# PATH FINDING ON A SPHERICAL SELF-ORGANIZING MAP USING DISTANCE TRANSFORMATIONS

A thesis submitted in fulfilment of the requirements for the degree of Master of Science in the School of Information Technologies at the University of Sydney

Michael Bui

August 2008

© Copyright by Michael Bui 2008
All Rights Reserved

# Abstract

Spatialization methods create visualizations that allow users to analyze high-dimensional data in an intuitive manner and facilitates the extraction of meaningful information. Just as geographic maps are simplified representations of geographic spaces, these visualizations are esssentially maps of abstract data spaces that are created through dimensionality reduction. While we are familiar with geographic maps for path planning/finding applications, research into using maps of high-dimensional spaces for such purposes has been largely ignored.

However, literature has shown that it is possible to use these maps to track temporal and state changes within a high-dimensional space. A popular dimensionality reduction method that produces a mapping for these purposes is the Self-Organizing Map. By using its topology preserving capabilities with a colour-based visualization method known as the U-Matrix, state transitions can be visualized as trajectories on the resulting mapping. Through these trajectories, one can gather information on the transition path between two points in the original high-dimensional state space. This raises the interesting question of whether or not the Self-Organizing Map can be used to discover the transition path between two points in an n-dimensional space.

In this thesis, we use a spherically structured Self-Organizing Map called the Geodesic Self-Organizing Map for dimensionality reduction and the creation of a topological mapping that approximates the n-dimensional space. We first present an intuitive method for a user to navigate the surface of the Geodesic SOM. A new application of the distance transformation algorithm is then proposed to compute the path between two points on the surface of the SOM, which corresponds to two points in the data space. Discussions will then follow on how this application could

be improved using some form of surface shape analysis. The new approach presented in this thesis would then be evaluated by analyzing the results of using the Geodesic SOM for manifold embedding and by carrying out data analyses using carbon dioxide emissions data.

# Preface

Below are a list of posters related to work in this thesis and the conferences they were presented at.

- Exploring high-dimensional landscapes using Self-Organizing Maps - Asia-Pacific Symposium on Information Visualization 2007 (APVIS 2007).
- Path finding on a spherical SOM using the distance transform and floodplain analysis - Workshop on Self-Organizing Maps 2007 (WSOM 2007). The paper corresponding to this poster was published in the proceedings.
- Path Finding on a Spherical SOM for Temporal Sequence Processing - IEEE VGTC Pacific Visualization Symposium 2008.

# Acknowledgements

I would like thank to my supervisor, Masahiro Takatsuka for providing me the opportunity to do this work. His guidance, patience and generosity has been greatly appreciated.

I would also like to thank Yingxin (Christine) Wu, who was kind enough to let me use her on the Geodesic SOM and answering any questions I had.

Finally, I'd like to thank Shea Goyette, who spared his time to proofread my thesis.

# Contents

**Abstract** **iii**

**Preface** **v**

**Acknowledgements** **vi**

**1 Introduction** **1**

1.0.1 Visual analytics . . . . . 2

1.0.2 Information visualization . . . . . 3

1.1 Motivation . . . . . 7

1.2 Project Goals . . . . . 8

1.3 Contributions . . . . . 9

1.4 Thesis Structure . . . . . 10

**2 Background** **11**

2.1 Introduction . . . . . 11

2.2 Spatialization methods . . . . . 12

2.2.1 Multidimensional scaling . . . . . 13

2.2.2 Self-Organizing Map . . . . . 14

2.2.3 Geodesic Self-Organizing Map . . . . . 19

2.3 Manifold learning . . . . . 22

2.4 Shortest path problem . . . . . 25

2.4.1 Distance Transformation . . . . . 26

2.5 Summary . . . . . 28

**3 Navigating the Geodesic SOM** **29**
3.1 Background . . . . . 29
3.1.1 Portal-based rendering . . . . . 29
3.1.2 Projection methods . . . . . 32
3.2 Proposal . . . . . 36
3.3 Results . . . . . 38
3.4 Discussion . . . . . 39
3.5 Summary . . . . . 39

**4 Path planning on the Geodesic SOM** **41**
4.1 Introduction . . . . . 41
4.2 Proposal . . . . . 46
4.3 Preliminary results . . . . . 47
4.4 Discussion . . . . . 52
4.5 Summary . . . . . 53

**5 Path planning on the Geodesic SOM with floodplain analysis** **54**
5.1 Introduction . . . . . 54
5.2 Background . . . . . 55
5.3 Proposal . . . . . 56
5.4 Results . . . . . 58
5.5 Discussion . . . . . 60
5.6 Summary . . . . . 60

**6 Experimental results: Manifold Learning - Path finding on Manifolds** **62**
6.1 Two-dimensional manifolds embedded in three-dimensions . . . . . 63
6.1.1 M-SOM . . . . . 65
6.2 Results . . . . . 66
6.2.1 Recovering manifolds from a cloud of points . . . . . 74
6.3 Two-dimensional manifold of face images . . . . . 75
6.4 Discussion . . . . . 77

**7 Experimental results: Socio-demographic data** **80**
7.1 Introduction . . . . . 80
7.2 Experimental data . . . . . 81
7.2.1 Attribute space visualization . . . . . 82
7.2.2 Temporal observation . . . . . 88
7.2.3 Trajectory finding . . . . . 89
7.3 Discussion . . . . . 93
7.4 Summary . . . . . 94

**8 Conclusions and Future Work** **95**
8.1 Conclusion . . . . . 95
8.2 Future work . . . . . 96

**Bibliography** **100**

# List of Tables

6.1 The mean difference in geodesic distances computed by linear interpolation and the distance transformation algorithm, compared to Dijkstra's algorithm applied to a manifold's k-NN graph. . . . . 69

7.1 The data values for Australia's emission from 1990 to 2004. . . . . 89
7.2 The calculated attribute values for Australia between 1990 and 1991 using the distance transformation algorithm. . . . . 90
7.3 The calculated attribute values for Australia between 1995 and 1996 using the distance transformation algorithm. . . . . 90
7.4 A comparison of the distances in the feature space computed using linear interpolation and the distance transformation algorithm. . . . . 92
7.5 The calculated attribute values for Australia between 1996 and 1997 using linear interpolation. . . . . 93

# List of Figures


1.1 Example of a mosaic plot. . . . . 5
1.2 Example of a graph. . . . . 6

2.1 An example of trajectory-based SOMs being used to investigate cases of parallel development in multi-temporal census data. . . . . 18
2.2 The geodesic dome opened. . . . . 20
2.3 The result of applying a transformation to the unwrapped lattice in Figure 2.2, creating the Geodesic SOM's 2D data structure. . . . . 21
2.4 An example of a visualization produced by the Geodesic SOM. . . . . 22
2.5 The S-curve manifold. . . . . 23
2.6 A visualization of the output created by the Isomap for the S-curve. . 24
2.7 A visualization of the output created by LLE for the S-curve. . . . . 24
2.8 An example of distance transformations being used for path planning. 27

3.1 A three-dimensional view of the Geodesic SOM. . . . . 31
3.2 A corresponding first person view for Figure 3.1. . . . . 31
3.3 A first person view from the current location in the Geodesic SOM with the user in the exact same position in Figure 3.2. . . . . 32
3.4 An explanation on how a user selects two points on the Geodesic SOM to obtain a projected, two-dimensional map. . . . . 35
3.5 The Geodesic SOM after the Wagner III pseudocylindrical projection has been applied. . . . . 35
3.6 The Geodesic SOM software interface. . . . . 37

3.7 The rendering configuration for creating a projection of the Geodesic SOM. . . . . 37
3.8 The results of allowing users to interact with the projected Geodesic SOM. . . . . 39

4.1 A synthetic 3D surface representing the SOM with trajectories formed using previous approachs plotted on the surface. . . . . 44
4.2 The binary tree used to construct the synthetic data set. . . . . 49
4.3 The calculated path from node 1 to 5 on the Geodesic SOM. . . . . 49
4.4 The calculated path from node 1 to 8 on the Geodesic SOM. . . . . 50
4.5 The calculated path from node 1 to 10 on the Geodesic SOM. . . . . 50
4.6 The calculated path from node 1 to 12 on the Geodesic SOM. . . . . 50
4.7 The calculated path from node 1 to 13 on the Geodesic SOM. . . . . 51
4.8 The calculated path from node 1 to 14 on the Geodesic SOM. . . . . 51
4.9 The calculated path from node 1 to 15 on the Geodesic SOM. . . . . 51
4.10 The calculated path from node 12 to 13 on the Geodesic SOM. . . . . 52
4.11 The calculated path from node 8 to 13 on the Geodesic SOM. . . . . 52

5.1 An example where a path may travel outside a data space. . . . . 56
5.2 A description about the application of the distance transformation algorithm to the Geodesic SOM. . . . . 57
5.3 The calculated path from node 8 to 13 on the Geodesic SOM without floodplain analysis. . . . . 58
5.4 The calculated path from node 8 to 13 on the Geodesic SOM with floodplain analysis. . . . . 60

6.1 The S-curve and the sampled points used for training the Geodesic SOM. 63
6.2 The swiss roll and the sampled points used for training the Geodesic SOM. . . . . 64
6.3 The torus and the sampled points used for training the Geodesic SOM. 64
6.4 The figure-8 Klein bottle and the sampled points used for training the Geodesic SOM. . . . . 65

6.5 A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the S-curve. . . . . 66
6.6 A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the swiss roll. . . . . 67
6.7 A comparison between using linear interpolation and the distance transformation algorithm on the Geodesic SOM for the S-curve data set. . 67
6.8 A comparison between using linear interpolation and the distance transformation algorithm on the Geodesic SOM for the swiss roll data set. 68
6.9 The calculated path using linear interpolation on the S-curve. The maximum amount of temporal vertices is used. . . . . 70
6.10 A comparison of two paths on the M-SOM that are parallel on the S-curve. . . . . 70
6.11 The corresponding two paths on the S-curve for the paths depicted in Figure 6.10. . . . . 71
6.12 A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the torus manifold. . . . . 71
6.13 A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the Klein bottle manifold. . . . . 72
6.14 A comparison between using distance transformations and linear interpolation to discover paths on the torus. . . . . 72
6.15 A comparison between using distance transformations and linear interpolation to discover paths on the Klein bottle. . . . . 73
6.16 A top-down view of the torus that reveals a number of points outside of the data space. . . . . 73
6.17 The Geodesic SOM after training it with 2000 points sampled from the torus. The cluster boundaries are within the drawn rectangles. . . . . 74
6.18 A comparison between the use of the distance transformation algorithm with and without floodplain analysis on the Geodesic SOM that has been trained with the torus data set. . . . . 74
6.19 Parts (a) and (b) of this figure depict the paths on the Geodesic SOM in part (a) and (b) of Figure 6.18 on the actual torus respectively. . . 75

6.20 The path from (-0.463182, 2.081766, -0.886263) to every other data point on the S-curve. . . . . 75

6.21 The path from (5.600684, 22.194967, 12.521035) to every other data point on the swiss roll. . . . . 76

6.22 The path from (0.408809, -0.50274, 0.355076) to every other data point on the torus. . . . . 76

6.23 The path from (1.177278, -0.038267, -0.180746) to every other data point on the torus. . . . . 77

6.24 The two-dimensional data structure of the M-SOM that shows what each output image is at each neuron and how they are arranged after training the M-SOM with the data set of various face images. The blue and red trajectories correspond to the results of using linear interpolation and the distance transformation algorithm as depicted in part (a) and (b) of Figure 6.25 respectively. . . . . 78

6.25 Parts (a) and (b) of this figure are examples of using linear interpolation and distance transformations on the M-SOM for image interpolation respectively. Both results are fairly similar. . . . . 79

7.1 The Geodesic SOM after it has been trained with the carbon emissions data set. . . . . 82

7.2 The first component plane of the Geodesic SOM that corresponds to population values. . . . . 83

7.3 The second component plane of the Geodesic SOM that corresponds to total $CO_2$ emission values. . . . . 83

7.4 The third component plane of the Geodesic SOM that corresponds to per capita emission values. . . . . 83

7.5 The fourth component plane of the Geodesic SOM that corresponds to petroleum emission values. . . . . 84

7.6 The fifth component plane of the Geodesic SOM that corresponds to natural gas emission values. . . . . 84

7.7 The sixth component plane of the Geodesic SOM that corresponds to coal emission values. . . . . 84
7.8 The seventh component plane of the Geodesic SOM that corresponds to total primary energy production values. . . . . 85
7.9 The eighth component plane of the Geodesic SOM that corresponds to total primary energy consumption values. . . . . 85
7.10 A close-up view of the US' location on the Geodesic SOM. . . . . 86
7.11 A close-up view of China's location on the Geodesic SOM. . . . . 86
7.12 A close-up view of Japan's location on the Geodesic SOM. . . . . 87
7.13 A close-up view of the EU's location on the Geodesic SOM. . . . . 87
7.14 A close-up view of the Australia's location on the Geodesic SOM. . . 88

# Chapter 1

# Introduction

Various technologies today are advancing at a rapid and alarming rate. Consequently, data that is being acquired is becoming significantly more complicated to analyze and understand. This has led to a greater demand for sophisticated data analysis techniques that extract knowledge about a system, environment or phenomena that is hidden within complex data. Data mining is one such method that inherently allows users to explore and analyze large quantities of data to discover meaningful patterns or rules through automatic or semi-automatic means [1]. This process is valuable to users as patterns and rules that were not obvious to users can be discovered. A typical data mining process generates a model of the domain of interest, allowing predictions to be made on the domain. This allows better ways of handling data to be discovered while also supporting future decision making.

Olaru and Wehenkel [2] argue that the reasons for the increasing interest towards data mining are due to:

- the emergence of very large amounts of data, which could, for example, result from automated data collection
- decreasing cost of mass storage devices
- advances in computer technology
- developments in automatic learning techniques

- possible presence of uncertainty in data, such as noise and missing information

This increasing interest has resulted in large variety of applications for data mining, ranging from market basket analysis [3] to fraud detection [4]. Although there is great interest in data mining as a promising solution with numerous potential benefits, data mining applications are said to focus too much on the underlying algorithms [5, 6]. Furthermore, such products have only been suitable for statisticians and expert analysts. It has also been argued that application design should be more user-centred and that visualization may aid non-expert analysts in detecting and understanding the extracted information [7]. Moreover, visualization allows users to process the data more quickly, which is vital in time-critical situations where decisions need to be made based on overwhelmingly large amounts of data. The events that occurred during September 11, 2001 highlighted the need for tools that would meet these demands, resulting in an emerging field known as visual analytics.

### 1.0.1 Visual analytics

Visual analytics is defined as the "science of analytical reasoning facilitated by interactive visual interfaces" [8]. Visual analytics tools enable analysts to gain insight into massive amounts of data that are often dynamic and conflicting. Information about the data that is already obvious should be detected by these tools, as well as other details that may not be expected. This multidisciplinary field requires research into various techniques for:

- Analytical reasoning: to allow users to obtain insights that would support the processes of assessment, planning and decision making.

- Visual representation and interaction: which can translate abstract data into a visible form that enables users to perceive, explore, and understand large amounts of information all at once.

- Data representation and transformation: for converting all types of data to a representation suitable for visualization and analysis.

- Production, presentation and dissemination of the results: which aids in communicating the results of an analysis to a variety of audiences in meaningful ways.

One field of research that is closely related to all of these areas is information visualization, whose techniques can represent and summarize a large amount of data using a relatively small amount of display space. Patterns and trends can therefore be recognized more easily, enabling users to make decisions based on this information.

### 1.0.2 Information visualization

Information visualization has provided many useful techniques for users to analyze high dimensional data. These techniques all generate an image of some sort, using various visual features, such as colour and size, to convey meaningful information about data and their attributes. Moreover, some information visualization systems integrate the user into the process of data exploration by providing interaction capabilities.

Despite the recent efforts in automated data mining techniques, visualization still provides great benefits for the purpose of exploratory data analysis. Visualization techniques have the ability to deal with non-homogeneous and noisy data. Furthermore, users are not required to have knowledge of complex mathematical algorithms.

With the different types of data that exist, visualization techniques can be classified according to the data type that needs to be visualized. Six categories of data have been identified by Keim [9]:

1. One-dimensional data: which usually has one dense dimension, such as temporal data.

2. Two-dimensional data: which consists of two distinct dimensions. A prime example is geographical data.

3. Multidimensional data: which can be described by more than three attributes and thus can not be easily visualized. Tables from relational databases are an example of such data.

4. Text: which can not be easily described by numbers and visualized using standard visualization techniques. Transformations are often applied first before using such techniques.

5. Graphs and hierarchies: pieces of data may be related to each other. Graphs and hierarchies are able to depict such relationships.

6. Algorithms and software: large software projects are difficult to manage. Visualization can support the process of software development by aiding users in understanding algorithms, such as the flow of data in a program.

Another method of classifying visualization techniques is based on the different approaches used to generate the visualization, namely (1) space-filling, (2) graph drawing and (3) spatialization. Space-filling approaches aim to utilize as much of the display space as possible to eliminate navigation issues that may cause users to be lost in the data [10]. One notable technique is the Tree-Map [11], which attempts to utilize the entire 2D display space to map hierarchical information. This is achieved by partitioning the display space into rectangular areas, with each node being mapped to a rectangle. Despite the efforts made to make the technique more effective [12, 13], the Tree-Map algorithm is primarily used for the display of hierarchical structures, such as a file directory, and when size is the most important feature to be displayed.

The mosaic plot [14] is another space-filling technique that represents multiway contingency tables by tiles whose size is proportional to the cell frequency, in a similar fashion to a stacked bar graph. These rectangles are laid out in a mosaic, hence the name of the algorithm. Figure 1.1 displays a mosaic plot of the Titanic data set [15].

For the plots to be informative to users though, they are required to be well-trained in interpreting mosaic plots. Experiments have shown that when more than two variables are involved, the plot becomes more confusing and misleading [16]. This is a prime example that illustrates how techniques that aim to achieve a high utilization of the display space may create visualizations that are confusing and difficult to interpret.

Graph drawings are used to model relational information; that is, the relationships between data elements [17]. A graph of an email transaction network that models

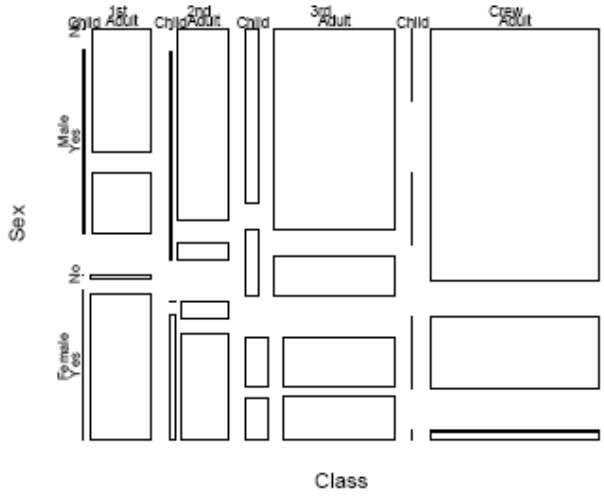


Figure 1.1: A conventional mosaic plot of Titanic data that was depicted in Figure 1 in [16] (used with kind permission of Springer Science and Business Media). The data consists of 2201 samples described by four attributes: class (1st/2nd/3rd/crew), gender (male/female), age (adult/child) and survived (yes/no).

the transfer of emails between clients and servers is depicted in Figure 1.2. The data elements are represented by the nodes in the graph, while relationships are represented by the edges between nodes. A typical graph drawing reveals the topology of the data in a pictorial forum. This allows users to interactively explore the relationships present in the data and insight into the structure of the phenomena being observed may gained. Unfortunately, graph drawings also suffer readability problems due to the features of the layout, such as size, and the extremely large amount of visible elements (nodes, edges, labels and colours) that are being displayed, even though there are techniques that may help avoid visual overload [18, 19]. These elements are mainly processed by short-term memory where capacity is limited and information is lost/forgotten quickly.

Finally, spatialization methods aim to visualize various features of the data in the form of spatial dependencies. Fuchs and Schumann [21] also argue that spatial

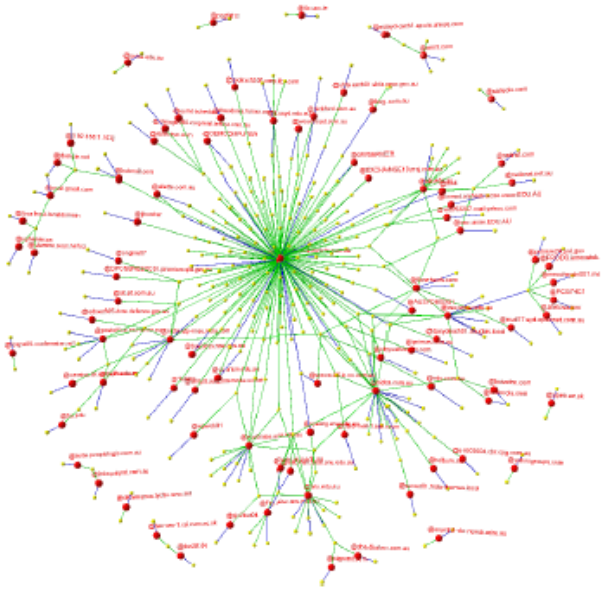

Figure 1.2: A graph of an email transaction network from [20] (©[2007] IEEE). Red nodes represent servers, while yellow nodes represent clients. Green and blue edges are used to distinguish between sending and receiving emails. The red node in the centre represents the main email server.

dependencies that are assumed to be non-existant by other techniques can be visualized along with the abstract, multivariate data. This relies on the use of spatial metaphors that exploit human spatio-cognitive skills for the exploration and analysis of non-geographic information [22]. As a result of spatializing the data, map-like visualizations are produced. Since nearly everyone has interacted with geographic maps through early education or everyday use of different maps (such as street directories or weather maps), users should be more familiar with such visualizations.

Regardless of the goals of these techniques and the various data types, researchers in the area of information visualization are ultimately faced with the challenge of creating intuitive visualizations that are easy to understand, allowing the data to be processed by the user more quickly, and interaction techniques that enables users to extract hidden and meaningful information that may be critical in various ways.

## 1.1 Motivation

Our society has been using maps as tools for exploration for a long period of time. Once a map is created, a simplified representation of some space can be obtained. This in turn reduces the complexity of discovering pathways within the space of interest. However, maps are not just used for geographical data, and the association between map metaphors and information visualization can be traced back to the 19th century [23]. This is an area of great relevance to this research, since information visualization techniques can be used to create maps of an abstract data space.

Various organizations also rely on what is known as roadmaps for the description of plans that require transitions from one state to another to arrive at a clearly defined goal. Such organizations or companies will often make decisions in the context of all previous decisions. These can be viewed as transition paths in an abstract knowledge space. Hence, if maps can be used to represent an abstract knowledge space, a path on these maps would correspond to transition paths in the knowledge space.

It would therefore be of great interest to study the usage of maps to spatialize multidimensional data onto a two-dimensional or three-dimensional space for the research presented in this thesis. These spatial visualizations are intuitive to users as concepts such as distance and direction are used to construct map-like visualizations. Moreover, most users should be familiar with geographic maps as our society relies on maps for activities such as exploration. The geographic maps that we often interact with are used to create simplified depictions of geographic spaces while attempting to preserving aspects such as area, distances and topological relationships. Map-like visualizations have a similar purpose by performing dimensionality reduction on a high-dimensional space while preserving certain characteristics of the data, such as the topology.

Although path finding is often applied to geographic maps to discover how to travel from one location to another, its applications to spatial visualizations have largely been ignored. Past research has shown that once data has been spatialized, temporal and state changes may be tracked through the use of maps. These changes would correspond to the movement of a data point in the original high-dimensional

space. Since spatial representations are, in principle, amenable to spatial analysis techniques, it follows that path finding techniques would be applicable to maps of high-dimensional spaces for discovering paths within the data space.

With the recent efforts on applying cartographic perspectives on non-geographic visualization [21, 22], it would be worth investigating the application of techniques that have typically been used on geo-referenced data to non-geographic data. Automated computational techniques should be used to perform these tasks, which can discover alternative and better paths that would otherwise remain hidden. Users could then gain more insight into the state transitions of an object or process. Research into a suitable method to achieve this objective though, would require a number of clearly defined goals to be met.

## 1.2 Project Goals

This project aims to investigate the use of information visualization methods and geometrical path planning methods to assist the process of decision making. This will involve discovering pathways in a high-dimensional feature space. In order to achieve this aim, the following goals will need to be completed:

- The selection of a suitable spatialization method for creating a map-like visualization of high-dimensional data.
- An investigation on how multidimensional path planning can be achieved once a data set has been spatially visualized through the use of a suitable technique.
- A study on the application of distance transformations to assist the path planning process using the resulting two-dimensional representation.
- An evaluation of the results to determine if the paths discovered on map-like visualizations are useful.

Above all, there must be some significance to our research, which would be of interest to the research community.

## 1.3 Contributions

There are several contributions from this research that give it some significance and distinguish it from previous work. They include:

- A critical assessment of past and related work: this will involve a review of various techniques for generating map-like visualizations and path planning methods. During this process, advantages and disadvantages of the different techniques will be made apparent.

- A proposal on how to perform path planning on visualized data: based on the results of the review, a suitable visual representation and path planning technique will be selected to provide a way for users to explore high-dimensional spaces using a two-dimensional map.

- A new application for the distance transformation algorithm: the distance transformation algorithm has primarily been used for image processing and robot motion planning. The results of our research will show that not only can the algorithm be used on grid maps representing a digital image or a robot's environment, but can also be applied on map-like visualizations describing an abstract data space.

