1.0 float value as follows:

  HeightmapUniform.brush_show.value = 1.0

- On other hand, if the brush is deactivated, the corresponding uniform value is set to 0.0 float value:

  HeightmapUniform.brush_show.value = 0.0;

- The new selected radius size of the brush is set like this:

  HeightmapUniform.brush_radius.value = radius;

- The chosen colour for the brush is set in such a way:

  HeightmapUniform.brush_color.value = newTHREE.Vector4(rgb,0.5);

- If the mouse is activated, the class Raycaster of three.js is used to get the exact world coordinates of the mouse position of a moving mouse in order to set the uniform value as follows:

HeightmapUniform.brush_center.value.x = vector.x;

HeightmapUniform.brush_center.value.y = vector.y;

HeightmapUniform.brush_center.value.z = vector.z

The code snippet of the fragment shader below checks if the pixel is inside the circle by incorporating the centre of the brush and the radius into the equation.

If a pixel is inside the brush and the brush is activated, the colour of the brush is added to the pixel and normalized afterwards.

```

uniform float brush_show;
uniform vec4 brush_color;
uniform vec3 brush_center;
uniform float brush_radius;
varying vec3 vPosition;


void main(){
  .
  .
  .
  if(brush_show == 1.0 && isInsideCircle(vPosition, brush_center,
      brush_radius) >0.0) {
      gl_FragColor.r += brush_color.r;
      gl_FragColor.b += brush_color.b;
      gl_FragColor.g += brush_color.g;
      gl_FragColor.a += 0.9;
      gl_FragColor = normalize(gl_FragColor);
   }
}
```

4.3.8 Implementation of rectangle-based selection tool

D3.js library has been used to implement the rectangle-based selection tool based on SVG, HTML5 and CSS standards. An additional div container, the only purpose of which is to hold the rectangle-based selection tool, is created and attached to the body of the application. The container is overlayed on canvas that contains the information landscape rendered in WebGL. The size of the container is always kept the same as the size of the canvas. The CSS of the container is set to “pointer-events: none”, which will have the effect that the container will not be able to catch any events at all; instead, all events will fall through to the element below it (i.e. canvas).

The rectangle element of SVG is defined by position, width, and height. The basic idea is to update these properties based on the mouse position of the received events by the canvas. The received position of the mouse by the canvas is in world coordinates, which has to be converted into screen coordinates before the position or width and height of the rectangle is calculated and updated.

4.3.9 Implementation of document and label selection

To select an object using the mouse, the following takes place:, the Projector and Raycaster classes of the three.js library are used together to determine whether a document or label have been clicked. Based on the position of the mouse click, a vector is created and its position is converted into world coordinates afterwards. The Raycaster class is used to send out a ray into the world from the position of the mouse click. Another Raycaster class is used to determine whether this ray hits any of the supplied objects. The result contains of list of hit objects, facilitating further operations.

4.3.10 Implementation of building perspective interactions

OrbitControls, a built-in functionality provided by three.js, has been used to control the camera throughout the scene. It allows the use of a mouse to move, pan and zoom around the scene. To make OrbitControls work properly, the position of the camera has to be updated. This update happens in the update method of the landscape module before a new frame is rendered. A three.js object called clock is used to calculate the elapsed time. To update the position of the camera, the OrbitControls.update() function is called.

4.4 Summary

This chapter described the implementation of the interactive and scalable information landscape in detail. The first section introduced the programming languages and technologies used as well as their general purpose, their advantages and drawbacks. Furthermore, the second section gave a broad overview of the software architecture of the application. The last section discussed the details of a selected core part of the implementation.

Chapter 5

Case Study

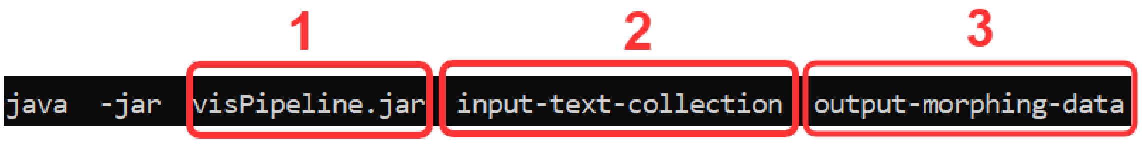
This chapter presents the use of the developed web-based application that visually renders an interactive information landscape. The chapter introduces the application using three use case scenarios. The first scenario begins by describing the steps needed to run the pipeline for generating the morphing data from a collection of text documents, followed by the steps needed to load the morphing data in the application. The second scenario introduces the individual selection of documents as well as selection based on content and position. The last scenario describes how to choose a suitable design, height and perspective for the information landscape before a demonstration of morphing between two time periods for information landscapes with different topographies.

5.1 Generating and Loading Morphing Data

All the use case scenarios use real-world data, i.e. a collection of text documents. As real-world data, 489 available papers that have been published by the Know-Center in recent years have been organized by year in order to determine how the topography of papers of one year changes to another year using the information landscape morphing concept. This use case scenario explains how to run the pipeline to generate morphing data from sequentially organized text documents in a collection. Furthermore, this use case describes how the user interface of the application is structured before the process of loading the periodic morphing data into the application.

5.1.1 Generating Periodic Morphing Data

The text processing pipeline has been exported as an executable JAR file, which compresses all the Java files of the used Know-Center framework into an archive. To generate the morphing data from a collection of text documents, the exported JAR file is run using the command line as shown in Figure 5.1:



```
java  -jar visPipeline.jar input-text-collection output-morphing-data
```

Figure 5.1: Command line for executing the pipeline: (1) name of the jar file, (2) input folder containing text documents, (3) output folder for generated periodic morphing data

After execution of the command, the pipeline runs all the steps as described in sections 3.2.1 and 3.2.2 and generates morphing data for every annual period, which can be loaded into the application in order to create an interactive information landscape

5.1.2 Structure of the Application

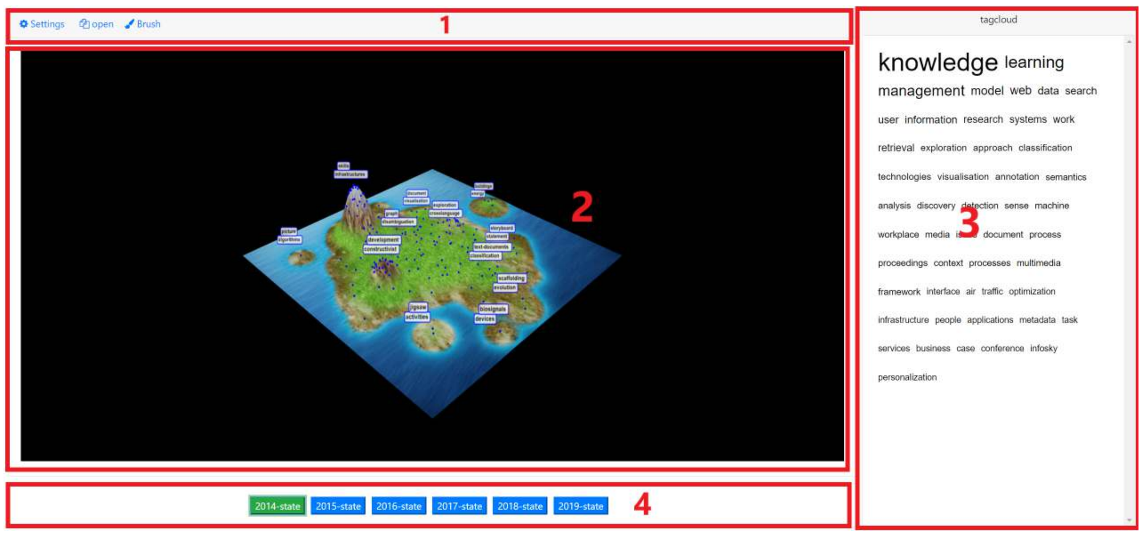

Settings
open
Brush
1
2
tagcloud
knowledge learning
management model web data search
user information research systems work
retrieval exploration approach classification
technologies visualisation annotation semantics
analysis discovery detection sense machine
workplace media document process
proceedings context processes multimedia
framework interface air traffic optimization
infrastructure people applications metadata task
services business case conference infosky
personalization
3
2014-state
2015-state
2016-state
2017-state
2018-state
2019-state
4

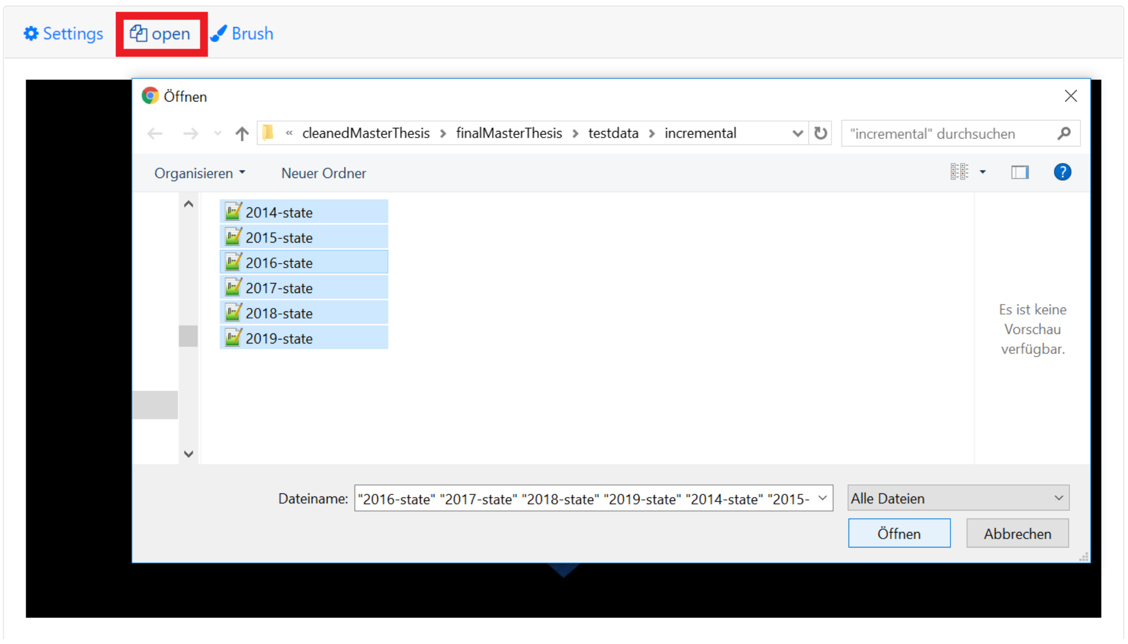
Figure 5.2: Structure of the visual application with highlighting of different functional areas

The UI of the application is organized into four different areas, as shown in Figure 5.2. The first area is a menu bar which provides the following functionality: (i) changing settings of the application, (ii) loading morphing data of different time periods into the application, (iii) activating and deactivating the brush tool for selecting the data. The second area is the main area, which contains the actual information landscape. The third area contains the tag cloud. The fourth area organizes the loaded periodic morphing data into a set of actions in the form of executable buttons in order to morph the information landscape of one period into another.