- A new application for the Self-Organizing Map: although the main goal of this research is to assist the decision making process, a notable contribution has been made in the area of manifold embedding. Through experiments, it will be shown that the (Geodesic) Self-Organizing Map combined with the distance transformation algorithm, computes a path that attempts to approximate the geodesic path between two points on a manifold. Furthermore, these paths would allow users to gain an understanding on how the set of points on the manifold are connected for manifold reconstruction.

## 1.4 Thesis Structure

In this chapter, a small overview of what the research presented in this thesis involves and the motivation behind it has been discussed. In the chapters to follow, the background relevant to the research, new proposed methods and an evaluation of the results will be covered. Thus, the structure of the thesis will be as follows:

- Chapter 2 will provide a critical assessment of various spatialization methods. This will be followed by a review of popular path finding algorithms. All of this would culminate in the selection of a suitable method for generating a map of a high-dimensional space and a method to perform path planning on the map.
- Chapter 3 will review the methods that could be used to navigate the Geodesic SOM in order to get a better view of the visualized data.
- Chapter 4 will contain detailed information on performing path finding on maps of a high-dimensional space. This will involve the application of distance transformations to the Geodesic SOM. Here, the following will be discussed (a) key concepts, (b) important points about previous work relevant to this thesis, (c) reasons for using distance transformations and (d) a detailed explanation of the proposed approach.
- Chapter 5 will address the problem that can be seen in the results in Chapter 4. A modification to the proposed method in the previous chapter will be presented to deal with cases where the path between two points in the same cluster would travel outside the data space.
- Evaluations of the proposed methods using data sets of various manifolds will then be provided in Chapter 6.
- Chapter 7 will evaluate the proposed methods on socio-demographic data.
- Chapter 8 will conclude this thesis and discuss the overall results that can be gathered from this research. This will then be followed by a discussion on the future direction that this research will follow.

# Chapter 2

# Background

## 2.1 Introduction

Advances in computing technology and the decreasing cost of mass storage devices has allowed massive amounts of data to be collected. This has led to a demand for automated methods to be developed that can handle and use such data. Data mining is one approach that a user can employ to achieve this. The knowledge discovery from databases (KDD) process distills the information obtained from data mining into knowledge, ideas or beliefs that can be used for decision making. However, visualization can significantly improve the process of exploratory data analysis by mapping the data to visual variables, which allow users to quickly gain insight into the data. This has generated a lot of interest in information visualization, where researchers have realized the benefits of using spatial metaphors to take advantage of human spatio-cognitive skills [22].

In this chapter, a review of the previous research in these areas that is relevant to the work presented in this thesis will be covered. In the process of doing so, a critical assessment of various techniques to create maps of multidimensional data will be conducted to determine the most suitable technique for the purpose of this research. Since some of the data spaces dealt with may actually be manifolds, a review of manifold learning techniques will be provided. This will be followed by a review of path planning techniques that would be applicable to the chosen data representation.

## 2.2 Spatialization methods

Tobler's First Law of Geography [24] is an influential principle that has been the premise behind most of the work done by geographers. It states that:"everything is related to everything else, but near things are more related than distant things" [24]. Geographic maps have thus been designed to preserve distance and/or topological relationships as much as possible while also creating a simplified depiction of some geographic space. Users then rely on their spatio-cognitive skills to explore these geographic spaces. The realization that these spatio-cognitive skills can be valuable in the exploration and analysis of non-geographic information has led to the emergence of spatial metaphors for information visualization [22]. Skupin also argues that visualizations with coordinate axes that reflect on all the input dimensions are more map-like [25]. Such visualizations would be more familiar to users and this has given rise to the development of techniques for generating map-like visualizations of non-geographic data that perform dimensionality reduction and preserve certain characteristics of the information space. These are typically two-dimensional visualizations, which eliminate navigation problems that occur with three-dimensional information spaces [26, 27, 28, 29].

With the relatively recent emergence of map-like visualizations for non-geographic information, one could easily be (mis)led to believe that the value of using spatial/map metaphors has only been realized in recent times. On the contrary, maps have been envisioned for the exploration of unknown information terrain back in the 19th century [23]. Furthermore, the application of metaphors, in general, results in a mapping from a familiar, source domain $D_s$ to a target domain $D_t$. Actions that are applicable in domain $D_s$ may therefore be applicable to domain $D_t$ [30, 31]. This realization led to the successful linking between spatial metaphors and geographic concepts, where it has been argued that geographic concepts may help users perform cognitive tasks [32]. Hence, it would be worth investigating the significance of performing tasks that have usually been associated with geographic maps on map-like visualizations.

One task that some geographic maps, such as street maps, are typically used for is discovering the (shortest) path between two locations. This would require taking into

account the existence of any structures in the geographic space. Although empirical investigations on the cognitive role of distance (the distance-similarity metaphor) for non-geographic information visualization have been carried out [33], no research has been conducted to discover what knowledge can be gained by performing this task on maps of non-geographic data. The most commonly used methods to generate such maps are multidimensional scaling, principal components analysis and the Self-Organizing Map. These techniques should be reviewed in order to select a suitable data representation for this research.

### 2.2.1 Multidimensional scaling

Multidimensional scaling (MDS) is a set of methods that is used to measure similarities or dissimilarities in the data, which can be visualized in the form of a geometrical picture. While its origins lie in psychometrics, where multidimensional scaling was first proposed to help people judge the similarities of objects [34], it has also been used in a diverse range of fields such as physics and marketing [35]. The goal of multidimensional scaling is to find an optimal configuration of points based on some measure of similarity. Two similar objects are then represented by two points close together in this space, while two dissimilar objects are represented by two points that are far apart. In other words, when trying to create a mapping for data items, these methods attempt to retain the spatial relationships between data items. The similarity measures used by these techniques are either qualitative or quantitive and are used to classify MDS techniques into two categories: non-metric MDS and metric MDS respectively [36].

Metric MDS techniques attempt to preserve distances between objects. Given an $n \times n$ distance matrix $D$, whose elements are the distances $d_{ij}$ between a pair of high-dimensional points $x_i$ and $x_j$, metric MDS aims to find a configuration of points in a low-dimensional space such that the difference between its Euclidean distance matrix and the distance matrix $D$ is minimized. This is identical to principal components analysis (PCA) (also known as the Karhunen-Love or Hotelling transform) [37], which is the most popular metric MDS technique. Through the use of an orthogonal linear

transformation, PCA can transform a data set to a new coordinate system. However, metric MDS techniques require the data to be metric so that distances can be derived from the dissimilarities and are therefore not equipped to deal with ordinal-scale data.

This has motivated the need for non-metric MDS techniques [38] [39], which approximate a nonlinear transformation of the dissimilarities in the data. These techniques typically try to minimize some sort of cost function and the cost function is what differentiates the various non-metric MDS techniques from each other. While these nonlinear methods perform quite well for lower dimensional data, performance rapidly decreases as the dimensionality of the data increases. Moreover, MDS techniques only focus on finding an optimal configuration of points based on some criteria. The maps would therefore be described by a discrete set of points and are not continuous, unlike geographic maps. A comparative analysis [40] has shown that a neural network algorithm, known as the Self-Organizing Map [41], generates maps that are easier to interpret and better clustered, and may outperform other methods for high-dimensional data.

### 2.2.2 Self-Organizing Map

The Self-Organizing Map (SOM) [41] is one of the most popular and widely used artificial neural network algorithms that can be used to visualize data. The SOM consists of an array of units (neurons) that are typically arranged on a rectangular or hexagonal grid. Each unit in the SOM is associated with two-dimensional screen coordinates that can be used for visualization, and a parametric real vector $w_i$ that has the same dimensions as the input vectors. In this thesis, these vectors will be referred to as weight vectors. Before training the SOM, these values are usually randomly initialized. During the training process, input vectors are presented to the SOM and the best matching unit (BMU) is then selected. This is usually done by extensively calculating which unit's weight vector is closest to the input vector according to some (typically Euclidean) distance function. Once the BMU $c$ is found, the input vector is mapped to that unit's sublist of data points. The values of the weight vectors of the units in the neighbourhood set of $c$ are then adjusted so that they are closer to

the values of the input vector as follows:

$$w_i(t+1) = w_i(t) + h_{ci}(t)[x(t) - w_i(t)] \tag{2.1}$$

where $w_i$ is the $i^{th}$ weight vector, $h_{ci}$ is the neighbourhood function, $x$ is the input vector and $t$ is a variable in the discrete time index. Typical implementations of the SOM apply a Gaussian neighbourhood function such that the further a unit in the neighbourhood set is from the centre, the less its weight vector values will be adjusted:

$$h_{ci}(t) = \alpha(t).exp(-\frac{||r_c - r_i||^2}{2\sigma^2(t)}) \tag{2.2}$$

where $\alpha(t)$ is the learning rate of scalar value, and $\sigma(t)$ is the neighbourhood radius. Both of these are monotonically decreasing values, while $r_i$ denotes the location (screen coordinates) of unit $i$ on the map grid.

This process is applied to the all input vectors and the SOM is usually trained over a large number of epochs until convergence is reached. Additional visualization techniques such as the U-Matrix [42], P-Matrix [43] or Smoothed Data Histograms [44] can then be applied to further aid the user in understanding the structure of the high-dimensional data through visual inspection. The U-Matrix is the most commonly used technique, which visualizes the local distances in high-dimensional space, known as U-heights, in the form of a landscape or colour. Given a unit $i$ and the set of its immediate neighbours $N(i)$, its U-height $uh_i$ is defined by the following equation:

$$uh_i = \frac{1}{n}\Sigma_j d(w_i, w_j), j \epsilon N(i), n = |N(i)| \tag{2.3}$$

These U-height values can then be used directly as height values for three-dimensional visualization. A colour-based visualization on the other hand, would use these U-height values to assign each unit a colour through the use of a lookup table. For instance, given a lookup table of length $n$ (that is, there are $n$ colours), the U-height values can first be normalized between the range of 1 and $n$. The result is then rounded off to the nearest integer and used as an index value to search the lookup table to assign a colour.

The SOM is recognized for having the characteristics of (1) being able to preserve topology in the data, (2) the capability to generalize data (since the weight vectors tend to approximate the probability function of the input vectors), (3) performing multidimensional scaling and (4) unsupervised clustering. Furthermore, they are able to mimic human cognitive mappers well [45].

One of the first applications of the SOM other than the study of neuroscience was in speech recognition [46]. Since then, the SOM has been greatly appreciated in other areas and applications. Li was one of the first to integrate the SOM with geographic information systems (GIS). Since then, studies on the application of the SOM for geospatial data analysis have been particularly lacking. However, one notable exception has been the work carried out on the GeoVISTA project [47], resulting in new forms of SOM visualizations that were devised by Takatsuka [48]. In order to perform temporal analysis with census data, he proposed two methods that were used, for example, to analyze gentrification in Harrisburg, Pennsylvania over three decades:

1. Chronological cluster analysis: where each decade is represented by a different SOM. The movement of gentrification over these three decades is studied by analyzing the location of the tracts relative to each other on the SOMs for the 1970, 1980 and 1990 data in chronological order.

2. Temporal cluster analysis: where a single SOM is used to represent the data for all three decades. The gentrification phenomenon is observed by tracking the movement of the tracts over a period of time using the SOM.

Koua also studied the use of the SOM to explore geospatial data [49]. Five different visualization techniques based on the SOM algorithm were implemented to provide analysts with different methods for extracting information from large geospatial data sets. The results showed improvement of geographical analysis and promising signs that the SOM may be a valuable alternative where conventional techniques fail.

Since the SOM creates maps that preserve the topology of the data, Ultsch states that such maps can regarded as roadmaps of the data space [50]. Exploiting this property, the SOM has been used to perform temporal sequence processing. Even

if time is not considered during the training process, it is still possible to recover temporal information from the SOM. Given a temporal sequence of feature vectors, the corresponding BMUs at each point of time $t$ can be located and connected to form a trajectory.

When combined with other visualization techniques such as the U-Matrix, these so-called "trajectory-based SOMs" [51] can also be used to visualize state transitions and have found applications in areas such as speech recognition [46] and process monitoring [52, 53, 54]. By using the SOM for process modelling, it is possible to monitor the state transitions of a system through the use of trajectories, since each cluster on the map corresponds to a certain behaviour. For example, given a SOM that represents a certain system, faults can be predicted to occur by analyzing a process' trajectory on the SOM to see if it is heading toward the region of the SOM representing the occurrence of various faults [53].

This approach can also be applied when visualizing the financial state of different companies [55]. Through the use of trajectories, the authors demonstrated that it is possible to observe the evolution of a bank over a number of years to see if the bank was entering the bankruptcy zone on the SOM. Such instances would indicate that the corresponding bank was in danger of going bankrupt, which was confirmed by the data used. Comparative, statistical analyses have also shown that the SOM is superior to traditional forecasting tools [56, 57, 58]. Moreover, the problems that are associated with statistical methods can be overcome, enabling companies to run tasks such as financial benchmarking [59, 60]. Consequently, the SOM may be utilized as a tool to support strategic management. This could help executives discover which characteristics would help lead a company to reach and maintain good overall performance.

Another application for trajectory-based SOMs is the visualization of temporal, demographic changes within population census data [61]. Takatsuka demonstrated in his earlier study of the SOM on census data that these changes could be represented as movement on the surface of the two-dimensional SOM [48]. This involves training a SOM with data from multiple, different time periods. From there, one can track the movement of a specific object by determining its location on the SOM at each time

period. One problem with this approach is that the user is required to visually detect these changes. Skupin addresses this problem by explicitly visualizing these changes on top of the SOM through the use of trajectories. For a given object of interest, its state in each time period is represented by a temporal vertex on the surface of the SOM. A trajectory is then formed by connecting these vertices in chronological order. Aspects of change such as parallelism, convergence, and divergence can then be observed by analyzing the trajectories. Figure 2.1 provides an example where the SOM can be used to depict cases of parallel development. The figure depicts the development of four counties (De Witt, Gonzalez, Collingwood and Hall) on the surface of the SOM through the use of trajectories. De Witt and Gonzales are heading towards stronger income growth and a higher percentage of employees with a commute time between 45 and 59 minutes. The trajectories for Collingwood and Hall indicate a lower percentage of rural farm population while maintaining low population density and a large percentage of rural population.

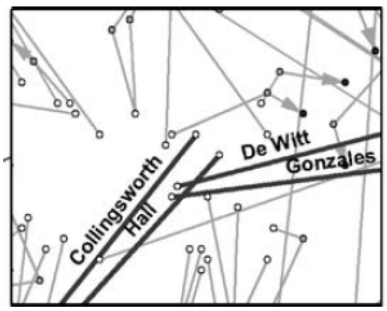


Figure 2.1: An example of the SOM being used to investigate cases of parallel development in multi-temporal census data (Figure 7 in [61]). This figure has been used with kind permission from Springer Science and Business Media.

However, there are two important, related flaws that are obvious in current applications of trajectory-based SOMs that arise when the trajectories formed are not associated with any distance metric:

1. The inability of analysts to derive from these trajectories a sense of the distance between two points on the SOM in the feature space.

2. The possibility for analysts to form a false impression of which trajectory is actually longer or shorter in the feature space when comparing different trajectories using the SOM.

Nevertheless, it is evident that the SOM in general has a wide applicability to many areas. Although wide applicability of a visualization technique is highly valued, another issue that needs to be taken into consideration is its ability to handle large data sets. With information overload being a critical problem for the World Wide Web, sophisticated tools are needed to help categorize the information, which could be used, for example, by search engines. Experiments have shown that the SOM is capable of categorizing a large Internet information space [62]. Furthermore, the user study revealed that users liked the visual and graphical aspects of the map generated. However, the details of the map become much more difficult to see as more categories are added. This led to further experimentation on the use of the SOM as a means to browse the World Wide Web [63]. By implementing a fish-eye and fractal view with the SOM in their prototype, results revealed that these two visualization techniques increase the effectiveness of the visualization. These applications clearly demonstrate that the SOM has a very high potential for mining large databases.

There is however, a notorious problem that has been associated with the SOM that is known as the "border effect". Due to the fact that neurons at the borders of these maps have less neighbours, these neurons have a reduced chance of being updated. This results in a map that appears to be less well-ordered near the borders of the map. Various suggestions have been made to eliminate the border effect problem. One of these suggestions is to implement the SOM on a spherical lattice eliminating the borders from the SOM [64]. Moreover, spherical SOMs are more suitable in handling directional data. This is a useful characteristic, since the work in this thesis is concerned with finding paths on map-like visualizations.

### 2.2.3 Geodesic Self-Organizing Map

After investigating various results from mathematicians and cartographers [65], Takatsuka and Wu came to the conclusion that an icosahedron-based geodesic dome would

be most suitable for implementing a spherical SOM [66]. Compared to the other platonic polyhedra (a tetrahedron, cube, octahedron and dodecahedron), it was observed to have the least variance in edge lengths after tessellation and resembled a sphere more than the other polyhedra.

These findings led to the development of the Geodesic Self-Organizing Map (Geodesic SOM), which maps n-dimensional data onto a geodesic dome. This eliminates the border effect while also being able to support fast neighbourhood searching on the 2D data structure that uses only O(n) space. This data structure is essentially a two-dimensional matrix (Figure 2.3) that is obtained by unwrapping the geodesic dome's lattice (Figure 2.2) and applying a transformation to it. Neurons on the boundaries of the data structure will thus have duplicates, which the Geodesic SOM keeps track of.

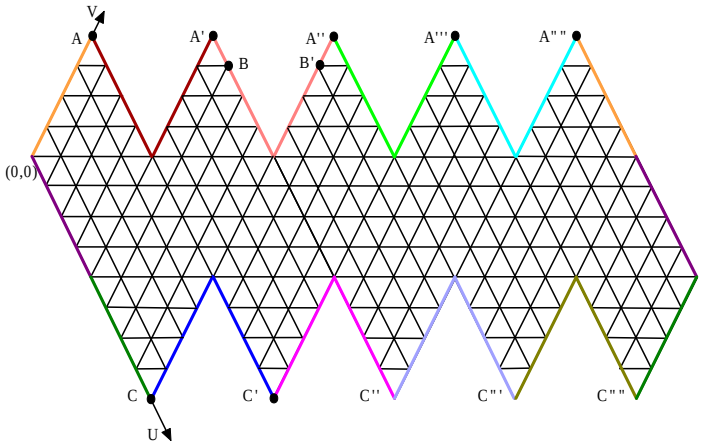


Figure 2.2: The geodesic dome opened.

Through their experiments, the authors showed that assigning two fixed extreme points on the SOM to be the north and south pole helped to provide a sense of direction that is helpful in comparing visualizations of different data sets. Furthermore, this allowed the data elements to be approximately ordered from the north pole to the south pole. Their belief was that this would aid users in the building of their mental maps. Analyses have also demonstrated that the Geodesic SOM has a more uniform error distribution and may have better performance when dealing with large

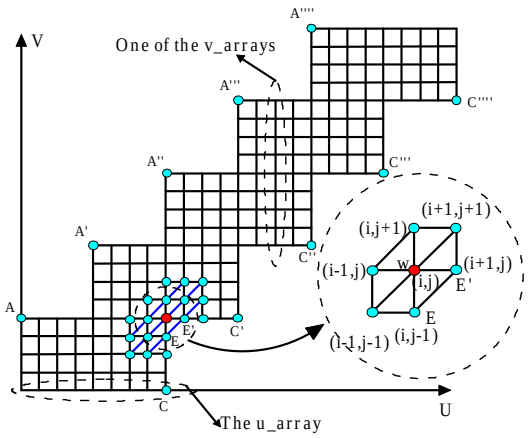


Figure 2.3: This figure illustrates the result of applying a transformation to the unwrapped lattice in the previous figure, creating the Geodesic SOM's 2D data structure. Duplicate points exist as a result of unwrapping the lattice. The diagram on the right side shows how the neighbours of a neuron can be obtained.

data sets [67]. Further work has also shown that the Geodesic SOM can be used to visualize multivariate networks using a hybrid approach [68]. This approach relies on the Geodesic SOM for multidimensional scaling and discovering the initial layout of the vertices in the network. A multivariate network can then be visualized as a graph on the Geodesic SOM through the use of graph drawing algorithms.

One final matter that needs to be discussed about the Geodesic SOM regards the issues faced with map design [22] [25]. As data sets become larger and more complex, more details need to be displayed on the map, which would occur at the cost of reduced visibility. Since users are only interested in a subset of the map at a particular point of time, the amount of information to be displayed for each region on the map can be adjusted with respect to the current area of interest. This can be achieved through the use of focus and context techniques [21]. Users also have a natural desire to place the region of interest in the centre of the screen for analysis. The spherical geometry of the Geodesic SOM addresses both of these points through its natural fish-eye effect and by allowing users to rotate the spherical visualization. These advantages are offset by the fact that a user can not get a full view of the entire data without rotating the 3D surface or projecting it onto a 2D surface. This is one of

the disadvantages of the Geodesic SOM and other three-dimensional visualizations. Complementary techniques for interacting and navigating the Geodesic SOM should therefore be considered and will be covered in detail in the next chapter.

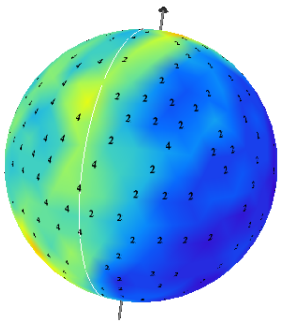

Figure 2.4: An example of a visualization produced by the Geodesic SOM. The arrow passes through the two extreme points from the south to the north pole. Regions are labelled with numbers to indicate clustering.

## 2.3 Manifold learning

Manifolds are topological spaces that are locally Euclidean. If one were to stand at any point on the manifold, it would appear as though the immediate neighbourhood is flat. In other words, manifolds can be described by lower-dimensional intrinsic/internal coordinates. The task of manifold learning is to learn these coordinates, which are embedded in high-dimensional observation coodinates. This process is interestingly analogous to dimensionality reduction.

Two popular manifold learning methods are the Isomap [69] and locally linear embedding (LLE) [70] algorithms. For the Isomap algorithm, a k-nearest neighbour graph is defined where the vertices are the set of points and the weights of the edges are the pairwise distances. All the shortest path distances in the graph are computed (for instance, Dijkstra's algorithm) and MDS is then used to find a set of point coordinates that preserves these distance constraints as best as possible. LLE on the other hand computes weights that would best reconstruct each data point from its k-nearest neighbours. These weights form a weight matrix W, and the output coordinates X

are calculated by solving the eigenvalue problem WX = X. These algorithms can then be used to create a topographic map based on the internal coordinates. For instance, a topographic map of face images can be created using the azimuth and elevation parameters of each face image. Figures 2.6 and 2.7 illustrate the results of applying the Isomap and LLE algorithm on the S-curve manifold (Figure 2.5) respectively. Source code on the Isomap [1] and LLE [2] homepages provided by the corresponding authors has been used to generate these visualizations. Both methods attempt to learn the internal coordinates on the S-curve manifold and the plotting of these coordinates effectively provides a visualization of an unrolled S-curve.

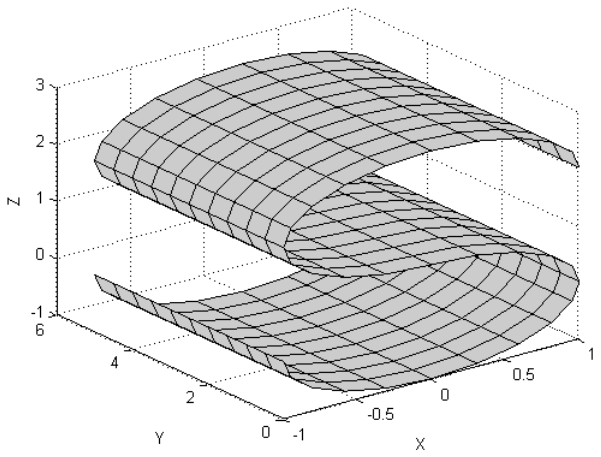


Figure 2.5: The S-curve manifold.

[1] http://isomap.stanford.edu/

[2] http://www.cs.toronto.edu/~roweis/lle/code.html

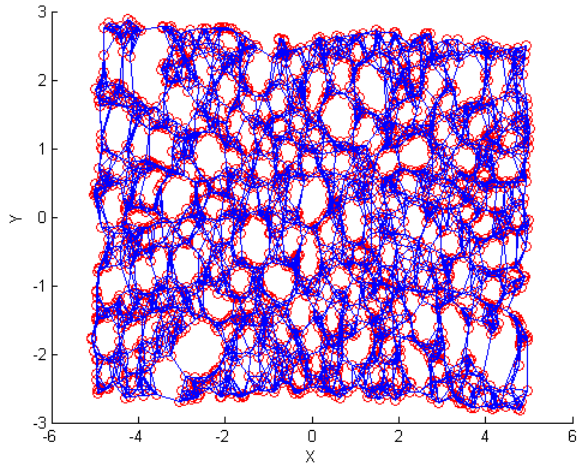


Figure 2.6: This example is a visualization of the output produced by the Isomap algorithm (k = 7) given 2000 sampled points on the S-curve manifold.

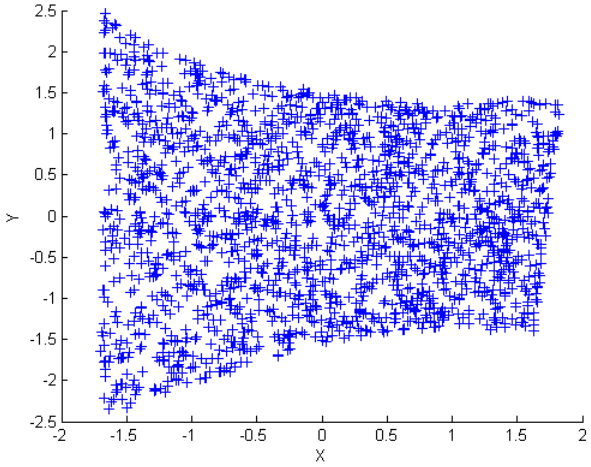


Figure 2.7: This example is a visualization of the output produced by the LLE algorithm (k = 12) given 2000 sampled points on the S-curve manifold.