5.1.3 Loading Morphing Data in the Application

The user performs the following actions to load the data into the application (Figure 5.3):

- Clicks on “open” in the menu bar, consequently a dialog appears
- Selects all the periodic morphing data that has to be loaded in the application and closes the dialog window with a confirmation


Settings
open
Brush
Öffnen
« cleanedMasterThesis › finalMasterThesis › testdata › incremental
"incremental" durchsuchen
Organisieren
Neuer Ordner
2014-state
2015-state
2016-state
2017-state
2018-state
2019-state
Es ist keine Vorschau verfügbar.
Öffnen
Abbrechen

Dateiname: "2016-state" "2017-state" "2018-state" "2019-state" "2014-state" "2015-

Alle Dateien

Figure 5.3: Steps needed to load morphing data

As soon as the user confirms the dialog window, the application reads all the periodic morphing data and combines the data to build a set of actions in the form of executable buttons in order to morph the information landscape of one period into another. Additionally, as shown in Figure 5.4, the application renders the information landscape along with a tag cloud representing the dataset with the most oldest period, i.e. the year 2014, and highlights the corresponding button in green.

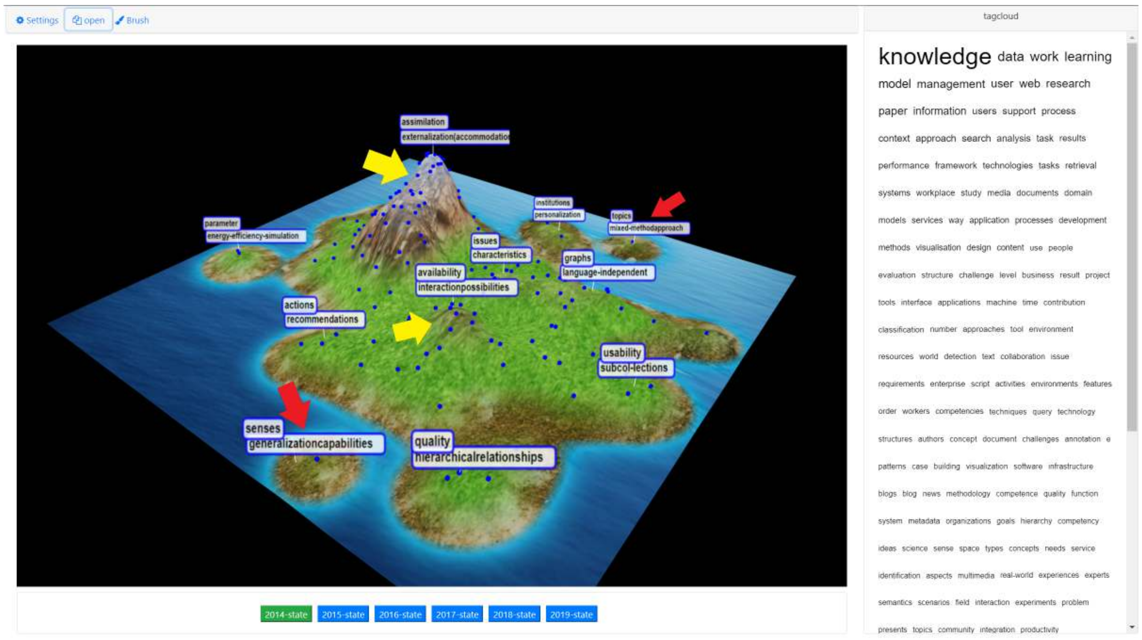

Settings
open
Brush
tagcloud
assimilation
externalization(accommodation
parameter
energy-efficiency-simulation
institutions
personalization
topics
mixed-methodapproach
issues
characteristics
graphs
language-independent
availability
interactionpossibilities
actions
recommendations
usability
subcol-lections
senses
generalizationcapabilities
quality
hierarchicalrelationships
2014-state
2015-state
2016-state
2017-state
2018-state
2019-state
knowledge data work learning
model management user web research
paper information users support process
context approach search analysis task results
performance framework technologies tasks retrieval
systems workplace study media documents domain
models services way application processes development
methods visualisation design content use people
evaluation structure challenge level business result project
tools interface applications machine time contribution
classification number approaches tool environment
resources world detection text collaboration issue
requirements enterprise script activities environments features
order workers competencies techniques query technology
structures authors concept document challenges annotation e
patterns case building visualization software infrastructure
blogs blog news methodology competence quality function
system metadata organizations goals hierarchy competency
ideas science sense space types concepts needs service
identification aspects multimedia real-world experiences experts
semantics scenarios field interaction experiments problem
presents topics community integration productivity

Figure 5.4: Information landscape with a tag cloud of the year 2014: the two papers classified as outliers are indicated by red arrows, and the two clusters containing most of papers are indicated by a yellow arrow

5.1.4 Conclusion

The first use case covered the steps when running the pipeline to generate morphing data from a collection of text documents, the structure of the user interface, and the process of loading a dataset into the application.

All the steps of the process are easy to follow. After completing this process, the visual representation of a landscape (Figure 5.4) provides an overview of all the papers that have been published by the Know-Center in 2014. The visual setup of the information landscape that roughly matches the real natural environment makes the landscape more intuitively comprehensible and interpretable. The user can understand the relatedness as well as topical distribution of the papers by looking at hills and labels (clusters). A user can also see that there are two topical clusters that contain most of the published papers. Furthermore, the user can identify two papers (see the red arrows in 5.4) that can be classified as outliers due to their dissimilarity to other papers in the collection. Based on the overall summary provided by the tag cloud, users can see that the main focus points of the research in the year 2014 were as follows: (i) knowledge, (ii), data, (iii) work, (iv) learning, (v) modal, (vi) management. Once the information landscape has been rendered, everything is set up for the user to interactively engage with the landscape and perform further actions.

5.2 Data Selection, Exploration And Filtering

This use case scenario allows the user to explore, filter and navigate documents of interest. The documents can be selected individually, based on their association with a keyword in the tag cloud or a displayed label in the landscape, as well as based on their positions. A brush-based selection tool as well a rectangle-based selection tool are used to select the documents that are within an area. The tag cloud is used to provide an overview of keywords for the selected documents.

5.2.1 Individual-Based Selection

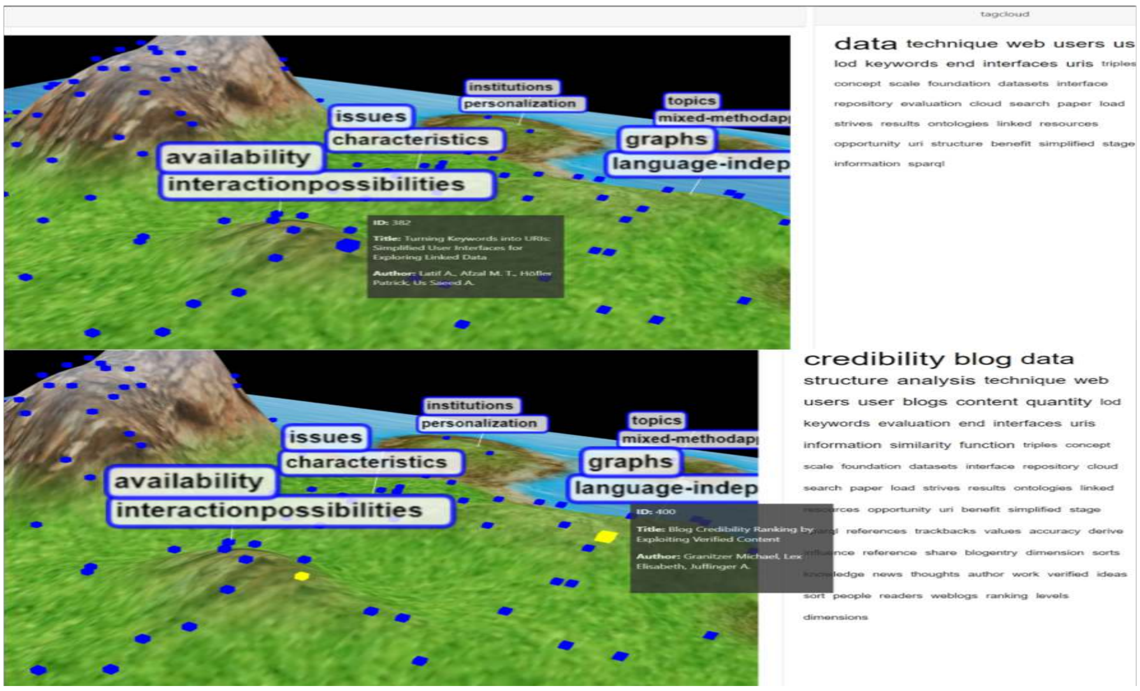
Once the user moves the mouse pointer over an individual document in the landscape, the corresponding document tooltip appears that gives basic information about the document, while at same time the contents of the whole document is represented in the tag cloud.

A click on an individual document will select it, whereas a second click removes the selection. The user can select multiple documents just by clicking on them, and the content of all selected documents will be represented in the tag cloud. An illustration of these actions is presented in the Figure 5.5.

5.2.2 Content-Based Selection

Label-Based Selection

As demonstrated in Figure 5.6, as soon as the user moves the mouse cursor over a label, the documents that contain the label are highlighted in the landscape by showing them enlarged, and at same time their content is reflected


tagcloud
institutions
personalization
topics
mixed-methodap
issues
characteristics
graphs
language-indep
availability
interactionpossibilities
data technique web users us
lod keywords end interfaces uris triples
concept scale foundation datasets interface
repository evaluation cloud search paper load
strives results ontologies linked resources
opportunity uri structure benefit simplified stage
information sparql
credibility blog data
structure analysis technique web
users user blogs content quantity lod
keywords evaluation end interfaces uris
information similarity function triples concept
scale foundation datasets interface repository cloud
search paper load strives results ontologies linked
references trackbacks values accuracy derive
reference share blogentry dimension sorts
news thoughts author work verified ideas
sort people readers weblogs ranking levels
dimensions

ID: 382
Title: Turning Keywords into URIs: Simplified User Interfaces for Exploring Linked Data
Author: Latif A., Afzal M. T., Höfler Patrick, Us Saeed A.

ID: 400
Title: Blog Credibility Ranking by Exploiting Verified Content
Author: Granitzer Michael, Lex Elisabeth, Juffinger A.

Figure 5.5: Individual document exploration: based on document mouse over (above, document shown enlarged), and based on the single selection method (below, 2 documents shown in yellow).

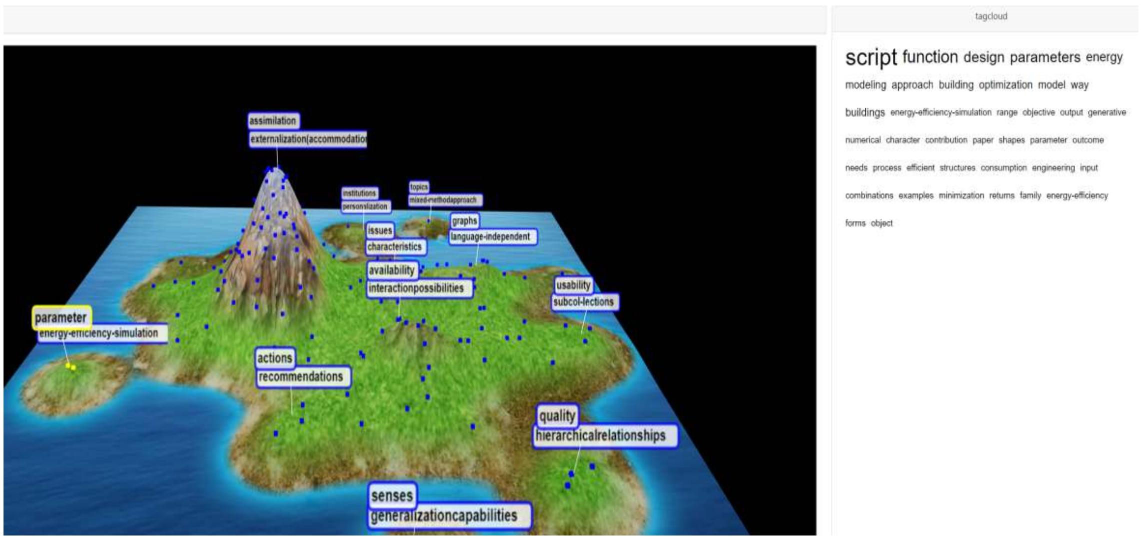

tagcloud
assimilation
externalization(accommodatio
institutions
personalization
topics
mixed-methodapproach
issues
characteristics
graphs
language-independent
availability
interactionpossibilities
usability
subcol-lections
parameter
energy-efficiency-simulation
actions
recommendations
quality
hierarchicalrelationships
senses
generalizationcapabilities
script function design parameters energy
modeling approach building optimization model way
buildings energy-efficiency-simulation range objective output generative
numerical character contribution paper shapes parameter outcome
needs process efficient structures consumption engineering input
combinations examples minimization returns family energy-efficiency
forms object

Figure 5.6: Exploration of documents based on a selected label ("parameter", on the lef side of the screen)

in the tag cloud. By clicking on the label, all the associated documents are selected, which shows them in yellow. A click on a selected document removes the selection.

Filtering of Documents via a Tag Cloud

As shown in Figure 5.7, as soon as the user moves the mouse over a keyword in the tag cloud, the documents that contain the keyword are highlighted enlarged in the landscape. Once the user moves the mouse pointer off the keyword, the associated documents are no longer highlighted and return to theire orginal size.

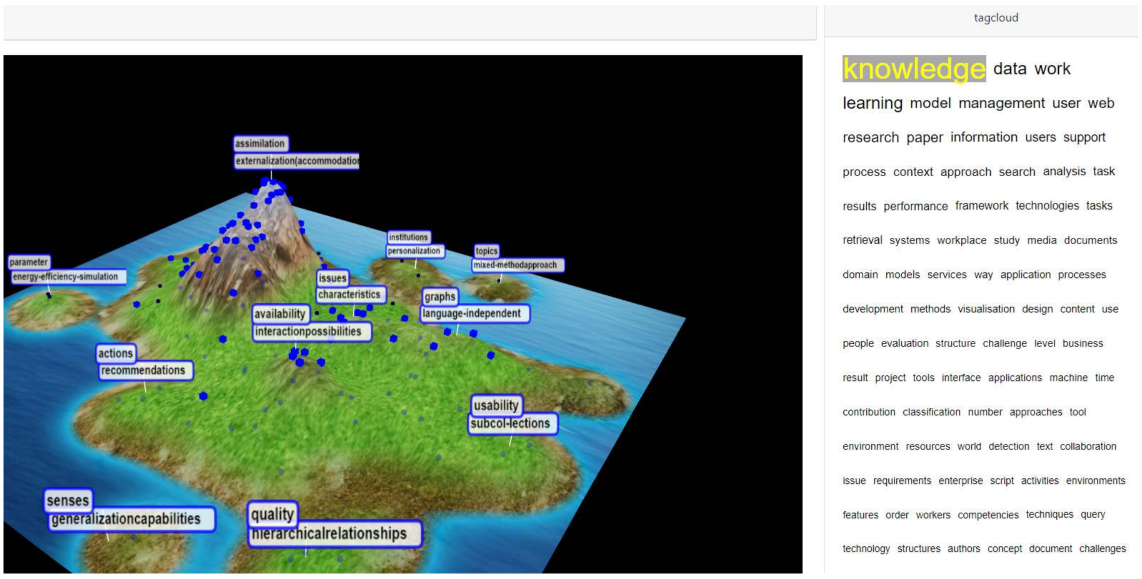

tagcloud
assimilation
externalization(accommodatio
institutions
personalization
topics
mixed-methodapproach
parameter
energy-efficiency-simulation
issues
characteristics
graphs
language-independent
availability
interactionpossibilities
actions
recommendations
usability
subcol-lections
senses
generalizationcapabilities
quality
hierarchicalrelationships
knowledge data work
learning model management user web
research paper information users support
process context approach search analysis task
results performance framework technologies tasks
retrieval systems workplace study media documents
domain models services way application processes
development methods visualisation design content use
people evaluation structure challenge level business
result project tools interface applications machine time
contribution classification number approaches tool
environment resources world detection text collaboration
issue requirements enterprise script activities environments
features order workers competencies techniques query
technology structures authors concept document challenges

Figure 5.7: Exploration of documents based on a selected keyword in the tag cloud

5.2.3 Position-Based Selection

Brush-Based Selection

To activate the brush, the user clicks on “Brush” in the menu bar. As soon as the brush is activated, it will appear in the landscape and will follow the movement of the mouse. All documents which are within the brush will be highlighted and the content of these documents will be represented in the tag cloud. Figure 5.8 shows the results of these actions. A click on the brush will select all the documents that are inside brush and clicking again deselects the documents. Furthermore, the size and the colour of the brush can be adjusted according to the preference as shown in Figure 5.9.

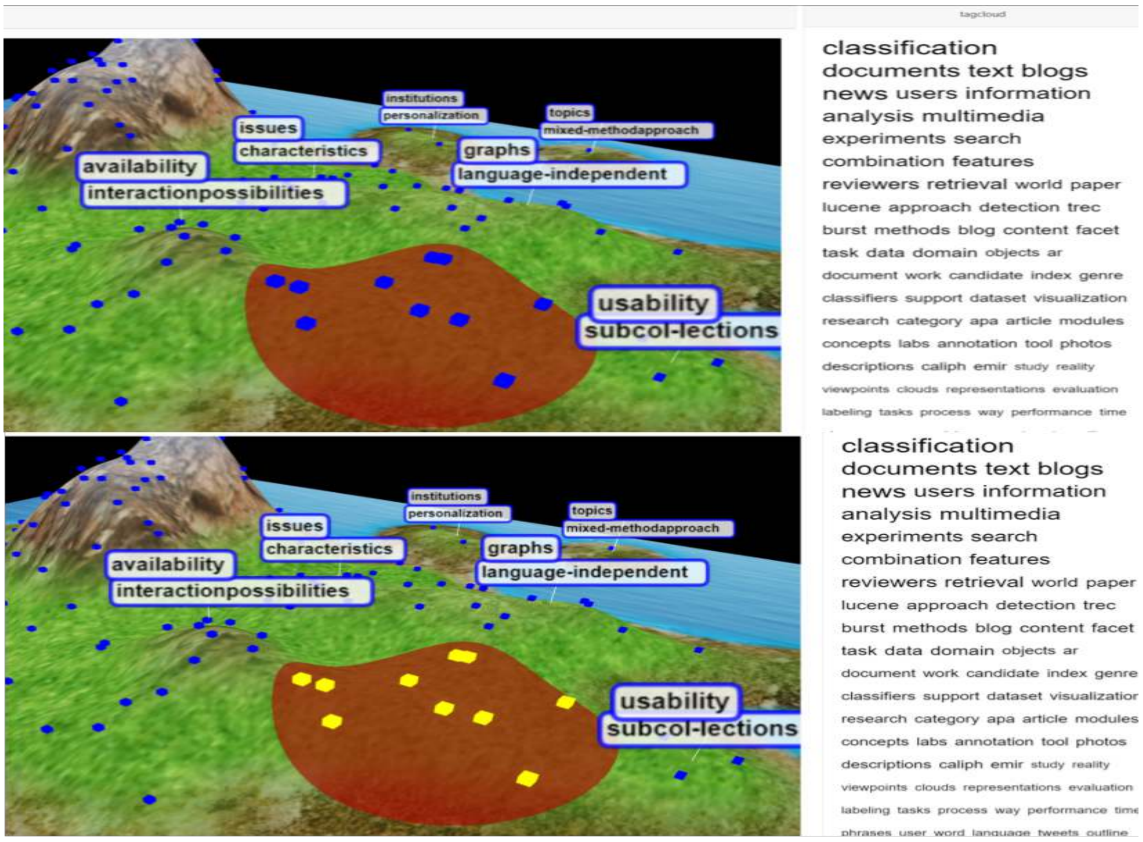
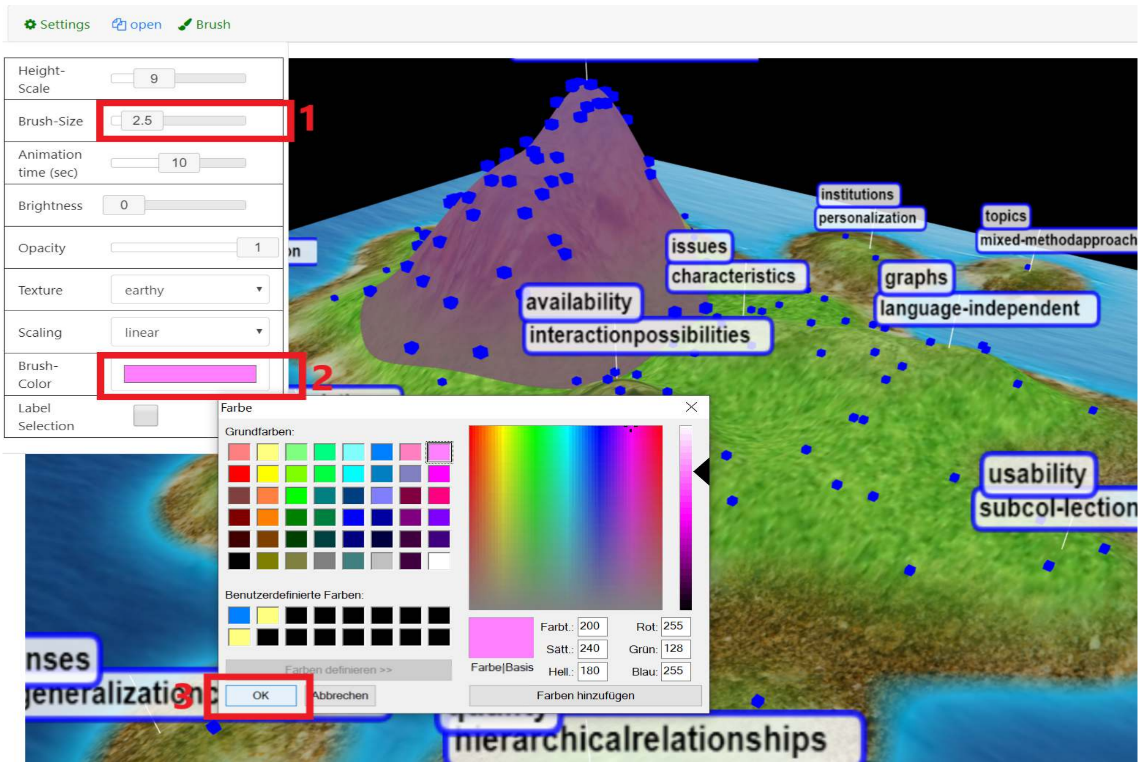