While SOMs can also be used for manifold learning, it has been shown that they can fail to model a visually obvious structure in the data [69]. However, this problem

could be overcome if the internal coordinates for each data point (that is described by observation coordinates) were learnt through the use of other manifold learning methods beforehand. BMUs would then be calculated using the internal coordinates, while weight vector adaptation would be done in both the internal and observation coordinates. This variation of the SOM is known as the M-SOM [71], which shares the same advantages as the SOM when it comes to summarizing large data sets through clustering.

## 2.4 Shortest path problem

In graph theory, the shortest path problem involves finding the path between two vertices such that the sum of the weights belonging to the edges on the path is minimized. There are basically three categories of such problems:

1. single-pair shortest path problem: find the shortest path between a pair of vertices

2. single-source shortest path problem: find the shortest path from a source vertex to all other vertices

3. all-pairs shortest path problem: find the shortest path between every pair of vertices

Note that the last two problems are actually generalizations of the single-pair shortest path problem. Some of the most popular algorithms for solving these problems to calculate a graph geodesic are Dijkstra's algorithm [72], the Bellman-Ford algorithm [73, 74] and the A* search algorithm [75]. However, another popular algorithm that has been used in robot motion planning and image processing is the distance transformation algorithm [76]. In its basic form, this algorithm is structurally similar to Dijkstra's algorithm but has the advantage of being able to avoid certain areas such as obstacles.

### 2.4.1 Distance Transformation

The distance transformation algorithm was originally proposed for image processing [76]. Given a binary image $I$, which is composed of pixels that have value of either 0 or 1, the algorithm computes a distance map $D$ by propagating distance values. The pixel value of *D(i, j)* corresponds to the distance of pixel *I(i, j)*, to the nearest zero pixel.

Since distance transformations can be used to find the skeleton of images, they are widely used in pattern recognition applications. For instance, Euclidean distance transformations can be combined with thinning algorithms to find the skeleton of handwritten words such as signatures in greyscale images [77]. These skeletons can then be used for character recognition or signature verification. Other applications for distance transformations include medical image processing [78], the implementation of mathematical morphological operations [79, 80, 81] and analyses of multidimensional data sets [82, 83].

However, the most important application for the algorithm that is relevant to our research is robot motion planning. By extending the original distance transformation algorithm so that distances are propagated through the free space around the obstacles, it has been shown that the algorithm was able to solve robot motion planning problems [84]. These environments are typically represented as a discretized grid map and as a result of applying the transformation, each cell in the free space is marked with an integer indicating the minimum amount of steps to the goal. This results in a complete algorithm that is guaranteed to find an optimal, collision-free path between any two points provided such a path exists. The distance transformation algorithm is prominent for discretized grid maps, while A* [75] is commonly used in continuous, real Euclidean space maps. Figure 2.8 illustrates an example of applying the distance transformation on a grid map. A path between the start and goal cell can then be traced by following the steepest descent from the start cell.

Since distance transformations require knowledge of the environment beforehand, they are classified as global path planners. Local navigation methods, on the other hand, do not require prior knowledge of the environment. As local navigation methods can lead to being trapped in a local minima, a combination of local navigation and

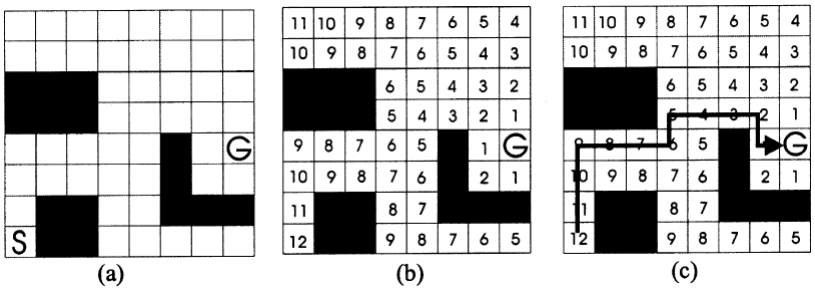


Figure 2.8: An example of distance transformations being used for path planning [85]. The map that needs to be navigated through is presented in (a), with the start and goal cells marked as $S$ and $G$ respectively. The distance transformation is applied to compute a distance map as seen in (b). The shortest path between $S$ and $G$ can then be found by following the steepest descent, as seen in (c)

global path planning methods has been proposed to solve some of these problems. The distance transformation algorithm could, for instance, be combined with the potential field method [86]. In this situation, the robot is represented by a point in configuration space, and becomes a particle under the influence of goal configuration and obstacle configuration. The resulting gradient of total potential then becomes an artificial force that is applied on the robot to guide it towards the goal.

Further work with distance transformations has shown that it can even be applied successfully in unknown environments, where sentries may exist, to discover paths that would minimize exposure to such sentries [87]. By using quadtrees for cell decomposition, it is also possible to use the distance transformation algorithm to find paths in situations where there are multiple goals that are considered equal [88]. This approach has been able to combine the advantages of both high resolution grid-based and quadtree-based approaches for computing Euclidean shortest paths in a 2D environment.

Due to the nature of these path planning problems, these algorithms are designed to be fast so that robots can react quickly, especially in dynamic environments. Interestingly, the robot motion planning approach has even been applied to study protein folding with some success [89]. In this case, the native fold is assumed to be known

so that the folding process can be studied. Although distance transformations were not used in these applications, it is interesting to see that techniques for finding a path from one point to another can be used in complex applications. Furthermore, using a high level, graph-based approach known as the image foresting transformation (IFT) [90] allows one to not only implement the distance transformation, but other image operators such as the watershed transformation, which is used for image segmentation.

## 2.5 Summary

In this chapter, spatialization, manifold learning and path planning methods were closely examined. The advantages and disadvantages of techniques in these areas were discussed with respect to the research described in this thesis. In the next chapter, methods for navigating the Geodesic SOM will be discussed to allow analysts to interact with the visualization in different ways.

# Chapter 3

# Navigating the Geodesic SOM

The Geodesic SOM's inherent spherical structure provides a natural fish-eye effect that allows users to focus on regions of interest by rotating the spherical visualization. However, its spherical geometry does not permit users to see the entire visualization unless the geodesic dome is rotated or flattened. Users may thus be lost in the data when trying to navigate the visualization. Techniques are hence required to complement these visualizations in order to enhance the way users interact with them. In this chapter, techniques that could be or have already been applied to the Geodesic SOM for navigation will be reviewed. This will lead to a proposal that enhances a previous approach that was used to create a two-dimensional projection of the Geodesic SOM.

## 3.1 Background

### 3.1.1 Portal-based rendering

The concept of portal rendering was first proposed by Jones for hidden-line removal [91]. The algorithm involves the division of models into convex, polyhedral cells bound by opaque walls or transparent, convex and polygonal portals. The fundamental idea behind portal-based rendering is that a room (cell), can only be visible if the "user" is inside it or if it can been seen through a doorway (portal). Through the process of

scene decomposition, a Cell and Portal Graph (CPG) can be created, which describes the adjacency information between cells. In this situation, cells are represented as nodes, while the edges of the graphs would correspond to portals that connect these cells. A visible set of cells would then be determined by traversing the CPG from the cell containing the eye point and following the edges that represent visible portals.

Early work on portal-based rendering concentrated on exact visibility determination. Later developments provided optimizations to Jones' original algorithm that resulted in fast visibility determination techniques that would compute what the viewer could *potentially* see from the cell they were in. A variation of Jones' algorithm, for example, was used to determine Potentially Visible Sets (PVS) in dynamic architectural models at runtime by using the screen-space bounding rectangles of projected portal geometry [92].

Recently, a general paradigm for portal-based rendering was proposed where the portals could be non-convex and non-planar [93]. This provides a more flexible framework for dynamic scene composition and more applications where portal-based rendering can be used. In contrast to previous work, this paradigm simply states that cells contain data that are connected by portals. Moreover, there are no binding geometric constraints. Meaningful, dynamic and visual links may thus be created between related data in a visualization environment, such as the the Geodesic SOM. Novel scene construction is also possible through portal-based rendering, by using transformative portals.

Portal-based rendering can therefore be used as a generic projection technique that is applicable to the Geodesic SOM and other three-dimensional closed surfaces [94]. These surfaces would need to be triangulated such that the cells (the triangles) are connected by transformative portals at each shared edge. This results in a generic technique for flattening three-dimensional closed surfaces that could easily be integrated into existing work. Furthermore, the application of this technique can be used to create what would essentially be virtual worlds of the high-dimensional space. Users may then explore the world from a first person perspective.

Figure 3.1 depicts the spherical SOM that the user is exploring from a first-person perspective. The user's current location is indicated by the white triangle in the figure

and facing towards the north. A first-person view without the use of transformative portals is illustrated in Figure 3.2, where it is clear that not much of the surface can be seen. The results of using transformative portals in this situation is displayed in Figure 3.3, where more of the surface is visible.

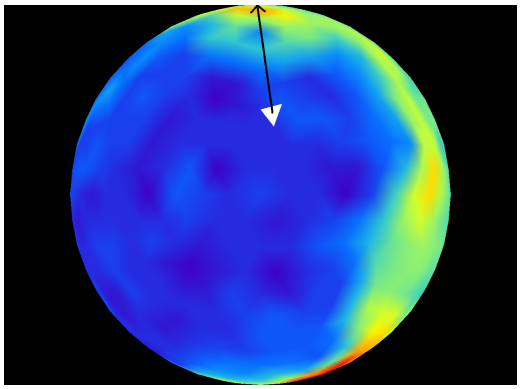

Figure 3.1: A three-dimensional view of the Geodesic SOM. The white triangle is used to indicate which cell the user is currently located in. Note that the user is facing towards the north direction.

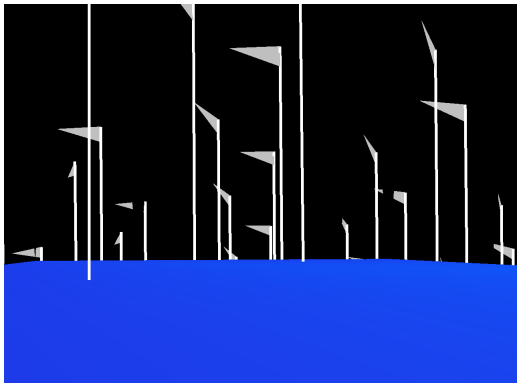

Figure 3.2: A corresponding first person view for Figure 3.1. Here only non-transformative portals are used. To measure the visibility, flags have been used, of which not many can be seen.

This approach is able to preserve the spatial dependencies in the data and can also be considered as a focus and context technique, since areas that are distant

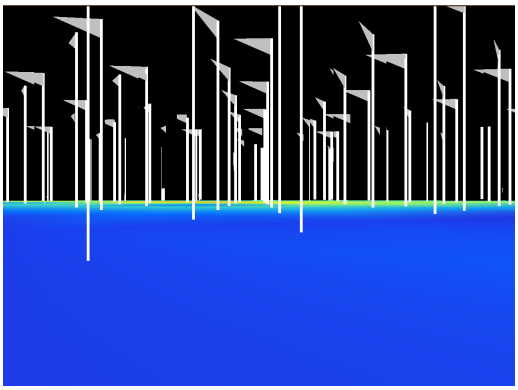

Figure 3.3: A first person view from the current location in the Geodesic SOM with the user in the exact same position in Figure 3.2. Here portal-based rendering is used with transformations enabled, resulting in a flat surface. Although portal artefacts can "mask" the distant flags, the results show that more flags are visible. This indicates that the flat surface allows the greater surface coverage.

from the current area of interest would naturally be distorted. Nevertheless, this novel technique can not generate a clear, full view of the visualization due to the limitations of perspective projections.

### 3.1.2 Projection methods

People are accustomed to rotating three-dimensional objects, such as a globe, in order to view other areas of an object. Since users need to keep track of areas that have already been visited, there is the possibility that users may lose track of these areas and tasks such as rotating objects may be an inconvenience. It would be more convenient to be able to view the entire object of interest, which would be possible if such an object was two-dimensional. Through the use of projection methods, two-dimensional representations of a high-dimensional space may be created. One of the major areas involved in the development of such techniques is cartography. Through the use of different mathematical equations, two-dimensional maps of the Earth can be created. Another approach that relies on computer graphics is multiperspective imaging. This approach results in an image containing different views of the same data. The following subsections will elaborate more on these methods.

### Multiperspective imaging

Human perception has long been known to have limitations in its ability to perceive the world. One of these shortcomings is that we can only see things in front of us due to our limited field of view. Visually inspecting a 3D object would therefore require us to rotate it to see the back of the object. However, we can easily overcome this problem by capturing multiple views of a single object to form a single image. This has become known as multiperspective imaging [95].

While multiperspective imaging may produce very interesting images, there are a few drawbacks to using this approach. Undesirable distortions can be introduced and the combination of multiperspective imaging with ray tracing for 3D scenes results in poor rendering performance, making it unsuitable in interactive environments [96].

### Cartography

The area of cartography has intrigued cartographers, mathematicians and navigators for over 2500 years [97, 98]. This long period of research has led to the development of hundreds of map projection techniques for various purposes. Their applications range from creating maps using large-scale data in a limited area, to a small-scale map of the world.

Two common methods for classifying map projection techniques are: (1) through the properties that they attempt to preserve, or (2) the surface used to project the map onto. If a projection technique is classified by what is known as its developable surface, they are either conic, cylindrical or planar. All of these techniques project the points of the spherical object to the surface of the selected geometric shape. These surfaces are then made flat to produce a 2D map, which requires the cut points on the surface (if appropriate) to be defined in order to open the surface up. The open surface would then be laid out to produce a flat map.

On the other hand, there are four types of projections based on the property being preserved: conformal, equal area, equidistant and true-direction projections. Conformal projections preserve the local shape of the data by maintaining all angles. Equal area projections preserve the area of displayed features while compromising

other properties of the data. Equidistant projections preserve distances between certain points. Finally, true-direction projections, also known as azimuthal projections, maintain the directions of all points with respect to the centre. Note that azimuthal and planar projections refer to the same group of projections.

In order to relate the spherical coordinates on the spherical object to coordinates on the flat planar object, mathematical formulas are needed. Furthermore, distortions are unavoidable and occur in either the shape, area, distance or direction of the data. Various techniques have been designed to minimize certain distortions. The only projection technique known to have no visible distortions is the Fuller Projection [99]. However, this technique was designed with the Earth in mind, which needs to be represented as an icosahedron with 20 faces.

Regarding the Geodesic SOM [66], one of its disadvantages is that a user can not see the entire visualization at once. This is not an issue with the traditional SOM, since it is already two-dimensional. Consequently, the authors used the Wagner III pseudocylindrical projection technique to obtain a two-dimensional, projected map of the Geodesic SOM. In order to let users define how the Geodesic SOM is to be projected to 2D, an interface is provided so that users can define the two points that are necessary for the projection to be done. Users select two points A and B, where A is the centre of the projection and together with B, defines a plane going through the centre of the sphere. This is used to determine the north and south poles of the sphere. The geodesic arc through the two poles is then used as the split line, which is where the geodesic dome is cut open. This procedure is described in Figure 3.4. Figure 3.5 illustrates an example of what the Geodesic SOM looks like after the Wagner III pseudocylindrical projection is applied.

The equations used to convert the spherical, longitude and latitude $(\theta, \phi)$ coordinates to 2D coordinates $(x, y)$ are described below. Here R is the radius of the sphere, which is 1 in this case, while c = 0.5 and p = 0.5.

$$x = R\frac{\beta\theta}{\sqrt{\alpha\beta}}\cos(\alpha\phi) \tag{3.1}$$

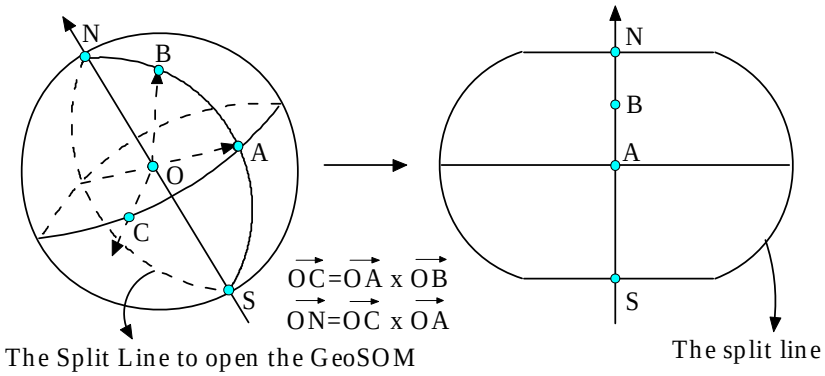


Figure 3.4: This diagram depicts how the process of projecting the Geodesic SOM to 2D works. Here users are required to select two points, A and B, in order to define where the geodesic dome should be cut open to produce a flat map.

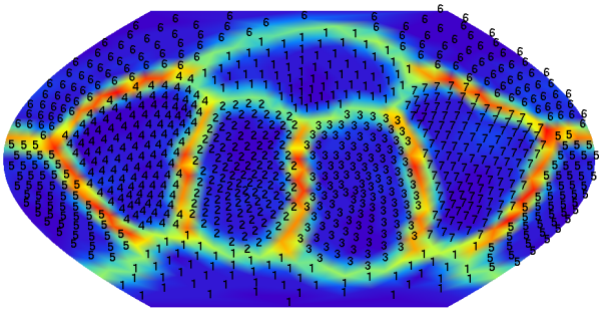

Figure 3.5: The Geodesic SOM after the Wagner III pseudocylindrical projection has been applied. Users are now able to see the entire Geodesic SOM without having to rotate the original geodesic dome.

$$y = R\frac{\alpha\phi}{\sqrt{\alpha\beta}} \tag{3.2}$$

$$\alpha = \frac{2\arccos c}{\pi} \tag{3.3}$$

$$\beta = \frac{\alpha}{2p} \tag{3.4}$$

The authors cite the reason for using the Wagner III pseudocylindrical projection

as being able to obtain a "more balanced representation of the spherical map's main features". The disadvantage of their approach however, is that the user may not obtain a satisfactory projection, since they are unable to tell what the projection would look like when selecting the two required points beforehand. Moreover, in order to change the region of interest so that it is located in the centre, the user must specify a different pair of projection points. Conversely, with the spherical Geodesic SOM, the user only needs to rotate the visualization.

## 3.2 Proposal

To combine the advantages of being able to view entire data and change the region of interest, one possible solution is to create a two-dimensional projection of the Geodesic SOM that can be "rotated" like the spherical Geodesic SOM. This can be achieved by creating a new two-dimensional projection when the Geodesic SOM is rotated. This can be implemented by rendering the Geodesic SOM (Figure 3.6) a certain distance away from the foreground with the colour mask disabled so that the Geodesic SOM is invisible. Figure 3.7 describes this rendering configuration. The user's eyes are looking toward the image screen that displays the two-dimensional projection of the Geodesic SOM. Unbeknownst to the user, the invisible, spherical Geodesic SOM is actually located behind the two-dimensional projection in world space. The spherical Geodesic SOM is used to compute a two-dimensional projection that is rendered in front of it.

Two 2D fixed points (A and B in Figure 3.6) are then specified so that the cut points used to determine the split line are actually the current north and south pole that the user sees. A projection is then created and rendered in front of where the Geodesic SOM is located in world space. A new projection would only be created when the user interacts with it in the same way the spherical Geodesic SOM is rotated. This process is summarized below (Algorithm 1) and users rotate the projection by left clicking on a focal point on the SOM and moving the mouse (with the left mouse button is kept pressed after clicking) in the direction of rotation. Each mouse movement would trigger the creation of a new projection that is displayed and as a

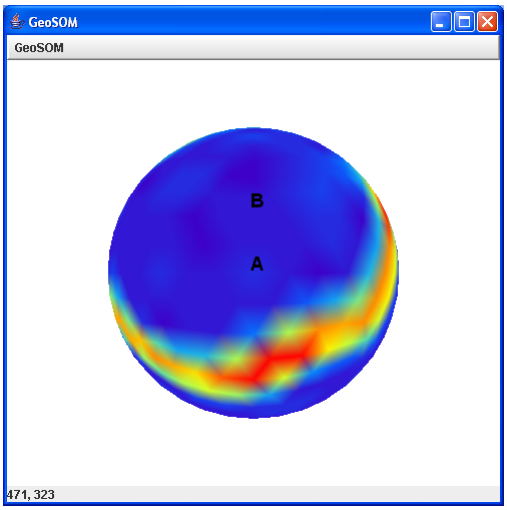


Figure 3.6: This image depicts what the Geodesic SOM interface looks like with the Geodesic SOM visible. The labels A and B indicate the position of the two 2D fixed points used to determine the split line and create a two-dimensional projection of the Geodesic SOM.

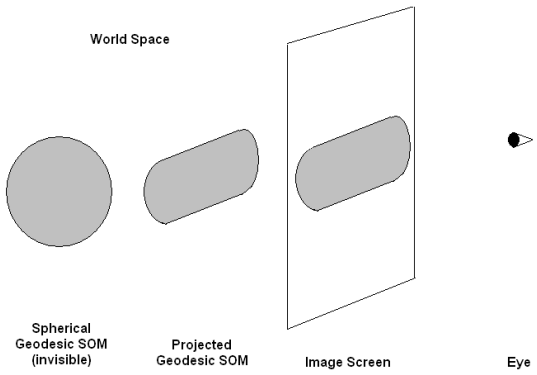


Figure 3.7: This schematic diagram describes the rendering configuration for creating a projection of the Geodesic SOM. The user uses their eyes to view the two-dimensional projection on the image screen. In the three-dimensional world space, the spherical Geodesic SOM (which is invisible to the user) is located in the far back (with respect to the direction the eye is looking at) and its corresponding two-dimensional projection is rendered in front of it.

result, creates a smooth animation of the rotation process.

**Algorithm 1** Navigating the Geodesic SOM.

**Require:** Two fixed 2D points A and B in screen coordinates
  Render Geodesic SOM
  **if** Map projection is not done OR Geodesic SOM is rotated **then**
    Convert points A and B to 3D world coordinates that represent points on the surface of the dome in world space
    Create a new map projection using the converted points as cut points
    Map projection is done
  **end if**
  Render projection

## 3.3 Results

In this section, the results of approach proposed above to lets users "rotate" the two-dimensional projection of the Geodesic SOM is shown. For the experiment, a simple synthetic data set corresponding to the distance matrix of a binary tree is used to train the Geodesic SOM. More details on the data set will be discussed in the next chapter as they are not relevant here. A two-frequency geodesic dome (42 neurons) is used for training (over 150 epochs) with an initial update radius of 3 and an initial learning rate of 0.8.

Figure 3.8a depicts the results after training the Geodesic SOM with the data set. If we assume that the red cluster border is an area of interest to the user, then its location should ideally be in the centre for the user to focus on. However, Figure 3.8a illustrates that it is currently located near the border. Consequently, it is difficult to analyze how the data has been mapped around the red cluster border. This problem can be solved by creating a new Wagner III projection of the Geodesic SOM as the user interacts with the spherical Geodesic SOM as proposed above. In this case, the user must rotate the two-dimensional projection to the left until the red cluster border is located in the centre of the screen. This results in the map projection depicted in Figure 3.8b. Since the red cluster border is now in the centre, this makes it easier to analyze the data located around it.

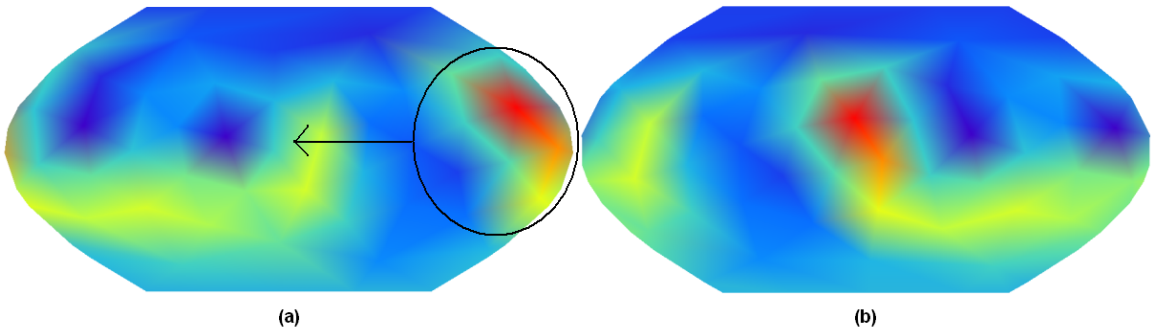


Figure 3.8: This image depicts the results of allowing users to interact with the projected Geodesic SOM. In the original visualization (a), the circled, red region is located near the edge of the map. To obtain a clearer view of the data mapped around the red region, the red region should be moved in the left direction so that it is position in the centre of the map (as indicated by the arrow). The visualization after the the user has actually rotated the spherical Geodesic SOM (in the background) to move the red region to the centre is depicted by part (b) of the figure.

## 3.4 Discussion

In the results above, it is clear the approach that has been proposed enhances the way users can interact with the Geodesic SOM. The original use of the Wagner III cartographic projection technique proposed by Wu and Takatsuka [66] created static maps that would allow users the see the entire visualization. However, if the data region of interest was not located at a desirable location, users were required to manually create new map projections until a satisfactory result is obtained. With the spherical Geodesic SOM, users were able to rotate the geodesic dome until the region of interested was located in the centre. By creating a new map projection every time the user rotates the geodesic dome, this allows users analyze the relationships within the data more easily.