tagcloud
institutions
personalization
topics
mixed-methodapproach
issues
characteristics
graphs
language-independent
availability
interactionpossibilities
usability
subcol-lections
classification
documents text blogs
news users information
analysis multimedia
experiments search
combination features
reviewers retrieval world paper
lucene approach detection trec
burst methods blog content facet
task data domain objects ar
document work candidate index genre
classifiers support dataset visualization
research category apa article modules
concepts labs annotation tool photos
descriptions caliph emir study reality
viewpoints clouds representations evaluation
labeling tasks process way performance time
phrases user word language tweets outline

Figure 5.8: Exploring documents using the brush tool (above), and selecting them (bellow)

Rectangle-Based Selection

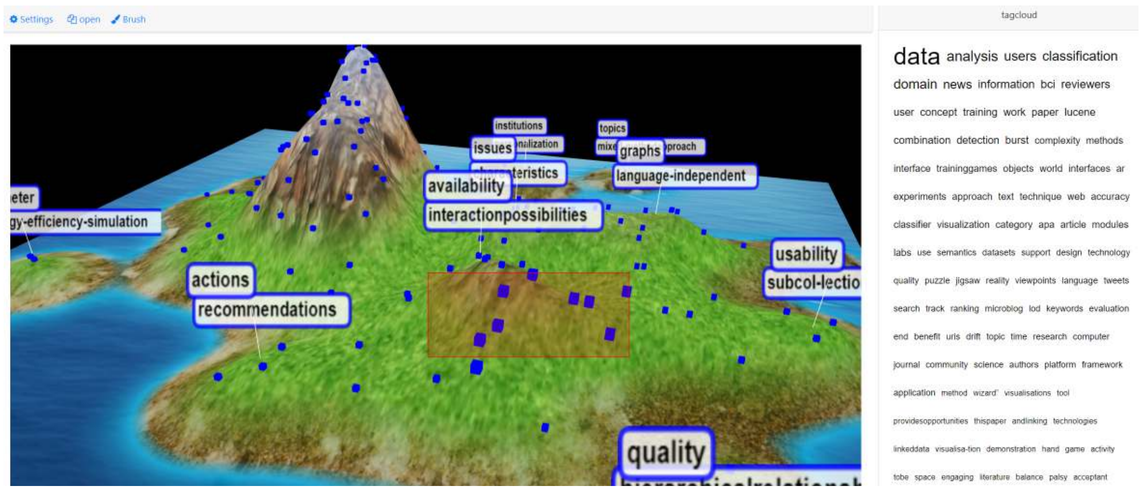
As soon as the users clicks the shift button, hold it pressed and then click the left mouse button the one corner of the selection rectangle is defined. As the user drags the mouse to the position of the diagonally opposite corner the selection rectangle shape changes accordingly. All the documents within the selection tool will be shown as highlighted and the content of these documents will be represented in the tag cloud. Once the user releases the mouse the documents within the selection tool will be selected and rectangle-based selection tool will vanish. Figure 5.10 depicts the action of this tool.



Height-Scale 9
Brush-Size 2.5
Animation time (sec) 10
Brightness 0
Opacity 1
Texture earthy
Scaling linear
Brush-Color
Label Selection

Farbe
Grundfarben:
Benutzerdefinierte Farben:
Farben definieren >>
OK Abbrechen
Farbe|Basis
Farbt.: 200 Rot: 255
Sätt.: 240 Grün: 128
Hell.: 180 Blau: 255
Farben hinzufügen

Figure 5.9: Setting the size and colour of the brush



data analysis users classification
domain news information bci reviewers
user concept training work paper lucene
combination detection burst complexity methods
interface traininggames objects world interfaces ar
experiments approach text technique web accuracy
classifier visualization category apa article modules
labs use semantics datasets support design technology
quality puzzle jigsaw reality viewpoints language tweets
search track ranking microblog lod keywords evaluation
end benefit urls drift topic time research computer
journal community science authors platform framework
application method wizard' visualisations tool
providesopportunities thispaper andinking technologies
linkeddata visualisa-tion demonstration hand game activity
tobe space engaging literature balance palsy acceptant

Figure 5.10: Exploring documents using the rectangle selection tool

5.2.4 Conclusion

This use case scenario covers the individual selection of documents as well as selection based on content and position.

The tag cloud is used very effectively for giving an overview of the selected documents and moreover an overview of the whole dataset when nothing is selected.

The brushing tool imitates the effect of a torchlight; i.e. the circle being highlighted adapts in shape accordingly to the elevation.

The labels are designed to be reactive whenever the user tilts or rotates the landscape so that they adapt their positions accordingly for the user and are always readable. However, this feature also has a disadvantage when the user wants to select the labels because the casting of the ray with labels is an approximation in the label. Due to this technical issue the user will find it very hard to select the label. The problem shall be addressed in a future version of the software.

5.3 Morphing the Information Landscape

This use case scenario demonstrates the morphing process between two periods of time using an information landscape to help the user follow and understand the changes. Before the morphing is applied, it is important that the user selects a suitable design, height and perspective for the information landscape, even though all of these settings can be specified during the morphing phase. Optimal settings will nonetheless make the morphing more understandable and easier to follow. This scenario thus covers all the steps needed to choose a suitable design, height, perspective, and the type of height scaling before the actual process of morphing is executed.

5.3.1 Perspective selection

The user uses the mouse scroll wheel to zoom in or out in the landscape to view the areas in more or less detail. Furthermore, the user can pan the landscape by holding the left mouse button and dragging the information landscape in any direction so that all areas are in the view and at the same time all the areas are displayed in required detail. Finally, by holding the Control (CTRL) keyboard button and dragging the mouse enables tilting and rotating. Figure 5.11 provides the view of the landscape before and after applaying perspective changes.

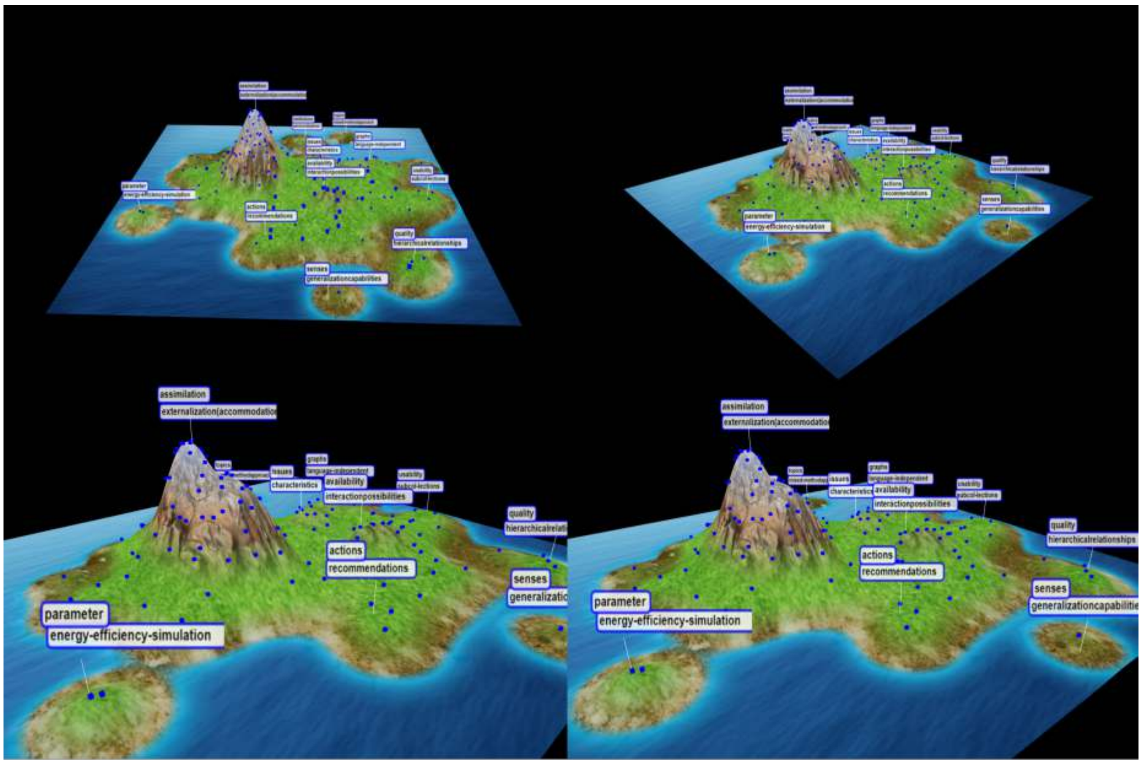
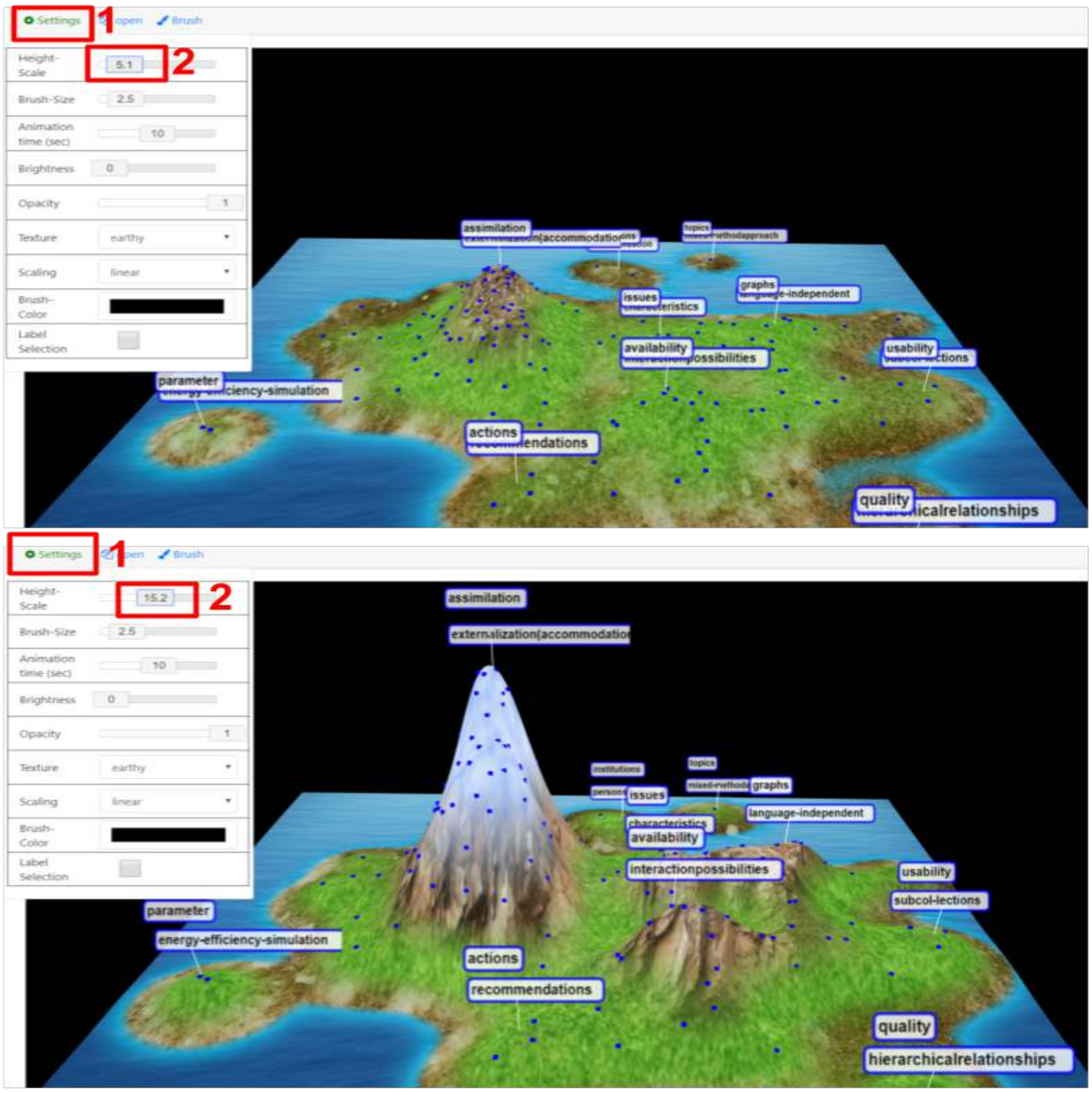


Figure 5.11: Representation of different perspectives: default view (top-left), view after tilting and rotating (top-right), view after zooming in (bottom-left), view after panning (bottom-right)

5.3.2 Height adjustment

In the first step the user clicks “Settings” in the menu bar, which consequently opens a drop-down menu. The drop-down menu provides a range slider as an option to set the maximum height of the information landscape. As the user moves the slider with the mouse, the information landscape is rendered with a newly scaled geometry between zero and the maximum height value. Furthermore, the height of the documents and labels is adjusted accordingly. Once the user clicks “Settings” in the menu bar again, the drop-down menu is closed. Figure 5.12 demonstrates the required steps.


Height-Scale 5.1
Brush-Size 2.5
Animation time (sec) 10
Brightness 0
Opacity 1
Texture earthy
Scaling linear
Brush-Color
Label Selection

Height-Scale 15.2
Brush-Size 2.5
Animation time (sec) 10
Brightness 0
Opacity 1
Texture earthy
Scaling linear
Brush-Color
Label Selection

Figure 5.12: Representation of information landscape after setting the maximum height of the information landscape: "5.1" maximum height (above) and “15.2” maximum height (below)

5.3.3 Height Scaling Selection

In the first step the user clicks “Settings” in the menu bar, which consequently opens a drop-down menu. The drop-down menu contains a selection box (i.e Scale), which provides the user with following four types of scales: (i) linear, (2) square root, (3) squared and (4) logarithmic. As soon as the user set one of the type of the scale, the landscape is rendered with immediate effect with newly selected type of scale.

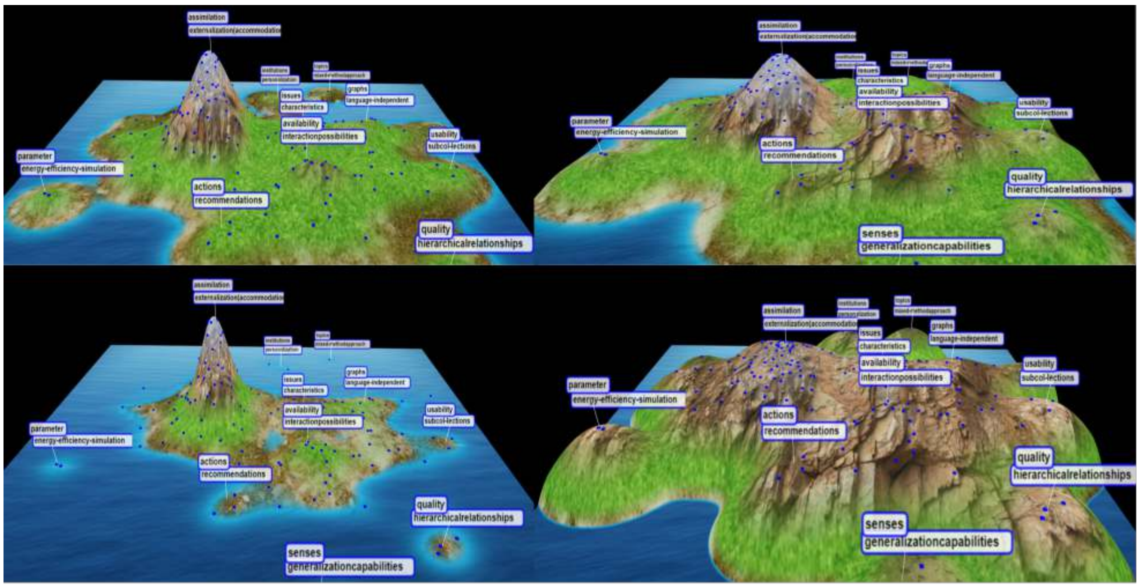


Figure 5.13: Representation of information landscape by setting different types of height scaling: linear scale (top-left), square root scale (top-right), squared scale (bottom-left), logarithmic scale (bottom-right)

5.3.4 Design Selection

The user opens the drop-down menu of the settings by clicking on “Settings” in the menu bar. The drop-down menu contains a selection box, which provides the user with three options for the design. As soon as the user chooses one of the designs, the landscape is rendered with immediate effect with the newly selected design. Figure 5.14 presents all the steps.

5.3.5 Morphing Process Execution

In order to demonstrate the morphing process, the information landscape for the year 2014 is morphed into the information landscape of the year 2015. As soon as the user clicks the corresponding execution button for the next period (i.e. 2015), a smooth animation sequence is initiated, which is rendered for every frame over a timespan of 10 seconds. Before the animation is started, the following steps are executed:

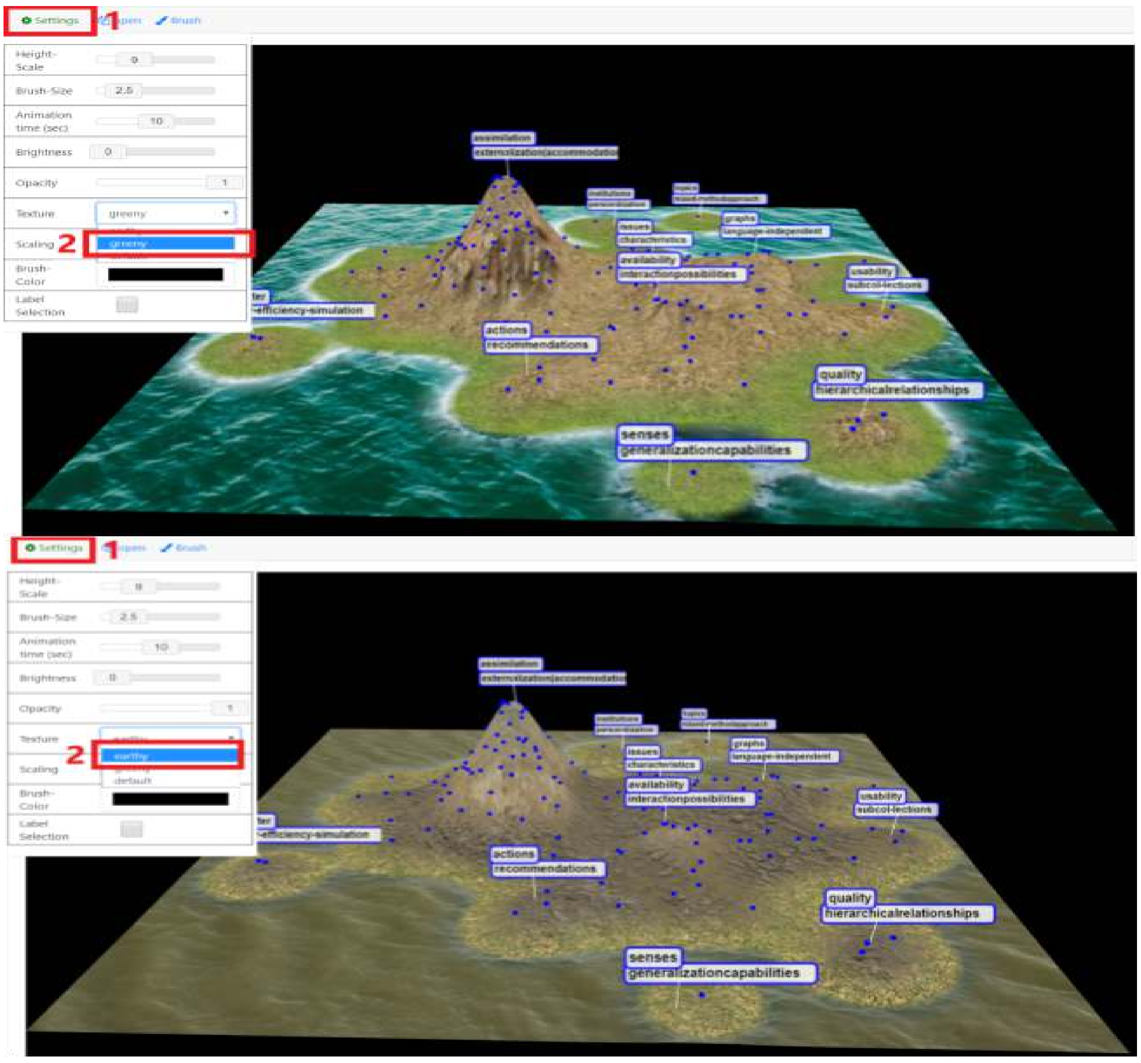


Figure 5.14: Representation of information landscape after applying new design: “greeny” design above and “earthy” design below

- All the new elements of the next period (new documents, new labels, orientation lines between labels and peaks) are added into the scene with an opacity of zero.
- For all documents that are part of the year 2014 as well as part of 2015, tracing lines starting with the position of old documents are added into the scene.
- The height map of the next period (i.e. 2015) is added to the scene as a wireframe, which will indicate where the new topics will emerge or

where topical convergence and divergence will take place.

Figure 5.15 shows the status of the information landscape before the animation is started.

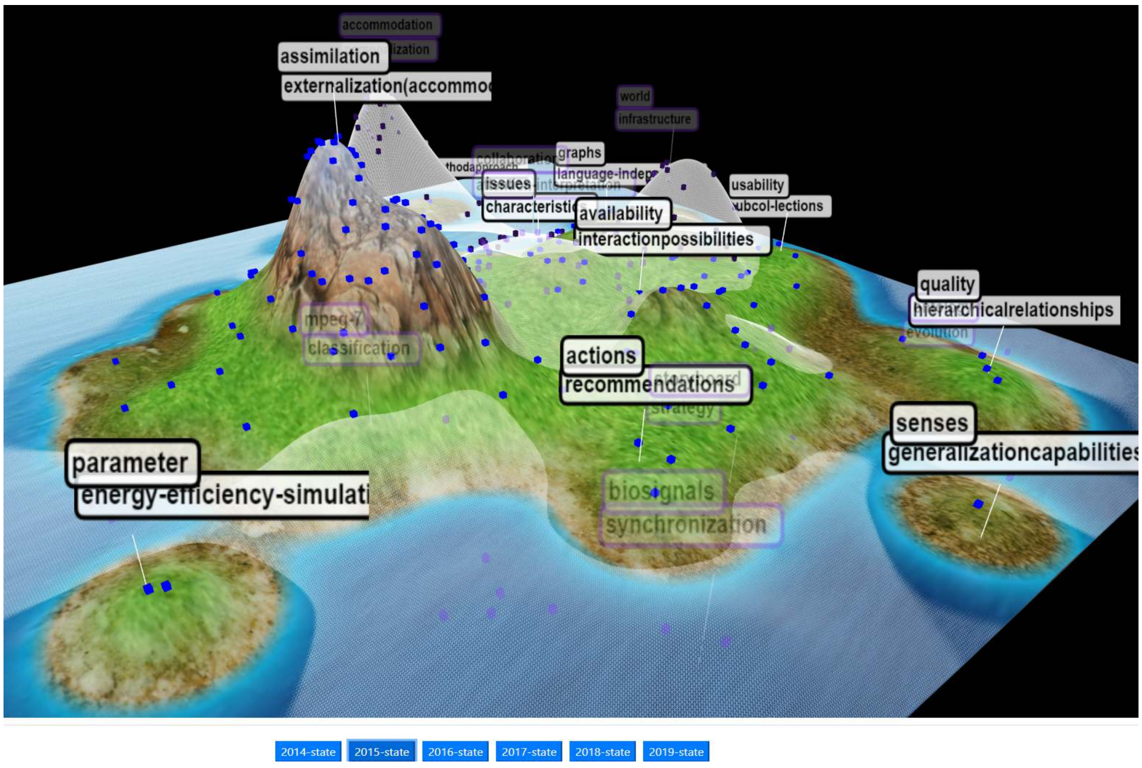


Figure 5.15: Status of morphing before the animation is started: The height map with textures represents the year 2014 and the height map in wireframe represents the year 2015, which clearly indicates the emergence of new topics as well the convergence and divergence of topics

For every frame of the animation, the following steps are executed:

- The height values (z-properties) of the current landscape (i.e. 2014) are moved towards the height values of the next landscape (i.e. 2015).
- The opacity of all the new elements of the next period (new documents, new labels, orientation lines between labels and peaks) is gradually scaled towards 1.0 in order to indicate the emergence of new topics, while the opacity of all the elements that are only part of the year 2014 scales towards 0.0 in order to show the fading of topics.

- The tracing lines of documents are moved towards new positions to show their contribution in the formation of new topics as well as in the convergence and divergence of topics.

Figure 5.16 depicts the status of the information landscape in the middle of the animation.

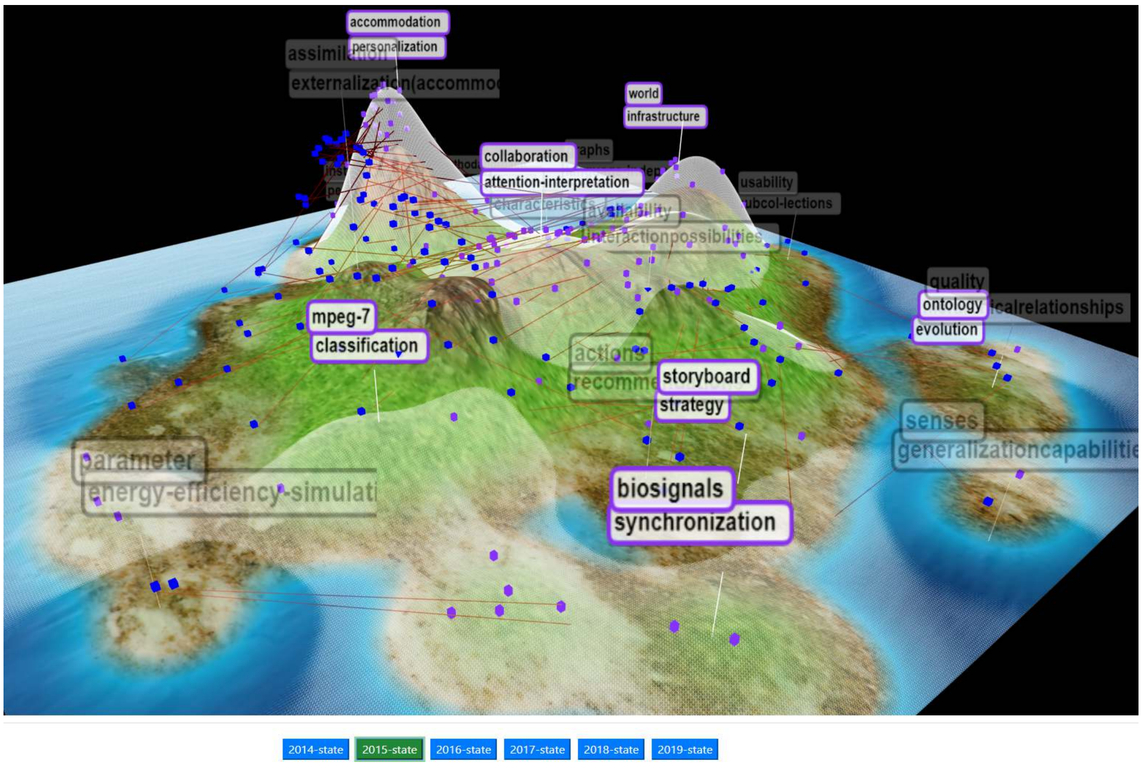


Figure 5.16: This is the middle phase of the animation, where one can see the emergence of new elements from 2015 (new documents, new labels, orientation lines) as well as the fading of elements which are not part of 2015; the tracing of documents represented as lines shows the documents that make up new topics as well as the divergence and convergence of topics

After the animation is finished, the following steps are executed:

- The elements (documents, labels, orientation lines between labels and peaks) that are not part of the periodic landscape (i.e. 2015) are removed from the scene.

- The added 2015 height map, depicted as a wireframe, is not needed anymore and it is removed from the scene. Instead, a height map of the previous period (i.e. 2014) is added to the scene in the form of a wire-frame in order indicate the faded topics as well as allow the user have to visual reference so that one can easily recognize what has changed.

Figure 5.17 shows the status of the information landscape after the animation.