## 3.5 Summary

The borders present in two-dimensional SOMs creates problems when data analysis is required. When data samples of interest end up being mapped to the neurons on

these borders, it is more difficult to analyze how these samples are related to the rest of the data. Since the Geodesic SOM has no boundaries, more information about the relationships within the data can be revealed. However, its spherical structure does not allow users to see the entire visualization at once. While the use of cartographic projection techniques can create a corresponding two-dimensional map for the Geodesic SOM, users are no longer able to rotate the Geodesic SOM to focus on different regions of the visualization.

In this chapter, a solution for navigating the Geodesic SOM was presented. It allows users to see the entire visualization while also being able to change the data region that needs to be focused on through rotation. Since users are able to "rotate" the projection, it is no longer necessary to continually specify different cut points until a satisfactory projection is reached. Users are thus able to interact with the projection using a more intuitive approach that combines the advantages of the spherical Geodesic SOM and its corresponding two-dimensional projection.

Having discussed the various methods that could be used to navigate the Geodesic SOM, the next chapter will focus on how to apply path planning techniques to the Geodesic SOM.

# Chapter 4

# Path planning on the Geodesic SOM

In this chapter, the proposal to use the Geodesic SOM with distance transformations to discover pathways in a high-dimensional space will be presented. The ultimate objective of this research is to be able to support the process of decision making. The approach used will be covered in a detailed section. This will be preceded by an assessment of previous research that is significant to the work presented in this thesis.

## 4.1 Introduction

In the editorial of the first Information Visualization (IVS) journal, Chen stated that "information visualization can be broadly defined as a computer-aided process that aims to reveal insights into an abstract phenomenon by transforming abstract data into visual-spatial forms" [100]. By transforming abstract data into a spatial representation, these spatial representations would, in principle, be amenable to spatial analysis techniques. Consequently, there has been a desire to apply geographic notions and principles to non-geographic information [32, 101]. This may assist in generating better visualizations of complex data that also have the advantage of being accessible, since we all have an "instinctive understanding of geographic relationships" [32]. The

need for users to be trained in making sense of these geographic representations would thus be minimized. Moreover, metaphors generally help relate two different domains to each other. They create mappings between a source domain and target domain such that thoughts and actions applicable in the source domain are also appropriate in the target domain. Using geographic metaphors can therefore open the door to using knowledge from the geographic domain.

For example, once data has been spatialized, it is then possible to track temporal changes in high-dimensional space [55, 61, 102]. This is similar to tracking the movement of a vehicle through the use of a map. The process requires temporal vertices to be created at each interval of time on the SOM. These vertices represent the state of the object being tracked at each time interval. A trajectory is then plotted by connecting these vertices together to depict the object's movement on the surface of the SOM.

Although the SOM's potential in geographic visualization and geocomputation has been discussed [61, 102], results from trajectory-based SOMs thus far have carried little actual meaning in the geographic domain. Given a trajectory on a geographic map that describes the path between two points, it is possible to derive the intermediate points that were reached or would need to be passed through. However, trajectory-based SOMs in general lack the capabilities that would enable analysts to gain insight into such information. When visualizing demographic trajectories, it has been suggested that additional temporal vertices can be inserted to depict the potential intermediate points [61]. Given two feature vectors corresponding to two different points in time, the values are linearly interpolated under the assumption that demographic developments are linear. The BMU for each n-dimensional vector calculated through linear interpolation is then inserted as an additional temporal vertex. However, there are still problems that have been identified with the approach used here and trajectory-based SOM applications in general:

- It has been argued that temporal information can be recovered from trajectory-based SOMs even though the SOM does not have knowledge of the temporal relations during training [51]. This implies that temporal information has been lost during this process. However, the "recovery" of this knowledge is merely

the visualization of all the *known* relations as trajectories.

- Trajectories on the surface of the SOM may not be indicative of the distance travelled and the intermediate points passed through in the feature space. Consequently, users can not determine which trajectory is longer or shorter in the feature space even if additional temporal vertices are inserted to allow analysts to make more informed judgements about the relationship between trajectories [61].

- The number of temporal vertices to add through (linear) interpolation is determined by the user. When finer details are required, this can become a tedious process for users when the SOM already performs (nonlinear) interpolation on the data. Although the insertion process can be automated by adding more temporal vertices until no new neurons are added, problems may arise with this approach as experiments will show later on.

- Many applications deal with data sets that are structurally nonlinear, where the use of linear interpolation may not be appropriate.

Figure 4.1 depicts a trajectory that has been plotted using previous approaches on a 3D heightmap visualization of the SOM. These types of visualizations were employed by Takatsuka for performing temporal analyses on geospatial data [48]. The heights of each neuron correspond to the local distances in the high-dimensional space that is calculated when constructing a U-Matrix. Furthermore, the geodesic distance between two points on these synthetic 3D surfaces generally correspond to their distances in the feature space (as mentioned by Takatsuka). Recent studies have shown that users expect that distances between points on spatializations to correspond to high-dimensional similarities [33]. However, the approaches used in trajectory-based SOMs thus far have either ignored or not addressed this point, including the work by Skupin and Hagelman. Although the form of the trajectories in their work is derived from cognitive plausibility [103], the comparison of path lengths is actually ill-advised. The methods that have been used would therefore create trajectories that are unnatural in the sense that they do not follow the artificial U-Matrix landscape

and pass through mountain ranges as seen in Figure 4.1. These regions correspond to large local distances and may potentially be perceived as obstacles that should be avoided when possible. Suppose there are two trajectories that appear to be parallel and close to the same length, such as those in Figure 4.1. One trajectory may actually travel a shorter distance in the feature space, that is, undergo fewer state transitions. An accurate representation of the path with the shorter distance in the feature space could, for example, be a trajectory that avoids the mountain range on the artificial landscape.

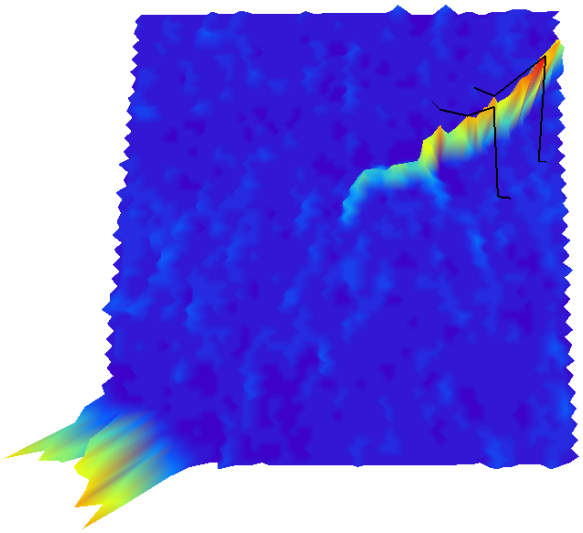

Figure 4.1: This figure depicts how trajectories created using methods in past work would appear on a synthetic 3D surface representing the SOM. Two parallel trajectories are depicted that appear to be of the same length when in actual fact, one trajectory may have a shorter distance in the data space.

The use of automated computational techniques can address these points by systematically connecting two temporal vertices and the potential intermediate vertices between them. This can be achieved by applying path finding algorithms on the SOM, which is a combination that has previously been suggested for process steering applications [52]. Geodesic distances on the surface of the SOM can then be computed. Furthermore, such techniques would actually recover temporal information, since they attempt to determine the intermediate states that were reached.

An important question is raised by applications that involve tracking temporal

changes. If a map describing an abstract data space can be used to track changes within this space, can these maps be used to discover the path between two points in the high-dimensional space? Most people in society are accustomed to using a map to discover the path between two geographic locations. With the recent efforts encouraging the use of cartographic and geographic metaphors to non-geographic visualization, the potential of using map-like visualizations for such purposes is certainly worth exploring. The utilization of automated computational techniques to solve these path planning problems could generate alternative pathways that may otherwise remain hidden and possibly be more optimal. With the analogies between path planning and decision making, there is the potential that the discovered paths may aid in the process of decision making. For instance, path finding algorithms could help steer a process to an optimal state while avoiding any defined forbidden states that have been marked on the SOM [52].

In order to create maps that spatialize the data and can be used for path finding, the algorithm used to generate such maps need to be considered. The SOM is a popular artificial neural network algorithm that is well established and has already been used in a large number of applications. Furthermore, its ability to produce a topological mapping of a data space has already been exploited to visualize the state transitions of an object through the use of trajectories on the surface of the SOM [55, 104, 61]. If the SOM can be used to track temporal changes in high-dimensional space, then it could potentially be used to find paths between two points in high-dimensional space. This would take advantage of the SOM's ability to approximate the distribution of the data. Therefore, if a suitable path finding algorithm is used, the neurons that lie on a path between two points on the SOM could potentially represent the intermediate states that were reached or need to be reached. These states can then be derived by analyzing the weight vector values of each intermediate neuron. According to our knowledge, very little work has been done to take the next step further to go from tracking paths to finding paths with SOMs of non-geographic data.

Another problem worth considering is the border effect inherent in traditional two-dimensional SOMs, which affects the accuracy of the mapping. The Geodesic

SOM is a variant of the SOM that uses a spherical lattice of neurons to remove the border effect. It has been shown to have a more uniform error distribution and the potential for better performance when dealing with large data sets. The latter point is particularly important as data sets from certain domains can be extremely large, such as financial data sets concerning the stock exchange. Therefore, it can be concluded that the SOM is the most suitable technique for generating maps of high-dimensional data, since it has been demonstrated that they can be used for tracking temporal changes. Moreover, the Geodesic SOM is the most appropriate variant to use, which eliminates problems (such as the border effect) that occur when using the conventional SOM.

## 4.2 Proposal

In order to reduce the complexity of performing path planning when given multi-dimensional data, the Geodesic SOM is first used for dimensionality reduction and visualization. Each neuron is associated with a distance value (initially -1) representing the shortest distance from that neuron to the goal. However, to compute the shortest distance between two neurons on the surface of the SOM, a distance metric must be defined.

The Geodesic SOM is visualized by constructing a U-Matrix. This requires calculating the local distances in high-dimensional space, which determines the colour used from a lookup table containing 30 colours. In this case, as the distance increases, the colours will change linearly from blue to cyan, to yellow, and finally to orange. Generally speaking, this means that the value of the red colour component will increase. Takatsuka has pointed out that the geodesic distances on the U-Matrix landscape could be perceived as the distances in the feature space [48]. The path between two points on these landscapes could therefore correspond to the path in the original data space. These landscapes are constructed by using these local distances as height values for each neuron and are referred to as U-heights [50]. The U-height $uh$ of each neuron is the average distance of the neuron's weight vector to the weight values of its immediate neighbours. The use of U-heights would therefore be appropriate to

use as distance values and can aid in constructing more accurate trajectories in the high-dimensional space.

To compute a distance map for path finding using these U-heights, distance transformations are used to propagate distances from one neuron to another. Note that for diagonal neighbours, the actual value that should be propagated is $uh\sqrt{2}$, according to Pythagoras' Theorem. Propagation then begins from the goal neuron, which has a distance value of zero. As there are duplicate neurons in the Geodesic SOM's data structure, when a neuron has its distance value updated, its duplicates will also need to be updated to have the same value. Fortunately, the Geodesic SOM keeps track of each neuron's duplicates, which eliminates the need to search through all the neurons to discover any duplicate neurons that need to be updated. Once the distance transformation is complete, the path can then be obtained by following the steepest descent from the starting neuron. The algorithm itself (Algorithm 2) is similar to Dijkstra's algorithm but also has the benefit of be able to discover paths around obstacles. These obstacles may represent marked, forbidden areas on the SOM as described by Tryba and Goser [52], who suggested that path finding algorithms could be use for process steering. Furthermore, the approach presented here is similar to those used in raster GIS where given a start and end point, an accumulated cost surface is defined and path finding algorithms can be used to calculated the shortest path [105].

Since the U-Matrix visualizes the average distance of a neuron to its direct neighbours, the path could in some cases be confirmed to be correct through visual analysis. For example, if two end points were on opposite sides of the mountain on the U-Matrix landscape (see Figure 4.1), analysts could determine if the path is visually correct by seeing if it navigates around the mountain.

## 4.3 Preliminary results

In order to test how the proposed approach works, it is necessary to run a small experiment. In this case, a synthetic data set was used which corresponds to a binary tree of depth 3, where there are a total of 15 nodes in the tree (see Figure 4.2). The

**Algorithm 2** Distance Transformation on the Geodesic SOM

```
Require: start and goal and Geodesic SOM
  Initialization
  Queue Q is empty
  for all neuron in neurons do
    neuron.distance = -1
  end for
  goal.distance = 0
  for all n in goal.sameVerticesList do
    n.distance = 0
  end for
  Q.push(goal)
  Start distance transformation
  while Q is not empty do
    current = pop(Q)
    neighbours = getNeighbours(current)
    for all neighbour in neighbours do
      d = neighbour.uh√2
      if neighbour.distance == -1 then
        neighbour.distance = current.distance + neighbour.avg_diff
        Q.push(n)
      else
        if neigbour is a diagonal neighbour and n.distance > current.distance + d
        then
          neighbour.distance = current.distance + d
          Q.push(n)
        else
          if neighbour.distance > current.distance + neighbour.uh then
            neighbour.distance = current.distance + neighbour.uh
            Q.push(n)
          end if
        end if
      end if
    end for
    for all v in neighbour.sameVerticesList do
      v.distance = neighbour.distance
    end for
  end while
  End of distance transformation
```

data set consists of 15 data samples described by 15 attributes and is essentially the tree's 15 x 15 distance matrix. If we denote this distance matrix as $M$, the value of $M(i, j)$ corresponds to the graph distance from node $i$ to node $j$. If the value of $M(i, j)$ is zero, this means that node $i$ and $j$ are actually the same node.

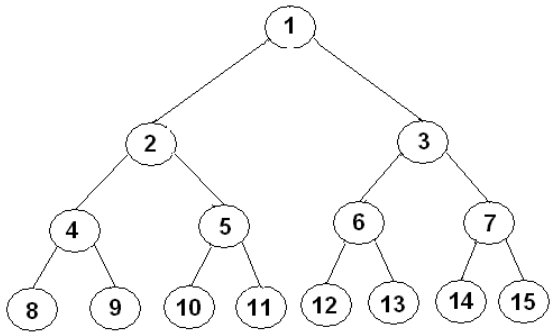


Figure 4.2: The binary tree used to construct the synthetic data set.

The training parameters were such that a two-frequency geodesic dome was used (42 nodes), while the initial update radius was set to 3, and the Geodesic SOM was trained for 150 epochs. The results of calculating paths from the root to all other nodes are depicted in Figures 4.3 to 4.9. Note that there are two circled red labels which are 8 and 10. This indicates that there was another node mapped to that same neuron as well, which are nodes 9 and 11 respectively. This is not a problem since the clustered nodes share the same parent.

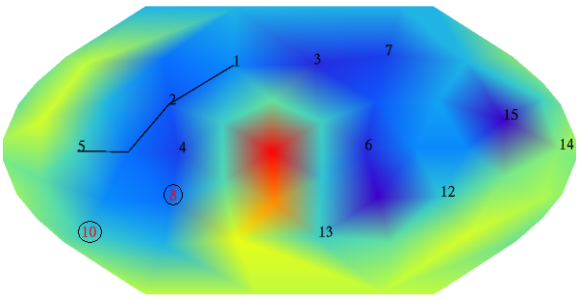


Figure 4.3: The calculated path (1-2-5) from node 1 to 5 on the Geodesic SOM. The path is correct.

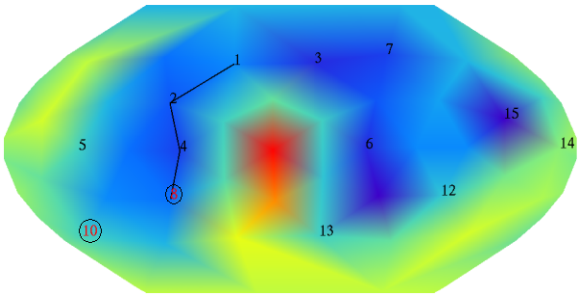


Figure 4.4: The calculated path (1-2-4-8) from node 1 to 8 on the Geodesic SOM. The path is correct.

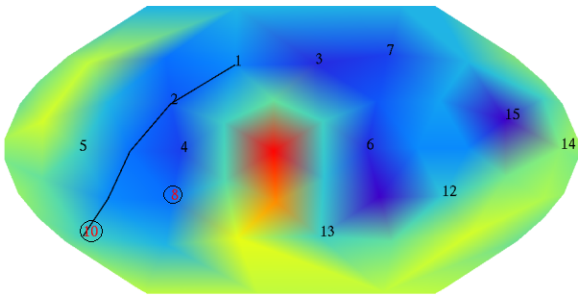


Figure 4.5: The calculated path (1-2-10) from node 1 to 10 on the Geodesic SOM. The path is incorrect.

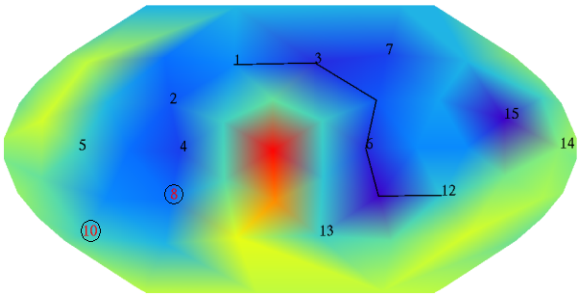


Figure 4.6: The calculated path (1-3-6-12) from node 1 to 12 on the Geodesic SOM. The path is correct.

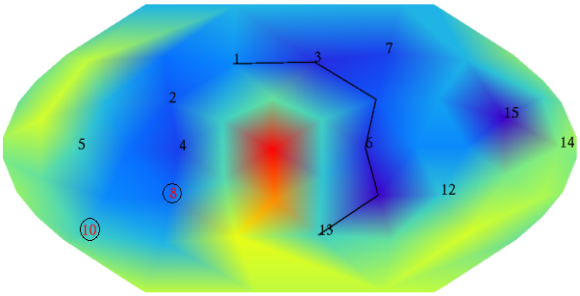


Figure 4.7: The calculated path (1-3-6-13) from node 1 to 13 on the Geodesic SOM. The path is correct.

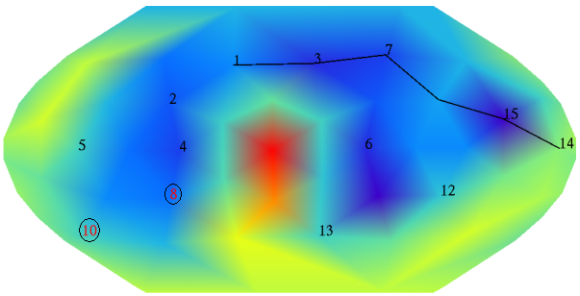


Figure 4.8: The calculated path (1-3-7-15-14) from node 1 to 14 on the Geodesic SOM. The path is incorrect.

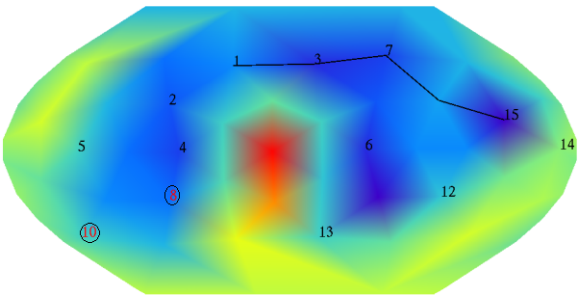


Figure 4.9: The calculated path (1-3-7-15) from node 1 to 15 on the Geodesic SOM. The path is correct.

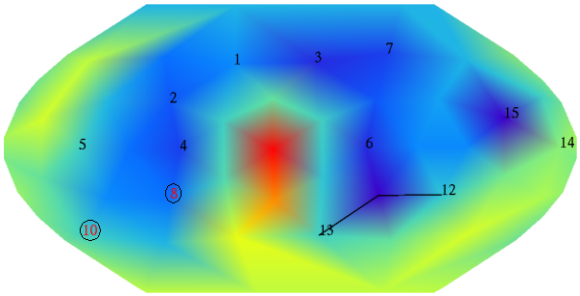


Figure 4.10: The calculated path (12-13) from node 12 to 13 on the Geodesic SOM. The path is incorrect as it does not pass through node 6, which is the parent of nodes 12 and 13.

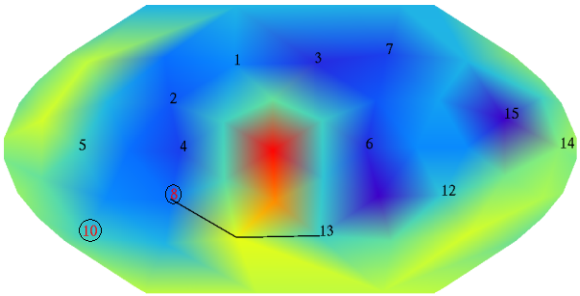


Figure 4.11: The calculated path (8-13) from node 8 to 13 on the Geodesic SOM. The path is incorrect and also passes through the mountain range.

## 4.4 Discussion

The results of our experiments show that the majority of the paths that were discovered from the root (node 1) were indeed the correct paths on the binary tree. This illustrates that applying distance transformations to the Geodesic SOM may reveal information about the structure of the data through an automated process. One of the minor incorrect results (Figure 4.5) could be attributed to the fact that the Geodesic SOM was not able to preserve the topology of the entire data. This may also be said for most of the incorrect results where the paths did not begin or end at the root. Paths between nodes that share the same parent may not discover the

parent between them, as the distances between these nodes may be quite small on the Geodesic SOM (Figure 4.10).

It can also be seen from the figures that there are some results which are incorrect by a significant margin (see Figure 4.11 for an example). In these cases, a path between two points may traverse through regions that represent large distances in the high-dimensional space; that is, a region containing neurons with large U-height values. This situation arises as the approximation of the data's distribution by the Geodesic SOM has resulted in neurons that represent points outside the data space.

## 4.5 Summary

In this chapter, a discussion was presented on the use of spatialization methods to create visualizations that could be used to track temporal changes within a data space. Trajectory-based SOMs have commonly been used to achieve this purpose and several key issues have been identified with the approaches used by trajectory-based SOMs. The use of the distance transformation algorithm on the Geodesic SOM was presented to address these problems. The use of the Geodesic SOM removes the border effect present in two-dimensional SOMs and reveals more information about the relationships within the data. Path finding is then be applied to visualize trajectories on the trained Geodesic SOM with the aid of distance transformations. Thus, the temporal lengths of the trajectories are computed without any user intervention. Experimental results suggest that the trajectories provide more accurate information on the path travelled in the feature space. However, the preliminary results also show that it is possible for trajectories to incorrectly pass through regions outside the data space. This action corresponds to travelling a large distance between cluster boundaries. In the next chapter, a modification to the approach proposed here will be presented to alleviate this problem.

# Chapter 5

# Path planning on the Geodesic SOM with floodplain analysis

## 5.1 Introduction

In the previous chapter, we detailed a proposal on the application of distance transformations to the Geodesic SOM to perform path planning. This involved using the U-heights of each neuron calculated by the U-Matrix as the distance values that need to propagated. Through this process, the problem of discovering the shortest path between two points on the Geodesic SOM can be solved.

However, there is a major flaw to this approach that was illustrated by the results in the previous chapter. In some situations, it was observed that the discovered paths would pass through the cluster boundaries on the Geodesic SOM when the start and goal points belong to the same cluster. Due to the SOM's grid structure, contraction may occur wherein two dissimilar points may be placed near each other on the SOM. Consequently, when two close data points that belong to the same cluster are mapped to opposite sides of a cluster boundary, the geodesic path for these two points may traverse through the cluster boundary. Furthermore, this region may contain neurons that represent points outside of the data space. By using the approach that we have presented to calculate the shortest path, these paths could end up travelling outside the data space. To ensure that the paths travel within the data space and travel

around cluster boundaries when appropriate, we propose to discover the flattest, shortest path.

## 5.2 Background

Robot motion applications operate in environments that may contain obstacles. Path planning solutions are thus required to calculate collision-free paths from a starting location to a goal location such they avoid all obstacles along the way. Moreover, the paths may also be the optimal path ie. the shortest path.

In the previous chapter, we proposed to find the shortest path on the Geodesic SOM using the distance transformation algorithm. While this provided a solution for finding the shortest path on the Geodesic SOM through the use of U-Matrices, there were paths that travelled through cluster boundaries. For points that belong to separate clusters, it is natural that the geodesic path would travel through the cluster boundaries. However, if the points belong to same cluster, then a computed path should travel around these boundaries. If a landscape metaphor were to be used to describe the U-Matrix, these cluster boundaries would correspond to mountain ranges that can be viewed as obstacles that should be avoided. The best route would thus be one that travels around these mountain ranges if possible.

Moreover, these boundaries may represent regions outside the data space: a region where no data samples exist. The weight vectors of the SOM tend to approximate the probability function of the input vectors. All the input vectors essentially form a cloud of points and the SOM functions like an elastic net (of weight vectors) that tries to wrap itself around this cloud as much as possible. Thus, this net is a surface that corresponds to the SOM's approximation of the probability density function. However, there is the possibility that the surface might cover regions of the data space where there are no data samples. This arises from the fact that the SOM tries to generalize the structure of the data. In other words, the structure of the aforementioned surface is governed by the connectivities of neurons. Therefore, when a path passes through a cluster boundary on the SOM, it could actually travel outside of the data space (Figure 5.1). Calculating the flattest, shortest path would help limit

the paths to travel along the region represented by the input data sets probability density function as much as possible, and create a sense of continuity.

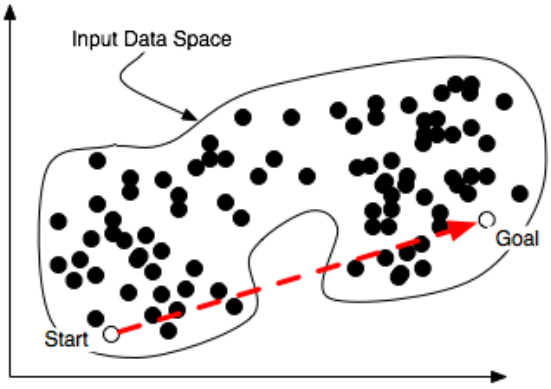


Figure 5.1: An example of a data space. The dashed line depicts a path that moves outside of the data space, which is the region enclosed by the curved shape.

## 5.3 Proposal

In order to find the flattest, shortest path between two points on the Geodesic SOM, we propose to use a modification of the approach that was presented in the previous chapter (described above by Algorithm 2). Figure 5.2 briefly describes the original approach using a small section of the Geodesic SOM's two-dimensional data structure. The nodes represent the neurons and the edges define the connectivity between the neurons. After training the Geodesic SOM, the distance value of each neuron is initalized to be -1. The goal neuron, on the other hand, has a distance value of 0. Both of these are depicted in Figure 5.2a. The distance transformation algorithm is then applied to create a distance map that could be used for path finding as shown in Figure 5.2b. The steepest descent is then used to calculate the path from any neuron to the goal (an example is shown by the bold edges).

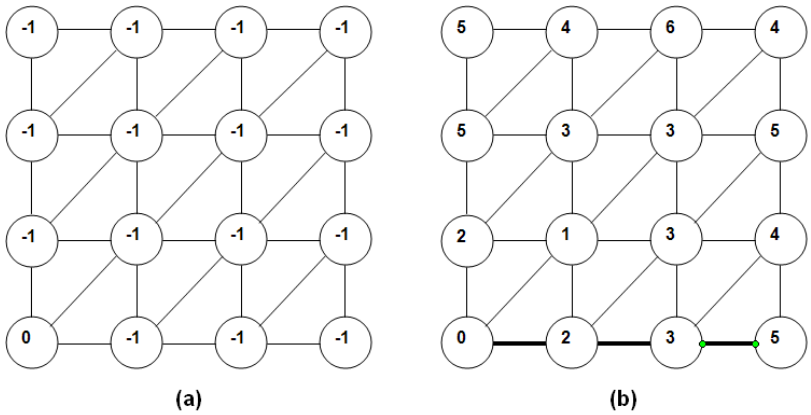


Figure 5.2: This figure describes how the distance transformation algorithm is applied to the Geodesic SOM. The distance values of the neurons would first be initialized to -1, while the goal neuron would have a distance value of 0 as shown in (a). The distance transformation algorithm is then used to propagate each neuron's U-height resulting in a distance map (b). Each neuron will be assigned a distance value that represents the cost of the shortest path to the specified goal. The path from any neuron to goal may then be computed using the steepest descent. An example of a path to the goal is highlighted by the bold edges in (b). In this situation, the start neuron (the bottom rightmost neuron) has a distance value of 5 as this is the sum of U-heights for each neuron that lies on the path, excluding the goal neuron's U-height but including the start neuron's U-height.

To solve the problem that has been described above, distance transformations are still applied to the Geodesic SOM. However, neurons with a colour index (denoted by $ci$) above a certain threshold ($ci_max$) will be ignored. These colour indices are the index values of the lookup table used by the Geodesic SOM to assign a neuron with a colour given its U-height. Hence, any neurons with the same colour index within a certain area will most likely belong to the same cluster. If U-heights were used instead, cases may occur where neurons could be incorrectly classified as belonging to separate cluster or part of a cluster boundary.

As before, the distance value of the goal neuron will be initialized to 0, while the rest of the neurons will have a distance value of -1. This distance value corresponds

to the shortest distance to the goal neuron, with a value of -1 meaning that no path exists from the corresponding neuron to the goal. The goal neuron is then added to a queue. It is then removed from the queue and its distance value is propagated to its direct neighbours so that their distance values can be updated. The direct neighbours are then added to the queue so that their distance values may also be propagated, and this process is repeated until the queue is empty. However, only the direct neighbours with a colour index value below or equal to the threshold will be added to the queue. If no path exists from the start to the goal; that is, the start neuron's distance value is -1 after applying the distance transformation algorithm, then the threshold is increased to be the next highest colour index that was found during the distance transformation. In this situation the distance transformation will be repeated, taking the threshold into consideration, until a valid path is found from the start neuron to the goal neuron.

## 5.4 Results

For this experiment, the same visualization produced by the Geodesic SOM in the last chapter will be used in order to see if the correct results may be produced using the modified approach that was discussed above. The image depicting the incorrect result of interest in the previous chapter (see Figure 5.3) will be displayed here again to make comparisons easier.

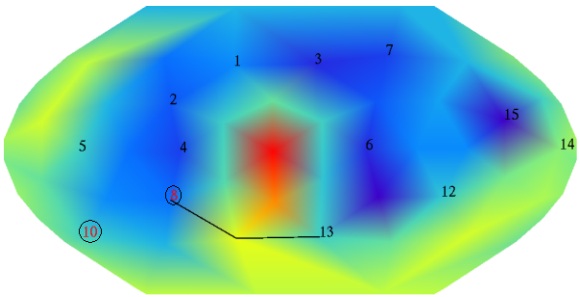


Figure 5.3: The calculated path (8-13) from node 8 to 13 on the Geodesic SOM. The path is incorrect and also passes through the mountain range.

By using the flattest, shortest path approach to recalculate the path between

**Algorithm 3** Distance transformation on the Geodesic SOM

```
Require: start and goal and Geodesic SOM
  ci_max = max(start.ci, goal.ci)
  while start.distance = - 1 do
    next_ci_max = ∞
    Queue Q is empty
    for all neuron in neurons do
      neuron.distance = -1
    end for
    goal.distance = 0
    Q.push(goal)
    Start distance transformation
    while Q is not empty do
      c = pop(Q)
      neighbours = getNeighbours(current)
      for all n in neighbours do
        if n.ci > ci_max then
          if n.ci < next_ci_max then
            next_ci_max = n.ci
          end if
          ignore n
        end if
        d = n.uh√2
        if n.distance == -1 then
          n.distance = current.distance + n.uh
          Q.push(n)
        else
          if n is a diagonal neighbour and n.distance > c.distance + d then
            n.distance = c.distance + d
            Q.push(n)
          else
            if n.distance > c.distance + n.uh then
              n.distance = c.distance + n.uh
              Q.push(n)
              for all v in n's duplicate neurons do
                v.distance = n.distance
              end for
            end if
          end if
        end if
      end for
    end while
    ci_max = next_ci_max
  end while
```

shown in Figure 5.3, the red cluster boundary was avoided and produced a result that is correct.

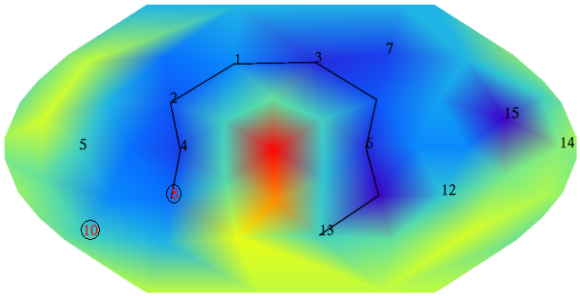


Figure 5.4: The path (8-4-2-1-3-6-13) from node 8 to 13 on the Geodesic SOM. The path is correct and travels around the red mountain range.

## 5.5 Discussion

As the results show, using the flattest, shortest path approach helps solves the problem where the path travels through outside the data space. These regions appear as mountain ranges on the Geodesic SOM's U-Matrix if viewed as a landscape. The paths that were computed using this approach were also the correct results on the binary tree represented by the data set. Therefore, in cases where the shortest path approach fails, this approach is a valuable alternative that is worth trying out to generate paths that may follow the structure of a data set more closely.

## 5.6 Summary

In this chapter, we presented a modified application of the distance transformation algorithm to the Geodesic SOM. This approach was proposed to solve cases where a path traverses through cluster boundaries on the SOM, which represent regions outside the data space. Experimental results suggest that this approach is worth experimenting with to deal with these situations to produce more accurate results. In the next chapter, more rigorous experiments will conducted on benchmark manifold

data sets to further evaluate the use of distance transformations on the Geodesic SOM.

# Chapter 6

# Experimental results: Manifold Learning - Path finding on Manifolds

In this series of experiments, the aim is to test how the proposed method can find valid and more truthful paths along or within the data space. Various data sets will be used that are sampled from different manifolds (such as a torus and Klein bottle) for this purpose. As discussed in Similä's work [71], it has been reported that such manifolds can be observed in areas like computer vision [106]. In these situations, the state of an object of interest may be constrained to transit between certain values. An example is also mentioned in the aforementioned work, where a dynamic process may only traverse between certain states such that the trajectories would form a manifold in the process variable space. In these environments where the data is obtained, the transitions between data points are often objects of great interest, besides typical analyses such as clustering. Furthermore, one task that is performed on manifolds is calculating the geodesic path between two points. By experimenting with manifolds using the proposed approach, an approximated geodesic path can be obtained that can be compared and analyzed to see if the path is meaningful to users.

## 6.1 Two-dimensional manifolds embedded in three-dimensions

The first experiment deals with manifolds that are embedded in 3D. Consequently, this allows the manifolds and trajectories to be visualized for analysis. The manifolds used here are the S-curve, swiss roll, torus and figure-8 Klein bottle. These manifolds and the sampled points used to create the data sets are displayed in Figures 6.1, 6.2, 6.3 and 6.4 respectively. Out of these 4 manifolds, only the S-curve and swiss roll can be unrolled onto a two-dimensional plane. For each manifold, 2000 points were randomly sampled on the manifold with a uniform distribution. A twenty-frequency geodesic dome (4002 neurons) is used for the Geodesic SOM, while the initial update radius was 39 and 1000 epochs were used for training.

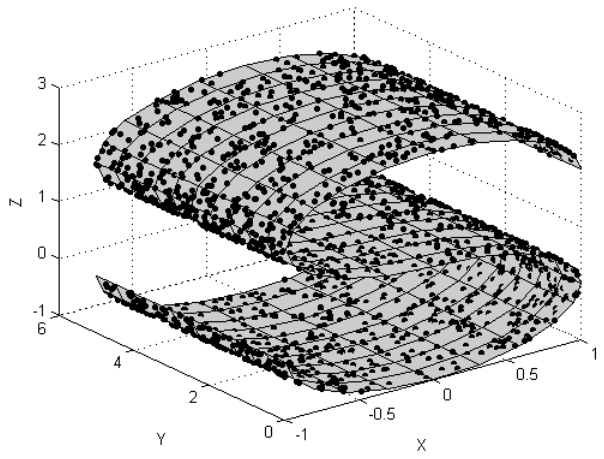


Figure 6.1: The S-curve and the sampled points used for training the Geodesic SOM.

Preliminary experiments with the S-curve and swiss roll manifolds illustrated that like the conventional SOM, the Geodesic SOM does not handle manifolds that are essentially a two-dimensional plane very well. The projection of the weight vectors reveals a significant number of points that lie outside of the manifold. Consequently, a number of calculated paths would ignore the actual topology of the manifold. To solve this problem, the Geodesic SOM was modified to use the M-SOM algorithm to

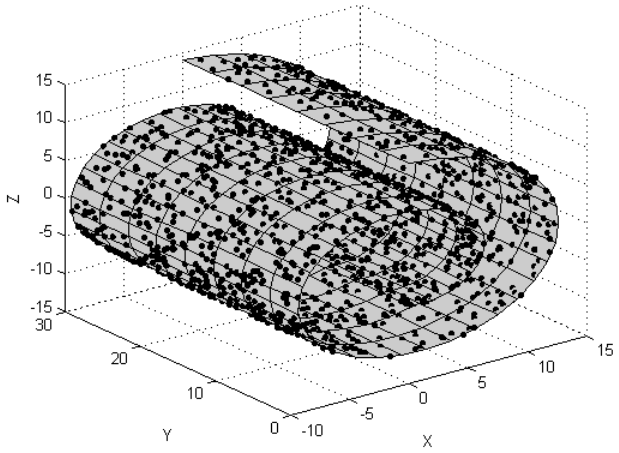


Figure 6.2: The swiss roll and the sampled points used for training the Geodesic SOM.

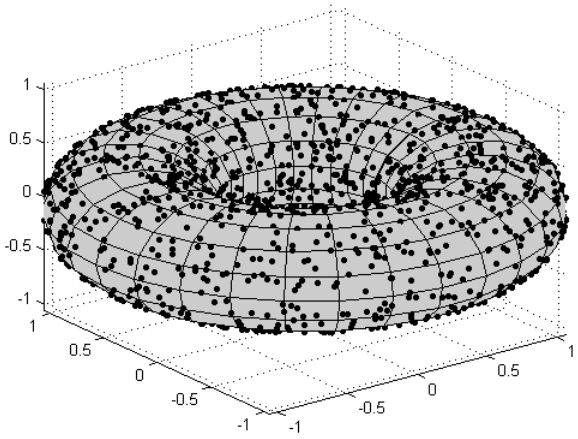


Figure 6.3: The torus and the sampled points used for training the Geodesic SOM.

deal with data that can be described by a manifold. In other words, the M-SOM has been implemented using a icosahedron-based geodesic dome as the neuron lattice. For reference, this will be known as the Geodesic M-SOM. The following subsection will describe the M-SOM algorithm in detail.

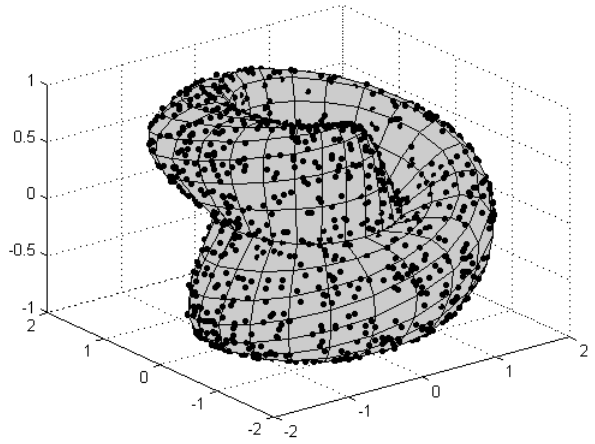


Figure 6.4: The figure-8 Klein bottle and the sampled points used for training the Geodesic SOM.

### 6.1.1 M-SOM

The M-SOM is an algorithm that is better suited than the conventional SOM when dealing with data that is known to form a manifold in the high-dimensional space. Given a set of points in the high-dimensional observation coordinates, a manifold learning algorithm is run to calculate their corresponding internal coordinates on the manifold. Since the goal of Isomap algorithm is to generate a set of points that preserves the geodesic distances on the manifold surface, it has been used in this situation (with k = 7) to learn the internal coordinates of each 3D point on the manifold. A new data set is then created, where each feature vector $(y, x)$ contains both the 2D internal coordinates $y$ and 3D observations coordinates $x$. Similarly, a neuron's weight vector consists of its position in the internal coordinates $w_j^1$ and the observation coordinates $w_j^2$, where $j$ denotes the neuron's index on the map. Calculating the BMU $c$ is done in the internal coordinates according to the Euclidean distance and is defined by the following equation:

$$c = \underset{j}{\operatorname{argmin}} \, ||y - w_j^1(t)|| \tag{6.1}$$

The BMU is therefore the neuron that has the shortest distance to the data point

on the manifold with respect to the internal coordinates. No modifications to the conventional SOM's updating rule are necessary.

## 6.2 Results

Figures 6.5 and 6.6 depicts the scatter plot of each weight vector's observation coordinates for the S-curve and swiss roll manifold respectively. These results show that the points closely respect the topology of the manifold.

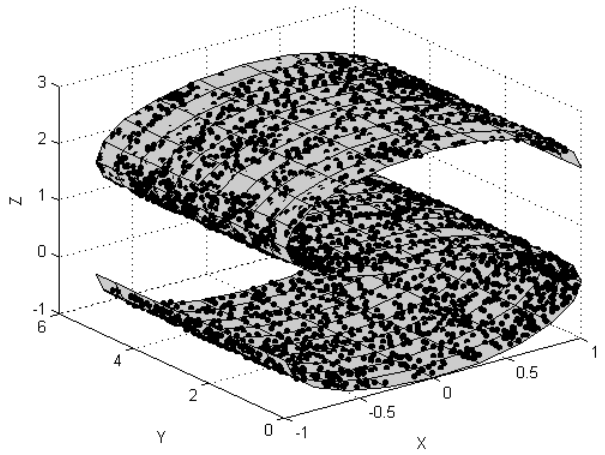


Figure 6.5: A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the S-curve.

Examples of paths on the S-curve and swiss roll calculated with the aid of distance transformations, compared to the results of using linear interpolation can be seen in Figures 6.7 and 6.8.

Since these manifolds are intrinsically a two-dimensional plane, the use of linear interpolation could be used on the internal coordinates to find a path on the manifold. For calculating a path of length $n$ between points $(y_0, x_0)$ and $(y_n, x_n)$ using linear interpolation, the equation to discover the BMU fo each data point $c_i$ ($i$ = 1, 2, 3, ... $n$) is as follows:

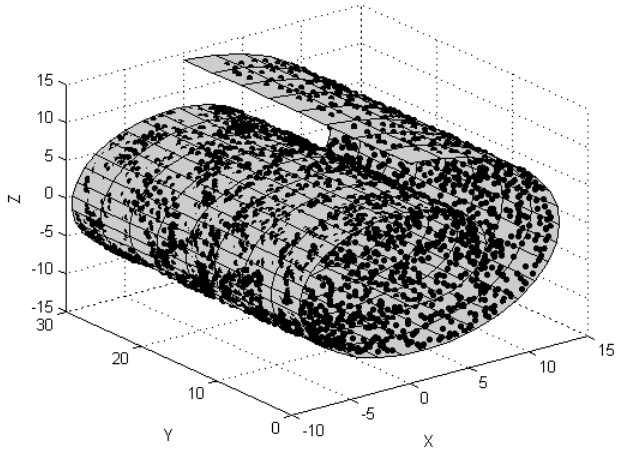


Figure 6.6: A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the swiss roll.

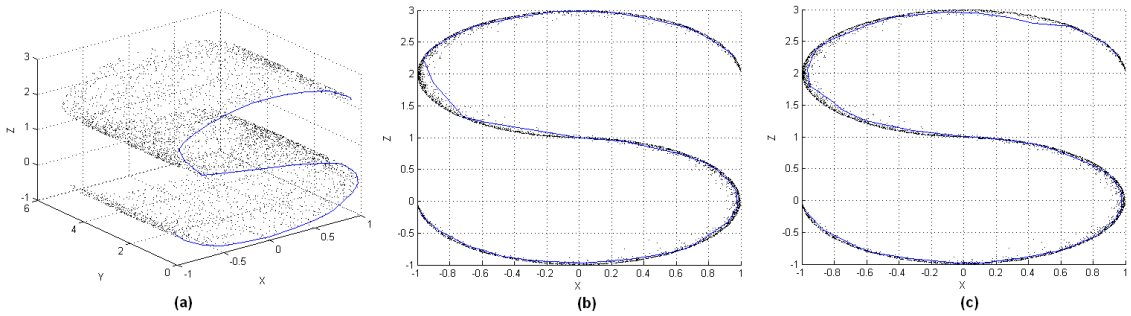


Figure 6.7: The calculated path from (-1, 0, 0) to (1, 0, 2) using distance transformations laid against the weight vectors for the S-curve (a). The distance travelled is 8.860698. A side view can be seen in (b), while (c) is the side view of the path with the same temporal length (36 vertices) and calculated with the use of linear interpolation. The distance travelled for the latter trajectory is 9.034972

$$c_i = \underset{i}{\operatorname{argmin}} \; ||y_0 + i \times \frac{y_n - y_0}{n+1}, w_j^1(t)|| \tag{6.2}$$

Similarly, the U-height $uh_i$ of each neuron $i$ used by the distance transformation algorithm is only calculated using the internal coordinates:

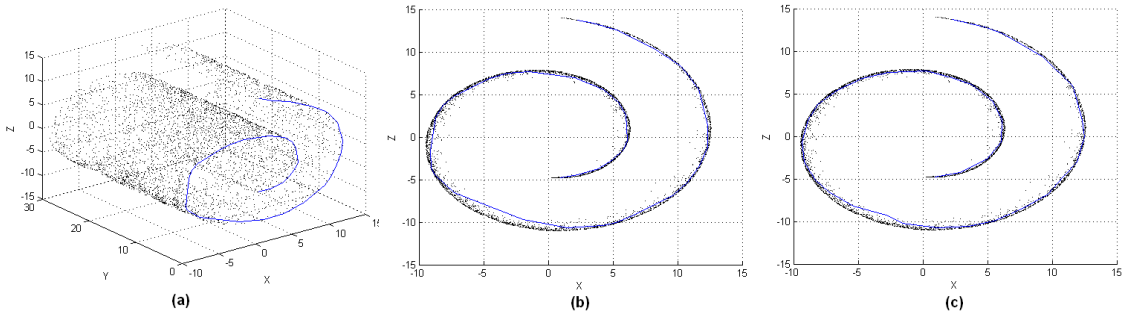


Figure 6.8: The calculated path from (0, 0, 14.14) between (0, 0, -4.712) using distance transformations laid against the weight vectors for the swiss roll (a). The distance travelled is 84.69305. A side view can be seen in (b), while (c) is the side view of the path with the same temporal length (37 vertices) and calculated with the use of linear interpolation. The distance travelled for the latter trajectory is 86.115944

$$uh_i = \frac{1}{n}\Sigma_j d(w_i^1, w_j^1), j\epsilon N(i), n = |N(i)| \tag{6.3}$$

where $N(i)$ is set the neurons in the neighbourhood of neuron $i$.

Table 6.1 presents a comparison of the approximated geodesic distances on the S-curve and swiss roll by the distance transformation algorithm and through linear interpolation. These distances were compared to the distances computed by Dijkstra's algorithm applied to the k-nearest neighbour (k = 7) graph of all 2000 sampled points; that is, the distance matrix fed to the Isomap algorithm. Note that the total number of temporal vertices for a trajectory computed by linear interpolation is set to be the same number of temporal vertices for the trajectory calculated with the distance transformation algorithm.

From these results, it can be gathered that the use of linear interpolation on the Geodesic M-SOM produces paths that are generally close to the real geodesic path. On the other hand, the use of distance transformations on the Geodesic M-SOM produces paths that are signficantly longer than the geodesic path. This occurs due to the Geodesic M-SOM's spherical structure, which is not suitable for manifolds that are intrinsically a two-dimensional plane. However, a two-dimensional M-SOM may handle such manifolds better as this would effectively attempt to unroll the

| SOM Type | Manifold | Distance transformation | Linear interpolation |
|---|---|---|---|
| Geodesic M-SOM | S-curve | 43.52% | 5.44% |
| Geodesic M-SOM | Swiss roll | 48.08% | 6.29% |
| 2D M-SOM | S-curve | 5.93% | 5.19% |
| 2D M-SOM | Swiss roll | 12.79% | 5.35% |

Table 6.1: The mean difference in geodesic distances computed by linear interpolation and the distance transformation algorithm, compared to Dijkstra's algorithm applied to a manifold's k-NN graph. Linear interpolation and distance transformations was applied to the Geodesic M-SOM and two-dimensional M-SOM for the S-curve and swiss roll data sets.

curved manifolds so they are flat. The results of using a two-dimensional M-SOM for approximating geodesics is also compared in Table 6.1. The size of the SOM is 64 x 64 (4096 neurons), with an initial update radius of 27 and an initial learning rate of 0.8. The comparison shows that the performance of the distance transformation algorithm and linear interpolation is very similar.

However, there is a significant problem with using linear interpolation: calculating the optimal temporal length of a trajectory. If the length is too small, the trajectory is more likely not to follow the structure of the data. While it would seem that increasing the temporal length would help make the path stay on the surface more, if the length is too long, the trajectory may actually not be as smooth and the distance travelled would be significantly larger. Figure 6.9 is an example where additional vertices were automatically added until no new neurons are able to be inserted. The resulting path is significantly longer and not as smooth. However, using distance transformations allows trajectories to be formed that like the work by Skupin and Hagelman [61], could be used to observe aspects of change such as parallelism (see Figures 6.10 and 6.11). Furthermore, the distance transformation algorithm produces trajectories that respect the topology of the data well, and has a similar performance to linear interpolation with respect to approximating the geodesic path.

For the torus and figure-8 Klein bottle, the Isomap algorithm was not used as the points would be embedded in two dimensions, whereby the manifolds would be cut open. Since the manifolds do not have boundaries, this would result in a loss of connectivity information. Thus, the regular Geodesic SOM algorithm was used. As the neuron lattice does not contain boundaries either, this may allow the Geodesic

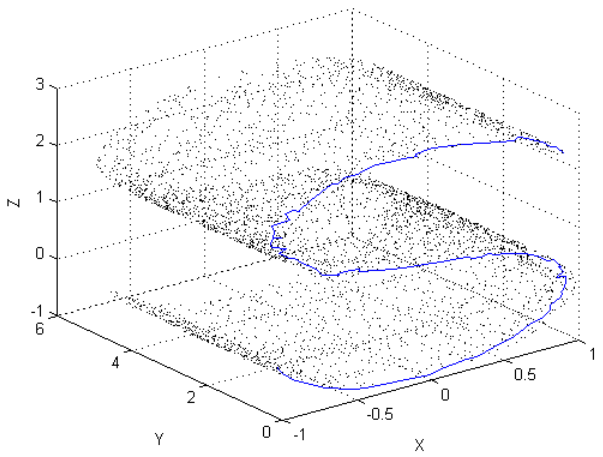


Figure 6.9: The calculated path from (-1, 0, 0) between (1, 0, 2) using linear interpolation on the S-curve. In this case, the maximum amount of temporal vertices is used (by temporal vertices until no new neurons are involved) and the trajectory is not as smooth as indicated by the "zig zags". Consequently, the distance travelled with this path much larger (14.15989)

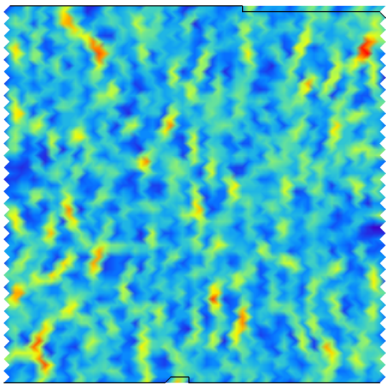

Figure 6.10: This figure shows the path from (-1, 5, 0) to (1, 5, 2) and the path from (-1, 0, 0) to (1, 0, 2) on the M-SOM. These paths are represented by the top and bottom trajectories respectively and are computed using distance transformations. Both paths are horizontal, which could indicate parallel movement.