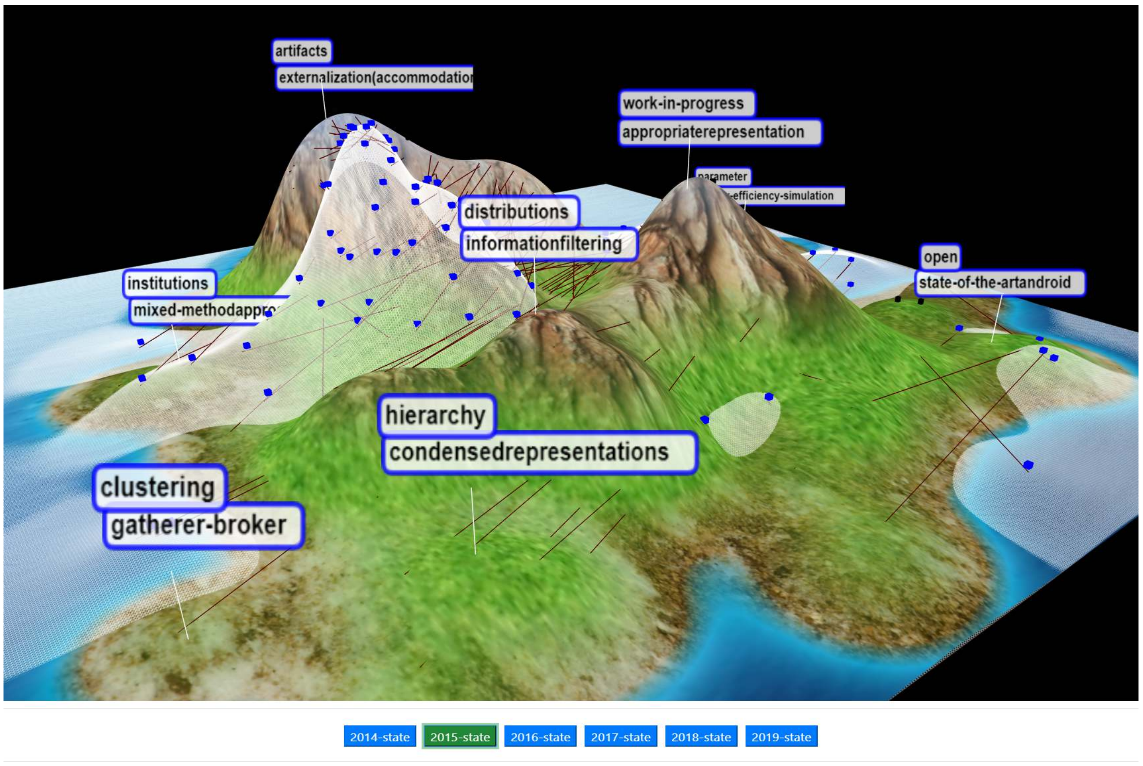


Figure 5.17: Status of morphing when the animation is finshed: The height map with textures represents the year 2015, whereas the height map in wire-frame represents the year 2014 and indicates the fading of topics; this allows the user to compare the states of both periods, where the tracing lines represents the movement of the documents from the old position (2014) to the new position (2015)

After the landscape morphing animation, the documents that are included in both of the periods are moved from the old position toward the new position in another animation lasting a few seconds (See the Figure 5.18). This step completes the morphing process.

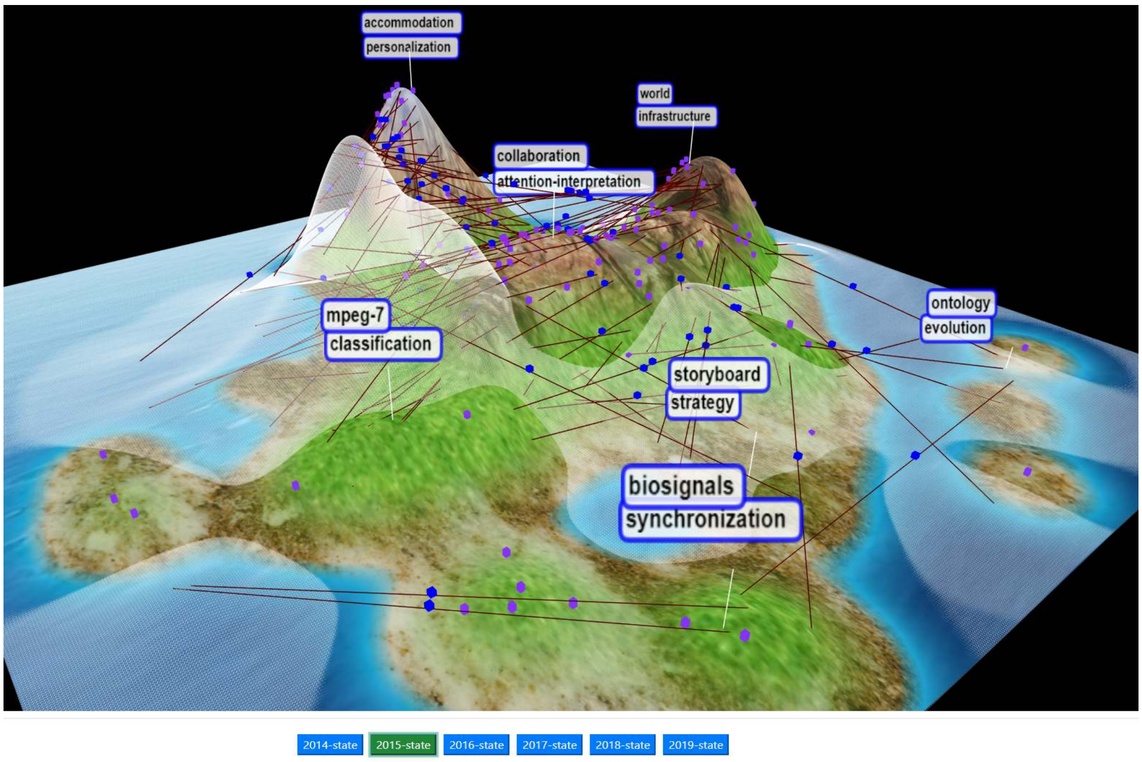


Figure 5.18: Representation after the animated movement of the documents from the old position (2014) towards to the new position (2015).

5.3.6 Conclusion

This use case scenario demonstrated the morphing process between two time periods in an information landscape. Furthermore, it showed how to choose a suitable design, height and perspective for the information landscape. The designed and developed concept of morphing offers the following main advantages in terms of visual design, performance and human interpretability:

- An efficient and highly parallel structure of computations is used to create landscapes that match the realistic natural environment as well as smooth 3D-accelerated morphing animation of landscape geometry.
- The visual design encompasses the fact that human beings interpret a landscape as different sets of layers, where a water texture defines the lowest points, with sand, grass, rocks and snow textures as the height increases. Therefore, the morphing of a fading topic follows the top-down approach in which the top layers of texture disappear first and then the bottom layers, whereas an emerging topic follows the bottom-

up approach.

- Even if the morphing is running and the user changes the design of the information landscape or adjusts the height of landscape at the same time, the morphing process is not hindered in any way and the new design or the new maximum height is applied with immediate effect the rest of the morphing process will continue with the new design or new maximum adjusted height.

One of the drawbacks may be that the application can run into a change blindness problem, which makes the users unable to identify the changes. An adapted method based on Nowell's recommendation has been implemented to overcome the blindness problem, however all the aspects still have to be investigated through the collection of user feedback in an extensive evaluation. The following steps have been taken to counter the change blindness problem as well as to improve user orientation in terms of emerging and fading topics:

- The morphing runs as smoothly as possible in order to make the changes more noticeable.
- The top-down approach as well as bottom-up approach for different sets of layers helps significantly when it comes to recognizing the changes.
- The users are provided with the option to increase the timespan of the animation to allow more time to grasp the changes.
- The height map of the next period is displayed as a wireframe from the beginning to the end of the morphing, showing the user where the new topics will emerge.
- The height map of the previous period is displayed as a wireframe as soon as morphing is completed, allowing the user to have access to the visual differences, the user can recognize what has changed.
- By increasing the opacity of new elements (documents, labels, orientation lines) of the new period and decreasing the opacity of elements of old period during the morphing phase, orientation in terms of which topics (peaks) will emerge or fade is provided to the user.
- Tracing lines are shown to follow document moves that occurs due to the topical shifts in the dataset.

Nevertheless, It has to be mentioned that unforeseen effects can happen in the short span of the animation which may confuse the user resulting in loss of focus.All of these aspects have to be considered in an extensive user evaluation.

5.4 Summary

Based on three scenarios, this chapter presented how to use the developed web-based application on real world data. The first scenario described the steps for running the pipeline in order to generate morphing data. Furthermore, the first scenario included an overview of the structure of the user interface as well as the process of loading the morphing data in the application. The second scenario introduced the individual selection of documents as well as selection based on content and position. The third scenario demonstrated morphing an information landscape over two periods of time having different topographies. Furthermore, the second scenario included the selection of a suitable design, height and perspective for an information landscape.

Chapter 6

Conclusion & Future Work

This Master's Thesis presented a web-based application for visualizing an interactive scalable information landscape using WebGL and HTML5 technologies. The application provides a comprehensive overview of an unfamiliar collection of data, and helps the user to understand the relationships, topical distribution and changes in topography of a complex, multidimensional and dynamically changing dataset. A text-processing pipeline was introduced to compute the layout when morphing an information landscape. The computed information by the pipeline (i.e. the height map, the locations, keywords of documents and the peak-related information) was used to visualize an information landscape that roughly matches the real natural environment.

The efficient and highly parallel structure of computations of graphic processing units has been utilized in carrying out following actions: (i) applying the height-based texture mapping method to each pixel of landscape using the shading language (i.e. fragment shader) in order to visualize the landscape in a set of layers of textures that is easy to understand, (ii) manipulating the geometry of the landscape using a shading language (i.e. vertex shader) to create a smooth 3D accelerated, smooth animation to help the user to follow and to understand the data changes. Furthermore, some functionality (i.e. zooming, rotation and titling) has been integrated to allow the user to interactively view the information landscape from different perspectives. Additionally, the user can select the data of interest in several ways (i.e. individually, based on their content and based on their position in the landscape). At same time an integrated interactive tag cloud provides a topical overview of the selected dataset or a topical overview of the whole dataset when nothing has been selected.

From a development point of view, the library three.js makes it very easy to attach the shader code, however it is very hard to debug. A developer has to pay attention to what she or he is doing and has to be precise when formulating the logic.