SOM to handle other topologies that share the same trait. The results of projecting the weight vectors of each neuron are shown in Figures 6.12 and 6.13. Once again, it can be observed that the points lie close to the surface of the corresponding manifold

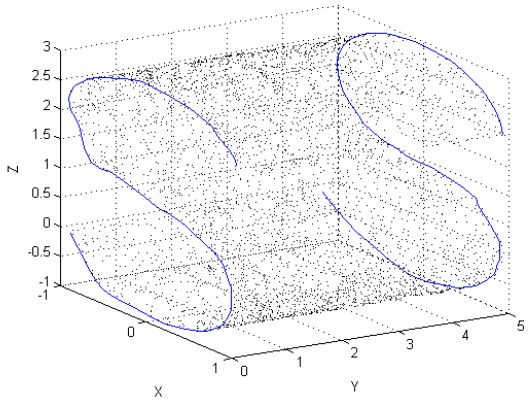


Figure 6.11: The figure depicts two paths on the projected weight vectors for the S-curve. The left and right paths correspond to the paths in part (a) and (b) of Figure 6.10 respectively. Both paths can be seen to be close to parallel on the S-curve.

and the resulting paths follow the topology of the data well. Figures 6.14 and 6.15 provide an example of a paths on the torus and Klein bottle respectively.

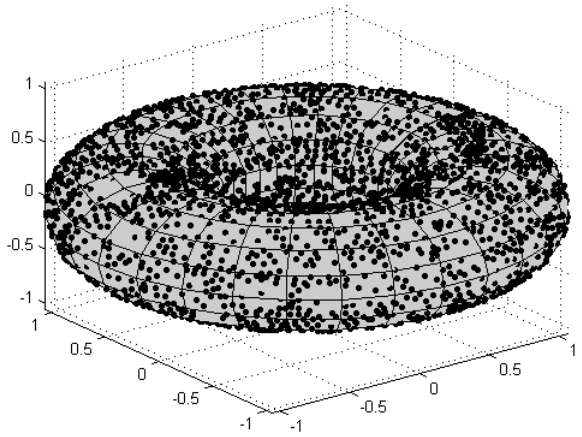

Figure 6.12: A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the torus manifold.

However, while testing with the torus manifold, it was observed that there were weight vectors that represented points located within the hole; that is, the points do not lie on the surface of the torus (Figure 6.16). These points generally lie in the

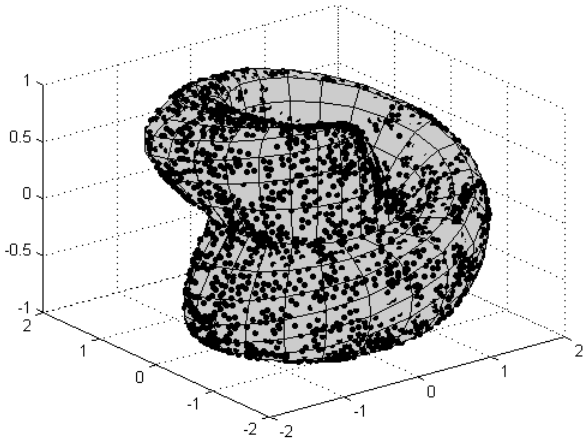

Figure 6.13: A scatter plot of the weight vectors after training the Geodesic M-SOM with points sampled from the Klein bottle manifold.

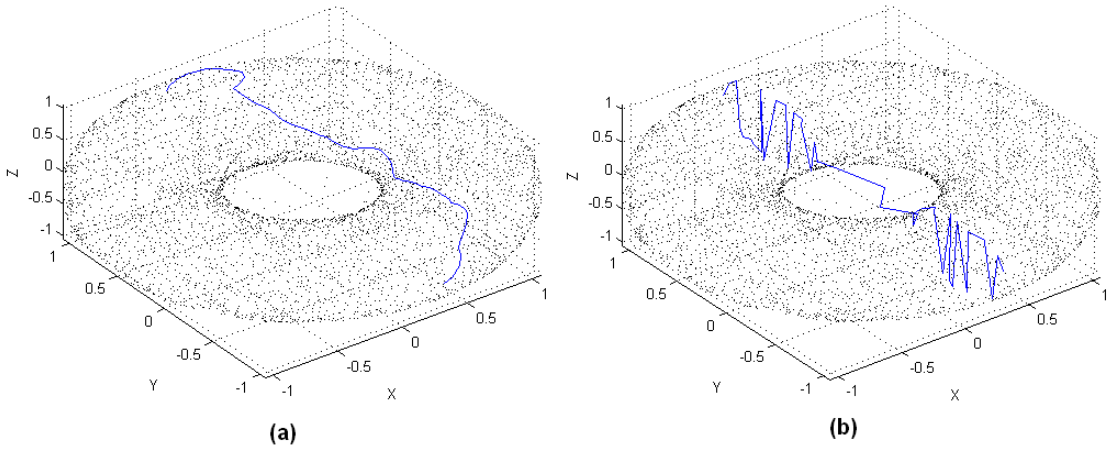


Figure 6.14: The calculated path from (-1.5, 0, 0) to (1.5, 0, 0) using distance transformations laid against the weight vectors for the torus manifold (a). The result of using linear interpolation produces a path that does not travel along the surface of the torus (b).

cluster boundaries that have been enclosed within the rectangles in Figure 6.17. In this situation, floodplain analysis could be used to avoid generating incorrect results.

Figures 6.18a and 6.18b illustrate a comparison of using distance transformations

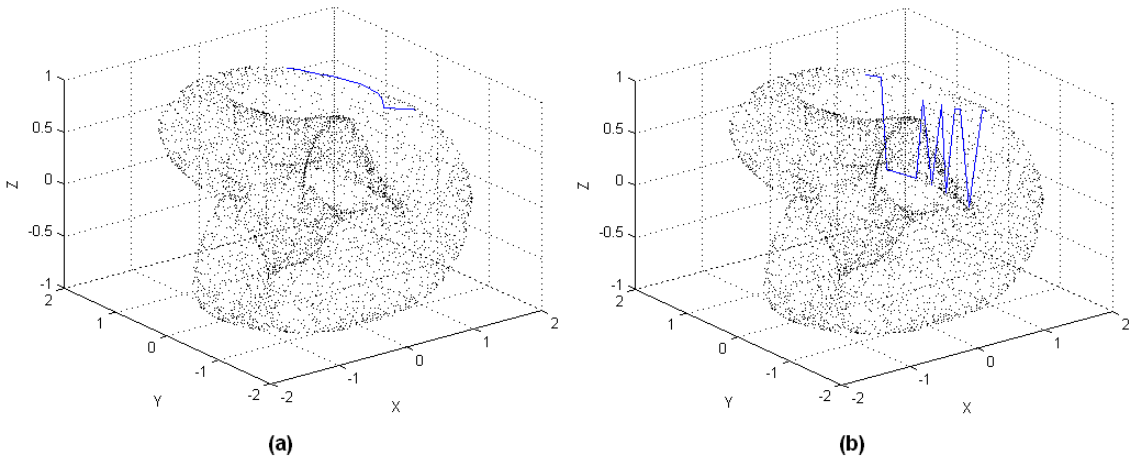


Figure 6.15: The calculated path from (0.6812, 1.18, 0.7866) to (1.707, 0, 0.5) using distance transformations laid against the weight vectors for the Klein bottle manifold (a). The result of using linear interpolation produces a path that does not travel along the surface of the Klein bottle (b).

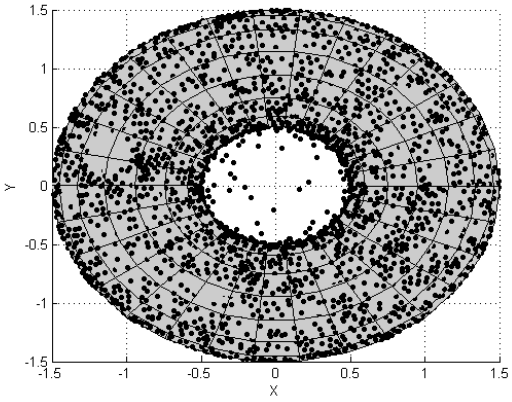


Figure 6.16: A top-down view of the torus that reveals a number of points outside of the data space.

without and with floodplain analysis on the Geodesic SOM respectively. With the use of floodplain analysis, the path in 6.18b navigates around the cluster boundary unlike the results in 6.18a. The corresponding paths for each result on the torus is depicted in 6.19. Visually comparing the two images shows that using floodplain analysis (Figure 6.19b) produces a path that travels along the torus, which would not

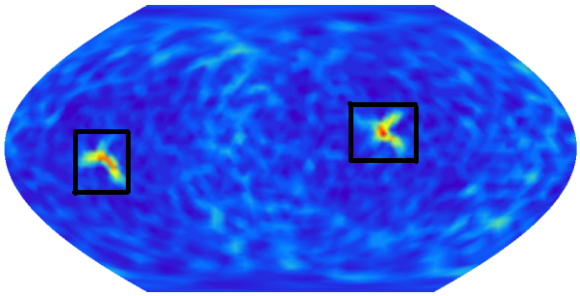

Figure 6.17: The Geodesic SOM after training it with 2000 points sampled from the torus. The cluster boundaries are within the drawn rectangles.

otherwise occur (Figure 6.19a).

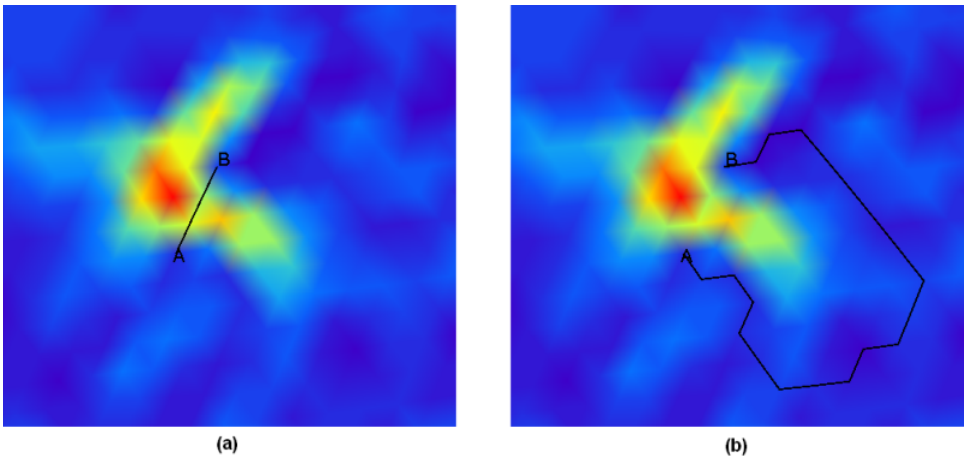


Figure 6.18: A close view of the right cluster boundary on the Geodesic SOM in Figure 6.17. Part (a) is the result of using only distance transformations whereby the path between (-0.22843947, -0.44011176, 0) and (-0.33318394 0.37217036 0.015714455) traverses through the cluster boundary. The result of using distance transformations with floodplain analysis produces a path that avoids the cluster boundary (b).

### 6.2.1 Recovering manifolds from a cloud of points

It can be observed the paths computed by the proposed method truthfully follow the structure of the data. Although the method was not intended to be use for this purpose, it may also be possible to use the resulting paths to recover the manifold from a cloud of points. Figures 6.20 to 6.23 depict the results of approximating

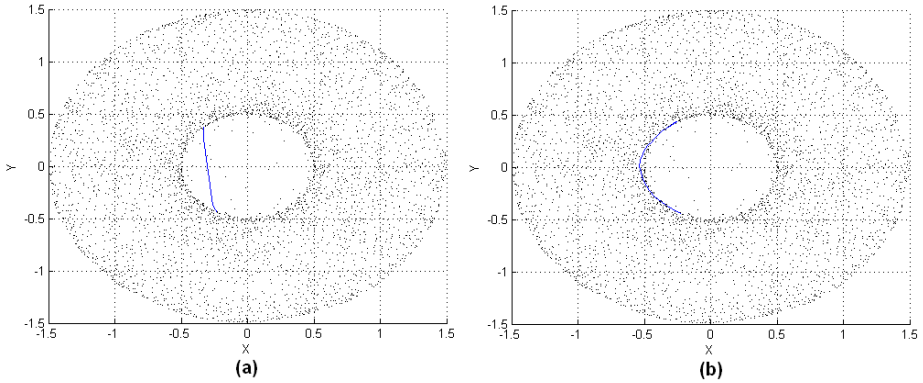


Figure 6.19: Parts (a) and (b) of this figure depict the paths on the Geodesic SOM in part (a) and (b) of Figure 6.18 on the actual torus respectively.

the geodesic path from just one point to every other data point on each manifold. From finding the path from each data point to a single point, a large part of the manifold's structure can already be seen that provides visual clues to how the points are connected. This information can be used to reconstruct the manifold.

Figure 6.20: The path from (-0.463182, 2.081766, -0.886263) to every other data point on the S-curve.

## 6.3 Two-dimensional manifold of face images

This section will illustrate how well the use of distance transformations on the M-SOM performs on a two-dimensional manifold of face images. The 698 images are taken

Figure 6.21: The path from (5.600684, 22.194967, 12.521035) to every other data point on the swiss roll.

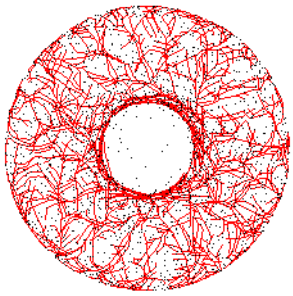

Figure 6.22: The path from (0.408809, -0.50274, 0.355076) to every other data point on the torus.

from the problem presented in [69]. Each 32 x 32 image is of a face in various poses that differ in two paramaters: azimuth and elevation. Each image is then submitted to the Isomap with k = 7 to compute the internal coordinates for the Geodesic M-SOM. A 10 x 10 hexagonal grid is used for the M-SOM and it is trained for 1000 epochs, with an initial update radius of 4 and an initial learning rate of 0.8. Using fewer neurons than the amount of data samples allows us to save on computation. The M-SOM's structure shown in Figure 6.24 reveals that the face images have been arranged in a fairly smooth manner.

As per the previous experiments, the use of linear interpolation will be compared to distance transformations. Both approaches are used to discover the transition from

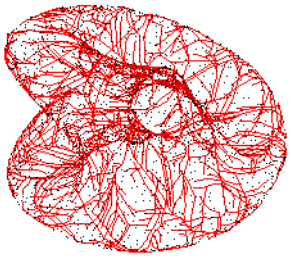

Figure 6.23: The path from (1.177278, -0.038267, -0.180746) to every other data point on the torus.

one face image to another; that is, image interpolation. An example of a result is shown in Figure 6.25. The start and goal are the left and right images respectively. To keep the comparison fair, the temporal lengths of each trajectory are once again the same. Figure 6.25a illustrates the use of linear interpolation, while 6.25b is the result of using distance transformations. Both methods can be observed to generate fairly similar results. Although linear interpolation may produce more accurate results, the previous experiments with manifolds revealed that automating the linear interpolation method can produce signficantly longer trajectories with respect to the distance travelled in the feature space.

## 6.4 Discussion

Through our experiments, we were able to show that there are limitations in the use of linear interpolation for trajectory formation on SOMs. The process can be automated by adding temporal vertices until no new neurons would be involved. However, this can result in trajectories that are not accurate representations of the path travelled in the feature space as the experiments have shown. The use of this approach would therefore require a suitable temporal length to be chosen. Short trajectories (that is, trajectories with a small temporal length) may not respect the topology of the data

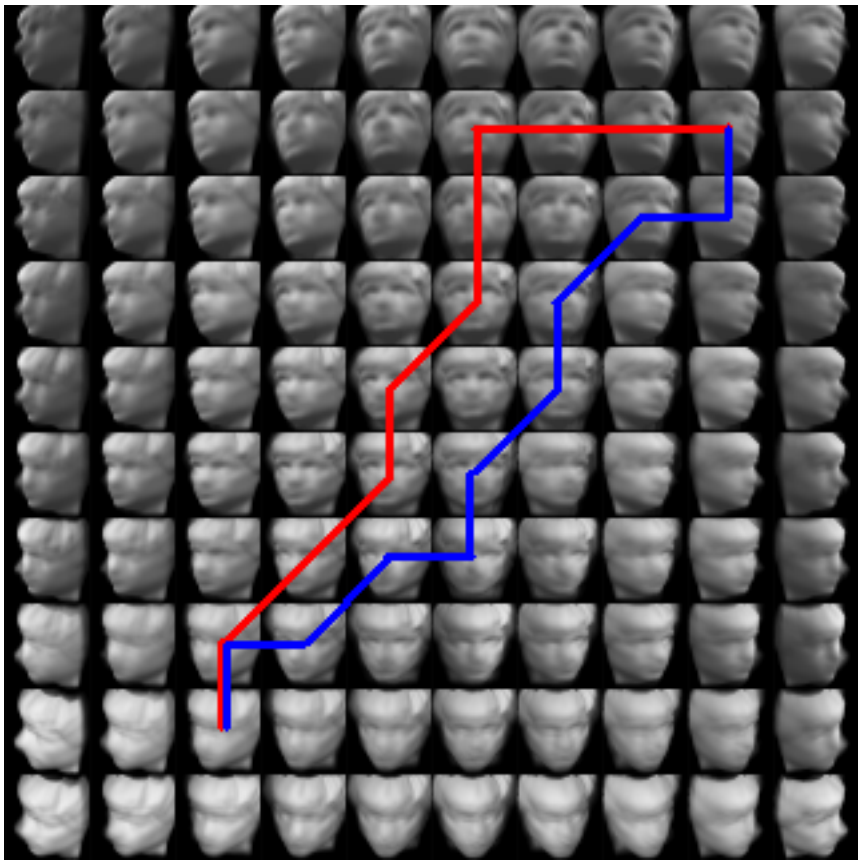

Figure 6.24: The two-dimensional data structure of the M-SOM that shows what each output image is at each neuron and how they are arranged after training the M-SOM with the data set of various face images. The blue and red trajectories correspond to the results of using linear interpolation and the distance transformation algorithm as depicted in part (a) and (b) of Figure 6.25 respectively.

and thus are not realistic representations of the path travelled in the original high-dimensional space. Long trajectories, on the other hand, may produce paths that are not as smooth and travel significantly larger distances in the high-dimensional space. Hence, the insertion of additional temporal vertices to trajectories on the SOM would need to be done manually.

The use of distance transformations on the Geodesic SOM has been shown to be a useful alternative that gives us the best of both worlds. The results approximate the geodesic distances in the feature space well and clearly respect the topology of

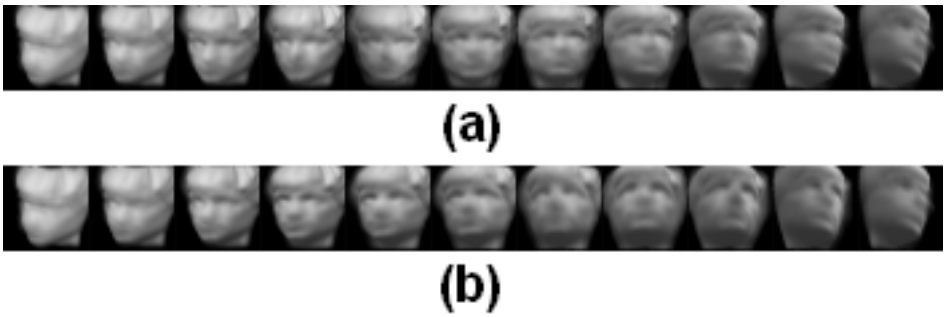


Figure 6.25: Parts (a) and (b) of this figure are examples of using linear interpolation and distance transformations on the M-SOM for image interpolation respectively. Both results are fairly similar.

the data. Furthermore, preliminary experiments have shown that because the paths respect the topology of the data, for a given manifold, the approximated geodesics between all pairs of points could possibly be used to reconstruct the manifold from an unorganized set of points. Therefore, when linear interpolation does not provide useful information, the use of distance transformations as an alternative method of (nonlinear) interpolation should be considered. This would allow the extraction of more information about intermediate states that may not otherwise be obtained.

# Chapter 7

# Experimental results: Socio-demographic data

## 7.1 Introduction

Trajectories have previously been used on the SOM to visualize temporal, demographic changes in population census data [61]. This involves connecting temporal vertices with a straight line. Linear interpolation was used to insert additional temporal vertices to trajectories so that more informed judgements about the relationships between trajectories could be made. This provides analysts the ability to compare different trajectories to observe if different objects are undergoing parallel, convergent or divergent development. However, one of the significant disadvantages of this approach is that analysts are unable to perceive the distances in the feature space through these trajectories.

In this chapter, a comparative analysis on the use of distance transformations and linear interpolation on socio-demographic data will be conducted. This will enable us to study whether or not distance transformations could further assist the process of temporal analysis and produce cognitively plausible trajectories whose lengths would correspond to distances in the high-dimensional space.

## 7.2 Experimental data

In this experiment, an evaluation on whether or not the trajectories using the proposed approach could also be used for examining the intermediate states and assisting the process of temporal analysis will be conducted. Data was obtained from the Energy Information Administration web site for the purpose of this experiment. The data describes the $CO_2$ emission levels (and related attributes) of a large number of countries between 1980 and 2004. This data was extracted to produce a data set that contains data on 21 selected countries (with $CO_2$ emission levels over 100 million metric tons) and their emission levels between 1990 and 2004. Altogether, there are 313 data samples described by 9 attributes. The attributes and their corresponding units are:

1. Population (millions)
2. Total $CO_2$ emissions (million metric tons of carbon dioxide)
3. Per capita emissions (metric tons of carbon dioxide)
4. Emissions from petroleum (million metric tons of carbon dioxide)
5. Emissions from natural gas (million metric tons of carbon dioxide)
6. Emissions from coal (million metric tons of carbon dioxide)
7. Total primary energy production (quadrillion BTU)
8. Total primary energy consumption (quadrillion BTU)

It should be noted that complete data is not available for Russia during 1990 and 1991, and that European Union is counted as a single entity since the Kyoto Protocol counts them as such.

In this experiment a twenty-frequency geodesic dome is used, with a learning rate of 0.8, an initial update radius of 39 and 1000 epochs is used for training the Geodesic SOM.

### 7.2.1 Attribute space visualization

Figure 7.1 depicts the resulting visualization when the Geodesic SOM is trained with the data set. The Wagner III cartographic projection technique is used so an entire view of the visualization can be seen. In the centre of the figure, it can be observed that there are a group of clusters. These are countries with extremely high emission levels that are in the thousands of million metric tons. The other countries have emission levels in the hundreds of million metric tons. The eight component planes are shown in figures 7.2-7.9 and can be used determine which attributes may be related to each other.

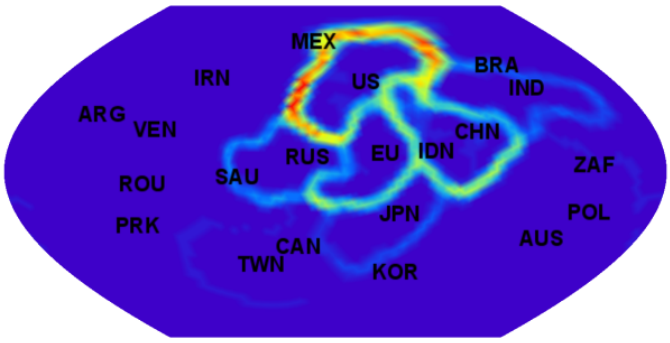


Figure 7.1: This figure depicts the Geodesic SOM after it has been trained with the carbon emissions data set. Large dark blue regions indicates that the Geodesic SOM has created a smooth distribution of the data. A group of clusters can also be seen where countries with high emission levels have been mapped to. Three-letter country codes following the ISO 3166-1 alpha-3 standard have been used to indicate where the data for each country is generally located. The exception here is that the European Union has been abbreviated as EU.

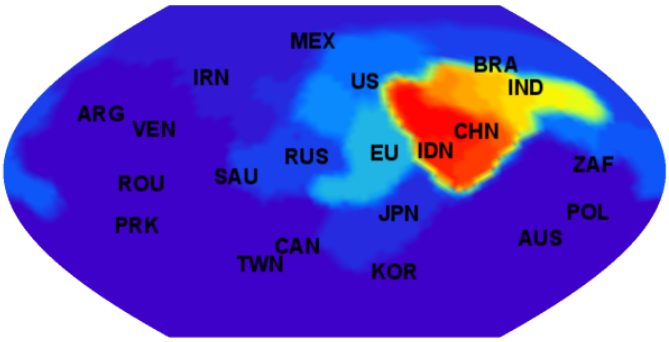


Figure 7.2: The first component plane of the Geodesic SOM that corresponds to population values.

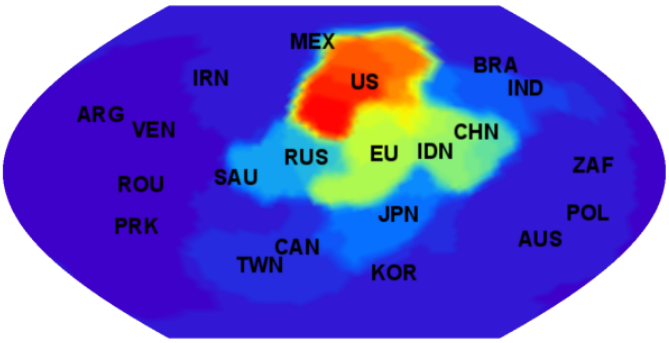


Figure 7.3: The second component plane of the Geodesic SOM that corresponds to total $CO_2$ emission values.

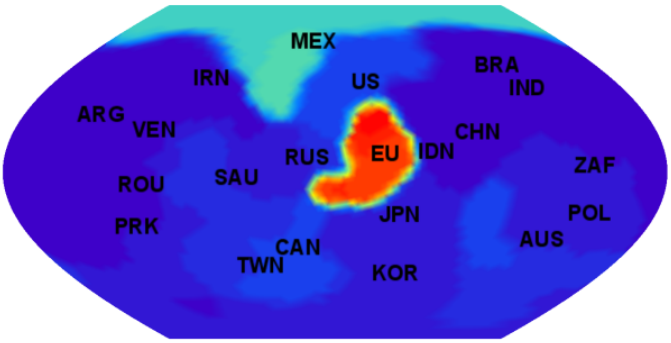


Figure 7.4: The third component plane of the Geodesic SOM that corresponds to per capita emission values.

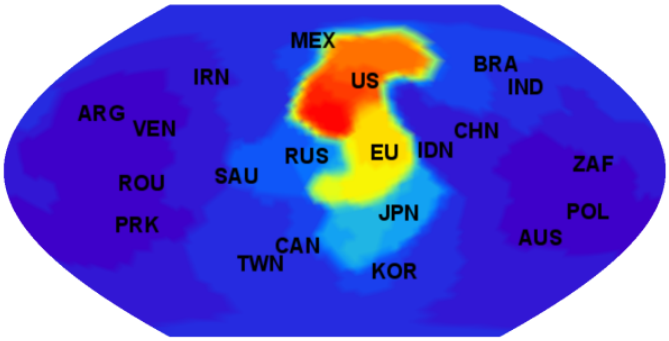


Figure 7.5: The fourth component plane of the Geodesic SOM that corresponds to petroleum emission values.

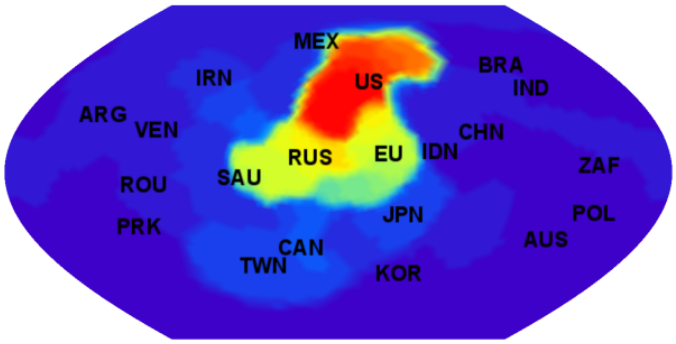


Figure 7.6: The fifth component plane of the Geodesic SOM that corresponds to natural gas emission values.

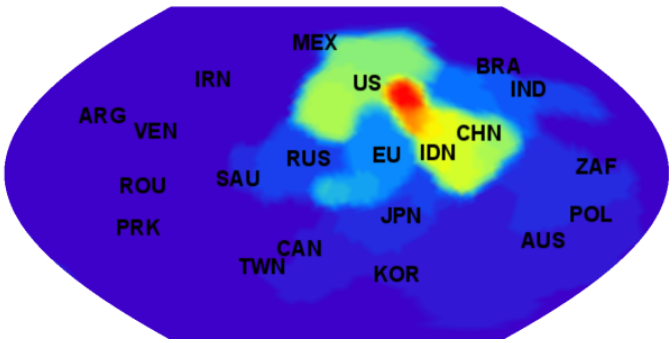


Figure 7.7: The sixth component plane of the Geodesic SOM that corresponds to coal emission values.

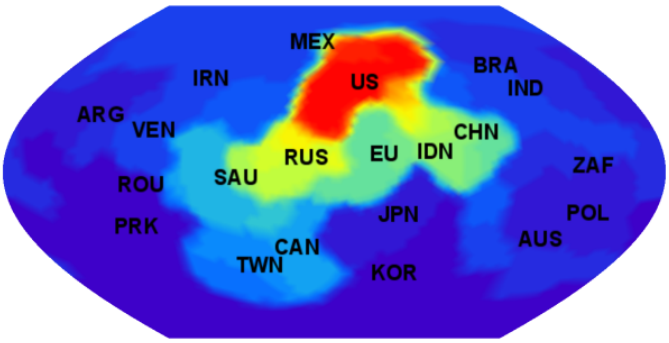


Figure 7.8: The seventh component plane of the Geodesic SOM that corresponds to total primary energy production values.

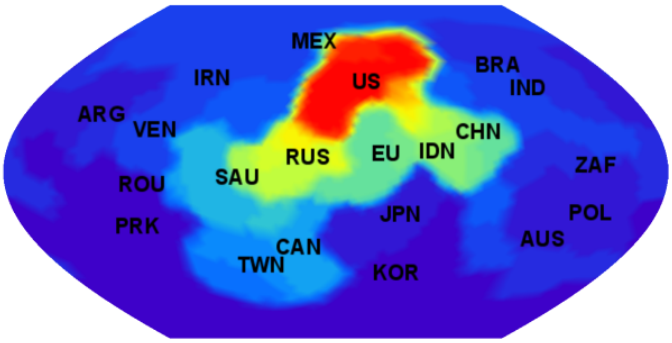


Figure 7.9: The eighth component plane of the Geodesic SOM that corresponds to total primary energy consumption values.

From the data, it is observed that the US has the highest emission levels out of all of the countries. Consequently, its data samples have been mapped to a single cluster with prominent cluster boundaries, which indicates that the data there is significantly different to the rest of the other data; that is, it has significantly higher emission levels than the other countries. Since the Geodesic SOM produces a topological mapping of the data space, countries with very high emission levels (such as China, Japan and the EU) have been placed in separate clusters located near the US' cluster. A close view of how the data of these countries (including the US' data) have been ordered within their clusters can be seen in Figures 7.10-7.13.

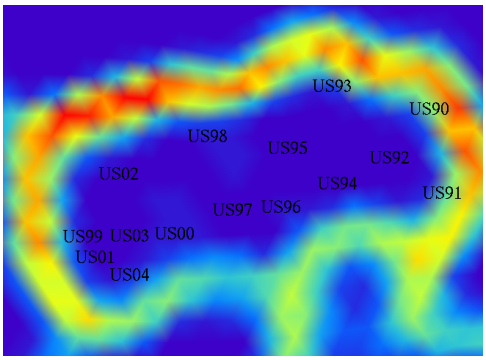


Figure 7.10: This figure displays a close-up view of the US' location on the Geodesic SOM. It's data has been placed in a cluster due to its high emission levels ( 4000-5000 million metric tons).

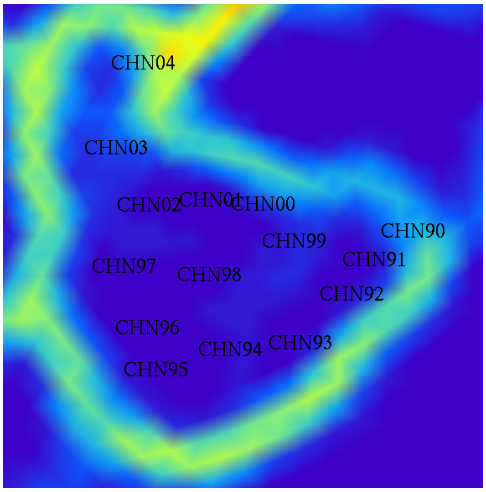


Figure 7.11: This figure displays a close-up view of the China's location on the Geodesic SOM. It's data has been placed in a cluster due to its high emission levels ( 2000-4000 million metric tons).

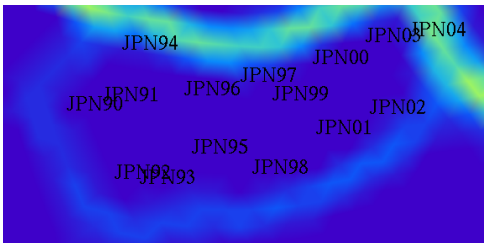


Figure 7.12: This figure displays a close-up view of Japan's location on the Geodesic SOM. It's data has been placed in a cluster due to its high emission levels ( 1000-1300 million metric tons).

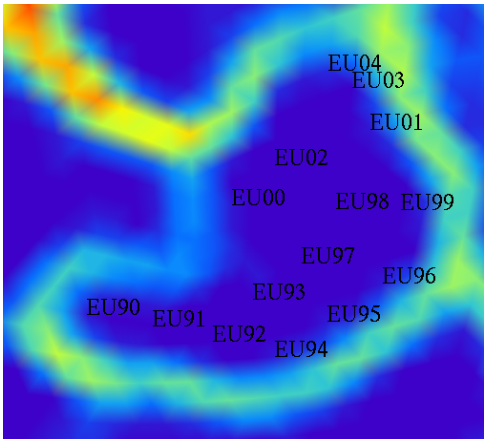


Figure 7.13: This figure displays a close-up view of the EU's location on the Geodesic SOM. It's data has been placed in a cluster due to its high emission levels ( 3200-3600 million metric tons).

Overall, the SOM distributes the data in a fairly smooth manner such that the emission levels generally decrease when moving away from the location of the US' cluster. The smooth distribution is indicated by the generally low distances between the neurons and their direct neighbours on the Geodesic SOM. This is also indicated by the large blue regions on the Geodesic SOM.

### 7.2.2 Temporal observation

Visual analytics allows us to perceive patterns and extract knowledge from visualizations. In the results in Figures 7.10 to 7.14, a close view is provided that depicts how the states of selected countries have been ordered on the SOM. These figures show that the states of each countries in these clusters have also been arranged in a direction relative to the US' cluster. In other words, the closer a country is to the US' cluster on the Geodesic SOM, the higher the emission level will be.

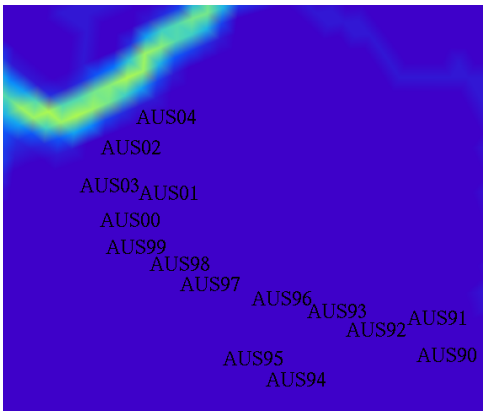


Figure 7.14: This figure displays a close-up view of the Australia's location on the Geodesic SOM. It's data has been placed in the large dark blue regions due to its relatively low emission levels ( $<$ 300 million metric tons).

Hence, trends can be identified that allows users to make predictions on events that may occur in the future. For instance, it is evident that Australia's emission levels are increasing as indicated by the direction its data is heading toward on the Geodesic SOM; that is, toward the group of clusters containing countries with high emission levels (see Figure 7.14). This information can be used to predict that Australia's emission levels in 2005 will be higher than they were in 2004. Similar predictions can be made for China, Japan and the EU since the direction of the data inside the corresponding clusters (Figures 7.11-7.13 respectively) is heading toward the US' cluster.

### 7.2.3 Trajectory finding

Having trained the Geodesic SOM with the data set, distance transformations were then used to calculate the trajectories between the BMUs that correspond to a country's emission levels over consecutive years (ie. 1990 to 1991, 1991 to 1992...). As an example, the application to Australia's data has been used to illustrate the results. The emission levels and other attributes can be seen in Table 7.1, where $a_i$ denotes the $i^{th}$ attribute as described above. Floodplain analysis is once again used here.

| Year | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ |
|---|---|---|---|---|---|---|---|---|
| 1990 | 17.022133 | 262.7667114 | 15.43676761 | 95.38250096 | 34.85799059 | 132.5262199 | 6.144310579 | 3.718075263 |
| 1991 | 17.257526 | 263.4723104 | 15.26709624 | 93.36012586 | 32.68014688 | 137.4320376 | 6.289699754 | 3.708916016 |
| 1992 | 17.481977 | 276.237352 | 15.80126504 | 98.01707624 | 33.73826649 | 144.4820093 | 6.565582797 | 3.817020878 |
| 1993 | 17.688687 | 281.0651078 | 15.88954046 | 101.2273867 | 35.09186584 | 144.7458552 | 6.609968594 | 3.91614524 |
| 1994 | 17.892557 | 279.0009996 | 15.59313181 | 104.1042977 | 36.58140673 | 138.3152953 | 6.909955523 | 3.91767409 |
| 1995 | 18.116171 | 284.8350815 | 15.72269777 | 105.002785 | 39.26470392 | 140.5675926 | 7.423951345 | 4.050353779 |
| 1996 | 18.348078 | 297.5430653 | 16.21657949 | 105.4695138 | 39.62123855 | 152.4523129 | 7.571375586 | 4.223340052 |
| 1997 | 18.565243 | 326.86827 | 17.60646332 | 107.0070946 | 39.86425812 | 179.9969173 | 8.310526864 | 4.560134692 |
| 1998 | 18.768789 | 333.2648046 | 17.75632965 | 111.154269 | 41.70808728 | 180.4024483 | 8.661854641 | 4.595390588 |
| 1999 | 18.968247 | 350.8170474 | 18.49496411 | 114.1509535 | 43.45502811 | 193.2110658 | 8.867776766 | 4.819522543 |
| 2000 | 19.16462 | 353.2025758 | 18.42992847 | 113.9574418 | 45.42068307 | 193.8244509 | 9.684649255 | 4.83251974 |
| 2001 | 19.357594 | 366.5098513 | 18.93364699 | 115.1666893 | 47.72197798 | 203.621184 | 10.26527802 | 4.993398957 |
| 2002 | 19.546792 | 374.3533436 | 19.15165126 | 116.6764917 | 50.3877261 | 207.2891258 | 10.51061268 | 5.097017823 |
| 2003 | 19.731984 | 371.702066 | 18.83754142 | 116.4360603 | 52.02929647 | 203.2367092 | 10.35485931 | 5.092657062 |
| 2004 | 19.913144 | 386.1775027 | 19.39309547 | 117.073242 | 52.35787241 | 216.7463883 | 10.55533594 | 5.266284743 |

Table 7.1: The data values for Australia's emission from 1990 to 2004.

Tables 7.2 and 7.3 contains the values of the weight vectors of the other neurons that lie on the trajectory between Australia's state in 1990 and 1991, and 1995 and 1996 respectively. These have been calculated through the use of distance transformations. The first row of Table 7.2 are the values of the weight vector associated with the BMU representing Australia's emission levels in 1990, while the last row corresponds to the BMU for its emission levels in 1991. Similarly, the first row of Table 7.3 corresponds to Australia's emission levels in 1995 and the last row corresponds to its levels in 1996. We will denote these BMUs in general as $x_{start}$ and $x_{goal}$ for easier reference, where $x_{start}$ represents the neuron at the beginning of the trajectory and $x_{goal}$ is the neuron at the end of the trajectory.

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ |
|---|---|---|---|---|---|---|---|
| 17.088438 | 262.7178 | 15.382145 | 95.082695 | 34.383934 | 133.25119 | 6.1376987 | 3.7160716 |
| 17.123491 | 263.00522 | 15.363072 | 94.567406 | 33.97294 | 134.46494 | 6.1965356 | 3.7139697 |
| 17.179424 | 263.2653 | 15.328686 | 94.10814 | 33.483253 | 135.67378 | 6.2360034 | 3.7125971 |
| 17.417149 | 263.77472 | 15.23209 | 93.38921 | 32.927113 | 137.45827 | 6.248512 | 3.7106044 |

Table 7.2: The calculated attribute values for Australia between 1990 and 1991 using the distance transformation algorithm. The first row contains the weight vector values for the BMU corresponding to Australia's data for the year 1990. The last row contains the weight vector values for the BMU corresponding Australia's data for the year 1991.

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ |
|---|---|---|---|---|---|---|---|
| 18.098314 | 284.5111 | 15.721142 | 104.93624 | 38.70803 | 140.86678 | 7.3129263 | 4.0355725 |
| 18.071562 | 284.27377 | 15.729849 | 104.729576 | 38.616154 | 140.9281 | 7.304681 | 4.0294037 |
| 18.148548 | 288.69827 | 15.905892 | 104.78141 | 38.880722 | 145.03613 | 7.377158 | 4.0928154 |
| 18.238848 | 293.63644 | 16.10159 | 104.833015 | 38.97587 | 149.82751 | 7.424095 | 4.1591787 |
| 18.662144 | 295.98477 | 16.017136 | 104.219315 | 38.997448 | 152.76799 | 7.4241176 | 4.186639 |

Table 7.3: The calculated attribute values for Australia between 1995 and 1996 using the distance transformation algorithm. The first row contains the weight vector values for the BMU corresponding to Australia's data for the year 1995. The last row contains the weight vector values for the BMU corresponding Australia's data for the year 1996

These results show that the Geodesic SOM has been able to interpolate the values between data samples fairly well as the weight vector values of intermediate neurons generally lie between the the weight vector values of $x_{start}$ and $x_{goal}$. Hence, the weight vectors of these neurons are meaningful in the sense that they are indicative of the progression between the two states and the intermediate states reached. In other words, if the values of $x_{goal}$ are higher than the values of $x_{start}$ as it is in these two cases, as we travel along the trajectory from $x_{start}$ to $x_{goal}$, the weight vector values of the intermediate neurons will increase as well. This behaviour can be observed by inspecting the results in the aforementioned two tables. If we look at each row, starting at the first row and go down to the last row, the values generally increase

as we go down the table. Furthermore, the Geodesic SOM is able to preserve the relationship between the attributes of the data where the total $CO_2$ emissions is generally the sum of the emissions from the consumption of petroleum, natural gas and coal (ie. $a_2 = a_4 + a_5 + a_6$).

This application of the distance transformation algorithm was then used to compute each country's trajectory to compare the results calculated through linear interpolation. Given a country and the feature vectors representing its state for each year, the feature vectors for two consecutive years would be interpolated to produce vectors that represent the country's state between this period. The BMUs of the feature vectors and the vectors obtained through linear interpolation would then be connected to form a trajectory. Assuming that linear development occurs between two consecutive years, the distance between two vectors in the high-dimensional space would be the Euclidean distance. Therefore, the distances of the trajectories computed by the distance transformation algorithm and linear interpolation in the feature space can be compared to the Euclidean distance. The results for each country's trajectories are depicted in Table 7.4. $\mu(|d_i - d_{DT_i}|)$ represents the mean difference between the Euclidean distance and the distance computed through distance transformations for all trajectories of a single country. Similarly, $\mu(|d_i - d_{LI_i}|)$ is the mean difference between the Euclidean distance and the distance computed through linear interpolation. $\mu_{DT}(\sigma(||w_i - w_{i+1}||))$ and $\mu_{LI}(\sigma(||w_i - w_{i+1}||))$ are the mean standard deviations of the interneuron distances for all trajectories of a country using distance transformations and linear interpolation respectively. Note that the temporal lengths of the trajectories created with the use of linear interpolation are the same as those generated through distance transformations.

| Country | $\mu(\|d_i - d_{DT_i}\|)$ (%) | $\mu(\|d_i - d_{LI_i}\|)$ (%) | $\mu_{DT}(\sigma(\|\|w_i - w_{i+1}\|\|))$ (%) | $\mu_{LI}(\sigma(\|\|w_i - w_{i+1}\|\|))$ (%) |
|---|---|---|---|---|
| Argentina | 29.40953 | 48.60662 | 72.1041 | 112.88268 |
| Australia | 9.078655 | 39.05893 | 71.67744 | 78.51406 |
| Brazil | 12.027331 | 10.543716 | 55.294853 | 56.924355 |
| Canada | 19.078484 | 84.95633 | 62.74848 | 131.57527 |
| China | 10.563775 | 18.250254 | 55.636246 | 53.567314 |
| EU | 49.153008 | 133.52766 | 69.662476 | 76.69917 |
| India | 17.04567 | 22.79891 | 57.474888 | 59.230793 |
| Indonesia | 37.762123 | 37.730553 | 68.15985 | 74.470665 |
| Iran | 14.1727 | 17.737558 | 47.78263 | 36.874268 |
| Japan | 35.371346 | 102.624344 | 72.35351 | 92.329254 |
| South Korea | 5.895685 | 226.55728 | 59.516376 | 73.67294 |
| Mexico | 25.719133 | 44.484814 | 66.98804 | 87.71016 |
| Poland | 19.600266 | 171.12245 | 68.541046 | 116.26843 |
| North Korea | 28.74262 | 284.84613 | 70.20034 | 86.63369 |
| Romania | 21.514257 | 160.75012 | 50.586857 | 89.37002 |
| Russia | 11.707836 | 123.47771 | 69.0686 | 82.590385 |
| Saudi Arabia | 12.528671 | 430.21994 | 72.054665 | 120.03248 |
| Taiwan | 11.068022 | 564.46606 | 61.13882 | 73.57899 |
| USA | 45.88477 | 63.094265 | 64.67646 | 75.47451 |
| Venezuela | 46.060486 | 138.79735 | 71.36755 | 114.84687 |
| South Africa | 18.404032 | 135.91045 | 72.08485 | 72.670685 |

Table 7.4: A comparison of the distances in the feature space computed using linear interpolation and the distance transformation algorithm. The distances for each region were computed using all the trajectories between BMUs corresponding to emission levels in consecutive years (1990 to 1991, 1991 to 1992, etc...).

This comparative analysis demonstrates that the trajectories computed with distance transformations generally represent the distances in the feature space more closely than when linear interpolation is applied. A thorough analysis of the results using linear interpolation reveals that the significant differences occur when a trajectory does not accurately represent the intermediate states in the feature space. In other words, the weight vector values of the BMUs are not nicely interpolated and are significantly different to the linearly interpolated values. Table 7.5 demonstrates an example where linear interpolation has been used to create a trajectory for Australia's state between 1996 and 1997. Looking at attribute $a_1$, the range of values should be between 18.662144 and 18.608067: the first and last values for attribute $a_i$, respectively. However, there are actually values that fall outside of this range, which

would affect the distances in the feature space. Another aspect that could be affected would be the distances between each consecutive temporal vertex. This is indicated by the values of $\mu_{LI}(\sigma(||w_i - w_{i+1}||))$. Although it may be possible to obtain more accurate results using linear interpolation with respect to these two points, this would be a time consuming process as it requires experimenting with a different amount of temporal vertices to achieve a satisfactory result.

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ |
|---|---|---|---|---|---|---|---|
| 18.662144 | 295.98477 | 16.017136 | 104.219315 | 38.997448 | 152.76799 | 7.4241176 | 4.186639 |
| 20.042826 | 302.06064 | 15.2126255 | 117.88931 | 32.12656 | 152.04477 | 4.846756 | 4.2916255 |
| 21.685394 | 310.2898 | 15.354114 | 98.01925 | 36.792103 | 175.4784 | 7.130263 | 4.2829957 |
| 18.565798 | 318.55502 | 17.165249 | 107.16122 | 39.61561 | 171.77824 | 8.014653 | 4.4631753 |
| 18.608067 | 326.29 | 17.540138 | 107.75241 | 40.224808 | 178.31284 | 8.332399 | 4.5450416 |

Table 7.5: The calculated attribute values for Australia between 1996 and 1997 using the distance transformation algorithm. The first row contains the weight vector values for the BMU corresponding to Australia's data for the year 1996. The last row contains the weight vector values for the BMU corresponding Australia's data for the year 1997.

## 7.3 Discussion

The results obtained on socio-demographic data has shown that besides linear interpolation, distance transformations can also create trajectories that are also cognitively plausible. The neurons on these trajectories are able to provide analysts an indication on the intermediate states that an object undergoes. Another important advantage compared to the results of using linear interpolation is that the distances of the trajectories in the feature space are closer to the real distance travelled in the feature space. These results suggest that the use of path finding techniques may thus create trajectories that are more accurate visualizations. However, one of the disadvantages is that both the SOM and Geodesic SOM create topological mappings of the data space that are not distance preserving, and the interneuron distances is therefore irregular. One possible solution to this would be use a variant of the SOM that preserves the

distance relationships, such as the ViSOM [107].

## 7.4 Summary

In this chapter, a comparative analysis was done on two methods for visualizing trajectories on a Geodesic SOM trained with socio-demographic data, namely distance transformations and linear interpolation. Results showed that it may possible to obtain more accurate trajectories using path finding techniques like the distance transformation algorithm. In the next and final chapter, conclusions will be drawn from the research presented in this thesis. This will be followed by suggestions on the future direction of this research, which may improve the results.

# Chapter 8

# Conclusions and Future Work

## 8.1 Conclusion

Everyday people are familiar with geographic maps. They are primarily used for:

- Creating simplified depictions of geographic spaces
- Tracking the movement of a certain object located in a geographic space
- Planning on how to get from one location to another

In this thesis, we explored the use of spatialization methods, particularly Self-Organizing Maps, to create maps of high-dimensional spaces. We envisioned that such maps could be used to perform these three tasks in the high-dimensional space. Spatialization methods already create simplifed representations of a n-dimensional space as dimensionality reduction is used to generate two-dimensional visualizations. These visualizations, which share similarities with maps, allow users to identify patterns (clusters, for example) in the data by organizing the data such that characteristics like distances or topology are preserved.