A lot of work can be done on further improving the information landscape application and adding additional features. The most reasonable tasks for future work are as follows:

- **User evaluation**: At least two user evaluations should be carried out in the future, first off a usability evaluation with the aim of identifying areas for improvement, strengths and limitations of the application, and second an evaluation to find out if the main feature of the application – the smooth morphing animation – helps the user to follow and to understand the changes and avoids the change blindness problem. The evaluation should find the answers to the following questions: (i) was the user able to identify what had changed from one time period to another, (ii) was the user able to remember what was different in the previous time period, (iii) did the user have to remember the changes because it was not possible for the user to tell what came before and what is going to appear next, (iv) did the seamless morphing with top-down approach/bottom-up approach and different set of layers have any impact in making changes noticeable (v) did increasing and decreasing the opacity of the labelling (peaks) during the morphing phase have any impact in recognizing the emergence or decline of topics.
- **Improving the interactions between labels in the information landscape**: The labels are not placed exactly above the corresponding peak and often overlap when the user tilts, pans or rotates the landscape. Another issue is that the user might find it hard to select the label, because the casting of the ray with labels is an approximation in relation to the label. Adequate steps will be taken to solve these issues in the future.
- **Converting client-based application to client-server-based application**: Currently, all required information, including a list of keywords and frequency information for every document, are loaded into the application in order to visualize the information landscape and tag cloud. This is a bad practice, especially when dealing with a larger data set. In the future, the client-based application should be transformed into a client-server-based application in which the keywords are loaded on demand by sending a request to a server.
- **Integration of lighting and shading**: The lighting and shading could be added to the information landscape in the future. The technique of Phong shading could be used to calculate shading per pixel (i.e. frag-

ment shader) for smoothing and improving the surface detail of the information landscape.

- **Implementation of drag and drop for custom design integration**: A convenient method of drag and drop functionality should be implemented in the future in order to let the user add a new design to the application. The functionality will consist of the following three steps: (i) the user drags and drops the textures of the new design, (ii) the user defines what texture should be used at which height, (iii) the user gives a unique name to the design.

List of Abbreviations

GPU	Graphics processing unit
HTML	Hypertext Markup Language
WebGL	Web Graphics Library
CSS	Cascading Style Sheets
DOM	Document Object Model
SVG	Scalabe Vector Graphic
PCA	Principal component analysis
MDS	Multidimensional scaling
NMDS	Non Metric Multidimensional scaling
FDP	Force Directed Placement
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
LOD	Linked Object Data
MIT	Massachusetts Institute of Technology
API	Application Programming Interface

Bibliography

[Andrews K. et al., 2002] 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 2012, volume 1, pp 166–181.

[BLOHM D. G.,] BLOHM, D. G. **Pca vs ica**. `http://compneurosci.com/wiki/images/4/42/Intro_to_PCA_and_ICA.pdf`. Last checked on 18.03.2019 08:30pm.

[Boyack K. et al., 2000] Boyack, K., Wylie, B., S. Davidson, G., and Johnson, D. (2000). **Analysis of patent databases using vxinsight**. New Paradigms in Information Visualization and Manipulation, a Workshop at the 9 th International Conference on Information and Knowledge Management (CIKM) 2000.

[Boyack K. W. et al., 2002] Boyack, K. W., Wylie, B. N., and Davidson, G. S. (2002). **Domain visualization using vxinsight for science and technology management**. J. Am. Soc. Inf. Sci. Technol. 2002, volume 53, pp 764–774.

[Chalmers M., 1995] Chalmers, M. (1995). **Design perspectives in visualising complex information**. Visual Database Systems 3: Visual information management 1995, pp 103–111.

[Chalmers M. & Chitson P., 1992] Chalmers, M. and Chitson, P. (1992). **Bead: Explorations in information visualization**. Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval 1992, pp 330–337.

[Daniel J. Simons R. A. R., 2005] Daniel J. Simons, R. A. R. (2005). **Change blindness: past, present, and future**.

[Davidson G. S. et al., 1998] Davidson, G. S., Hendrickson, B., Johnson, D. K., Meyers, C. E., and Wylie, B. N. (1998). **Knowledge mining with vxinsight: Discovery throughinteraction**. J. Intell. Inf. Syst. 1998, volume 11, pp 259–285.

[Davis J. C., 2002] Davis, J. C. (2002). *Statistics and Data Analysis in Geology, Edition 3.* Wiley.

[Eades P., 1984] Eades, P. (1984). **A heuristic for graph drawing**. Congressus Numerantium 1984, volume 42, pp 149–160.

[Endert A. et al., 2013] Endert, A., Bradel, L., and North, C. (2013). **Beyond control panels: Direct manipulation for visual analytics**. IEEE Comput. Graph. Appl. 2013, volume 33, pp 6–13.

[Fahlén L. E. et al., 1993] Fahlén, L. E., Brown, C. G., Ståhl, O., and Carlsson, C. (1993). **A space based model for user interaction in shared synthetic environments**. Proceedings of the INTERACT and CHI Conference on Human Factors in Computing Systems 1993, pp 43–48.

[Fraga R. et al., 2014] Fraga, R., Stow, A., Magnusson, W., and Lima, A. (2014). **The costs of evaluating species densities and composition of snakes to assess development impacts in amazonia**. PLoS ONE 2014, volume 9.

[Fruchterman T. M. J. & Reingold E. M., 1991] Fruchterman, T. M. J. and Reingold, E. M. (1991). **Graph drawing by force-directed placement**. Software - Practice and Experience 1991, volume 21, pp 1129–1164.

[Granitzer M. et al., 2003] Granitzer, M., Kienreich, W., Sabol, V., and Dösinger, G. (2003). **Webrat: Supporting agile knowledge retrieval through dynamic, incremental clustering and automatic labelling of web search result sets**. Proceedings of the Twelfth International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises 2003, pp 296– 301.

[Jackendoff R. S., 1985] Jackendoff, R. S. (1985). **Semantics and cognition (current studies in linguistics)**.

[Jolliffe I., 2002] Jolliffe, I. (2002). **Principal component analysis**. Springer Verlag.

[Kienreich W. et al., 2003] Kienreich, W., Sabol, V., Granitzer, M., Kappe, F., and Andrews, K. (2003). **Infosky: A system for visual exploration of very large, hierarchically structured knowledge spaces**. Proceedings der GI Workshopwoche, Workshop der Fachgruppe Wissensmanagement 2003.

[Klieber W. et al.,] Klieber, W., Sabol, V., Muhr, M., Kern, R., and Granitzer, M. **Knowledge discovery using the knowminer framework**. IADIS International Conference on Information Systems 2009.

[Kohonen T., 1997] Kohonen, T. (1997). **Self-organizing maps**. Berlin, Heidelberg. Springer-Verlag.

[Krishnan M. et al., 2007] Krishnan, M., Bohn, S., Cowley, W., Crow, V., and Nieplocha, J. (2007). **Scalable visual analytics of massive textual datasets**. IEEE International Parallel and Distributed Processing Symposium 2007.

[Krista Lagus S. K. & Kohonen T.,] Krista Lagus, Timo Honkela, S. K. and Kohonen, T. **Websom - a status report**. Last checked on 16.08.2019 15:56.

[Kruskal J., 1964] Kruskal, J. (1964). **Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis**. Psychometrika 1964, volume 29, pp 1–27. Springer.

[Kuhn W. & Blumenthal B., 1996] Kuhn, W. and Blumenthal, B. (1996). **Spatialization: Spatial metaphors for user interfaces**.

[Muhr M. et al., 2010] Muhr, M., Sabol, V., and Granitzer, M. (2010). **Scalable recursive top-down hierarchical clustering approach with implicit model selection for textual data sets**. Workshops on Database and Expert Systems Applications 2010, pp 15–19.

[Nowell L. et al., 2001] Nowell, L., Hetzler, E., and Tanasse, T. (2001). **Change blindness in information visualization: a case study**. IEEE Symposium on Information Visualization (INFOVIS) 2001., pp 15–22.

[Okabe A. et al., 2000] Okabe, A., Boots, B., Sugihara, K., and Chiu, S. (2000). **Spatial tessellations: Concepts and applications of voronoi diagrams**. Wiley Series in Probability and Mathematical Statistics 1992, volume 43.

[Rennison E., 1994] Rennison, E. (1994). **Galaxy of news: An approach to visualizing and understanding expansive news landscapes**. Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology 1994, pp 3–12.

[Sablol V., 2012] Sablol, V. (2012). **Visual analysis of relatedness and dynamics in complex, enterprise-scale repositories**. PhD thesis 2012, Technische Universität Graz.

[Sabol V. et al., 2010] Sabol, V., Ali Ahmad Syed, K., Scharl, A., Muhr, M., and Hubmann-Haidvogel, A. (2010). **Incremental computation of information landscapes for dynamic web interfaces**. HIMI 2010, pp 205–208.

[Sabol V. et al., 2002] Sabol, V., Kienreich, W., Granitzer, M., Becker, J., Tochtermann, K., and Andrews, K. (2002). **Applications of a lightweight, web-based retrieval, clustering, and visualisation framework**. Proceedings of the 4th International Conference on Practical Aspects of Knowledge Management 2002, pp 359–368.

[Samuel Kaski K. L. T. K., 1998] Samuel Kaski, Timo Honkela, K. L. T. K. (1998). **Websom – self-organizing maps of document collections**. pp 101–117.

[Shepard R. N., 962a] Shepard, R. N. (1962a). **The analysis of proximities: Multidimensional scaling with an unknown distance function. i.** Psychometrika 1962a, volume 27, pp 125–140.

[Shepard R. N., 962b] Shepard, R. N. (1962b). **The analysis of proximities: Multidimensional scaling with an unknown distance function. ii**. volume 27, pp 219–246.

[Simons D. J. & Levin D. T., 1997] Simons, D. J. and Levin, D. T. (1997). **Change blindness. trends cogn. sci. 1**. pp 261–267.

[Simons D. J. & Levin D. T., 1998] Simons, D. J. and Levin, D. T. (1998). **Failure to detect change to people during a real-world interaction**. *Psychonomic Bulletin Review*, 5:644–649.

[SPIRE-PNNL,] SPIRE-PNNL. **Spire-pnnl**. `https://in-spire.pnnl.gov/faq.stm`. Last checked on 28.03.2019 10:00pm.

[Thomas J. et al., 2001] Thomas, J., Cowley, P., Kuchar, O., Nowell, L., Thompson, J., and Wong, P. (2001). **Discovering knowledge through visual analysis**. J. UCS, volume 7, pp 517–529.

[Torgerson W. S., 1952] Torgerson, W. S. (1952). **Multidimensional scaling: I. theory and method**. Psychometrika 1952, volume 17, pp 401–419.

[Ulbrich E. et al., 2015] Ulbrich, E., Veas, E., Singh, S., and Sabol, V. (2015). **Reading through graphics: Interactive landscapes to explore dynamic topic spaces**. HIMI 2015, pp 127–137.

[V. S. & Scharl A., 2008] V., S. and Scharl, A. (2008). **Visualizing temporal-semantic relations in dynamic information landscapes**.

[vacommunity,] vacommunity. **vacommunity (in-spire)**. `http://vacommunity.org/IN-SPIRE`. Last checked on 28.03.2019 09:00pm.