Literature has shown that the topology preserving SOM can be used to track temporal and state changes. Applications for this include process monitoring, financial analysis and geographic analysis. The fact that the SOM can be used for both dimensionality reduction and tracking temporal/state changes has provided motivation

for the work presented in this thesis. Here, the application of path planning to the SOM was explored to investigate whether or not abstract, high-dimensional spaces could be navigated through using this approach.

One of the problematic issues associated with the traditional SOM though, is the border effect, which decreases the accuracy of the mapping. The problem arises due to the two-dimensional, grid structure of the SOM. Hence, the Geodesic SOM was chosen so that the SOM algorithm could be implemented on a geodesic dome to remove the border effect. The distance transformation algorithm was then selected as the method to perform path finding on the SOM. The application of this algorithm would help discover the shortest path on the SOM, where the distance measure is the local distances in the high-dimensional space as calculated by the U-Matrix.

Experiments were conducted to evaluate the results of using this approach. Results showed that this approach compares well against linear interpolation. Furthermore, it can be seen that there are problems associated with the use of linear interpolation. It is difficult and time consuming to determine the amount of vertices that should be used to generate a path such that its distance reflects the distance in the feature space as close as possible while also following the structure of the data closely. We presented in this thesis an alternative method to compute and plot the intermediate states through an automated process that performs comparably well with respect to both of these aspects.

Overall, our work provides new opportunities to the process of visual analytics. The proposed method allows users not only to visually inspect the high-dimensional data, but also to obtain extra visual cues. This might trigger the user to ask different analytical questions that would not have been asked otherwise.

## 8.2 Future work

In this thesis, distance transformations were used to plan paths from one point to another on the SOM. There are a few aspects that could be improved or deserve further research. Both the conventional SOM and Geodesic SOM are known to produce topology preserving mappings that do not preserve the distance relationships. The

interneuron distances are therefore inconsistent. When dealing with temporal data, this makes it difficult to associate a time period to a neuron that lies on the path computed by the method we have presented. Further work would look into modifying the Geodesic SOM so that it is also distance preserving. Preliminary work has been done where the ViSOM updating rule [107] has been adapted for use on the Geodesic SOM. The ViSOM updating rule decomposes the updating force $[x(t) - w_k(t)]$ into two components. One represents the updating force from the winner (BMU) $v$ to the input $x(t)$ ($[x(t) - w_v(t)]$. The other is a contraction force that brings any $k$ in the neighbourhood closer to the winner ($[w_v(t) - w_k(t)]$). A constraint is applied to this contraction force so that distances between neurons on the map are proportional to the distances in the feature space. This results in the following updating rule:

$$w_k(t+1) = w_k(t) + h_{ci}(t)([x(t) - w_v(t)] + [w_v(t) - w_k(t)] \times \beta) \tag{8.1}$$

where

$$\beta = \frac{d_{vk}}{\Delta_{vk}\lambda} - 1 \tag{8.2}$$

where $d_{vk}$ and $\Delta_{vk}\lambda$ are the distances between the neurons $v$ and $k$ in the feature space and on the map respectively. $\lambda$ is a resolution parameter that controls the distance between adjacent neurons in the feature space.

While experimenting with the "Geodesic ViSOM" and ViSOM, problems were encountered where the weight vectors could reach erroneous values when the value of $\beta$ was greater than 2. Attempts were made to contact the author of the ViSOM to first request source code for validation and experimentation. The author replied and mentioned that they do not have a satisfactory, general implementation of the ViSOM but only had code specific to certain data sets. While the author has offered to provide further assistance, we have yet to hear back from them for a long period of time after reporting our findings regarding the erroneous values.

Another aspect that could be improved is in creating paths that avoid areas on the SOM (namely, cluster boundaries) that represent regions outside of the data space when appropriate. The method presented in this thesis to solve this problem

effectively is to perform segmentation on the SOM to restrict the region where path finding is applied. One problem that may occur with this approach is that it may incorrectly classify a neuron to be part of another cluster or cluster boundary when it actually belongs to the same cluster as the start and goal neuron. The application of the watershed transformation to the U-Matrix could help in this regard. This would help separate the visualization such that watersheds would be represented by cluster boundaries and catchment basins by the rest of the visualization. This would help better distinguish which regions would be more desirable to travel through, which in this case would be the catchment basins. Using the distance transformation algorithm for path planning makes implementing the watershed transformation more simple, since both of these could then be implemented using an image foresting transformation-based approach.

Another area worth looking into involves dealing with manifolds. During our experiments, it was seen that it was possible to determine what the structure of the manifold appears like from the set of data points. This can be achieved as our method uses a topology preserving mapping and the local distances in the high-dimensional space calculated by the U-Matrix. This poses an interesting problem for research: given a Geodesic SOM corresponding to an n-dimensional data space, can all the paths between each pair of point be used to find a manifold that fits this data? Solving this problem would enable researchers to understand more about the data's structure and provide an alternative visualization of the data space. Since cartographic projection techniques can be applied to the Geodesic SOM, it would be worth investigating their use in creating two-dimensional embeddings of manifolds. Thus, when the data set contains points sampled from a manifold, users could interactively create different two-dimensional embeddings by manipulating the two-dimensional projection of the Geodesic SOM.

Tryba and Goser have also suggested that path finding techniques could be used for process steering [52]. This particular area of research remains to be explored despite the fact that such techniques could information about the transition path between two states. The results presented in this thesis show promise for such applications. The use of the method proposed in this thesis for benchmarking applications, for

example, is one area that is certainly worth exploring. This could help analysts to track and compare the progress of an object or process. Moreover, the results could help analysts to discover characteristics that would lead toward some desirable state and support the decision making process.

# Bibliography


[1] M. J. Berry and G. Linoff, *Data Mining Techniques: For Marketing, Sales, and Customer Support*. New York, NY, USA: John Wiley & Sons, Inc., 1997.

[2] C. Olaru and L. Wehenkel, "Data mining," *IEEE Computer Applications in Power*, vol. 12, pp. 19–25, July 1999.

[3] Z. Hu, W.-N. Chin, and M. Takeichi, "Calculating a new data mining algorithm for market basket analysis," in *Practical Aspects of Declarative Languages*, vol. 1753 of *Lecture Notes in Computer Science*, (London, UK), pp. 169–184, Springer-Verlag, 2000.

[4] F. Yu, Z. Qin, and X.-L. Jia, "Data mining application issues in fraudulent tax declaration detection," in *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, vol. 4, (Washington, DC, USA), pp. 2202–2206, IEEE Computer Society, November 2003.

[5] J. Hipp, U. Güntzer, and U. Grimmer, "Data quality mining - making a virute of necessity.," in *DMKD*, 2001.

[6] K. Thearling, "Thought on the current state of data mining software applications," 1998.

[7] N. Robinson and M. Shapcott, "Data mining information visualisation - beyond charts and graphs.," in *IV '02: Proceedings of the Sixth International Conference on Information Visualisation (IV'02)*, (Washington, DC, USA), pp. 577–583, IEEE Computer Society, July 2002.

[8] J. J. Thomas and K. A. Cook, eds., *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE CS Press, 2005.

[9] D. A. Keim, "Information visualization and visual data mining," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1–8, 2002.

[10] D. V. Beard and J. Q. Walker II, "Navigational techniques to improve the display of large two-dimensional spaces," *Behaviour and Information Technology*, vol. 9, no. 6, pp. 451–466, 1990.

[11] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proceedings of IEEE Visualization '91*, (San Diego, CA, USA), pp. 284–291, IEEE Computer Society, IEEE Computer Society, October 1991.

[12] J. J. Van Wijk and H. Van de Wetering, "Cushion treemaps: Visualization of hierarchical information," in *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, (Washington, DC, USA), p. 73, IEEE Computer Society, 1999.

[13] F. Vernier and L. Nigay, "Modifiable treemaps containing variable shaped units," 2000. IEEE Information Visualization 2000.

[14] J. A. Hartigan and B. Kleiner, "Mosaics for contingency tables," in *Proceedings of the 13th Symposium on the Interface*, pp. 268–273, Springer-Verlag, 1981.

[15] R. J. M. Dawson, "The "unusual episode" data revisited," *Journal of Statistics Education*, vol. 3, no. 3, pp. 1–7, 1995.

[16] M. Y. Huh, "Line mosaic plot: Algorithm and implementation," in *Proceedings in Computational Statistics*, pp. 277–285, 2004.

[17] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, "Algorithms for drawing graphs: An annotated bibliography," *Computational Geometry: Theory and Applications*, vol. 4, no. 5, pp. 235–282, 1994.

[18] W. Li, S.-H. Hong, and P. Eades, "A framework for visualising large graphs," in *IV '05: Proceedings of the Ninth International Conference on Information Visualisation (IV'05)*, (Washington, DC, USA), pp. 528–535, IEEE Computer Society, 2005.

[19] M. Balzer and O. Deussen, "Level-of-detail visualization of clustered graph layouts," in *Proceedings of Asia Pacific Symposium on Information Visualisation 2007 (APVIS2007)*, pp. 133–140, February 2007.

[20] X. Fu, S.-H. Hong, N. Nikolov, X. Shen, Y. Wu, and K. Xu, "Visualization and analysis of email networks," in *Proceedings of Asia Pacific Symposium on Information Visualisation 2007 (APVIS2007)*, pp. 1–8, February 2007.

[21] G. Fuchs and H. Schumann, "Visualizing abstract data on maps," in *IV '04: Proceedings of the Information Visualisation, Eighth International Conference on (IV'04)*, (Washington, DC, USA), pp. 139–144, IEEE Computer Society, 2004.

[22] A. Skupin, "From metaphor to method: Cartographic perspectives on information visualization," in *INFOVIS '00: Proceedings of the IEEE Symposium on Information Vizualization 2000*, (Washington, DC, USA), p. 91, IEEE Computer Society, 2000.

[23] W. B. Rayward, "Visions of xanadu: Paul otlet (18681944) and hypertext," *Journal of the American Society for Information Science and Technology*, vol. 45, no. 4, pp. 235–250, 1994.

[24] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, vol. 46, no. 2, pp. 234–40, 1970.

[25] A. Skupin, "On geometry and transformation in map-like information visualization," in *Visual Interfaces to Digital Libraries [JCDL 2002 Workshop]*, (London, UK), pp. 161–170, Springer-Verlag, 2002.

[26] D. Charitos, "Designing space in virtual environments for aiding wayfinding behaviour," in *Proceedings of the 4th international conference of the UK Virtual Reality Special Interest Group*, 1997.

[27] R. P. Darken, T. Allard, and L. B. Achille, "Spatial orientation and wayfinding in large-scale virtual spaces ii: Guest editors' introduction," *Presence: Teleoperators and Virtual Environments*, vol. 8, no. 6, pp. iii–vi, 1999.

[28] R. P. Darken and J. L. Sibert, "Wayfinding strategies and behaviors in large virtual worlds," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (New York, NY, USA), pp. 142–149, ACM, 1996.

[29] T. T. Elvins, "Visfiles: virtually lost in virtual worlds wayfinding without a cognitive map," *ACM SIGGRAPH Computer Graphics*, vol. 31, no. 3, pp. 15–17, 1997.

[30] G. Lakoff and M. Johnson, *Meatphors We Live By*. University of Chicago Press, 1980.

[31] W. Kuhn, "Metaphors create theories for users," in *Spatial Information Theory: A theoretical basis for GIS* (A. U. Frank and I. Campari, eds.), vol. 716 of *Lecture Notes in Computer Science*, pp. 366–376, 1993.

[32] H. Couclelis, "Worlds of information: The geographic metaphor in the visualization of complex information," *Cartography & Geographic Information Systems*, vol. 25, pp. 209–220, October 1998.

[33] D. R. Montello, S. I. Fabrikant, M. Ruocco, and R. S. Middleton, "Testing the first law of cognitive geography on point-display spatializations," in *Spatial Information Theory: Foundations of Geographic Information Science* (W. Kuhn, M. Worboys, and S. Timpf, eds.), vol. 2825 of *Lecture Notes in Computer Science*, pp. 316–331, Springer-Verlag: Berlin, 2003.

[34] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, pp. 401–419, 1952.

[35] F. W. Young and R. M. Hamer, *Theory and Applications of Multidimensional Scaling*. Hillsdale, NJ: Eribaum Associates, 1994.

[36] F. W. Young, "Multidimensional scaling," in *Encyclopedia of Statistical Sciences* (Kotz-Johnson, ed.), vol. 5, pp. 649–659, John Wiley & Suns, Inc., 1985.

[37] I. T. Jolliffe, *Principal Component Analysis*. New York: Springer Verlag, 2002.

[38] J. W. Sammon Jr., "A nonlinear mapping for data structure analysis," *IEEE Transactions on Computers*, vol. C-18, pp. 401–409, May 1969.

[39] J. B. Kruskal, "Nonmetric multidimensional scaling: A numerical method," *Psychometrika*, vol. 29, no. 2, pp. 115–129, 1964.

[40] S. Li, O. de Vel, and D. Coomans, "Comparative performance analysis of non-linear dimensionality reduction methods," tech. rep., James Cook University, 1995.

[41] T. Kohonen, *Self-Organizing Maps, Third Edition*. Berlin Heidelberg: Springer-Verlag, 2001.

[42] A. Ultsch and H. P. Siemon, "Exploratory data analysis: Using kohonen networks on transputers," Tech. Rep. 329, University of Dortmund, 1989.

[43] A. Ultsch, "Maps for the visualization of high-dimensional data spaces," in *Proceedings Workshop on Self-Organizing Maps (WSOM 2003)*, pp. 225–230, 2003.

[44] E. Pampalk, A. Rauber, and D. Merkl, "Using smoothed data histograms for cluster visualization in self-organizing maps," in *Proceedings of the International Conference on Artificial Neural Networks*, (London, UK), pp. 871–876, Springer-Verlag, 2002.

[45] R. Lloyd, "Self-organized cognitive maps," *The Professional Geographer*, vol. 52, no. 3, pp. 517–531, 2000.

[46] T. Kohonen, "The 'neural' phonetic typewriter," *Computer*, vol. 21, no. 3, pp. 11–22, 1988.

[47] G. M., T. M., W. M., and H. F., "Introducing geovista studio: an integrated suite of visualization and computational methods for exploration and knowledge construction in geography," *Computers, Environment and Urban Systems*, vol. 26, pp. 267–292, July 2002.

[48] T. Takatsuka, "An application of the self-organizing map and interactive 3-d visualization to geospatial data," in *Proceedings of the 6th International Conference on GeoComputation*, September 2001.

[49] E. L. Koua, "Using self-organizing maps for information visualization and knowledge discovery in complex geospatial datasets," in *Proceedings of the 21st International Cartographic conference (ICC)*, (Durban, South Africa), pp. 1694–1701, The International Cartographic Association (ICA), Document Transformation Technologies, August 2003.

[50] A. Ultsch, "U*-matrix: a tool to visualize clusters in high dimensional data," Tech. Rep. 36, CS Department, Philipps-University Marburg, 2004.

[51] G. Guimarães, V. S. Lobo, and F. Moura-Pires, "A taxonomy of self-organizing maps for temporal sequence processing," *Intelligent Data Analysis*, vol. 7, no. 4, pp. 269–290, 2003.

[52] V. Tryba and K. Goser, "Self-organizing feature maps for process control in chemistry," in *Proceedings of the International Conference on Artificial Neural Networks* (T. Kohonen, K. Mkisara, O. Simula, and J. Kangas, eds.), pp. 847–852, 1991.

[53] M. Kasslin, J. Kangas, and O. Simula, "Process state monitoring using self-organizing maps," in *Artificial Neural Networks* (I. Aleksander and J. Taylor, eds.), vol. 2, pp. 1531–1534, 1992.

[54] A. Ultsch, "Self-organized feature maps for monitoring and knowledge aquisition of a chemical process," in *Proceedings of the International Conference on Artificial Neural Networks 1993* (S. Gielen and B. Kappen, eds.), pp. 864–867, Springer-Verlag, 1993.

[55] C. Serrano-Cinca, "Let financial data speak for themselves," in *Visual Explorations in Finance* (G. J. Deboeck and T. Kohonen, eds.), ch. 1, pp. 3–23, Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1998.

[56] B. Martin and C. Serrano-Cinca, *Self Organizing Neural Netwoks: The Financial State of Spanish Companies*, ch. 23, pp. 341–357. London, UK: John Wiley & Sons, 1995.

[57] K. Kiviluoto and P. Bergius, "Analyzing financial statements with the self-organizing map," in *Proceedings Workshop on Self-Organizing Maps (WSOM 2007)*, pp. 362–367, 1997.

[58] S. Kaski, J. Sinkkonen, and J. Peltonen, "Bankruptcy analysis with self-organizing maps in learning metrics," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 936–947, 2001.

[59] B. Back, K. Sere, and H. Vanharanta, "Analyzing financial performance with self-organizing maps," in *1998 IEEE International Joint Conference on Neural Networks* (A. Stubberud and J. Dayhoff, eds.), 1998.

[60] T. Eklund, B. Back, H. Vanharanta, and A. Visa, "Financial benchmarking using self-organizing maps - studying the international pulp and paper industry," in *Data mining: opportunities and challenges*, pp. 323–349, Hershey, PA, USA: Idea Group Publishing, 2003.

[61] A. Skupin and R. Hagelman, "Visualizing demographic trajectories with self-organizing maps," *Geoinformatica*, vol. 9, no. 2, pp. 159–179, 2005.

[62] H. Chen, A. L. Houston, R. R. Sewell, and B. R. Schatz, "Internet browsing and searching: User evaluations of category map and concept space techniques,"

*Journal of the American Society for Information Science*, vol. 49, no. 7, pp. 582–603, 1998.

[63] C. C. Yang, H. Chen, and K. Hong, "Exploring the world wide web with self-organizing map," in *Proceedings of the International World Wide Web Conference*, (Honolulu, Hawaii, USA), May 2002.

[64] H. Ritter, "Self-organizing maps on non-euclidean spaces," in *Kohonen Maps* (E. Oja and S. Kaski, eds.), pp. 97–110, Amsterdam: Elsevier, 1999.

[65] K. Sahr, D. White, and A. Jon Kimerling, "Geodesic discrete global grid systems," *Cartography and Geographic Information Science*, vol. 30, no. 2, pp. 121–134, 2003.

[66] Y. Wu and M. Takatsuka, "Spherical self-organizing map using efficient indexed geodesic data structure," *Neural Networks*, vol. 19, pp. 900–910, July–August 2006.

[67] Y. Wu and M. Takatsuka, "The geodesic self-organizing map and its error analysis," in *Proceedings of Australian Computer Science Conference 2005*, January/February 2005.

[68] Y. Wu and M. Takatsuka, "Visualizing multivariate network on the surface of a sphere," in *Proceedings of the Asia Pacific Symposium on Information Visualisation (APVIS 2006)*, (Darlinghurst, Australia), pp. 77–83, Australian Computer Society, Inc., 2006.

[69] J. B. Tenenbaum, "Mapping a manifold of perceptual observations," in *NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10*, (Cambridge, MA, USA), pp. 682–688, MIT Press, 1998.

[70] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, December 2000.

[71] T. Similä, "Self-organizing map learning nonlinearly embedded manifolds," *Information Visualization*, vol. 4, no. 1, pp. 22–31, 2005.

[72] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[73] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[74] L. R. Ford Jr. and D. R. Fulkerson, *Flows in Networks*. Princeton University Text, 1962.

[75] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[76] A. Rosenfeld and J. L. Pfaltz, "Sequential operations in digital picture processing," *Journal of the ACM*, vol. 13, no. 4, pp. 471–494, 1966.

[77] K. W. Lau, P. C. Yuen, and Y. Y. Tang, "Edt based tracing maximum thinning algorithm on grey scale images.," in *ICPR '00: Proceedings of the Fifthteenth International Conference on Pattern Recognition*, (Los Alamitos, CA, USA), pp. 2863–2866, IEEE Computer Society, September 2000.

[78] M. E. Rettmann, X. Han, and J. L. Prince, "Watersheds on the cortical surface for automated sulcal segmentation," in *Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis*, (Washington, DC, USA), p. 20, IEEE Computer Society, 2000.

[79] D. W. Paglieroni, "Distance transforms: properties and machine vision applications," *CVGIP: Graphical Models and Image Processing*, vol. 54, no. 1, pp. 56–74, 1992.

[80] I. Ragnemalm, "Fast erosion and dilation by contour processing and thresholding of distance maps," *Pattern Recognition Letters*, vol. 13, no. 3, pp. 161–166, 1992.

[81] C. T. Huang and O. R. Mitchell, "A euclidean distance transform using grayscale morphology decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 4, pp. 443–448, 1994.

[82] V. Starovoitov, "A clustering technique based on the distance transform," *Pattern Recognition Letters*, vol. 17, no. 3, pp. 231–239, 1996.

[83] S. Warfield, "Fast k-nn classification for multichannel image data," *Pattern Recognition Letters*, vol. 17, no. 7, pp. 713–721, 1996.

[84] R. A. Jarvis, "Collision-free trajectory planning using distance transforms," *Mechanical Engineering Transactions of the Institution of Engineers*, vol. ME10, no. 3, pp. 187–191, 1985.

[85] M. Williams and D. I. Jones, "A rapid method for planning paths in three dimensions for a small aerial robot," *Robotica*, vol. 19, no. 2, pp. 125–135, 2001.

[86] L. C. Wang, L. S. Yong, and J. Ang, M.H., "Hybrid of global path planning and local navigation implemented on a mobile robot in indoor environment," in *Proceedings of the 2002 IEEE International Symposium on Intelligent Control*, pp. 821–826, October 2002.

[87] M. Marzouqi and R. A. Jarvis, "Covert path planning in unknown environments with known or suspected sentry location," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Edmonton Alberta, Canada), pp. 1772–17728, August 2005.

[88] D. Z. Chen, R. J. Szczerba, and J. Uhran, J. J., "A framed-quadtree approach for determining euclidean shortest paths in a 2-d environment," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 5, 1997.

[89] G. Song and N. M. Amato, "Using motion planning to study protein folding pathways," in *RECOMB '01: Proceedings of the fifth annual international conference on Computational biology*, (New York, NY, USA), pp. 287–296, ACM Press, 2001.

[90] A. X. Falc ao, J. Stolfi, and R. A. Lotufo, "The image foresting transform: Theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[91] C. B. Jones, "A new approach to the hidden line problem," *The Computer Journal*, vol. 14, no. 3, pp. 232–237, 1971.

[92] D. Luebke and C. Georges, "Portals and mirrors: Simple, fast evaluation of potentially visible sets," in *Proceedings of the 1995 symposium on Interactive 3D graphics*, (California, USA), pp. 105–106, ACM Press, 1995.

[93] N. Lowe and A. Datta, "A new technique for rendering complex portals," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 1, pp. 81–90, 2005.

[94] M. Bui, N. Lowe, and M. Takatsuka, "Visualization of a closed three-dimensional surface using portal-based rendering," in *Proceedings of Asia Pacific Symposium on Information Visualisation 2006 (APVIS2006)* (K. Misue, K. Sugiyama, and J. Tanaka, eds.), vol. 60 of *CRPIT*, (Tokyo, Japan), pp. 35–38, ACS, 2006.

[95] S. M. Seitz and J. Kim, "Multiperspective imaging," *IEEE Computer Graphics and Applications*, vol. November/December, pp. 16–19, November 2003.

[96] S. Vallance and P. Calder, "Multi-perspective images for visualisation," in *CRPITS'11: Proceedings of the Pan-Sydney Workshop on Visual Information Processing*, (Darlinghurst, Australia, Australia), pp. 69–76, Australian Computer Society, Inc., 2001.

[97] L. M. Bugayeviskiy and J. P. Snyder, *Map Projections: A Reference Manual*. London: Taylor and Francis, 1995.

[98] M. Kennedy and S. Kopp, *Understanding Map Projections*. United States of America: ESRI Press, 2000.

[99] B. Fuller, "The fuller projection," 1954. Accessed 27th Oct 2005.

[100] C. Chen, "Information visualization," *Information Visualization*, vol. 1, no. 1, pp. 1–4, 2002.

[101] S. I. Fabrikant and B. P. Buttenfield, "Formalizing semantic spaces for information access," *Annals of the Association of American Geography*, vol. 91, no. 2, pp. 263–280, 2001.

[102] A. Skupin and S. I. Fabrikant, "Spatialization methods: A cartographic research agenda for non-geographic information visualization," *Cartography and Geographic Information Science*, vol. 30, no. 2, pp. 95–119, 2003.

[103] S. I. Fabrikant and A. Skupin, "Cognitively plausible information visualization," in *Exploring Geovisualization* (J. Dykes, M.-J. Kraak, and A. M. MacEachren, eds.), pp. 667–690, Elsevier: Amsterdam, Oxford, 2005.

[104] A. Skupin and R. Hagelman, "Attribute space visualization of demographic change," in *GIS '03: Proceedings of the 11th ACM international symposium on Advances in geographic information systems*, (New York, NY, USA), pp. 56–62, ACM Press, 2003.

[105] D. H. Douglas, "Least-cost path in gis using an accumulated cost surface and slopelines," *Cartographica*, vol. 31, no. 3, pp. 31–51, 1994.

[106] A. Elgammal and C.-S. Lee, "Nonlinear manifold learning for dynamic shape and dynamic appearance," *Computer Vision and Image Understanding*, vol. 106, no. 1, pp. 31–46, 2007.

[107] H. Yin, "Data visualisation and manifold mapping using the visom," *Neural Networks*, vol. 15, no. 8-9, pp. 1005–1016, 2002.